THE UNIVERSITY OF TULSA

THE GRADUATE SCHOOL

NOVEL APPLICATIONS OF LEAST-SQUARES SUPPORT-VECTOR AND GAUSSIAN PROCESS REGRESSION PROXIES TO LIFE-CYCLE PRODUCTION OPTIMIZATION PROBLEMS; CO_2 HUFF-AND-PUFF, WATER ALTERNATING GAS, AND WELL SHUTOFF

by
Azad Almasov

A dissertation submitted in partial fulfillment of
the requirements for the degree of Doctor of Philosophy
in the Discipline of Petroleum Engineering

The Graduate School
The University of Tulsa

2021

T H E U N I V E R S I T Y O F T U L S A

THE GRADUATE SCHOOL

NOVEL APPLICATIONS OF LEAST-SQUARES SUPPORT-VECTOR AND GAUSSIAN PROCESS REGRESSION PROXIES TO LIFE-CYCLE PRODUCTION OPTIMIZATION PROBLEMS; CO_2 HUFF-AND-PUFF, WATER ALTERNATING GAS, AND WELL SHUTOFF

by
Azad Almasov

A DISSERTATION

APPROVED FOR THE DISCIPLINE OF

PETROLEUM ENGINEERING

By Dissertation Committee

Mustafa Onur, Chair
Rami Younis
Reza Rastegar
Sandip Sen
Evren Ozbayoglu

COPYRIGHT STATEMENT

ABSTRACT

Azad Almasov (Doctor of Philosophy in Petroleum Engineering)

Novel Applications of Least-Squares Support-Vector And Gaussian Process Regression Proxies to Life-Cycle Production Optimization Problems; CO_2 Huff-and-Puff, Water Alternating Gas, and Well Shutoff

Directed by Mustafa Onur

257 pp., Chapter 8: Conclusions

(642 words)

In this work, we investigate the applications of least-squares support-vector regression (LS-SVR)- and Gaussian process regression (GPR)-based iterative-sampling-refinement optimization methods on the optimization of different life-cycle production problems, where the objective function to be maximized is the net present value (NPV). The life-cycle production optimization problems considered here are the CO_2 Huff-and-Puff (HnP) problem in unconventional reservoirs, the CO_2 water alternating gas injection (WAG) problem, and the well shutoff problem. For the HnP problem, a multi-staged fractured single-horizontal well is considered in unconventional oil reservoirs. It accounts for the natural fracture and geomechanical effects. Both the deterministic and robust production optimization strategies are considered. The injection rate of CO_2, production bottom-hole pressure (BHP), duration of injection and production periods in each cycle of the HnP process, and cycle lengths for a predetermined life-cycle time are included in the set of design variables. For the WAG problem, we consider a channelized reservoir with three layers, where we have 9 production wells and 4 injectors. Design variables are: gas injection and water injection rates for each injection well at each WAG cycle, production BHP for each production well at each WAG half-cycle, and inflow control valve (ICV) for each well at each WAG half-cycle and at each valve. We consider a rectangular tight oil reservoir with 2 water injectors and 4 producers

for the well shutoff problem. The total duration of life-cycle production of injection wells is divided into uniform injection control-time steps. However, the total duration of life-cycle production of production wells is divided into what we call production cycles where, in each cycle, we have a production period and a shutoff period for each production well. Thus, the design variables considered for the well shutoff problem are water injection rate for each injection well at each injection control time step; production BHP for each production well at each production cycle; production cycle lengths of each production wells; and production time fraction for each production well at each production cycle. For the well shutoff problem, we slightly modify the NPV formulation to include operational expenditures (OPEX) of each well, where shutting off the production wells that will have high expenditure costs becomes crucial. For each of the production optimization problems considered in this study, we consider different scenarios where we fix some of the design variables and try to optimize NPV using other design variables to check the importance of design variables for different production optimization problems.

During optimization, the NPV is calculated by a machine learning (ML) proxy model trained to accurately approximate the NPV that would be calculated from a reservoir simulator run. Given a set of forward simulation runs with a commercial simulator, a proxy is built based on the ML method chosen. Having the proxy model, we use it in an iterative-sampling-refinement optimization algorithm directly to optimize the design variables. As an optimization tool, the sequential quadratic programming (SQP) method is used inside this iterative-sampling-refinement optimization algorithm. The computational efficiencies of these two ML proxy-based optimization methods are compared with that of the conventional stochastic simplex approximate gradient (StoSAG)-based methods.

Results, in general, show that the LS-SVR and GPR models can be efficiently used for the production optimization problems involving the HnP, WAG, and well shutoff with small training data generated from a high fidelity numerical simulator. Higher computational efficiency was achieved using the LS-SVR- and GPR-based iterative-sampling-refinement optimization methods over the finite-difference, simplex, and StoSAG-based optimization

methods using a high-fidelity numerical simulator. It was found that the LS-SVR was computationally more efficient (about 1.5 times) than the GPR for the large-scale optimization problems considered in this study. For the problems where the number of design variables is small, these two ML-based iterative optimization methods are at least 3 to 6 times computationally more efficient, depending on the cases considered, than the gradient-based optimization methods using a numerical simulator.

"If you want to improve, be content to be thought foolish and stupid."
–Epictetus

ACKNOWLEDGEMENTS

I would like to thank my advisor, Dr. Mustafa Onur, for his guidance and patience throughout this journey. His dedication, hard work, curiosity, and friendship will always be an example in my life. I also would like to thank Dr. Albert Reynolds, for his support during this research. I would like to thank my professors Drs. Rami Younis, Stefan Miska, Ovadia Shoham, Richard Redner, Bill Coberly, and Cem Sarica, for the knowledge I gained from their lectures. I also thank Drs. Reza Rastegar, Evren Ozbayoglu, and Sandip Sen, for serving in my dissertation committee and providing useful comments and suggestions.

I cannot forget the support of my friends during my Ph.D. study. I thank Farid, Emin, Aykut, Piri, Abuzar, Brady, Israyil, and Walid for turning me toward reality when I was lost in the things that I exaggerated, and for making life fun. Importantly, I thank my family for supporting me always, and God for keeping the people who love me safe and healthy always, especially during COVID pandemic. I thank God again for keeping my mind, soul, and body healthy and strong, and aligning me with my true self. I dedicate this work to my family.

This research was conducted under the auspices of TUPREP, the Tulsa University Petroleum Reservoir Exploitation Projects, and it was prepared with financial support from this research group.

"Memento Mori, Amor Fati."
–Marcus Aurelius

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

1.1 Background

Closed-loop reservoir management consists of two main sections: model-based optimization and computer-assisted history matching. Fig. 1.1 illustrates a schematic diagram of the closed-loop reservoir management. As can be seen from the diagram, closed-loop reservoir management (also known as "closed-loop optimization") is a sequential process. The "System" section of Fig. 1.1 is the physics of the system, which is a reservoir with wells and facilities. We can use multiple reservoir models which are represented by the system models section in the center of the diagram. Different kinds of reservoir models come from geology, seismic, well logs, well tests, etc., which creates the prior model of the reservoir. This prior model is a statistical model that we use to quantify the uncertainty of our knowledge about the subsurface. We use this prior model to build a reservoir model by using a numerical simulator which is used in the section of optimization algorithms. The "Sensor" section of the diagram can be thought of as real sensors measuring flow rate, bottom hole pressure (BHP), etc. The blue colored section on the left is for production optimization where by using the uncertain reservoir model in the system (which reservoir simulator), production optimization is performed to find optimal well controls as well as other design variables by maximizing an objective function such as recovery factor or net present value (NPV). In Fig. 1.1, we see optimum design variables represented as controllable inputs. Then using this input in the system (field) we obtain the output. Input can contain noise due to uncertainty of system boundaries and initial conditions, etc. The output can be interpreted as noise due to the noise in the sensors. The output contains information regarding uncertain reservoir model parameters. Therefore, using the section of data assimilation algorithms, the uncertainty of the reservoir parameters decreases as the output from the field is used with the

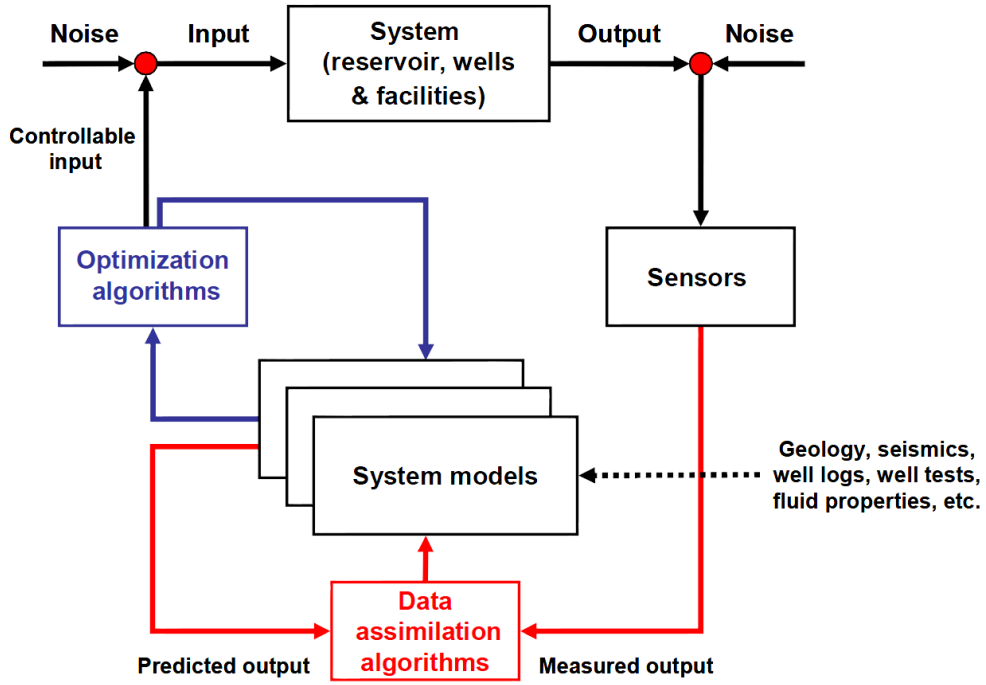


Figure 1.1: Schematic diagram of closed-loop reservoir management (Jansen et al., 2009).

prior distribution on reservoir parameters to obtain a posterior distribution. This can be thought of as calibration (or also referred to as the computer-assisted history matching) of the reservoir model parameters in the reservoir simulator, which is a predictive model, based on the comparison of the predicted output and measured output (Jansen et al., 2009).

In this research, however, we mainly focus on the production optimization section of the closed-loop reservoir management diagram where we also consider the uncertainty in the reservoir model as well. This is also known as life-cycle production optimization. When we consider only a single realization of the reservoir model, it is referred to as the deterministic life-cycle production optimization, while considering multiple realizations of the reservoir model during optimization is called robust life-cycle optimization. The system here corresponds to one of three production problems we would like to optimize. We consider three different improved-oil-recovery (IOR) processes. The objective function to be optimized is the NPV for each of these processes, but the formulation of NPV for each of the processes may differ due to different economical considerations and parameters. Most optimization techniques used for the life-cycle production optimization are gradient-based approaches, in

which derivative information of the objective function is required at every iteration. Some of the optimization techniques for life-cycle production optimization use gradient-free optimization methods such as Genetic Algorithm (GA), Branch and Bound (B&B), Mesh Adaptive Direct Search (MADS), Particle Swarm Optimization (PSO) (Isebor et al., 2014; Lu and Reynolds, 2020; Tavakkolian et al., 2004; Tabatabaei Nejad et al., 2007). In this work, we consider gradient-based optimization. In the past, many researchers mainly focused on optimization of the reservoir performance using conventional high-fidelity reservoir model-based optimization where they use a stochastic approximation of the gradient to be used in the gradient-based optimizations which usually require thousand or more of simulation runs such as StoSAG and EnOpt (ensemble optimization) (Do and Reynolds, 2013; Chen et al., 2015; Chen and Reynolds, 2017; Chen et al., 2017; Chen and Reynolds, 2018; Hewson et al., 2017; Bahagio, 2013; Fonseca et al., 2015c,d, 2017; Chen et al., 2009; Lu et al., 2017; Feng et al., 2017). Very few of them modified the source code of the reservoir simulator to use adjoint gradient in the gradient-based optimization, which is computationally more efficient than using the stochastic gradient approximation, but it requires significant time and effort to modify the source code (Kraaijevanger et al., 2007; Forouzanfar et al., 2010). In addition, adjoint gradient needs to be coded for different production problems separately. Therefore, the main objective of this study is to explore the use of efficient optimization methods which do not require modifying the source code, can be used for any optimization problem independently, and are also efficient than using high fidelity reservoir model-based optimization. Liu and Reynolds (2020) also states that using the full-physical reservoir simulation in production optimization is a computationally expensive process. They also mention that significant computational time can be saved if the appropriate proxy models are selected to replace the reservoir simulation during the production process. Using learning-based data-driven models is one of the most popular low-fidelity models (proxy models). In their study, they used gradient-enhanced support-vector regression (GE-SVR) for robust life-cycle production optimization. However, to use GE-SVR, they needed to compute the gradient for each training point using the adjoint gradient, which requires modifying source code to compute

the adjoint gradient. Their computational results show that with GE-SVR-based iterative optimization method requires on the order of five times fewer reservoir simulation runs than required for the simulator-based optimization with the adjoint solution.

In this study, an iterative-sampling-refinement optimization method, which is based on the proxy models constructed by two different ML methods; least-squares support-vector regression (LS-SVR) and Gaussian process regression (GPR), is applied to three different production optimization problems which their optimization by these two proxies have not been considered previously in the literature. These three production optimization problems are CO_2 Huff-and-Puff (HnP) in unconventional oil reservoirs, CO_2 water-alternating-gas (WAG) in conventional reservoirs and production optimization including well shutoff option to investigate applicability and efficiency of this method over conventional gradient-based optimization methods such as EnOPT and StoSAG. As mentioned previously, the use of two different ML-based iterative-optimization methods such as LS-SVR- and GPR-based proxy models in the iterative-sampling-refinement optimization to life-cycle production optimization problems are investigated, and their differences are compared and documented. With the use of the iterative-sampling-refinement optimization method, our aim is to achieve more computational efficiency over the use of the StoSAG optimization method. The efficiency to be gained can change depending on the production optimization problem, we are considering because different production problems have different numbers and types of design variables as well as different physical processes. For instance, HnP and WAG processes are usually modeled using a compositional reservoir simulator which uses different PVT models and physics in the treatment of fluid flow in porous media than those of a black oil reservoir model. Also, considering the HnP problem is in an unconventional reservoir, we considered geomechanical effects. We also include molecular diffusion. For the well shutoff production optimization problem, we consider the waterflooding process in conventional reservoirs where we use a black oil simulator. Here the physics of the production problem is different from both the HnP problem in unconventional reservoirs and the WAG problem in conventional reservoirs. The NPV formulation as well as the design variables considered for each pro-

duction optimization problem are different as well. Therefore, the production optimization problems considered in this research have significant differences. For each production optimization problem, we also consider different cases at which a number of design variables change even though the physics of the problem is the same. The reason for considering a variety of production optimization problems is to be able to generalize the application of the iterative-sampling-refinement optimization problem so that we can make solid and general conclusions on its efficiency over StoSAG optimization. For the production optimization of the HnP, we also consider the geological uncertainty of the reservoir to perform robust optimization, where the objective function is the expectation of NPV over the uncertain reservoir model parameters.

For the HnP problem, we also investigate the effects of the miscibility and molecular diffusion and discuss the physics of the mass transport mechanism in unconventional reservoirs since it is our first production optimization problem we consider. We perform sensitivity analysis on HnP process to investigate the physics of the problem and to check the effect of each design variable on production optimization so that we can relate our production optimization results to whether they are consistent with our sensitivity analysis or not. We also mention the problems we face during using a commercial compositional reservoir simulator (GEM, 2020) and gave guidelines for the readers on how to properly choose numerical variables of the solver so that they do not face convergence issues when they use this commercial compositional simulator. However, for other production optimization problems, we do not conduct sensitivity analysis, and we mainly focus on the application of the iterative-sampling-refinement optimization problem.

1.2 Literature Review

1.2.1 Life-cycle Production Optimization of Huff-and-Puff Process

Production of oil from unconventional shale-oil/tight-liquid sources is of big interest to industry and governments across the globe due to the increasing scarcity of conventional

oil reserves. A good review and description of such resources can be found in Bohacs et al. (2013). One of the disadvantages of producing oil from these reservoirs is that the oil production rate declines sharply due to low permeability of the matrix system after a few years of primary production. The primary oil recovery factor (RF) of shale-oil formations is less than 10% (Clark, 2009), even though long horizontal wells are drilled and fractured (Christensen et al., 1998). Therefore, IOR/EOR methods are usually applied to improve RF after primary production. This has motivated us to investigate the challenging problem of life-cycle production optimization for efficient production oil from such reservoirs by considering cyclic miscible gas such as CO_2 injection.

In this study, we consider unconventional shale-oil/tight-liquids systems. In this kind of reservoir, permeability ranges from 50 to 10^6 nD. The Bakken formation is considered to be an important source of light shale oil in North America (Jin and Sonnenberg, 2014). Therefore, we considered the formation and composition of the Bakken reservoir in our research.

The oil production rate declines sharply due to the low permeability of the matrix system after a few years of primary production. The primary oil recovery factor (RF) of shale-oil formations is usually less than 10% (Clark, 2009), despite the fact that long horizontal wells are drilled and fractured (Christensen et al., 1998). Therefore, IOR/EOR methods are usually applied to improve RF after primary production.

Cyclic CO_2 injection, also known as the CO_2 huff-and-puff (HnP) process, appears to be an attractive EOR/IOR method that could be applied to provide additional recovery from unconventional oil reservoirs. HnP is a single-well process in that both injection and production are performed in the same well, unlike waterflooding or gas flooding processes where more than one well is involved. However, in a field, individual HnP processes can be conducted at multiple wells. Although hydrocarbon gases could also be used in an injection period of an HnP process, the CO_2 can be preferred as an injection fluid. The reasons for the CO_2 to be used as injection gas are: at high pressures, higher than 1160 psi, its density is high enough to avoid significant gravity segregation; its viscosity is low enough so

that it is highly mobile, but not so low that it causes fingering issue (Jarrell et al., 2002). Performing experimental studies on tight core samples by considering four different recovery schemes; namely, waterflooding, immiscible CO_2 HnP, near-miscible CO2 HnP, and miscible CO_2 HnP processes, Song et al. (2013) found that waterflooding showed better performance than immiscible CO_2 injection, but miscible CO_2 injection performed the best. However, to achieve miscibility for CO_2, we have to maintain reservoir pressure above the minimum miscibility pressure (MMP) (Yu et al., 2014). Each cycle of the CO_2 HnP process consists of three periods: injection, soaking where we shut in the well for some time, and production. Many cycles may be used during a reservoir's life. HnP starts after reservoir pressure has been depleted to some level. A compositional model should be used to handle mass transport rigorously in a miscible process including molecular diffusion. We use the CMG-GEM (2016) compositional simulator to simulate a single-well CO_2 HnP process (GEM, 2016).

Song et al. (2013); Gamadi et al. (2014); Yu et al. (2016); Alharthy et al. (2018); Alfarge et al. (2017); Tovar et al. (2018a,b) conducted experimental studies and Artun et al. (2011); Yu et al. (2014); Sun et al. (2016); Yu et al. (2019); Joslin et al. (2018); Fragoso et al. (2018); Gala et al. (2018); Pankaj et al. (2018); Wang et al. (2019); Ganjdanesh et al. (2019) performed numerical simulation studies for investigation of HnP process. Field case-based simulation studies can be found in Todd et al. (2016); Hoffman et al. (2018); Wang et al. (2018); Orozco et al. (2019). All these cited works present sensitivity studies investigating the effect of different design variables; e.g., the injection time, injection rate, soaking time, production time, and production well bottomhole pressure (BHP), as well as the effect of the different reservoir and fracture flow parameters such as fracture network and conductivity, matrix permeability, etc., on RF during CO_2 HnP process, which is a forward problem. However, some of these studies (Todd et al., 2016; Yu et al., 2014; Wang et al., 2018; Orozco et al., 2019) consider history matching of oil production rate and cumulative oil production, which is an inverse problem. To the best of our knowledge, Artun et al. (2011) were the first to develop an artificial neural network (ANN)-based proxy models for predicting cumulative oil production and oil production rate screening and optimization. However, they did not

use an iterative-sampling-refinement optimization framework to perform the optimization. In this work, we prefer to consider kernel-based machine learning methods such as least-squares support-vectors and Gaussian regression processes as they are computationally less expensive than an ANN-based machine learning method which requires more training data to construct an appropriate proxy model. Besides, Artun et al. (2011) used a genetic algorithm (GA) coupled with ANN-based proxy models to maximize the discounted oil volume produced per injected volume for a specified period of operation. In this study, we prefer to use a kernel-based proxy coupled with a gradient-based optimization method (namely sequential quadratic programming - SQP) method due to its significant efficiency over a gradient-free GA method for optimization problems based on continuous design (or optimization) variables. To optimize the HnP control variables by maximizing the lower confidence interval ($\mu - \sigma$) of cumulative oil production Hamdi et al. (2019) used Gaussian Processes.

In this study, the objective function considered is NPV, and the design variables are used are different from what was used in the literature. New design variables for the HnP process are defined as the length of each cycle, and injection and production time fractions of each cycle besides well-controls. Thus, the formulation NPV objective function is also different from what was used in the literature. Linear equality constraints on cycle lengths are handled using the gradient projection method.

1.2.2 Life-cycle Production Optimization of Water-Alternating-Gas Injection Process

WAG is a cyclic process as well. However, in this process, unlike the HnP process, there is no soaking period, and water is injected after the gas injection alternately. So, each injection cycle of a given injection well has two injection half-cycles; gas injection half-cycle and water injection half-cycle. Furthermore, it is not a single-well process because it requires injection wells and production wells separately. There can be multiple injection and production wells. WAG is a well-known EOR method applies in the oil industry to improve recovery factor. WAG process is mainly used for conventional reservoirs.

Most literature on CO_2 WAG optimization is based on reservoir simulation experi-

ments, that is sensitivity analysis (Bender et al., 2014; Zhou et al., 2012; Ghaderi et al., 2012; Johns et al., 2003). However, with an experimental approach, the well-operating conditions may not be close to the optimal solution. Therefore, it is important to have an automatic algorithm to find the optimal solution using the information of the NPV such as the gradient information, Hessian information, etc. Most of the researchers performed linear programming which requires only gradient information for the optimization problem (Bahagio, 2013; Chen and Reynolds, 2018; Hewson et al., 2017). Bahagio (2013) used the EnOPT algorithm for the CO_2 WAG injection process. However, he did not investigate the effect of the duration of each injection half-cycle for each injection wells on production optimization. Hewson et al. (2017) performed the ensemble-based optimization of robust life-cycle production optimization of CO_2 WAG including cycle lengths for each injection wells as design variable as well as well controls for a full-field model. The drawback of their methodology is that they used a chopping strategy to handle the linear equality constraints which can lead to the suboptimal solution. Later, Chen and Reynolds (2018) also included the duration of each injection half-cycle as design variables besides the well control variables using StoSAG optimization with equality constraints. However, they handled these constraints using the augmented Lagrangian method instead of using the chopping strategy.

By controlling design variables of the WAG process such as water injection rate, gas injection rate, and production BHP, one can achieve a higher NPV. However, inflow-control-valves (ICVs) (also known as inflow-control-devices (ICDs)) have shown to be important as their use improves oil production and recovery factor when petro-physical properties of the reservoir layers are crucially different such as the significant difference of permeability field between layers (Naroso et al., 2010; Alsyed and Yateem, 2013; Al-Muailu et al., 2013). Dossary et al. (2012); Al-Khalifa et al. (2013); Chan et al. (2014), in their research widely discuss how using smart completion, such as ICVs help to achieve higher recovery factor. Some of the researchers considered ICV as a continuous variable varying between 0 and 1 (Naus et al., 2004), some of them used ICV as an on/off option taking value only 1 and 0 only (Pari and Kabir, 2009; Fonseca et al., 2015a). Naus et al. (2004) modeled ICV as

a multiplication factor of the productivity index (PI) in the production optimization using sequential linear programming (SLP). Chen and Reynolds (2017) showed the importance of including ICVs as design variables besides the well controls such as gas injection rate water injection rate and production BHPs by introducing the smart completion strategy They performed gradient ascent optimization method with StoSAG to perform robust life-cycle production optimization of WAG process including ICVs to maximize NPV. Later, they also included the duration of the WAG injections (e.g., half-cycle length) for each injector as design variables besides well-controls and ICVs to maximize NPV as mentioned above (Chen and Reynolds, 2018). They observed that including the length of each half-cycle as a design variable improves NPV only when optimal well controls are close to their bounds. However, when optimal well controls are far from their bounds, this improvement of NPV is negligible.

In this research, we also include the ICV as a design variable and show its importance to maximize NPV. Different optimization problems are considered under the production optimization of the WAG process such as including ICVs as design variables and not including. As mentioned in Section 1.1, as part of the main focus of this research, LS-SVR- and GPR-based iterative-sampling-training optimization method are applied to different production optimization cases of the WAG process to investigate the applicability of this optimization method to the WAG process under consideration of a different combination of design variables. As the second focus of this research, the computational efficiency of iterative-sampling-refinement optimization is compared with the StoSAG method, where we use a high-fidelity reservoir model. It should be noted that a different formulation of the NPV objective function from the NPV formulation used in the literature is proposed which is named as the general NPV formulation for the production optimization of the WAG process. Similar to the formulation used for the HnP process in this research, the design variables considered are the duration of each injection cycle, gas injection time fraction at each cycle beside the design variables such as ICVs and well controls. The gradient projection method is used to handle linear equality constraints on the cycle lengths.

1.2.3 Life-cycle Production Optimization Including Well Shutoff

The well shutoff optimization problem refers to optimizing the time at which a particular well is either open or closed for maximizing the net present value. It is the focus of most of the operating companies as the companies desire the shutoff to be part of the optimization procedure since the NPV formulation they use also includes the operation expenditures (OPEX) for each well. Once OPEX of each well is included in the NPV formulation, to shut off production wells that do not produce to cover OPEX of that production well should become part of the life-cycle production optimization. There are different approaches to include the well shutoff option for the life-cycle production optimization. One of them is to define a binary design variable to shut well on/off, which constitutes a discrete optimization problem. However, it is not efficient to handle this optimization problem by using gradient-free methods such as a genetic algorithm or particle swarm, etc, especially well shutoff optimized along with the other well controls such as injection rate and production bottomhole pressures (BHP) at each well. To be able to apply a gradient-based optimization, we need to have a continuous optimization problem, i.e., we need all design variables to be continuous. One approach is using production BHPs or production flow rates at each control time step of each production well. When the BHP increases to a value higher than that of the well-block pressure, the well automatically shutoffs due to a small pressure difference. However, this leads to a discontinuous NPV with respect to BHP at the points close to the well-block pressure. This is because once we shut off the well, the NPV jumps due to zero OPEX. Similarly, when we use production rate as a design variable at each control time step of each production well, at zero rates at given control time step of given production well, OPEX of that production well will be zero and there will be a jump in the NPV value. To be able to understand the reason for this discontinuity with respect to BHP, we suggest the reader check Section 2.3. In that section, the formulation of the NPV including OPEX is formulated, and the discontinuity is visualized using the example of waterflooding case. Another approach to this is to define switching-times for the well shutoff as design variables which is going to be continuous. (Sudaryanto and Yortsos, 2000) looked at the optimization of fluid

front dynamics in porous media using rate control. They observe that the rate was better to be controlled either fully open or closed, not continuously. To do so, they defined switching-time as the change of the rate from off to on, and on to off. They did not use this option to deal with discontinuity, but to improve their objective function by replacing continuous rate control with on/off. Even though they do not have NPV as an objective function, and they considered a totally different optimization problem from the one considered in this study, this method can be applied to any optimization problem that requires an on/off option. For example, Fonseca et al. (2015a) used switching-time-interval parameterization to shut-off the inflow-control devices (ICDs) modeled after the switching-time-optimization method provided by Sudaryanto and Yortsos (2000). However, in this approach, the last switching-time-interval needs to be truncated so that summation of all switching-time-intervals must be equal to the life of the production. The objective of this research is to handle discontinuity in the NPV objective function so that gradient-based optimization can be applied. In this research, therefore, a similar method to that of Fonseca et al. (2015a) is used, but the difference is in different approach for the parameterization of the switching-time-intervals, and constraints on design variables as well as the optimization problem. For this purpose, we use a similar parameterization to the HnP problem (Almasov et al., 2020b), that is we define a fixed number of cycles, length of each cycle, and production time fraction. The remaining fraction of each cycle is the shutoff time fraction. We use linear equality constraints so that summation of the length of each cycle is equal to the life of the production, and thus, we do not need to truncate the length of the last cycle. Linear equality constraints are handled using the gradient projection method. Both the stochastic simplex gradient optimization and the machine learning-based iterative-sampling-refinement optimization methods such as LS-SVR and GPR for the life-cycle optimization of well shutoff problems are considered, and the computational efficiencies of the methods are compared.

1.2.4 Iterative-Sampling-Refinement Optimization

To the best of our knowledge, the use of an LS-SVR machine learning method for

the life-cycle optimization problem is first considered by Guo and Reynolds (2018) who presented a new and general workflow for efficient estimation of the optimal well controls for the robust production-optimization problem using support-vector regression (SVR), where the objective function is the net present value (NPV). They used the so-called iterative-sampling-refinement optimization algorithm. All the examples presented by Guo and Reynolds (2018) consider production under waterflooding in conventional oil reservoirs which are simulated by a using black oil commercial simulator. However, Nwachukwu et al. (2018) performed machine learning (ML)-based optimization of well locations and WAG parameters. They used the Extreme Gradient Boosting method (XGBoost) as the ML method to build a proxy for making predictions given a set of observations. Then they introduce hyperdimensional, simultaneous optimization of well location and well controls of WAG using a new optimization scheme which is similar to Mesh Adaptive Direct Search (MADS). With this algorithm, they achieved more efficiency than using high-fidelity reservoir model-based optimization. In their applications, the proxy-based optimization required 880 simulation runs, 500 simulation runs being required for constructing the proxy model. They achieved 11 times more computational efficiency over joint optimization with a reservoir simulator. However, the optimization problem they looked at was a discrete optimization problem. In our research, we look at continuous optimization, where all our variables are continuous. The XGBoost method requires a lot of training data to have an accurate enough model (500 training data in their application) to perform optimization and prediction. Since discrete optimization using a high-fidelity reservoir model requires much higher reservoir simulation runs (ten thousand) than continuous optimization, using this framework achieves higher computational efficiency. However, continuous optimization using a high-fidelity reservoir model requires thousands of simulation runs (most of the time up to 2000 simulation runs). So, the WAG production optimization problems we consider are different from their problem. Kernel-based ML methods require less training data even for very high dimensional problems, such as the number of design variables being more than 200. Therefore, we used kernel-based proxies under the iterative-sampling-refinement optimization framework to achieve higher computational

efficiency in gradient-based production optimization problems we considered.

As the kernel-based ML methods, we consider the use of the Gaussian regression process (GPR) in addition to the LS-SVR on different life-cycle production optimization problems. There are several books and papers published in the machine learning literature; for example see Williams (1998); Seeger (2004); Williams and Rasmussen (2006); Liu et al. (2020); Gramacy (2020), advocating the use of the GPR methods to solve hard machine learning problems. Based on the ML literature, it seems that the GPR-based methods are attractive because of their flexible nonparametric nature and more importantly, treated within a Bayesian framework, they offer valid estimates of uncertainties in our predictions and generic model selection procedures cast as nonlinear optimization problems. Hence, in this work, we investigate the use of two different ML-based methods; the GPR- and LS-SVR-bases proxy models, for the production optimization problem of a miscible CO_2 HnP problem in unconventional oil reservoirs, WAG injection problem, and well shutoff problem. In the well shutoff problem, we consider the waterflooding problem. However, the NPV formulation used is different from the NPV formulation used for HnP and WAG problems considering OPEX for each well. Thus, the NPV objective function, as well as the design variables defined for each production optimization problem, are all different from what were defined in the literature.

1.2.5 Robust Optimization

As mentioned in Section 1.1, reservoir parameters are uncertain. The uncertainty of the reservoir parameters at the initial stage of the reservoir is quantified by the geological, geophysical, well-log data, well test, etc., as shown in Fig. 1.1, which creates the prior probability distribution function (PDF) over the uncertain reservoir parameters. The variance of this uncertainty narrows down at the later stages of the reservoir when we start producing and getting the data from sensors. However, there will always be uncertainty in the reservoir models. NPV is sensitive to the reservoir model parameters, especially the permeability field. Therefore, chosen reservoir model affects the optimization results, optimum design

variables significantly. Since we never know the true reservoir model, it is important to include the reservoir uncertainty in the life-cycle production optimization. When production optimization incorporates geological uncertainty into the life-cycle production optimization, it is called robust optimization. Many researchers showed the importance of considering the uncertainty of the geological model on the production optimization results (Capolei et al., 2015; Chang et al., 2015; Chen and Oliver, 2010; van Essen et al., 2009).

The commonly used objective function to be optimized in robust optimization is the expectation of the NPV over the uncertain reservoir parameters. Most of the researchers performed gradient-based robust optimization, using the steepest ascent algorithm. To calculate the gradient, very few of them used gradients computed with the adjoint method (van Essen et al., 2009; Capolei et al., 2015; Chen, 2011; Forouzanfar et al., 2013; Jansen, 2011; Sarma et al., 2006). However, as mentioned in Section 1.1, implementation of the adjoint gradient is difficult since it requires modifying the source code of the simulator. Therefore, using an approximate gradient (numerical gradient) caught the attention of most researchers. The finite-difference (FD) method is one of the numerical approximation methods for gradients. However, when the number of design variables is more than ten, it is not efficient to use the FD method. Therefore, in the literature, the stochastic gradient method is frequently used. This method is inspired by the Ensemble Kalman Filter (EnKF) method (Lorentzen et al., 2006; Nwaozo, 2006). For robust optimization, Chen and Oliver (2010) formulated EnOpt method first time. They suggested that for robust optimization one-to-one pairing of each perturbation of design variable with each of the realization of the uncertain reservoir model parameters should be enough for gradient approximation for the expectation of the NPV, which is the objective function of the robust production optimization problem. However, Fonseca et al. (2017) mathematically showed that standard formulation of the EnOpt for robust optimization, where one-to-one correspondence is used, gives poor gradient approximation. However, increasing the number of perturbations corresponding with each realization of the reservoir model can yield a better result. However, this increases the computational costs (Fonseca et al., 2015b). Therefore, later Do and Reynolds

(2013) modified EnOpt and formulated modified-EnOpt. Fonseca et al. (2017) modified the EnOpt to a more accurate and efficient formulation for robust production optimization and referred to it as a stochastic simplex approximate gradient (StoSAG). In their research, they also discussed singly- and doubly-smoothed StoSAG and gave the theoretical explanation for each formulation as well as the relation between formulations. In this study, we will use the StoSAG method for robust optimization cases. For the deterministic optimization cases, where the single realization of the reservoir model is used, we used the simplex method to approximate gradient for production optimization problems of HnP and well shutoff, but for the WAG problem, we used the singly-smoothed cross-covariance StoSAG (ss-cc-StoSAG) with the single realization of the reservoir model. The reason for using ss-cc-StoSAG for the optimization of the WAG problem is because we want to consider the correlation between well controls for each production and for each injection well. This reason will be explained more in Section 3.2.

However, in this study, the ML-based iterative-sampling-refinement optimization method is utilized to perform robust optimization, where the objective function is the expectation of the NPV over the uncertain reservoir model parameters. The computational efficiency of this optimization method is compared with that of StoSAG method. As ML model, the GPR and LS-SVR are used and their computational efficiencies, as well as the accuracy, are compared over each other.

1.3 Research Objectives and Dissertation Outline

1.3.1 Research Objectives

We can briefly summarize the objectives of this research as follows:

1. Give theoretical background on physics of the HnP process in unconventional reservoirs with natural fractures and including geomechanical effects.

2. Perform sensitivity analysis for HnP process.

3. Investigate convergence issues in GEM (2016) and discuss how to solve them.

4. Formulate NPV for optimization problem of:
    (a) HnP problem;
    (b) WAG problem;
    (c) Well shutoff problem.
5. Give theoretical background on:
    (a) stochastic gradient-based optimization methods;
    (b) LS-SVR;
    (c) GPR;
    (d) iterative-sampling-refinement optimization method.
6. Perform iterative-sampling-refinement optimization on life-cycle production optimization problems of different production processes to check the applicability of this method and compare the efficiency of the iterative-sampling-refinement optimization method with stochastic gradient-based optimization methods. For the HnP, perform robust optimization as well as deterministic optimization.
7. Compare the performance of LS-SVR-based iterative-sampling-refinement optimization with GPR-based iterative-sampling-refinement optimization.
8. Provide useful guidelines for constructing ML-based proxies that can be used to perform efficient deterministic and robust life-cycle optimization.
9. Provide different formulations to handle discontinuity to perform gradient-based life-cycle production optimization including well shutoff option.

1.3.2 Dissertation Outline

The thesis is organized with eight chapters as follows:

In Chapter 2, we present the formulations of the NPV objective function for production optimization problems of HnP, WAG, and well shutoff. In that chapter, we define

the design variables as well as the constraints on those design variables for each production optimization problem.

In Chapter 3, we introduce the theory of stochastic gradient-based optimization methods, where we discuss: gradient approximation methods, such as simplex, StoSAG in general (including singly-smoothed cross-covariance StoSAG (ss-cc-StoSAG) and double-smoothed cross-covariance StoSAG (ds-cc-StoSAG)); constraint handling methods, such as truncation and gradient projection methods; gradient ascent algorithm and convergence criteria.

In Chapter 4, we focus on the ML-based iterative-sampling-refinement optimization method. First, we give a theoretical background of two different kernel-based machine learning methods; LS-SVR and GPR. Then, we introduce the iterative-sampling-refinement optimization algorithm.

In Chapter 5, we apply stochastic gradient-based optimization and iterative-sampling-refinement optimization methods on the HnP process for both deterministic cases and for robust cases, and compare the computational efficiency of these two optimization methods. Before doing that, we discuss the physical process of the miscible CO_2 injection process in unconventional reservoirs. We also perform sensitivity analysis of the HnP process with respect to design variables as well as reservoir parameters. The reason we investigate the HnP process deeper than other production optimization problems considered here is that it is our first optimization problem considered here and therefore we want to investigate the physical reason for the optimization results.

In Chapter 6, we perform stochastic gradient-based optimization and iterative-sampling-refinement optimization methods on the WAG process, and we compare the computational efficiency of two optimization methods. First, we investigate the effect of choice of the upper and lower bounds of the BHP of each production well on the training data and the accuracy of the initial proxy model as well as the optimization results of the iterative-sampling-refinement optimization method. Then we develop the strategy of choosing bounds of BHP based on these results. Later we consider three main sub-cases such as considering water and gas injection rates at each control time step of each injection wells as only design

variables; then we include BHP as design variable; at the last, we also include the ICVs as design variables besides well controls. We compare different sub-cases to see the importance of including BHP and ICV as a design variable and their effect on the optimization results as well as the accuracy of the proxy models such as LS-SVR and GPR.

In Chapter 7, first, we introduce the novel methodology to solve production optimization problems including well shutoff by defining new design variables such as production cycle and production time fraction with the same analogy used in the optimization problem of the HnP. Then we perform both stochastic gradient-based optimization and iterative-sampling-refinement optimization methods to find optimum design variables to maximize NPV, and we compare optimization results of these two methods as well as their computational efficiencies. We have to mention for the NPV of well shutoff optimization we also consider operational expenditures (OPEX) of each well.

In Chapter 8, we summarize the conclusions and give useful discussion based on the results of this research.

CHAPTER 2

NET PRESENT VALUE (NPV) FORMULATIONS AND DESIGN VARIABLES

Life-cycle production optimization of an oil field is one of the main objectives of petroleum companies. The objective of life-cycle production optimization is an estimation of the optimal design variables that maximize the net present value (NPV) relevantly formulated as an objective (cost) function to obtain the maximum economic benefit. To find the maximum is an optimization problem given the relevant formulation of the objective function (NPV). The size and the types of design variables, as well as formulation of the objective function, can change depending on the production problem. Examples of design variables are production BHPs, injection rates, injection times, well locations, type of well (injector or producer), and shutoff option of the wells. Recently, oil and gas companies are interested in optimizing the time to shut off a well to eliminate its operating cost in the overall NPV of the life-cycle production operation. In the following sections, for each production optimization problem, first, we define the design variables relevant to that optimization problem, then formulated NPV. We also define constraints on the design variables for each production optimization problem. Finally, the normalization of the design variables for each problem is given for each production optimization problem.

2.1 NPV Formulation for the CO_2 HnP Problem

To formulate the NPV objective function for the single-well HnP process, the following definitions and assumptions are used:

- N_c, the total number of cycles, is fixed during the optimization process.
- Δt^n is the duration of the n^{th} cycle, where $n = 1, 2, ..., N_c$.

- The total duration of the HnP process, $t_{total} = \sum_{n=1}^{N_c} \Delta t^n$, is known and fixed during the optimization process.
- $\widehat{\Delta t}_i^n$, $\widehat{\Delta t}_{soak}^n$, and $\widehat{\Delta t}_p^n$ represent time fractions of injection, soaking, and production periods, respectively, for the n^{th} cycle. They can be treated as unknowns in optimization, and defined by

$$\widehat{\Delta t}_p^n = \frac{\Delta t_p^n}{\Delta t^n}, \qquad \widehat{\Delta t}_i^n = \frac{\Delta t_i^n}{\Delta t^n}, \quad \text{and} \qquad \widehat{\Delta t}_{soak}^n = 1 - \widehat{\Delta t}_p^n - \widehat{\Delta t}_i^n, \tag{2.1}$$

  where Δt_p^n and Δt_i^n (days) are the duration of production period and injection period in the n^{th} cycle, respectively. Note that specifying the duration of injection and production periods determines the duration of the soaking period.
- $\delta\Delta t_{i,j}^n$ (days), for $j = 1,2,\ldots,N_i^n$, is the length of the j^{th} control-time step for the injection period in the n^{th} cycle, and $\delta\Delta t_{p,j}^n$ (days), for $j = 1,2,\ldots,N_p^n$, is the length of the j^{th} control-time step for the production period in the n^{th} cycle. N_i^n and N_p^n denote the numbers of injection and production control-time steps, respectively, and they are known in optimization. In the case we use uniform control-time steps for injection and production periods, then $\delta\Delta t_{i,j}^n = \delta\Delta t_i^n$, for $j = 1,2,\ldots,N_i^n$ and $\delta\Delta t_{p,j}^n = \delta\Delta t_p^n$, for $j = 1,2,\ldots,N_p^n$

$$N_p^n = \frac{\Delta t_p^n}{\delta\Delta t_p^n} \quad \text{and} \quad N_i^n = \frac{\Delta t_i^n}{\delta\Delta t_i^n}. \tag{2.2}$$

  In all our applications to be given in this dissertation, for simplicity, we consider uniform lengths of control-time steps for injection and production periods.
- We define $t_{i,j}^n$ to denote the total time, relative to t_0, marking the end of the j^{th} injection control-time step in the n^{th} cycle, and $t_{p,j}^n$ to denote the total time, relative to t_0, marking the end of the j^{th} production control-time step in the n^{th} cycle, and, respectively, as

$$t_{i,j}^n = t_0 + \sum_{l=1}^{n-1} \Delta t^l + \sum_{k=1}^{j} \delta\Delta t_{i,k}^n, \text{ for } j = 1,2,\ldots,N_i^n, \tag{2.3}$$

and

$$t^n_{p,j} = t_0 + \sum_{l=1}^{n-1} \Delta t^l + \Delta t^n (1 - \widehat{\Delta t}^n_p) + \sum_{k=1}^{j} \delta \Delta t^n_{p,k}, \text{ for } j = 1,2,...,N^n_p. \tag{2.4}$$

To create a schedule to be used in the simulator (CMG-GEM (GEM, 2016) in this research), we need to order the control times in such a way that they follow each other in time order. It means that, first, we create time schedule set $t_{sc} = \{t^n_{i,j}, t^n_{p,j}\}$ for all i, j and n indices, and then we order the set in ascending order. In that way, the well controls are ordered corresponding to their control times.

A schematic illustration of the definitions for the CO_2 HnP process is shown in Fig. 2.1.

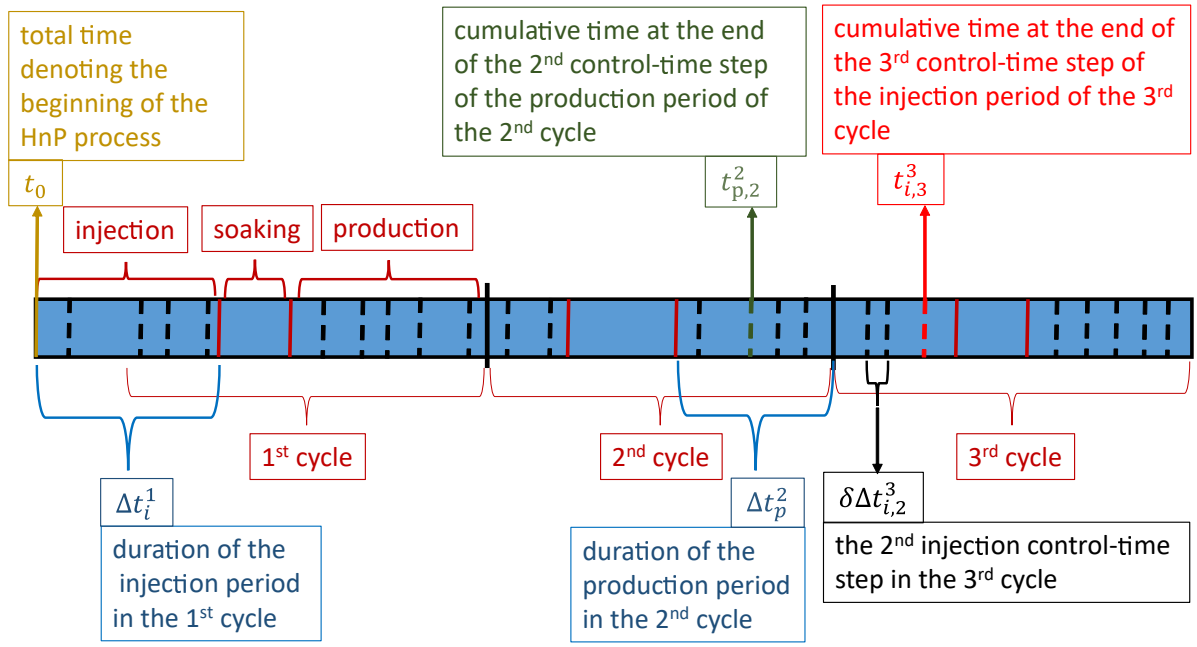


Figure 2.1: A schematic illustration for a 3-cycle HnP process.

Using the above notation and Fig. 2.1, we define the objective function for the NPV of the HnP process in an unconventional oil formation as

$$J(\mathbf{m}, \mathbf{u}) = \sum_{n=1}^{N_c} \left[\sum_{j=1}^{N^n_p} \frac{\delta \Delta t^n_{p,j}}{(1+b)^{\frac{t^n_{p,j}}{365}}} (r_o \bar{q}^n_{o,j} - c_{CO_2,p} \bar{q}^n_{CO_2,p,j}) - \sum_{j=1}^{N^n_i} \frac{\delta \Delta t^n_{i,j}}{(1+b)^{\frac{t^n_{i,j}}{365}}} (c_{CO_2,i} \bar{q}^n_{CO_2,i,j}) \right], \tag{2.5}$$

where b is the annual discount rate; r_o (\$/STB) is the oil revenue; $c_{CO_2,p}$ (\$/MSCF) is the dis-

posal cost of CO_2; $c_{CO_2,i}$ (\$/MSCF) is the cost of injection of CO_2; $\bar{q}^n_{o,j}$ (STB/Day) is the average produced oil rate at the j^{th} control-time step of the n^{th} cycle; $\bar{q}^n_{CO_2,p,j}$ (MSCF/Day) is the average produced CO_2 rate at the j^{th} control-time step of the n^{th} cycle; $\bar{q}^n_{CO_2,i,j}$ (MSCF/Day) is the average injected CO_2 rate at the j^{th} control-time step of the n^{th} cycle.

In Eq. 2.5, the time steps are general and do not have to be uniform. If we give the time steps with the following equations:

$$\delta\Delta t^n_{i,j} = \frac{\Delta t^n \widehat{\Delta t}^n_i}{N^n_i} \tag{2.6}$$

and

$$\delta\Delta t^n_{p,j} = \frac{\Delta t^n \widehat{\Delta t}^n_p}{N^n_p}, \tag{2.7}$$

then they are uniform. For instance, let us take our injection period of the first cycle as 183 days, and N_i is 19. If we want to have uniform time steps, we will use Eq. 2.6, and $\delta\Delta t^1_{p,j}$ will be 9.63 days for $j = 1, ..., 19$. If you want you can choose each of the time step values as you want. For example, $\delta\Delta t^1_{p,j} = 10$ for $j = 1, ..., 18$, and $\Delta\delta t^1_{p,19} = 3$, in which again we have 19 time steps and the summation of these time steps of injection period of the first cycle is equal to the duration of the injection period of the first cycle.

Once we control design variables in these time steps, we call them control time steps. Even if we do not control any design variable during the production period or injection period, we might as well have small time steps so that we can accurately compute the NPV since we have inflation multipliers of $1/(1+b)^{\frac{t^n_{i,j}}{365}}$ and $1/(1+b)^{\frac{t^n_{p,j}}{365}}$. But, this makes the difference in the NPV values small enough that it can be neglected. However, to be more accurate, we can divide the injection and production period of each cycle into smaller time steps.

In Eq. 2.5, $\mathbf{m}$ is an N_m-dimensional vector of reservoir-model parameters that represents one realization of the uncertain reservoir models, where N_m denotes the total number of uncertain reservoir model parameters, and $\mathbf{u}$ represents the N_u-dimensional vector of the design variables, where N_u denotes the total number of design variables. The design variable

vector includes CO_2 injection rate ($q^n_{CO_2,i,j}$) at each injection control-time step ($\delta\Delta t^n_i$), production BHP ($p^n_{bh,j}$) at each production control-time step ($\delta\Delta t^n_p$), the fraction of injection ($\widehat{\Delta t}^n_i$) and production ($\widehat{\Delta t}^n_p$) periods at each of the N_c cycles, and the length of each cycle (Δt^n's). The order of design variables in the design vector is

$$\mathbf{u} = [(\mathbf{u}^{\mathrm{IC}})^T, (\mathbf{u}^{\mathrm{PC}})^T, (\mathbf{u}^{\mathrm{PT}})^T, (\mathbf{u}^{\mathrm{IT}})^T, (\mathbf{u}^{\mathrm{C}})^T]^T, \tag{2.8}$$

where $\mathbf{u}^{\mathrm{IC}}$ contains the injection controls (gas injection rates) at all control steps; $\mathbf{u}^{\mathrm{PC}}$ contains the production controls (production BHPs) at all control steps; $\mathbf{u}^{\mathrm{PT}}$ and $\mathbf{u}^{\mathrm{IT}}$ contains production time fractions and injection time fractions at all cycles, respectively; $\mathbf{u}^{\mathrm{C}}$ contains the cycle length of all cycles. Thus, the dimension of the design variable vector,

$$N_u = \sum_{n=1}^{N_c} \left(N^n_p + N^n_i\right) + 3N_c. \tag{2.9}$$

The NPV function can also be defined in terms of simulation time steps where production and injection time steps are given in that function, implicitly

$$J(\mathbf{m}, \mathbf{u}) = \sum_{k=1}^{N_s} \frac{\Delta t^k}{(1+b)^{\frac{t^k}{365}}} (r_o \bar{q}^k_o - c_{CO_2,p}\bar{q}^k_{CO_2,p} - c_{CO_2,i}\bar{q}^k_{CO_2,i}), \tag{2.10}$$

where N_s is number of simulation time steps; b is the annual discount rate; Δt^k is time difference between each simulation time steps (Days); t^k is simulation time at the k^{th} simulation time step (Days). Then, the time difference between each simulation time step can be calculated as follows:

$$\Delta t^k = t^k - t^{k-1}. \tag{2.11}$$

To make sure that both NPV equations (Eqs. 2.5 and 2.10) yield the same result, a computational example is given in Appendix A.

Though the NPV formulation given by Eq. 2.5 is more general, here, we focus on the specific subproblem of it, where we fix the number of cycles, N_c, equal to 5, and the

injection and production periods are not divided into control-time steps, i.e., $N_p^n = N_i^n = 1$ for each cycle. Furthermore, we fix the total duration of the HnP process to 3000 days, i.e., $\sum_{n=1}^{5} \Delta t^n$=3000 days, though the duration of each cycle, Δt^n could be treated as a design variable. Thus, the maximum number of design variables at each cycle is five; namely, $q_{CO_2,i}^n$, p_{bh}^n, $\widehat{\Delta t}_i^n$, $\widehat{\Delta t}_p^n$, and Δt^n, $n = 1, 2, ..., 5$. It is worth noting that the number of cycles (N_c) may be treated as unknown in optimization, but we have not done it, and it is a subject of future study because treating N_c as a design variable poses a more difficult optimization problem in the sense that it requires one to work with the vectors of design variables that change sizes in each iteration.

2.1.1 Constraints on Design Variables

We have bound constraints, linear inequality, and linear equality constraints. In the following equation, we give the bound constraints for production BHPs and injection rates as

$$p_{bh}^{low} \leq p_{bh}^n \leq p_{bh}^{up} \quad \text{for} \quad n = 1 : N_c \tag{2.12}$$

and

$$q_{CO_2,i}^{low} \leq q_{CO_2,i}^n \leq q_{CO_2,i}^{up} \quad \text{for} \quad n = 1 : N_c. \tag{2.13}$$

In our applications for the CO_2 HnP process, we use $p_{bh}^{low} = 1500$ psi, $p_{bh}^{up} = 2400$ psi, $q_{CO_2,i}^{low} = 40$ MSCF/D, and $q_{CO_2,i}^{up} = 250$ MSCF/D unless otherwise stated.

The time fraction of the production period is constrained by the following bound constraint:

$$\widehat{\Delta t}_p^{low} \leq \widehat{\Delta t}_p^n \leq \widehat{\Delta t}_p^{up} \quad \text{for} \quad n = 1 : N_c. \tag{2.14}$$

We set $\widehat{\Delta t}_p^{up} = 1$, which means we can produce for the entire life of the cycle n, and for $\widehat{\Delta t}_p^{low} = 0.3$ in Eq. 2.14. The reason for choosing a lower bound different from zero for $\widehat{\Delta t}_p^{low}$ is based on our engineering judgment that a lower bound of 0 is unreasonably low. Based on our sensitivity analysis (Section 5.2) and applications, this lower bound seems to be reasonable for the production time fraction. However, one could set a lower limit for $\widehat{\Delta t}_p^n$ as

low as zero, but in this case, the sample space will increase to construct a proxy model. Also, having a constraint such that $\widehat{\Delta t_p}^{low} = 0$ means no production. Our results with $\widehat{\Delta t_p}^{low} = 0$ show that optimization using a proxy built with $\widehat{\Delta t_p}^{low} = 0$ takes more computational time to get the same optimum that would be obtained with a proxy built with $\widehat{\Delta t_p}^{low} = 0.3$ (see Subsection 5.3.8). So, the logical and experience-based choice of bounds saves a lot of computational time, and hence, it is fair to choose the bounds based on engineering common sense.

We use the following linear inequality constraint for the duration of the injection period for a given cycle:

$$\frac{\Delta t^n - \Delta t_p^n}{2} \leq \Delta t_i^n \leq \Delta t^n - \Delta t_p^n \quad \text{for} \quad n = 1 : N_c. \tag{2.15}$$

It is worth noting that we derived the above linear inequality constraint for the injection time based on two requirements:

1. The sum of production and injection periods must be less than or equal to the length of cycle, i.e.,

$$\Delta t_i^n + \Delta t_p^n \leq \Delta t^n \quad \text{for} \quad n = 1 : N_c \tag{2.16}$$

and

2. the minimum length of the injection period must be equal to the half-length of the remaining portion of the cycle after extracting the production period so that the duration of the soaking period is no more than that of the injection period, i.e.,

$$\Delta t_i^n \geq \frac{\Delta t^n - \Delta t_p^n}{2} \quad \text{for} \quad n = 1 : N_c. \tag{2.17}$$

Here, one may ask the question of why we require the injection time fraction to be higher than the soaking time fraction. Our examples showed that the duration of the injection period is always bigger than that of the soaking period to achieve a higher recovery factor (RF). Our sensitivity analysis conducted in Section 5.2 also justifies

our use of linear inequality constraint given by Eq. 2.17.

We can express the linear inequality constraint given by Eq. 2.15 as two linear inequality constraints in terms of injection and production time fractions as

$$\widehat{\Delta t}_i^n + \widehat{\Delta t}_p^n \leq 1 \quad \text{for} \quad n = 1 : N_c \tag{2.18}$$

and

$$-\widehat{\Delta t}_i^n - 0.5\widehat{\Delta t}_p^n \leq -0.5 \quad \text{for} \quad n = 1 : N_c. \tag{2.19}$$

Since we include each cycle length as a design variable, we constraint it as

$$\Delta t^{low} \leq \Delta t^n \leq \Delta t^{up} \quad \text{for} \quad n = 1 : N_c, \tag{2.20}$$

for $n = 1, 2, ..., 5$, and

$$\sum_{n=1}^{N_c} \Delta t^n = t_{total}, \tag{2.21}$$

where t_{total} is the total lifetime of the HnP process, and as mentioned previously, for our applications given here, we set $t_{total} = 3000$ days and $N_c = 5$.

We can write each of the bound constraints (Eqs. 2.12, 2.13, 2.14 and 2.20) as two linear inequality constraints. For example, Eq 2.12 can be written as two linear inequality constraints: $p_{bh}^n \leq p_{bh}^{up}$ and $-p_{bh}^n \leq -p_{bh}^{low}$ for $n = 1 : N_c$.

2.1.2 Normalization of Design Variables, NPV, and Constraints

The magnitude of each design variable affects the performance of optimization methods and the training process of an ML algorithm (Crone et al., 2006). Therefore, we normalize the features (features are the design variables in our problem) before training (Suykens et al., 2002). This normalization is important in the gradient-based optimization process itself (actually a training process is, eventually, an optimization process). Normalizing design variables makes the step-size selection process stable in a line-search method (Wright and Nocedal, 1999). There are different normalization techniques in the literature. In this study,

design variables are normalized using their lower and upper bounds so that they are scaled between zero and one as

$$\bar{u}_k = (u_k - u_k^{low})/(u_k^{up} - u_k^{low}), \quad \text{for} \quad k = 1 : N_u. \tag{2.22}$$

The NPV can also be normalized using min-max normalization given as

$$\bar{J} = (J - J^{min})/(J^{max} - J^{min}), \tag{2.23}$$

where J^{min} and J^{max} represent minimum and maximum values of the NPV, respectively. They are obtained from the training set. As the training set changes during the iterative-sampling-refinement optimization process, J^{min} and J^{max} may change from iteration to iteration. In this work, we do not normalize NPVs for the conventional optimization methods (simplex, StoSAG), but normalize for the ML-based optimization methods. We normalize the design variables for both conventional and ML-based optimization methods since it is important for better performance of a gradient-based optimization method. After normalization, all bound constraints become between zero and one:

$$0 \leq \bar{u}_k \leq 1 \tag{2.24}$$

Similarly, we write our constraints given by Eqs. 2.18, 2.19, and 2.21 in terms of the normalized variables, respectively, as

$$\overline{\widehat{\Delta t}}_i^n (\widehat{\Delta t}_i^{up} - \widehat{\Delta t}_i^{low}) + \overline{\widehat{\Delta t}}_p^n (\widehat{\Delta t}_p^{up} - \widehat{\Delta t}_p^{low}) \leq -\widehat{\Delta t}_i^{low} - \widehat{\Delta t}_p^{low} + 1 \quad \text{for} \quad n = 1 : N_c, \tag{2.25}$$

$$-\overline{\widehat{\Delta t}}_i^n (\widehat{\Delta t}_i^{up} - \widehat{\Delta t}_i^{low}) - 0.5\overline{\widehat{\Delta t}}_p^n (\widehat{\Delta t}_p^{up} - \widehat{\Delta t}_p^{low}) \leq \widehat{\Delta t}_i^{low} + 0.5\widehat{\Delta t}_p^{low} - 0.5 \quad \text{for} \quad n = 1 : N_c, \tag{2.26}$$

and

$$\sum_{n=1}^{N_c} \overline{\Delta t}^n (\Delta t^{up} - \Delta t^{low}) = t_{total} - N_c \Delta t^{low}. \tag{2.27}$$

We can write all normalized linear inequality constraints (Eqs. 2.25 and 2.26) in a matrix-vector multiplication form

$$\mathbf{A}\bar{\mathbf{u}} \leq \mathbf{b}, \tag{2.28}$$

where $\mathbf{A}$ contains coefficients of all the linear inequality constraints and is an $M \times N_u$ matrix, $\mathbf{b}$ is an M-dimensional column vector, where M is the number of linear inequality constraints. To better understand we also visualized the Eq. 2.28 as

$$\mathbf{A} = \begin{bmatrix} a_{11} & a_{12} & \dots & a_{1N_u} \\ a_{21} & a_{22} & \dots & a_{2N_u} \\ \vdots & \vdots & \ddots & \vdots \\ a_{M1} & a_{M2} & \dots & a_{MN_u} \end{bmatrix} \begin{bmatrix} \bar{u}_1 \\ \bar{u}_2 \\ \vdots \\ \bar{u}_{N_u} \end{bmatrix} \leq \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_M \end{bmatrix} \tag{2.29}$$

where each normalized linear inequality equation can be written with those coefficients as follows:

$$a_{i1}\bar{u}_1 + a_{i2}\bar{u}_2 + \cdots + a_{iN_u}\bar{u}_{N_u} \leq b_i \quad \text{for} \quad i = 1, \dots, M. \tag{2.30}$$

The matrix-vector multiplication formula for the normalized linear equality constraint given by Eq. 2.27 must be written separately as

$$\hat{\mathbf{A}}\bar{\mathbf{u}} = \hat{\mathbf{b}}, \tag{2.31}$$

where $\hat{\mathbf{A}}$ is a $M_{le} \times N_u$ matrix containing the coefficients of the normalized linear equality constraints, and $\hat{\mathbf{b}}$ is the M_{le}-dimensional vector containing the values of the normalized linear equality constraints. One can intuitively visualize Eq. 2.31 similarly to the Eq. 2.29. For our applications here, since we have single well HnP process, we have only one linear equality constraint (Eq. 2.21) and hence $M_{le} = 1$ and $\hat{b}_1$ is equal to t_{total}. More precisely, we create two separate functions for constraints: one for normalized equality and one for

normalized inequality constraints. When we create a sample set for training as well as a perturbation for Simplex and StoSAG, we truncate our design variables for each sample to satisfy all these constraints. We deal with constraints via the gradient projection method where we project our gradient on the hyperplane of the intersection of active constraints. The gradient projection method will be briefly discussed in Section 3.3.

2.2 NPV Formulation for the CO_2 WAG Problem

To formulate a general NPV objective function for the WAG process, the following definitions and assumptions are used:

- N_{ic}, the total number of cycles for each injection well, is the same for each injection well and fixed during the optimization process. It corresponds to the number of injection control time steps. Remember that at each cycle we control gas injection and water injection. Therefore, we call each injection time period a half-cycle of the injection well. At the gas injection half-cycle, we control gas injection rate, and at the water injection half-cycle we control water injection rate. Thus, we will have N_{ic} number of gas injection time steps and N_{ic} number of water injection time steps.
- N_c, the total number of cycles for each production well, is the same for each production well and fixed during the optimization process. It corresponds to the number of production control time steps.
- $\Delta t^{n,m}$ is the duration of the n^{th} cycle of m^{th} injection well for $n = 1, 2, ..., N_{ic}$ and $m = 1, 2, ..., N_I$, where N_I is the number of the injection wells.
- $\Delta t^{n,k}$ is the duration of the n^{th} cycle of k^{th} production well for $n = 1, 2, ..., N_c$ and $k = 1, 2, ..., N_P$, where N_P is the number of the production wells.
- The total duration of the WAG process ($t_{total} = \sum_{n=1}^{N_{ic}} \Delta t^{n,m} = \sum_{n=1}^{N_c} \Delta t^{n,k}$ for each injection well m and production well k) is known and fixed during the optimization process.

- $\widehat{\Delta t}_g^{n,m}$ and $\widehat{\Delta t}_w^{n,m}$ represent time fractions of gas injection and water injection periods, respectively, for the n^{th} cycle of m^{th} injection well. They can be treated as unknowns in optimization and defined by

$$\widehat{\Delta t}_g^{n,m} = \frac{\Delta t_g^{n,m}}{\Delta t^{n,m}}, \qquad \widehat{\Delta t}_w^{n,m} = \frac{\Delta t_w^{n,m}}{\Delta t^{n,m}} = 1 - \widehat{\Delta t}_g^{n,m}, \tag{2.32}$$

  where $\Delta t_g^{n,m}$ and $\Delta t_w^{n,m}$ (days) are the duration of gas injection period and water injection period at the n^{th} cycle of m^{th} injection well, respectively. Note that specifying the duration of the gas injection period determines the duration of the water injection period.

- $t^{n,m}$ (days) is the cumulative time at the end of the n^{th} cycle of the injection well m, and $t^{n,k}$ (days) is the cumulative time at the end of the n^{th} cycle of the production well k:

  - For gas injection time period of cycle n of injection well m,

$$t^{n,m} = \sum_{l=1}^{n} (\Delta t^{l-1,m}) + \Delta t^{n,m} \cdot (1 - \widehat{\Delta t}_g^{l,m}), \tag{2.33}$$

  - for water injection time period of cycle n of injection well m,

$$t^{n,m} = \sum_{l=1}^{n} \Delta t^{l,m}, \tag{2.34}$$

  - for production time of cycle n of production well k,

$$t^{n,k} = \sum_{l=1}^{n} \Delta t^{l,k}. \tag{2.35}$$

A schematic illustration of the definitions for the CO_2 WAG process is shown in Fig. 2.2.

Using the above notation and Fig. 2.2, we define the objective function for the general NPV of the WAG process in the Eq. 2.36. In Eq. 2.36, we ignored gas production revenue

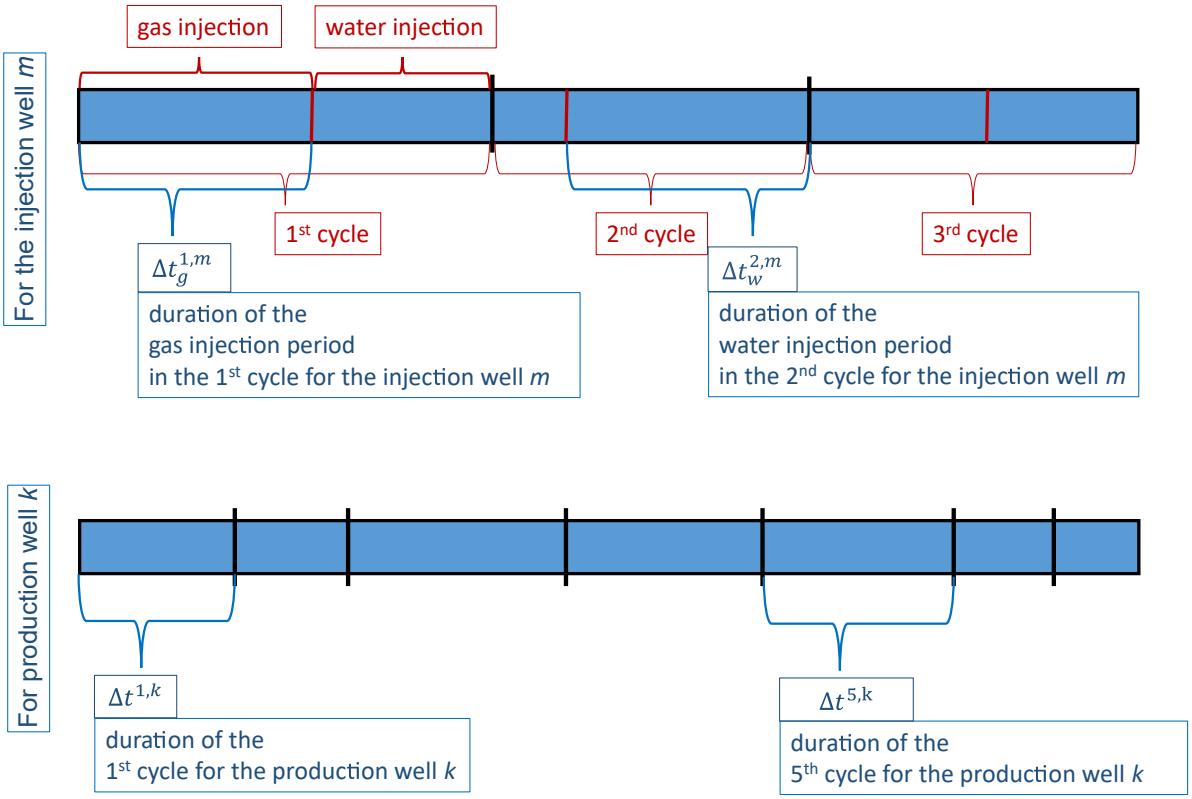


Figure 2.2: A schematic illustration for a 3-cycle WAG process.

and gas disposal cost, since they both almost compensate each other, and therefore their difference is negligible (is almost zero).

$$J(\mathbf{m},\mathbf{u}) = \sum_{n=1}^{N_c}\sum_{k=1}^{N_P} \frac{\Delta t^{n,k}}{(1+b)^{\frac{t^{n,k}}{365}}}(r_o \bar{q}_o^{n,k} - c_{wp}\bar{q}_{wp}^{n,k}) - \sum_{n=1}^{N_{ic}}\sum_{m=1}^{N_I}\left(\frac{\Delta t^{n,m}\widehat{\Delta t}_g^{n,m}}{(1+b)^{\frac{t^{n,m}}{365}}}(c_{gi}\bar{q}_{gi}^{n,m}) + \frac{\Delta t^{n,m}\widehat{\Delta t}_w^{n,m}}{(1+b)^{\frac{t^{n,m}}{365}}}(c_{wi}\bar{q}_{wi}^{n,m})\right), \tag{2.36}$$

where c_{gi} (\$/MSCF) is the gas injection cost; c_{wi} (\$/STB) is the water injection cost; $\bar{q}_o^{n,k}$ (STB/Day) is the average produced oil rate at the n^{th} cycle of the production well k; $\bar{q}_{gi}^{n,m}$ (MSCF/Day) is the average gas injection rate at the n^{th} cycle of the injection well m; $\bar{q}_{wi}^{n,m}$ (STB/Day) is the average water injection rate at the n^{th} cycle of the injection well m.

In Eq. 2.36, in contrast to the Eq. 2.5, we do not further divide each cycle into control time step, we assume each half-cycle is control-time step. In Eq. 2.36, $\mathbf{m}$ is an N_m-dimensional vector of reservoir-model parameters that represents one realization of the

uncertain reservoir models, where N_m denotes the total number of uncertain reservoir model parameters, and $\mathbf{u}$ represents the N_u-dimensional vector of the design variables, where N_u denotes the total number of design variables.

We have to mention that for production optimization of the WAG problem, we also considered an optimization case where we use production oil rate (STB/D) at each cycle of each production well ($q_o^{n,k}$) as design variable instead of production BHPs. The WAG problem is the only problem that we considered the optimization case where production oil rate is in the set of design variables, and the reason will be explained in Section 6.

In Eq. 2.5, for the case where we do not optimize inflow control valves (ICVs) the design variable vector, $\mathbf{u}$ includes gas injection rate at each cycle of each injection well ($q_{gi}^{n,m}$), water injection rate at each cycle of each injection well ($q_{wi}^{n,m}$), production BHP (or production oil rate, $q_o^{n,k}$) at each cycle of each production well ($p_{bh}^{n,k}$), the fraction of gas injection period at each cycle of each injection well ($\widehat{\Delta t}_g^{n,m}$), the length of each cycle of each injection well ($\Delta t^{n,m}$), and the length of each cycle of each production well ($\Delta t^{n,k}$). The dimension of the design variable vector, N_u,

$$N_u = 4 \cdot N_{ic} \cdot N_I + 2 \cdot N_c \cdot N_P. \tag{2.37}$$

However, as mentioned in Section 1.2.2, ICVs are important design variables, especially when petro-physical properties of reservoir change significantly. At every perforation, meaning that at each layer we will have ICV. There are v^m valves for injection well m, and v^k for production well k. For injection wells we call them injection ICVs (IICVs), and for production wells it is production ICVs (PICVs). At each cycle of production and injection wells and at each valve ICV becomes design variable. Thus the design vector of $\mathbf{u}^{\text{ICV}} = [((\mathbf{u}^{\text{IICV}})^m)^T, ((\mathbf{u}^{\text{PICV}})^k)^T]^T$ for $m = 1 : N_I$ and $k = 1 : N_P$, where $(\mathbf{u}^{\text{IICV}})^m = [\text{IICV}_n^l]$ for $n = 1 : N_{ic}$, $l = 1 : v^m$, and $(\mathbf{u}^{\text{PICV}})^k = [\text{PICV}_n^l]$, for $n = 1 : N_c$, $l = 1 : v^k$. This is the most general case where all possible design variables are considered. The design variables

are ordered in the design vector as follows:

$$\begin{aligned}\mathbf{u} = [&((\mathbf{u}^{\mathrm{GIC}})^1)^T, \ldots, ((\mathbf{u}^{\mathrm{GIC}})^m)^T, \ldots, ((\mathbf{u}^{\mathrm{GIC}})^{N_I})^T, ((\mathbf{u}^{\mathrm{WIC}})^1)^T, \ldots, ((\mathbf{u}^{\mathrm{WIC}})^m)^T,\\ &\ldots, ((\mathbf{u}^{\mathrm{WIC}})^{N_I})^T, ((\mathbf{u}^{\mathrm{PC}})^1)^T, \ldots, ((\mathbf{u}^{\mathrm{PC}})^k)^T, \ldots, ((\mathbf{u}^{\mathrm{PC}})^{N_P})^T, ((\mathbf{u}^{\mathrm{GIT}})^1)^T,\\ &\ldots, ((\mathbf{u}^{\mathrm{GIT}})^m)^T, \ldots, ((\mathbf{u}^{\mathrm{GIT}})^{N_I})^T, ((\mathbf{u}^{\mathrm{CI}})^1)^T, \ldots, ((\mathbf{u}^{\mathrm{CI}})^m)^T, \ldots, ((\mathbf{u}^{\mathrm{CI}})^{N_I})^T,\\ &((\mathbf{u}^{\mathrm{CP}})^1)^T, \ldots, ((\mathbf{u}^{\mathrm{CP}})^k)^T, \ldots, ((\mathbf{u}^{\mathrm{CP}})^{N_P})^T, ((\mathbf{u}^{\mathrm{IICV}})^1)^T, \ldots, ((\mathbf{u}^{\mathrm{IICV}})^m)^T,\\ &\ldots, ((\mathbf{u}^{\mathrm{IICV}})^{N_I})^T, ((\mathbf{u}^{\mathrm{PICV}})^1)^T, \ldots, ((\mathbf{u}^{\mathrm{PICV}})^k)^T, \ldots, ((\mathbf{u}^{\mathrm{PICV}})^{N_P})^T],\end{aligned} \tag{2.38}$$

where $(\mathbf{u}^{\mathrm{GIC}})^m$ is the vector containing the gas injection controls (rates) for injection well m at all cycles; $(\mathbf{u}^{\mathrm{WIC}})^m$ is the vector containing the water injection controls (rates) for injection well m at all cycles; $(\mathbf{u}^{\mathrm{PC}})^k$ is the vector containing the production controls (production BHPs or production oil rates) for production well k at all cycles; $(\mathbf{u}^{\mathrm{GIT}})^m$ is the vector containing gas injection time fractions of injection well m at all cycles; $(\mathbf{u}^{\mathrm{CI}})^m$ is the vector containing cycle lengths of the injection well m; $(\mathbf{u}^{\mathrm{IICV}})^m$ is the vector containing IICVs of the injection well m at all cycles; $(\mathbf{u}^{\mathrm{PICV}})^k$ is the vector containing PICVs of the production well k at all cycles. The elements of $(\mathbf{u}^{\mathrm{IICV}})^m$ and $(\mathbf{u}^{\mathrm{PICV}})^k$ are

$$(\mathbf{u}^{\mathrm{IICV}})^m = [\mathrm{IICV}_1^1, \mathrm{IICV}_2^1, \ldots, \mathrm{IICV}_{N_{ic}}^1, \ldots \mathrm{IICV}_n^l, \ldots, \mathrm{IICV}_{N_{ic}}^{v^m}] \tag{2.39}$$

and

$$(\mathbf{u}^{\mathrm{PICV}})^k = [\mathrm{PICV}_1^1, \mathrm{PICV}_2^1, \ldots, \mathrm{PICV}_{N_c}^1, \ldots \mathrm{PICV}_n^l, \ldots \mathrm{PICV}_{N_c}^{v^k}], \tag{2.40}$$

where IICV_n^l is IICV at the valve l of the injection well m at the cycle n, where $l = 1 : v^m$; and PICV_n^l is PICV at the valve l of the production well k at the cycle n, where $l = 1 : v^k$. Thus, the total number of design variables is

$$N_u = 4 \cdot N_{ic} \cdot N_I + 2 \cdot N_c \cdot N_P + \sum_{m=1}^{N_I} 2 \cdot N_{ic} \cdot v^m + \sum_{k=1}^{N_P} N_c \cdot v^k. \tag{2.41}$$

The NPV function defined with simulation time steps, where all the design variables

are given inside the NPV implicitly, is the same as the Eq. 2.10.

For the case where cycle lengths and gas injection time fractions are no longer in the set of design variables, we divide the total life-time of the WAG process into N_c cycles, uniformly for each injection well. We assume that gas injection and water injection half-cycles at each cycle of each injection well are also uniform; that is $\widehat{\Delta t}_g^{n,m} = 0.5$ for all n and m. Thus, we define total number of half-cycle, $N_t = 2 \cdot N_{ic}$. We also make another assumption that $N_c = N_t$. So, the control time step is the same at each half-cycle of each injection well and production well and equal to

$$\Delta t = \frac{t_{total}}{N_t}. \tag{2.42}$$

Using these assumptions, we can formulate a simpler version of NPV, which is mostly used in the literature:

$$J(\mathbf{u}) = \sum_{n=1}^{N_t} \left\{ \frac{\Delta t}{(1+b)^{\frac{t^n}{365}}} \left[\sum_{k=1}^{N_P} (r_o \bar{q}_o^{n,k} - c_{wp} \bar{q}_{wp}^{n,k}) - \sum_{m=1}^{N_I} (c_{gi} \bar{q}_{gi}^{n,m} + c_{wi} \bar{q}_{wi}^{n,m}) \right] \right\}, \tag{2.43}$$

where t^n is the cumulative time at n^{th} half-cycle and is the same for all wells:

$$t^n = n \cdot \Delta t. \tag{2.44}$$

We call this formulation of NPV a simple NPV. For the simple NPV, the number of design variables, considering $N_t = 2 \cdot N_{ic} = N_c$ in Eq. 2.37 for the case without ICVs is

$$N_u = N_t \cdot N_I + N_t \cdot N_P = N_t \cdot (N_I + N_P), \tag{2.45}$$

and in Eq. 2.41 for the case with ICVs it is:

$$N_u = N_t \cdot (N_I + N_P) + \sum_{m=1}^{N_I} N_t \cdot v^m + \sum_{k=1}^{N_P} N_t \cdot v^k = N_t \cdot (N_I + N_P + \sum_{m=1}^{N_I} v^m + \sum_{k=1}^{N_P} v^k). \tag{2.46}$$

In all production optimization cases of the WAG problem considered in this study, we consider only vertical wells, all of which are perforated at each layer. Therefore, the number of valves are the same for each well, both injection and production wells, and equal to the number of layers; that is, $v^m = v^k = N_z$ for $m = 1 : N_I$ and $k = 1 : N_P$, where N_z is the number of layers of the reservoir. However, since we formulated a general NPV objective function, in the following section we discuss constraints for all of the design variables of the most general case, where the well controls, time fractions, cycle lengths, and ICVs are in the set of the design variables.

2.2.1 Constraints on Design Variables

For all design variables we have bound constraints as follows:

$$\begin{aligned}
q_{gi}^{low} \le q_{gi}^{n,m} \le q_{gi}^{up} \quad &\text{for} \quad n = 1 : N_{ic} \quad \text{and} \quad m = 1 : N_I, \\
q_{wi}^{low} \le q_{wi}^{n,m} \le q_{wi}^{up} \quad &\text{for} \quad n = 1 : N_{ic} \quad \text{and} \quad m = 1 : N_I, \\
p_{bh}^{low} \le p_{bh}^{n,k} \le p_{bh}^{up} \quad &\text{for} \quad n = 1 : N_c \quad \text{and} \quad k = 1 : N_P, \\
\widehat{\Delta t}_g^{low} \le \widehat{\Delta t}_g^{n,m} \le \widehat{\Delta t}_g^{up} \quad &\text{for} \quad n = 1 : N_{ic} \quad \text{and} \quad m = 1 : N_I, \\
\Delta t^{low} \le \Delta t^{n,m} \le \Delta t^{up} \quad &\text{for} \quad n = 1 : N_{ic} \quad \text{and} \quad m = 1 : N_I, \\
\Delta t^{low} \le \Delta t^{n,k} \le \Delta t^{up} \quad &\text{for} \quad n = 1 : N_c \quad \text{and} \quad k = 1 : N_P, \\
\mathrm{IICV}^{low} \le \mathrm{IICV}_n^{l,m} \le \mathrm{IICV}^{up} \quad &\text{for} \quad n = 1 : N_{ic}, \ m = 1 : N_I \quad \text{and} \quad l = 1 : v^m, \\
\mathrm{PICV}^{low} \le \mathrm{PICV}_n^{l,k} \le \mathrm{PICV}^{up} \quad &\text{for} \quad n = 1 : N_c, \ k = 1 : N_P \quad \text{and} \quad l = 1 : v^k,
\end{aligned} \tag{2.47}$$

and for the case where the production oil rate is a design variable instead of the production BHP at each cycle of each production well we have

$$q_o^{low} \le q_o^{n,k} \le q_o^{up} \quad \text{for} \quad n = 1 : N_c \quad \text{and} \quad k = 1 : N_P. \tag{2.48}$$

In our all production optimization cases of the WAG problem, $\widehat{\Delta t}_g^{low} = 0.2$, $\widehat{\Delta t}_g^{up} = 1$, $\Delta t^{low} = t_{total}/(5N_c)$, $\Delta t^{up} = t_{total}/(0.5N_c)$, $\mathrm{IICV}^{low} = \mathrm{PICV}^{low} = 0$, $\mathrm{IICV}^{up} = \mathrm{PICV}^{up} = 1$,

$q_{gi}^{low} = 0$ (SCF/D), $q_{gi}^{up} = 2 \times 10^7$(SCF/D), $q_{wi}^{low} = 0$ (STB/D), $q_{wi}^{up} = 4000$ (STB/D). Bound constraints of all other design variables change depending on the production optimization cases. The reason for choosing these upper and lower bound values for injection well controls is the capacity of the reservoir we are using for our all production optimization cases as well as the fluid properties since we also want to achieve multi-contact miscibility of injected gas, which is CO_2 in our case, and reservoir fluid. For the cycle lengths of both injection wells and production wells, we choose lower bound and upper bound heuristically. We do not want the lower bound to be zero since, in that case, well controls at that cycle as well as the gas injection time fraction in the case of injection wells will vanish and will no longer be important whatever value they choose. However, the upper bound for the cycle lengths can be chosen larger than this value. When we discuss the training procedure of the machine learning models (GPR and LS-SVR, in our case, but that discussion is general), we will see that to sample design variables from the smaller region, where we believe that NPV is going to be higher than other regions, helps us to get accurate model around optimum with less amount of data. Furthermore, in the optimization procedure, if we narrow our area to perform optimization; that is, choosing reasonable bounds, instead of performing optimization on all regions of the NPV surface, helps us to converge to the higher local optimum. However, as mentioned, these constraints should be chosen based on knowledge of the process knowing the physics of the process either through sensitivity analysis or through lab experiments. In our cases, the bounds for cycle lengths are chosen based on the optimization we performed with and without those bounds to see the importance of choosing bounds for cycle lengths.

Since we include each cycle length as a design variable, we have to consider linear equality constraints so that the summation of the cycle lengths of each production and injection wells is equal to the total life of the WAG process:

- for injection wells

$$\sum_{n=1}^{N_{ic}} \Delta t^{n,m} = t_{total} \quad \text{for} \quad m = 1 : N_I, \tag{2.49}$$

- for production wells

$$\sum_{n=1}^{N_c} \Delta t^{n,k} = t_{total} \quad \text{for} \quad k = 1 : N_P, \tag{2.50}$$

where t_{total} is the total lifetime of the WAG process, and for our applications given here, we set $t_{total} = 2880$ days, $N_{ic} = 8$ and $N_c = 16$.

As we did in the HnP problem, we can write each of the bound constraints (Eq. 2.47) as two linear inequality constraints.

2.2.2 Normalization of Design Variables, NPV, and Constraints

The importance of normalization in the gradient-based optimization and in the training process of the ML model is discussed in Section 2.1.2 as well as the formulation of the min-max normalization of design variables and NPV (Eqs. 2.22 and 2.23). As the linear inequality constraints, we have only linear bound constraints. The normalized version of bound constraints is given in Eq. 2.24.

Like Eq. 2.27, the normalized linear equality constraints become as follows:

- for injection wells

$$\sum_{n=1}^{N_{ic}} \overline{\Delta t}^{n,m} (\Delta t^{up} - \Delta t^{low}) = t_{total} - N_{ic} \Delta t^{low} \quad \text{for} \quad m = 1 : N_I, \tag{2.51}$$

- for production wells

$$\sum_{n=1}^{N_c} \overline{\Delta t}^{n,k} (\Delta t^{up} - \Delta t^{low}) = t_{total} - N_c \Delta t^{low} \quad \text{for} \quad k = 1 : N_P. \tag{2.52}$$

All normalized linear inequality and equality constraints can be formulated as the matrix-vector products as given in Eq. 2.28 and Eq. 2.31, respectively. We have to mention that, unlike in the HnP problem, in the WAG problem, we have more than one linear equality constraint, since it is a multi-well problem. Therefore, in Eq. 2.31, the number of rows of the matrix $\hat{\mathbf{A}}$ is a $M_{le} = N_I + N_P$. When we create a sample set for training as well as a perturbation for Simplex and StoSAG, we truncate our design variables for each sample to

satisfy all these constraints. We deal with constraints via the gradient projection method where we project our gradient on the hyperplane of the intersection of active constraints. However, for the production optimization cases where the simple NPV is considered which means cycle lengths are no longer design variables (see Eq. 2.43), we deal with the constraints with truncation only since we will have only bound constraints. The truncation method will also be discussed in Section 3.3.

2.3 NPV Formulation for the Well Shutoff Problem

As discussed in Section 1.2.3, different methods can be considered for treating the shutoff option as an optimization variable in life-cycle optimization. The methodology here is very similar to how we treated soaking time in the HnP problem (see Section 2.1); that is, we use cycle lengths and time fraction design variables to define switching time for production wells to shut off. Also as noted in Section 1.2.3, we include OPEX for each production well. Therefore, in our production optimization problems including well shutoff, we only shut off production wells.

Well shutoff is an option that can be used for any production optimization problem. In our application, we consider production optimization of the waterflooding process. For waterflooding process including the well shutoff option, we define the following design variables and assumptions:

- N_c, the total number of cycles of production wells, is fixed for each production well during the optimization process.
- N_{ic}, the total number of cycles of injection wells, is fixed for each injection well during the optimization process and can be different from N_c.
- $\Delta t^{n,k}$ is the duration of the n^{th} cycle of production well k, for $n = 1, 2, ..., N_c$ and $k = 1, 2, ..., N_P$.
- $\Delta t^{n,m}$ is the duration of the n^{th} cycle of injection well m, for $n = 1, 2, ..., N_c$ and $m = 1, 2, ..., N_I$.

- The total duration of the waterflooding process ($t_{total} = \sum_{n=1}^{N_{ic}} \Delta t^{n,m} = \sum_{n=1}^{N_c} \Delta t^{n,k}$ for each injection well m and production well k) is known and fixed during the optimization process.
- $\widehat{\Delta t}_p^{n,k}$ and $\widehat{\Delta t}_s^{n,k}$ represent time fractions of production and shutoff periods, respectively, for the n^{th} cycle of k^{th} production well. They can be treated as unknowns in optimization, and defined by

$$\widehat{\Delta t}_p^{n,k} = \frac{\Delta t_p^{n,k}}{\Delta t^{n,k}}, \qquad \widehat{\Delta t}_s^{n,k} = \frac{\Delta t_s^{n,k}}{\Delta t^{n,k}} = 1 - \widehat{\Delta t}_g^{n,k}, \tag{2.53}$$

  where $\Delta t_p^{n,m}$ and $\Delta t_s^{n,m}$ (days) are the duration of production period and shutoff period at the n^{th} cycle of k^{th} production well, respectively. Note that specifying the duration of the production period determines the duration of the shutoff period.
- $t^{n,k}$ (days) is the cumulative time at the end of the n^{th} cycle of the production well k, and $t^{n,m}$ (days) is the cumulative time at the end of the n^{th} cycle of the injection well m:
  - for the production time period of cycle n of production well k:

$$t^{n,k} = \sum_{l=1}^{n} (\Delta t^{l-1,k}) + \Delta t^{n,k} \cdot (1 - \widehat{\Delta t}_p^{l,k}), \tag{2.54}$$

  - for the shutoff time period of cycle n of production well k:

$$t^{n,k} = \sum_{l=1}^{n} \Delta t^{l,k}, \tag{2.55}$$

  - for the cumulative time of cycle n of injection well k:

$$t^{n,m} = \sum_{l=1}^{n} \Delta t^{l,m}. \tag{2.56}$$

In our optimization problems, we do not consider the cycle length of injection wells

($\Delta t^{n,m}$) as a design variable. The total life of each injection wells is divided into cycles, uniformly: $\Delta t^{n,m} = t_{total}/N_{ic}$. A schematic illustration of the definitions for the waterflooding production optimization including well shutoff is shown in Fig. 2.3 where only one of the production well and 3 cycles are considered.

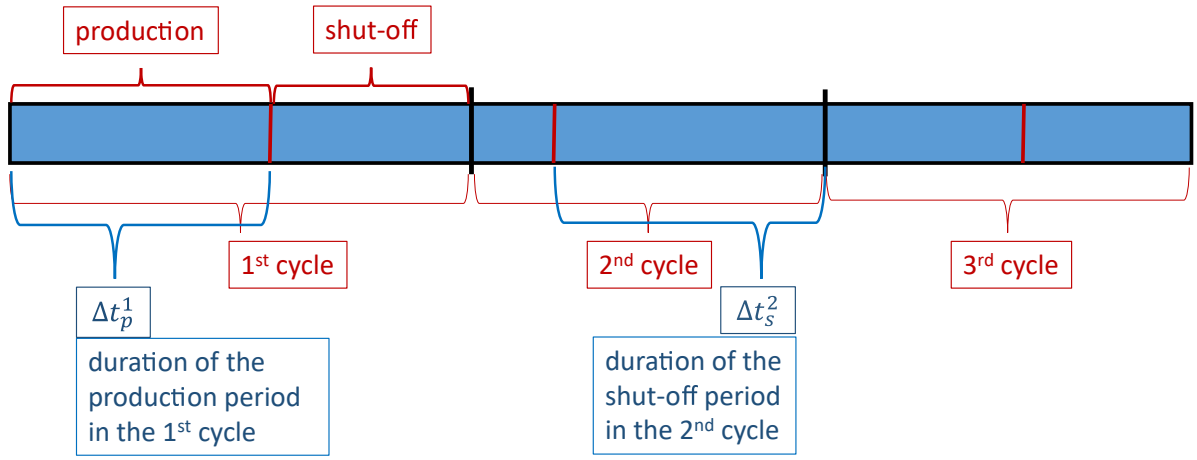


Figure 2.3: A schematic illustration for a 3-cycle well shutoff problem.

Using the above notation and Fig. 2.3, we define the objective function for the NPV of the waterflooding process including well shutoff as

$$\begin{aligned} J(\mathbf{m},\mathbf{u}) = &\sum_{n=1}^{N_c}\sum_{k=1}^{N_P} \frac{\Delta t^{n,k}\widehat{\Delta t}_p^{n,k}}{(1+b)^{\frac{t^{n,k}}{365}}}(r_o\bar{q}_o^{n,k} - c_{wp}\bar{q}_{wp}^{n,k} - cp_j \cdot l(\bar{q}_o^{n,k})) \\ &- \frac{\Delta t^{n,m}}{(1+b)^{\frac{t^{n,m}}{365}}}\sum_{n=1}^{N_{ic}}\sum_{k=1}^{N_I}(c_{wi}\cdot\bar{q}_{wi}^{n,m}), \end{aligned} \tag{2.57}$$

where c_{wi} (\$/STB) is the water injection cost; $\bar{q}_o^{n,k}$ (STB/Day) is the average produced oil rate at the n^{th} cycle of the production well k; $\bar{q}_{wi}^{n,m}$ (STB/Day) is the average water injection rate at the n^{th} cycle of the injection well m. In Eq. 2.57, in contrast to the Eq. 2.5, we do not further divide each cycle into control time step, we assume each half-cycle is control-time step. Note that each cycle of production well becomes a control time step for that production well. Similarly, each cycle of injection well becomes the control time step for that injection well.

In Eq. 2.57 the function $l(x)$ is the Heaviside function taking the value 1 if a particular

production well is open, meaning it is producing and it is 0 when they are shut off

$$l(x) = \begin{cases} 1 \text{ if } x > 0 \\ \\ 0 \text{ if } x \leq 0 \end{cases}, \tag{2.58}$$

where x is oil production rate of a given well at a given time step.

If we check the objective function with respect to the BHP of a particular well for a given control time step, it can be seen that Eq. 2.57 is discontinuous at the point where the BHP is very close to the well-block pressure of that particular well. The reason for this is because at that pressure, the well will stop producing and the term $l(\bar{q}_0^{n,k})$ will disappear from the Eq. 2.57 (since its value is going to be zero); and thus, Eq. 2.57 will change with the BHP before well-block pressure and after well-block pressure, differently. As the BHP approaches the well-block pressure, NPV decreases, and when the BHP becomes equal to the well-block pressure, NPV suddenly jumps because the production well cost term disappears for that particular control time step of a particular production well since we stop producing. For the values of the BHP more than well-block pressure, NPV remains constant since the further increase of the BHP will not make any difference to NPV. We illustrated this discontinuity in Fig. 2.4, where the total life of waterflooding is 7000 days and we have 4 production wells and 2 water injection wells. To plot Fig. 2.4, we considered 10 cycles (control time steps) for production wells and 20 cycles (control time steps) for injection wells, and fixed all design variables except BHP of production well PW4 at 5^{th} control step. We see discontinuity at the point where BHP is equal to 1800 psi. Discontinuity of NPV with respect to BHP at a given cycle for a given well is induced by the Heaviside function in NPV, $l(\bar{q}_o^{n,k})$. Since the function $l(x)$ is not a continuous function of BHPs, Eq. 2.57 is not continuous function of the BHPs. Therefore, working with BHP to shut off the well induces discontinuity. As we know, the derivative at the point where the function is discontinuous is undefined. Even though the calculation of the numerical gradient will give some value at that point, this value will be highly inaccurate for the gradient direction. For the case of having multiple production wells and more than one control time step, we will have a discontinuity at more than one

point; that is, those points will create a hyperplane. Therefore, we used the approach of using production and shutoff time fractions with cycle lengths to mitigate the discontinuity of NPV with respect to any design variable so that we can apply gradient-based optimization.

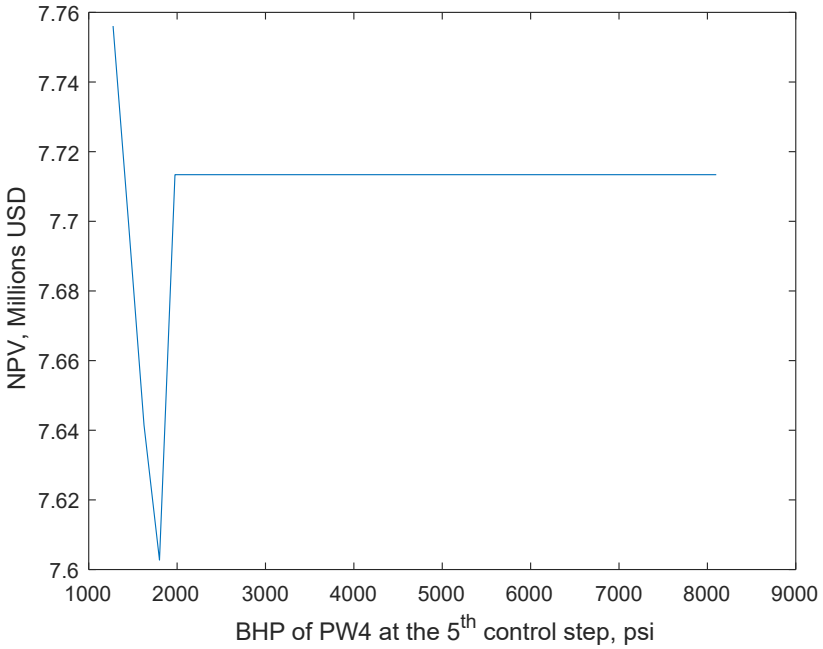


Figure 2.4: NPV vs BHP of production well PW4 at 5^{th} control step; total life of production is 7000 days; we have 4 production and 2 water injection wells. Figure shows the discontinuity of NPV with respect to BHP.

In Eq. 2.57, $\mathbf{m}$ is an N_m-dimensional vector of reservoir-model parameters, which represents one realization of the uncertain reservoir models, where N_m denotes the total number of uncertain reservoir model parameters, and $\mathbf{u}$ represents the N_u-dimensional vector of the design variables, where N_u denotes the total number of design variables. The design variable vector, $\mathbf{u}$, includes water injection rate at each cycle of each injection well ($q_{wi}^{n,m}$), production BHP at each cycle of each production well ($p_{bh}^{n,k}$), the fraction of production period at each cycle of each production well ($\widehat{\Delta t}_p^{n,k}$), and the length of each cycle of each production well ($\Delta t^{n,k}$). The dimension of the design variable vector, N_u,

$$N_u = N_I N_{i,c} + 3 N_P N_c, \tag{2.59}$$

And the design variables are ordered in the design vector as follows:

$$\begin{aligned}\mathbf{u} = [&((\mathbf{u}^{\mathrm{IC}})^1)^T, \ldots, ((\mathbf{u}^{\mathrm{IC}})^m)^T, \ldots, ((\mathbf{u}^{\mathrm{IC}})^{N_I})^T, ((\mathbf{u}^{\mathrm{PC}})^1)^T, \ldots, \\ &((\mathbf{u}^{\mathrm{PC}})^k)^T, \ldots, ((\mathbf{u}^{\mathrm{PC}})^{N_P})^T, ((\mathbf{u}^{\mathrm{CPT}})^1)^T, \ldots, ((\mathbf{u}^{\mathrm{CPT}})^k)^T, \ldots, \\ &((\mathbf{u}^{\mathrm{CPT}})^{N_P})^T, ((\mathbf{u}^{\mathrm{CP}})^1)^T, \ldots, ((\mathbf{u}^{\mathrm{CP}})^k)^T, \ldots, ((\mathbf{u}^{\mathrm{CP}})^{N_P})^T],\end{aligned} \tag{2.60}$$

where $(\mathbf{u}^{\mathrm{IC}})^m$ contains the injection controls (rates) for injection well m at all cycles; $(\mathbf{u}^{\mathrm{PC}})^k$ contains the production controls (production BHPs or production oil rates) for production well k at all cycles; $(\mathbf{u}^{\mathrm{CPT}})^m$ is production time fractions of production well m at all cycles; and $(\mathbf{u}^{\mathrm{CP}})^m$ is the cycle lengths of the production well k.

The NPV function defined with simulation time steps where all the design variables are given inside the NPV, implicitly, is the same as Eq. 2.10. In this production optimization problem, we have the production optimization case where we fix the BHP of the production wells at each cycle. We also have the production optimization case where we fix production time fractions to 1 at each cycle of each production well, which means that none of the production wells shut off. The reasons for considering these cases will be clear when we see the applications of the production optimization of waterflooding with well shutoff (Section 7). For the case, where BHPs are no longer in the set of design variables, the number of design variables is

$$N_u = N_I N_{i,c} + 2N_P N_c, \tag{2.61}$$

and for the case, where the production time fractions are no longer design variables in addition to BHPs, the number of design variables is

$$N_u = N_I N_{i,c} + N_P N_c. \tag{2.62}$$

We also consider a case where we have only a single cycle of the production wells. As one can guess, in the case of just one cycle, we do not need to consider cycle lengths as part of the design variable since they are fixed and equal to the life of the waterflooding

process, $\Delta t^{n,m} = t_{total}$. Considering well controls of production and injection wells as well as production time fractions of each production wells for those single cycles of production wells, since cycle lengths will no longer be in the set of design variables, the number of design variables will be calculated from Eq. 2.61.

2.3.1 Constraints on Design Variables

For the well shutoff optimization problem, have bound constraints for all design variables

$$\begin{aligned}
q_{wi}^{low} \leq q_{wi}^{n,m} \leq q_{wi}^{up} \quad &\text{for} \quad n = 1 : N_{ic} \quad \text{and} \quad m = 1 : N_I, \\
p_{bh}^{low} \leq p_{bh}^{n,k} \leq p_{bh}^{up} \quad &\text{for} \quad n = 1 : N_c \quad \text{and} \quad k = 1 : N_P, \\
\widehat{\Delta t}_p^{low} \leq \widehat{\Delta t}_p^{n,k} \leq \widehat{\Delta t}_p^{up} \quad &\text{for} \quad n = 1 : N_c \quad \text{and} \quad m = 1 : N_P, \\
\Delta t^{low} \leq \Delta t^{n,k} \leq \Delta t^{up} \quad &\text{for} \quad n = 1 : N_c \quad \text{and} \quad k = 1 : N_P.
\end{aligned} \tag{2.63}$$

We must note that even though for the well shutoff option we use cycle lengths and production time fractions at each cycle length of each production well to be able to perform gradient-based optimization, BHPs at each production period of each production well are still design variables. It means that even at the production period of each production well a higher value of BHP can shut off the production well. Therefore, we have chosen the upper bound for BHPs as low as possible to avoid this kind of situation since our ultimate goal here is to eliminate a discontinuity of the NPV with respect to any design variable. Considering the reservoir and fluid properties of the production optimization case, we choose $p_{bh}^{low} = 750$ (psi), $p_{bh}^{up} = 2500$, $q_{wi}^{low} = 0$ (STB/D), $q_{wi}^{up} = 10$ (STB/D), $\widehat{\Delta t}_p^{low} = 0$, $\widehat{\Delta t}_p^{up} = 1$, $\Delta t^{low} = 30$ (Days) and $\Delta t^{up} = 1000$ (Days). At higher water injection rates, for a given reservoir, we observe that the average reservoir pressure often maintains a high enough level so that any of the production wells do not shut off to achieve higher maximum NPV as a result of optimization. Therefore, we choose a low value for q_{wi}^{up} to be able to observe well shutoff; that is, to create a situation for at least one of the production well so shut off to increase the maximum NPV during the optimization. However, choosing a lower bound and an upper

bound for cycle lengths for production wells is a heuristic as explained in Section 2.2.1, and the reason for choosing narrow bounds is also discussed in Section 2.2.1 since the WAG problem is also similar to the well shutoff production optimization problem.

Since we include each cycle length as a design variable, we must consider linear equality constraints so that the summation of the cycle lengths of each production wells is equal to the total life of the waterflooding process:

$$\sum_{n=1}^{N_c} \Delta t^{n,k} = t_{total} \quad \text{for} \quad k = 1 : N_P, \tag{2.64}$$

where $t_{total} = 2880$ days, and $N_c = 10$.

As we did in the HnP problem, we can write each of the bound constraints (Eq. 2.63) as two linear inequality constraints.

2.3.2 Normalization of Design Variables, NPV, and Constraints

As linear inequality constraints, we have only the linear bound constraints. The normalized version of bound constraints is given by Eq. 2.24.

Like Eq. 2.27, the normalized linear equality constraints become as follows:

$$\sum_{n=1}^{N_c} \overline{\Delta t}^{n,k} (\Delta t^{up} - \Delta t^{low}) = t_{total} - N_c \Delta t^{low} \quad \text{for} \quad k = 1 : N_P. \tag{2.65}$$

We note that for the case where only a single cycle is considered (since cycle length at each production well is fixed and equal to the life of the waterflooding problem), we will have only bound constraints.

All normalized linear inequality and equality constraints can be formulated as the matrix-vector products as given by Eq. 2.28 and Eq. 2.31, respectively. Like the WAG problem, we also have more than one linear equality constraint, since it is a multi-well problem, we have more than one production well. Therefore, in Eq. 2.31, the number of rows of the matrix $\hat{\mathbf{A}}$ is a $M_{le} = N_P$. When we create a sample set for training as well as a perturbation for Simplex and StoSAG, we truncate our design variables for each sample

to satisfy all these constraints. We deal with constraints via the gradient projection method where we project our gradient on the hyperplane of the intersection of active constraints. However, for the production optimization cases, where a single cycle is considered for each production well, which means cycle lengths are no longer design variables, we deal with the constraints with truncation only since we will have only bound constraints. The truncation method will also be discussed in Section 3.3.

CHAPTER 3

STOCHASTIC GRADIENT-BASED OPTIMIZATION METHODS

There are various gradient approximation methods. They are also referred to as perturbation methods (Glasserman and Ho, 1991). In most of the early works of the researchers, we can see the use of a finite-difference method (FD) as a gradient approximation method. For the small-sized problems where the number of design variables is small, this method is computationally not expensive. However, when the size of the problem becomes larger, the FD method is not efficient even though it is the most accurate gradient approximation method. It is because, in the FD method, we have to perturb each design variable at least once (once in the forward and backward difference method, and twice in the central difference method), which means that the number of perturbations is at least equal to the number of design variables. As mentioned before, for the large-sized problems, the adjoint method (for example see, Jansen (2011) and Forouzanfar et al. (2013)) provides the most accurate gradient and is computationally the most efficient method when the cost function is a differential function of the optimization variables. As the adjoint gradient is not generally available in most commercial simulators, some researchers use the stochastic gradient to approximate the true gradient and thus calculate a search direction. Calculating stochastic gradients does not require the objective function to be differentiable. Therefore, new perturbation methods which do not perturb each design variable individually but perturb the design vector, which is referred to as stochastic gradient estimation, are suggested. This decreases the number of perturbations required for gradient approximation. In this section, as the stochastic gradient estimation methods, we discuss simplex, EnOpt, and StoSAG methods.

We provide a review of the optimization methods used in this study to maximize the NPV function of each production optimization problems, such as HnP (Eq. 2.5), WAG (Eq. 2.36 and 2.43), and well shutoff (Eq. 2.57). As mentioned earlier, in this study, both

deterministic optimization based on a single reservoir model and robust optimization based on an ensemble of reservoir models will be performed. For the deterministic production optimization case, where we use only one realization of the reservoir model, we use the simplex method for production optimization cases of HnP and well shutoff. However, for the WAG problem, we also use the StoSAG method with one realization of the reservoir models using the correlation between the well-controls of each well. For the robust case, we use the StoSAG method. When we used the StoSAG method for optimization, we use singly-smoothed cross-covariance StoSAG (ss-cc-StoSAG), which is the first formulation suggested by Fonseca et al. (2015a) as a modified EnOpt. Later in the work of (Fonseca et al., 2017), they referred to it as StoSAG, where they also give the derivation of StoSAG and its modifications as well as the theoretical relationship with simplex approximate gradient and EnOpt. Therefore, first, we review the simplex gradient and then we discuss the EnOpt method. Later, we give the derivation of StoSAG. We have to mention that all these methods directly use the high-fidelity numerical simulator to perform the maximization of a given NPV function. We also discuss the gradient ascent optimization algorithm as well as the constraint handling methods such as gradient projection and truncation.

3.1 Simplex Gradient Approximation Method

The simplex gradient is one of the gradient approximation methods, and its derivation is mentioned by various researchers (Bortz and Kelley, 1998; Conn et al., 2009; Custódio and Vicente, 2007; Kelley, 1999). To derive the simplex gradient formulation, one performs perturbation of the design variables. It is worth noting that we perform optimization in terms of the normalized design variables. However, when we calculate NPV with the simulator, we directly use the unnormalized design variables. We define $\mathbf{u}$ and $\bar{\mathbf{u}}$ as the vectors of design and normalized design variables, respectively, and $\tilde{\mathbf{u}}$ and $\tilde{\bar{\mathbf{u}}}$ are the vectors of perturbed design variables and the normalized perturbed design variables, respectively. Let l denote the iteration index.

For the deterministic optimization case, the gradient at the iteration level l (denoted

by $\mathbf{g_l}$) in the simplex method is calculated from

$$\mathbf{g}_l = \left(\Delta\widetilde{\mathbf{U}}_l(\Delta\widetilde{\mathbf{U}}_l)^T\right)^+ \Delta\widetilde{\mathbf{U}}_l(\Delta\mathbf{j}_l) \approx \nabla_{\mathbf{u}} J(\mathbf{u}_l), \tag{3.1}$$

where the superscript "+" indicates the generalized (or the Moore-Penrose pseudo) inverse which can be obtained by singular value decomposition (Golub et al., 1989), and $\Delta\widetilde{\mathbf{U}}_l$ is the matrix of the differences between normalized design variables at the l^{th} iteration and their perturbed values, given by

$$\Delta\widetilde{\mathbf{U}}_l = \left[\tilde{\bar{\mathbf{u}}}_{l,1} - \bar{\mathbf{u}}_l, \tilde{\bar{\mathbf{u}}}_{l,2} - \bar{\mathbf{u}}_l, \cdots, \tilde{\bar{\mathbf{u}}}_{l,N_p} - \bar{\mathbf{u}}_l\right]. \tag{3.2}$$

Here, $\tilde{\bar{\mathbf{u}}}_{l,1}, \cdots, \tilde{\bar{\mathbf{u}}}_{l,N_p}$ are the vectors of normalized perturbed design variables. We have N_p perturbations at each iteration l. $\Delta\mathbf{j}_l$ in Eq. 3.1 is the vector of difference of the NPV at iteration l and given as follows:

$$\Delta\mathbf{j}_l = \left[J(\tilde{\mathbf{u}}_{l,1}) - J(\mathbf{u}_l), J(\tilde{\mathbf{u}}_{l,2}) - J(\mathbf{u}_l), \cdots, J(\tilde{\mathbf{u}}_{l,N_p}) - J(\mathbf{u}_l)\right]^T. \tag{3.3}$$

The perturbation does not have to be stochastic. In the normal simplex gradient, one might deterministically choose perturbations of design variables. However, when Do and Reynolds (2013) obtain the simplex gradient from the fundamental equation of G-SPSA gradient, they end up with the simplex gradient where perturbation of design variables come from the normal distribution. They start their derivation assuming $\tilde{\bar{\mathbf{u}}}_{l,j} \sim \mathcal{N}(\bar{\mathbf{u}}_l, \mathbf{C_U})$, where $\mathbf{C_U}$ is the covariance matrix of the design variable vector. Usually, a correlation between well controls of each well is assumed since changing the well controls abruptly is not manageable and not preferred. However, mathematically speaking, we do not have to assume a correlation between the design variables when we perturb them from a normal distribution; that is, one can use diagonal covariance, $\mathbf{C_U} = \sigma^2\mathbf{I}$, where σ is the standard deviation, also known as perturbation size for gradient approximation.

For the production optimization of HnP, we use only diagonal covariance matrix,

$\mathbf{C_U} = \sigma^2\mathbf{I}$, where $\sigma = 0.03$. The reason for neglecting the correlation between design variables is because, for the HnP problem, we do not divide production and injection periods into control time steps, and therefore, there is no need to assume a correlation. We choose the standard deviation (σ) to be equal to 0.03 for the HnP problem. For the well shutoff problem, we also do not assume a correlation between BHPs at each cycle of each production well since at each cycle after every production period there is a shutoff period, which means BHPs are not correlated. However, for the WAG problem, since every half-cycle becomes our control time step, we assume correlations between well controls, BHPs for each production well, and water and gas injection rates for each injection well. Note that there is no correlation between the two control time steps of different wells since different wells can easily be assigned different schedules of production or injection. However, there is a correlation between well controls for each individual wells. We model correlation by using a covariance matrix which is a block diagonal matrix in which block matrix represents the covariance matrix of each well:

$$\mathbf{C_U} = \begin{bmatrix} {\mathbf{C_U}}^1 & 0 & \dots & 0 \\ 0 & {\mathbf{C_U}}^2 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & {\mathbf{C_U}}^{N_w} \end{bmatrix}, \tag{3.4}$$

where N_w is the total number of wells, $N_w = N_P + N_I$. For the calculation of this correlation in this work, we used spherical covariance formulation. Denoting the (i,j) entry of ${\mathbf{C_U}}^n$ by $C^n_{i,j}$, for $n = 1 : N_w$, spherical covariance is calculated as follows:

$$C^n_{i,j} = \sigma_n^2 \left(1 - \frac{3}{2}\left(\frac{|i-j|}{N_t^n}\right) + \frac{1}{2}\left(\frac{|i-j|}{N_t^n}\right)^3\right), \tag{3.5}$$

if $|i-j| \leq N_t^n$, and $C^n_{i,j} = 0$ if $|i-j| > N_t^n$, where N_t^n is the number of control step of the well n, i and j are the i^{th} and j^{th} control steps of the given well n, respectively. Note that, in the WAG problem, the standard deviation is the same for all wells,$\sigma_n^2 = \sigma^2$ for $n = 1 : N_w$. Recall that for other design variables, in the WAG production optimization

problem, we do not assume correlation for some of the design variables such as ICVs, gas injection time fractions, and cycle lengths. There is no practical need and advantage of assuming correlation for such design variables. Also for the production optimization case of the WAG problem, where we include gas injection time fractions and cycle lengths into the set of design variables, we do not consider correlation for any of the design variables, even for well controls. It is because we do not know the control time steps beforehand since they are also part of the optimization being an implicit function of gas injection time fractions and cycle lengths.

We used the simplex gradient for the deterministic production optimization problems. Note that when we perturb the design variables, all samples need to be inside the bound constraints as well as the design variables linear inequality and equality constraints. To do so, we truncate the samples so that they do not violate any of the constraints.

3.2 EnOpt and StoSAG Gradient Approximation Methods

The concept of gradient approximation using an ensemble of control vectors for a single reservoir model is first mentioned in the work done by Lorentzen et al. (2006). However, the name of ensemble optimization (EnOpt) was thought up by Chen et al. (2009) (also Chen and Oliver (2010)). Therefore, when we refer to EnOpt, we refer to the implementation of Chen et al. (2009), where it refers to robust optimization (ensemble optimization). For robust optimization, we maximize the expectation of NPV over the uncertain reservoir parameters given by

$$J_E(\mathbf{u}_l) = \frac{1}{N_e} \sum_{k=1}^{N_e} J(\mathbf{m}_k, \mathbf{u}_l), \tag{3.6}$$

where N_e is the number of realizations of reservoir parameters (geological models), known as ensemble size; $\mathbf{m}_k$ is the k_{th} realization of the reservoir parameter vector. $J(\mathbf{m}_k, \mathbf{u}_l)$ is calculated for each production optimization problem considered here, using their corresponding NPV formulations (Eqs. 2.5, 2.36, 2.43 and 2.57). To approximate the gradient of Eq. 3.6, the naive approach would be choosing an ensemble of control vectors for each member of the ensemble of geological models. This means a single approximation of this gradient would re-

quire the number of simulations equal to the multiplication of ensemble size and the number of perturbations of the design variable vector. Since this approach is computationally very expensive, Chen et al. (2009) suggest one to one correspondence of random design variable and realization of geological model:

$$\mathbf{g}_l = \frac{1}{N_p - 1} \sum_{j=1}^{N_p} (\tilde{\tilde{\mathbf{u}}}_{l,j} - \overline{\tilde{\tilde{\mathbf{u}}}_l})(J(\mathbf{m}_j, \tilde{\tilde{\mathbf{u}}}_{l,j}) - \overline{J(\mathbf{m}, \bar{\mathbf{u}}_l)}), \tag{3.7}$$

where $\overline{\tilde{\tilde{\mathbf{u}}}_l}$ is the sample mean of the perturbations of design variable given as

$$\overline{\tilde{\tilde{\mathbf{u}}}_l} = \frac{1}{N_p} \sum_{j=1}^{N_p} \tilde{\tilde{\mathbf{u}}}_{l,j}, \tag{3.8}$$

and $\overline{J(\mathbf{m}, \bar{\mathbf{u}}_l)}$ defined as

$$\overline{J(\mathbf{m}, \bar{\mathbf{u}}_l)} = \frac{1}{N_p} \sum_{j=1}^{N_p} J(\mathbf{m}_j, \tilde{\tilde{\mathbf{u}}}_{l,j}). \tag{3.9}$$

With that suggestion of Chen et al. (2009), the number of simulations required for the gradient approximation counts down to the ensemble size of the geological model. However, the method suggested by Chen et al. (2009) did not provide a gradient accurate enough to find the optimum of the production optimization problem according to Raniolo et al. (2013). They observed that a better gradient approximation can be achieved if five perturbations of the design variable were paired with each geological model (5 to 1 correspondence). Fonseca et al. (2017) theoretically investigated the formulation of EnOpt. They observed that the standard EnOpt formulation (Eq. 3.7) requires two assumptions. The first one assumes that mean of the perturbed design vector at iteration l is a good approximation of the design variable at that iteration, which was used as mean to generate perturbations:

$$\overline{\tilde{\tilde{\mathbf{u}}}_l} = \frac{1}{N_p} \sum_{j=1}^{N_p} \tilde{\tilde{\mathbf{u}}}_{l,j} \approx \bar{\mathbf{u}}_l. \tag{3.10}$$

However, in production optimization problems, we always have constraints. Therefore, we

need to truncate some of our perturbed design variables. Considering this modification to the perturbed design variables and that the N_p is not always sufficiently large, this assumption does not hold anymore.

Another assumption of Chen et al. (2009) according to Fonseca et al. (2017) is that for any reservoir model parameter $\mathbf{m}$, we have

$$J(\mathbf{m}_1, \bar{\mathbf{u}}_l) \approx J(\mathbf{m}_2, \bar{\mathbf{u}}_l) \approx \cdots J(\mathbf{m}_{N_e}, \bar{\mathbf{u}}_l). \tag{3.11}$$

This assumption is invalid because there is variance in the prior model of the reservoir model parameter.

Instead of using these assumptions which can potentially lead inaccurate gradient approximation, Fonseca et al. (2017) defined the singly-smoothed cross-covariance StoSAG (ss-cc-StoSAG) formulation following the works by Fonseca et al. (2015a,b)

$$\mathbf{g}_l = \frac{1}{N_e} \sum_{k=1}^{N_e} \left(\frac{1}{N_p} \sum_{j=1}^{N_p} (\tilde{\tilde{\mathbf{u}}}_{l,j,k} - \bar{\mathbf{u}}_l)(J(\mathbf{m}_k, \tilde{\mathbf{u}}_{l,j,k}) - J(\mathbf{m}_k, \mathbf{u}_l)) \right). \tag{3.12}$$

The reason Eqs. 3.7 to be called singly-smoothed cross-covariance is because Do and Reynolds (2013) showed that

$$\mathbf{g}_l \approx \mathbf{C_U} \nabla_{\mathbf{u}} J(\bar{\mathbf{u}}_l), \tag{3.13}$$

where $\mathbf{g}_l$ is given by Eq. 3.7, which means that gradient is further multiplied by the covariance matrix (therefore, it is called singly-smoothed). Similarly, the reason for Eq. 3.12 to be called singly-smoothed is because Fonseca et al. (2017) showed that it is approximately equal to multiplication of the simplex gradient of the expectation of NPV over reservoir model parameters with the covariance matrix:

$$\mathbf{g}_l \approx \mathbf{C_U} \left(\sum_{k=1}^{N_e} \nabla_{\mathbf{u}} J(\mathbf{m}_k, \bar{\mathbf{u}}_l \right) = \mathbf{C_U} \nabla J_E(\bar{\mathbf{u}}_l). \tag{3.14}$$

To derive doubly-smoothed cross-covariance StoSAG (ds-cc-StoSAG), we can multiply Eq.

3.12 with $\mathbf{C_U}$.

For the robust optimization cases, we use ss-cc-StoSAG. For the deterministic case of the WAG problem, we use ss-cc-StoSAG, just assuming one realization ($N_e = 1$) in Eq. 3.12 to compare its performance over using a simplex gradient, which is not smoothed gradient, as well as the ds-cc-StoSAG gradient.

3.3 Handling of Constraints

We consider a constrained optimization problem with linear inequality and equality constraints. For some of the cases of the production optimization problems considered in this study, we only have bound constraints. In that case, we just use the truncation method. However, when the linear equality and inequality constraints are included, to handle constraints, we use the gradient projection method (Luenberger et al., 1984; Forouzanfar et al., 2010).

3.3.1 Truncation

When we have bound constraints, we can truncate the design variables at a new iteration (at iteration $l + 1$) within their upper and lower bounds. Since we work with the min-max normalized design variables, we truncate the normalized design variables between zero and one, i.e.,

$$\bar{u}_k = \begin{cases} 1 \text{ if } \bar{u}_k > 1 \\ \\ 0 \text{ if } \bar{u}_k < 0 \end{cases}. \tag{3.15}$$

This makes the gradient to be inside the feasible region, but it affects the gradient direction.

3.3.2 Gradient Projection

Gradient projection is the method to change the direction of the gradient to stay in the feasible region by projecting the gradient onto the hyperplane of the active constraints. With the estimated gradient if we are on any of the linear constraints, we project the gradient onto the hyperplane of the active constraints. In the case of linear inequality constraints, when the updated design variable $\bar{\mathbf{u}}$ is on the one or more constraint hyperplanes, it means that those

constraints are active. In the case of linear equality constraint, our design variable during the optimization must be on this hyperplane as well so that the linear equality constraint is always an active constraint during the optimization procedure. In the gradient projection algorithm, we project the gradient on the hyperplanes of active constraints only. Therefore, for gradient projection, we use the matrix $\mathbf{B}$ and the vector $\mathbf{c}$ to consider only the coefficients of active constraints ($\mathbf{B}\bar{\mathbf{u}} = \mathbf{c}$). Let us denote the number of active constraints with M_a, then the dimensions of $\mathbf{B}$ and $\mathbf{c}$ are $M_a \times N_u$ and M_a, respectively. Denoting the gradient calculated at iteration l by $\mathbf{g}_l$, we can write the equation for the projected gradient as

$$\mathbf{d}_l = \left(\mathbf{I} - \mathbf{B}^T(\mathbf{B}\mathbf{B}^T)^{-1}\mathbf{B}\right)\mathbf{g}_l, \tag{3.16}$$

where $\mathbf{g}_l$ is approximate gradient calculated either simplex gradient method (Eq. 3.1) or the ss-cc-StoSAG method (Eq. 3.12). Then we use this modified gradient inside the optimization algorithm. As an optimization algorithm, we use a gradient ascent algorithm. In the next section, we describe the gradient ascent algorithm, in which we construct the gradient projection algorithm.

3.4 Gradient Ascent Algorithm

Derivation of the gradient ascent comes from the first order Taylor expansion of the objective function (in this study, NPV) at each iteration, at each point $\bar{\mathbf{u}}_l$. Below *a gradient ascent algorithm* is provided, in which the gradient is approximated with the either *simplex gradient* method or *ss-cc-StoSAG* method, linear constraints are handled with the *gradient projection* method, bound constraints are handled with *truncation*, and the step size β is chosen by a *backtracking algorithm* to compute $\bar{\mathbf{u}}_{l+1}^{-}$ from

$$\bar{\mathbf{u}}_{l+1} = \bar{\mathbf{u}}_l + \beta\frac{\mathbf{d}_l}{||\mathbf{d}_l||_\infty}. \tag{3.17}$$

1. Set iteration index l to zero, $l = 0$. Choose an initial guess for the vector of normalized design variables, $\bar{\mathbf{u}}_0$.

2. Calculate $J(\mathbf{u}_l)$ using the simulator.

3. Using the simplex method (Eq. 3.1) or ss-cc-StoSAG method (Eq. 3.12) calculate the gradient ($\mathbf{g}_l$) at a given iteration, l.

4. Start with the highest step size (initial backtracking step size), $\beta = \beta^0 = 1$, and apply the *gradient projection algorithm* described at step 5 to find $\bar{\mathbf{u}}_{l+1}$ and $J(\mathbf{u}_{l+1})$ for the chosen step size β:

5. Gradient projection algorithm: To better understand the gradient projection algorithm, we illustrated this procedure for a 2D design variable case in **Fig. 3.1**.

   (a) Check if at the point of given design vector $\bar{\mathbf{u}}_l$ plus the small step (γ) of the gradient we activate any of the constraints (we take this small arbitrary step as $\gamma = 0.001$). This means we find $\mathbf{B}$ and $\mathbf{c}$ which satisfies the following equation:

$$\mathbf{B}(\bar{\mathbf{u}}_l + \gamma \mathbf{g}_l) \geq \mathbf{c}. \tag{3.18}$$

   If we activate any constraint (means that with this gradient we would violate at least one of the constraints (then those constraints become active constraints), use Eq. 3.16 to calculate the projected gradient $\mathbf{d}_l$ (in Fig. 3.1, constraint c_1 is activated, and therefore, gradient $\mathbf{g}_l$ is projected on constraint plane c_1 to get direction $\mathbf{d}_l$). Note that with assuming γ different from zero (arbitrarily small number) we take not only the constraints, which $\bar{\mathbf{u}}_l$ is on, but also the constraints, which $\bar{\mathbf{u}}_l$ is very close to, as active constraints. Note that if $\bar{\mathbf{u}}_l$ is not on any of the constraints, $\mathbf{B} = \mathbf{0}$, where $\mathbf{0}$ is a null matrix, in Eq. 3.16, meaning that $\mathbf{d}_l = \mathbf{g}_l$. Now $\mathbf{d}_l$ is our search direction.

   (b) Starting from $\bar{\mathbf{u}}_l$ and $\mathbf{d}_l$, find whether we activate any constraints with the step size β. To do so, solve the gradient ascent equation (Eq. 3.17) for step size which will activate any linear constraints. Denote i^{th} row of matrix $\mathbf{A}$ with $\mathbf{a}_i^T$, which contains coefficients of design variables for the i^{th} constraint equation. Then,

calculate the step size α_i, which activates i^{th} constraint by

$$\alpha_i = \frac{b_i - \mathbf{a}_i{}^T \bar{\mathbf{u}}_l}{\frac{\mathbf{a}_i{}^T \mathbf{d}_l}{||\mathbf{d}_l||_\infty}}, \tag{3.19}$$

which is derived from:

$$\mathbf{a}_i{}^T \left(\bar{\mathbf{u}}_l + \alpha_i \frac{\mathbf{d}_l}{||\mathbf{d}_l||_\infty} \right) = b_i. \tag{3.20}$$

Then, choose the minimum of these steps, $\alpha_{min} = \min_i(\alpha_i)$ and compare it with the step size from backtracking, β to check to see any of the constraints are activated with step β (in Fig. 3.1, constraint c_2 is activated). Note that when we find the minimum of α, ignore any negative values of α_i since a negative α_i indicates a descent direction, and we are not interested in that direction. Ignore α_is equal to zero and infinite values as well because this indicates that we are on one of the constraint hyperplanes already. The latter case means that we have already projected gradient on that hyperplane and with any step size we will not violate that constraint, so we do not need to care about that constraint anymore. We just search for the step size on that hyperplane that will violate one or more of the other constraints.

(c) If $\beta \leq \alpha_{min}$, it means that we are in a feasible region with given step size β. Thus accept the given step size β, and calculate the next update of $\mathbf{u}$ with the gradient ascent equation given by Eq. 3.17. Go to step 6.

(d) If $\beta > \alpha_{min}$, it means that with given step size β we violate at least one of the constraints. In this case, we continue to the following step:

(e) Using α_{min} and direction $\mathbf{d}_l$ calculate the next point $\bar{\mathbf{u}}_{l,1}$, which is called the middle point (in Fig. 3.1, this point is on the intersection of the constraint planes c_1 and c_2), using $\bar{\mathbf{u}}_l$ in the gradient ascent equation (Eq. 3.17). This middle point should not be confused with $\bar{\mathbf{u}}_{l+1}$. Since we are on constraints with this point, we need

to project the direction. It is worth noting that when we project the direction, project the gradient direction ($\mathbf{g}_l$) to get the direction at this middle point $\mathbf{d}_{l,1}$. After having the middle point ($\bar{\mathbf{u}}_{l,1}$) and the direction ($\mathbf{d}_{l,1}$), we repeat step b to calculate $\alpha_{min,1}$. Then, using this step size and direction, we calculate a second middle point, $\bar{\mathbf{u}}_{l,2}$ (in Fig. 3.1, this point is in the intersection of the constraint planes (c_2 and c_3). Then we test step size β with the following condition:

(f) If $||\bar{\mathbf{u}}_{l,2} - \bar{\mathbf{u}}_l||_\infty \leq \beta$ stop here and choose second middle point as update for the next iteration of the optimization process, i.e., set $\bar{\mathbf{u}}_{l+1} = \bar{\mathbf{u}}_{l,2}$.

(g) If $||\bar{\mathbf{u}}_{l,2} - \bar{\mathbf{u}}_l||_\infty > \beta$, it means calculated $\bar{\mathbf{u}}_{l+1}$ will be between points $\bar{\mathbf{u}}_{l,1}$ and $\bar{\mathbf{u}}_{l,2}$ which satisfies $||\bar{\mathbf{u}}_{l+1} - \bar{\mathbf{u}}_l||_\infty = \beta$. We replace $\bar{\mathbf{u}}_{l+1}$ using the last updated direction $\mathbf{d}_l$ and the first middle point $\bar{\mathbf{u}}_{l,1}$ in the gradient ascent equation (Eq. 3.17) to obtain

$$\left|\left|\left(\alpha\frac{\mathbf{d}_l}{||\mathbf{d}_l||_\infty} + \bar{\mathbf{u}}_{l,1}\right) - \bar{\mathbf{u}}_l\right|\right|_\infty = \beta. \tag{3.21}$$

So, we want to solve this equation for α. However, since it is very difficult to solve it for an infinity norm, we can replace the infinity norm with a two norm (Eq. 3.22). In this case, however, $||\bar{\mathbf{u}}_{l+1} - \bar{\mathbf{u}}_l||_\infty$ will be less or equal to β because the infinity norm provides a value which is always less than or equal to a value computed from a two norm. This update is satisfactory though

$$\left|\left|\left(\alpha\frac{\mathbf{d}_l}{||\mathbf{d}_l||_\infty} + \bar{\mathbf{u}}_{l,1}\right) - \bar{\mathbf{u}}_l\right|\right|_2 = \beta. \tag{3.22}$$

(h) Using α computed from Eq. 3.22 and the first middle point $\bar{\mathbf{u}}_{l,1}$ in the gradient ascent equation (Eq. 3.17), obtain design vector for the next iteration $\bar{\mathbf{u}}_{l+1}$.

The motivation of applying this gradient projection algorithm twice is to have as big update of $\bar{\mathbf{u}}$ as possible. As can be seen from Fig. 3.1, we try to get a big update of the design variable as backtracking step size β. If we go with the smaller steps of update, we will need more iterations to find the optimum. This means more simulation runs.

Therefore, we apply the gradient projection algorithm twice.

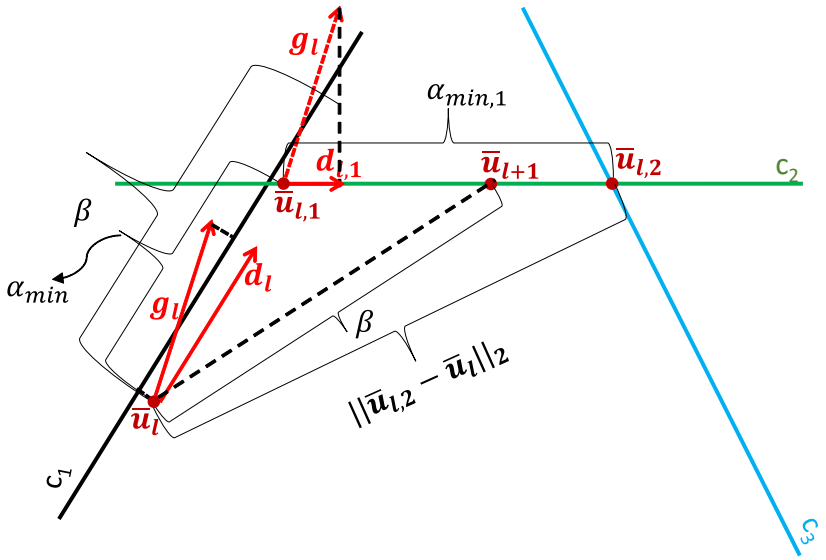


Figure 3.1: Illustration of the gradient projection procedure for a 2D design variable case. α_{min}, $\alpha_{min,1}$, β are step sizes; $\mathbf{g}_l$ is the gradient direction at the point $\bar{\mathbf{u}}_l$; $\mathbf{d}_l$ and $\mathbf{d}_{l,1}$ are the projected directions of gradient $\mathbf{g}_l$ at points $\bar{\mathbf{u}}_l$ and $\bar{\mathbf{u}}_{l,1}$ on constraint planes c_1 and c_2 respectively; c_1, c_2, and c_3 are constraint planes.

6. Check if $J(\mathbf{u}_{l+1}) \geq J(\mathbf{u}_l)$. If it is satisfied, accept $\mathbf{u}_{l+1}$ and go to the next step.

7. Check convergence by using the following convergence criteria:

$$\frac{|J(\bar{\mathbf{u}}_{l+1}) - J(\bar{\mathbf{u}}_l)|}{max[|J(\bar{\mathbf{u}}_l)|, 1]} \leq \epsilon_J, \tag{3.23}$$

and

$$\frac{||\bar{\mathbf{u}}_{l+1} - \bar{\mathbf{u}}_l||_2}{max[||\bar{\mathbf{u}}_l||_2, 1]} \leq \epsilon_u, \tag{3.24}$$

where $||\cdot||_2$ is the l_2 norm of a vector. In our research, we used $\epsilon_J = 0.0001$ and $\epsilon_u = 0.001$. If converged, $\bar{\mathbf{u}}_{l+1}$ is our optimum design vector, and optimization process ends. If not converged, set $l := l + 1$, $\bar{\mathbf{u}}_l := \bar{\mathbf{u}}_{l+1}$ and check whether we exceed the maximum allowed simulation runs, N_{sim}^{max}, otherwise go to step 2. Calculation of the number of the simulation runs is given end of this section

8. If $J(\mathbf{u}_{l+1}) < J(\mathbf{u}_l)$, update the step size, $\beta := \beta/2$.

9. Using this updated step size in the gradient projection algorithm (step 5) find $\bar{\mathbf{u}}_{l+1}$ again. Then, check the condition at step 6. We repeat step cutting until either satisfy the condition at step 6 or we reach the maximum allowed number of step cuttings N_{cut}^{max}, which is $N_{cut}^{max} = 5$ in our backtracking algorithm.

10. If backtracking cannot find a design vector that increases the value of J within the maximum allowable number of backtracking iterations (number of step cuttings), this means our stochastic gradient can be in the wrong direction. In this case, we follow one of the following approaches which are named as *Backtracking Approach* 1 and *Backtracking Approach* 2:

    - *Backtracking Approach* 1: We simply accept the new design variable which gives the highest J, set $l = l + 1$ and go to step 7 (Chen and Reynolds, 2017).
    - *Backtracking Approach* 2: Reject the gradient of this iteration ($\mathbf{g}_l$), and generate a new set of perturbed design vectors and compute the gradient $\mathbf{g}_l$ either from Eq. 3.1 for simplex gradient or 3.12 for ss-cc-StoSAG. Repeat steps 4 to 10 with this new gradient until we reach the maximum allowable number of gradient re-estimations, (denoted by N_{grad}^{max}), which can be taken as five. If the maximum allowable number of gradient estimation is reached, this may indicate that $\bar{\mathbf{u}}_l$ is very close to the local optimum (because with the stochastic gradient approximation method it is hard to estimate gradient around local optimum). Therefore, we accept $\bar{\mathbf{u}}_l$ as our optimum design vector and end optimization (Do and Reynolds, 2013).

In this work, for the production optimization problem of HnP, we mainly use *Backtracking Approach* 2 for the termination of the gradient ascent algorithm. However, for some of the cases of the HnP problem we also use *Backtracking Approach* 2 for comparison purposes. We found that the former approach usually takes more iterations but terminates with a higher NPV value than does the latter approach for those cases. For production optimization of the WAG and well shutoff problems, we use only *Backtracking Approach* 1.

In all production optimization problems, $N_{sim}^{max} = 2000$, $N_{cut}^{max} = 5$, $\beta^0 = 1$ unless otherwise stated. Other parameters change depending the production optimization problem. For the HnP problem:

- Standard deviation (or perturbation size) is $\sigma = 0.03$.
- The number of perturbations, N_p, ranges from 10 to 20 depending on the production optimization case.
- $N_{grad}^{max} = 5$. Note that only for this production optimization problem, we use *Backtracking Approach* 2.

For the WAG problem:

- σ ranges from 0.03 to 0.1, depending on the production optimization case.
- N_p ranges from 10 to 20, depending on the production optimization case.

For the well shutoff problem:

- $\sigma = 0.1$.
- $N_p = 10$.

We should note that Fonseca et al. (2015b) show that the number of perturbations and the size of the perturbations affects the quality of the stochastic gradient (its direction compared to the direction of the true gradient). Unfortunately, we do not know a way to determine a priori the best choice for either the number of perturbations or the perturbations. Hence, we have chosen the above values, heuristically, based on the suggestions given by Chen and Reynolds (2017).

It is worth noting that depending on which gradient approximation method is used, we call the optimization method either the simplex optimization method or the StoSAG optimization method to distinguish them in the Results section given later.

For some of the production optimization cases, where the number of design variables is not big, we also use the FD method to compare the results with simplex optimization

and StoSAG optimization. We choose the FD method for computational efficiency. The perturbation size used for the FD method changes depending on the production optimization case.

3.5 Number of Simulations

For the deterministic optimization based on *Backtracking Approach* 2, the number of simulations required by the simplex method at "convergence" can be calculated from

$$N_{sim,d} = 1 + \sum_{l=1}^{N_{iter}} [(N_{cut}^{max} + 1) \times (N_{grad}^{l} - 1) + N_{cut}^{l} + 1 + N_{grad}^{l} \times N_{p}^{l}], \tag{3.25}$$

where $N_{sim,d}$ is the number of simulation runs, N_{iter} is the total number of iterations until "convergence", N_{grad}^{l} is the number of gradient estimation required to find the ascent direction at the l^{th} iteration, N_{cut}^{l} is the number of step-size cuts took to find $J(\bar{\mathbf{u}}_{l+1}) > J(\bar{\mathbf{u}}_{l})$, and N_{p}^{l} is the number of perturbations to estimate the gradient at the l^{th} iteration. This formula is based on the following computations:

1. We need 1 simulation run for our initial guess

2. At each iteration l, we perform the following simulation runs:

   (a) Gradient estimation. Each gradient estimation takes N_{p}^{l} simulation runs, i.e., we need $N_{grad}^{l} \times N_{p}^{l}$ simulation runs.

   (b) For each of the failing gradient estimation cases (out of N_{grad}^{l}, $(N_{grad}^{l} - 1)$ gradient estimation failed, only the last one succeeded), we need to run $(N_{cut}^{max} + 1)$ simulation runs in the backtracking algorithm, where we cut our initial step-size with the maximum number of times N_{cut}^{max}. Here, we add 1 for the case where we did not cut the step-size yet and then we add N_{cut}^{max}, since we performed N_{cut}^{max} cuts for the previous failed gradient. Thus, we run $(N_{cut}^{max} + 1) \times (N_{grad}^{l} - 1)$ simulations in the case of failing gradient estimation.

(c) For the case of the successful gradient estimation, using that gradient in the backtracking algorithm, we run $N_{cut}^{l} + 1$ simulation runs inside the backtracking algorithm.

For the deterministic optimization based on *Backtracking Approach* 1, we set $N_{grad}^{l} = 1$ for $l = 1 : N_{iter}$ in Eq. 3.25.

For robust optimization, we multiply $N_{sim,d}$ computed by Eq. 3.25 by N_e, the number of ensembles of reservoir models, i.e., $N_{sim,e} = N_e \times N_{sim,d}$.

CHAPTER 4

MACHINE LEARNING-BASED ITERATIVE-SAMPLING-REFINEMENT OPTIMIZATION METHODS

As stated before, both simplex and StoSAG optimization methods use directly the simulator and approximate the gradient numerically by perturbation and consequently often require a large (on the order of hundreds or thousands of depending on the scale of the problem) number of reservoir simulation runs. Hence, there is a need for efficient optimization methods for the life-cycle production optimization problems. This is the main objective of our research.

We apply the machine learning (ML)-based iterative-sampling-refinement optimization method to solve the production optimization problems we considered. We use two different kernel-based ML methods; LS-SVR and GPR. The reason to use a kernel-based method is that it can approximate an objective function with a high dimension of design variables with a small training data set so that higher computational efficiency is achieved.

Another point that is worth mentioning is that there are parametric and nonparametric proxy models. The parametric ones assume a functional form, or shape of the function to be predicted, such as proxy models based on linear, polynomial functions of inputs. On the other hand, nonparametric (e.g., GPR) or semiparametric (e.g., LS-SVR) proxy models do not make explicit assumptions about the functional form of the model. A parametric model has the disadvantage that the functional form used to estimate the function is usually very different from the true function, in which case the resulting model will not fit the data well. In contrast, nonparametric and semiparametric proxies avoid this danger since essentially no assumption about the form of the function is made. Here, we refer the readers to Chapter 2 of the book by James et al. (2013) who provide a comprehensive discussion on the advantages of the parametric versus nonparametric proxy models, and Guo and Reynolds (2018)

who provide a comprehensive review and discussion on the advantages of LS-SVR over the response surface models (see Introduction section of Guo and Reynolds 2018). For example, Guo and Reynolds presented a comparison of a linear response-surface-based proxy with an LS-SVR-based proxy for a robust waterflooding production optimization problem (Example 2 of Guo and Reynolds 2018). It is shown that the linear response-surface proxy does not provide an accurate proxy that can yield a higher NPV than the LS-SVR proxy.

In the following two sections, we discuss the theoretical backgrounds of GPR and LS-SVR and show mathematical derivations of the training procedure of LS-SVR and GPR. We try to be brief in theory and derivations leaving the reader with useful references for reading and investigation more since the theory for each of these ML methods is quite comprehensive. Then, we provide a review of the theory on the LS-SVR-based and GPR-based proxy models that are used in the iterative proxy-based-optimization method (referred to here as the iterative-sampling-refinement optimization) presented by Guo and Reynolds (2018) who use the ideas given in Queipo et al. (2005) and Forrester and Keane (2009) for solving production optimization problems. Note that Guo and Reynolds (2018) used this optimization method only with the LS-SVR method.

4.1 LS-SVR Proxy Model

Support-vector machines (SVMs) for classification and nonlinear function estimation were introduced by Vapnik et al. (1995) and Vapnik (1998). Least-squares (LS) versions of SVM's have been used for classification (Suykens and Vandewalle, 1999) and function estimation (Drucker et al., 1997; Saunders et al., 1998; Suykens et al., 2002). LS-SVR is a kernel-based method and is widely used for solving nonlinear regression problems. Here, we provide the theory and derivation of LS-SVR that can also be found in Saunders et al. (1998), Suykens et al. (2002), and Guo and Reynolds (2018).

4.1.1 Review of Basic Theory

Given a number of training input and output data where the output data are gen-

erated from the "true" model, LS-SVR "learns" how to map the input data to the output response and generates a function to predict the output response of the true model for any given input data. In our applications, the "true" model corresponds to the NPV (denoted by J here) defined by Eq. 2.5, and the training output (denoted by J_o) corresponds to the NPV generated from a numerical simulator. As the training NPV data are generated from a numerical reservoir simulator, they contain numerical noise resulting from inexact solutions of both linear and nonlinear solvers used in the simulator (Guo and Reynolds, 2018). So, the output response is related to the true model as a function of the vector of input data (denoted by the N_u-dimensional vector $\mathbf{u}$) by

$$J_o = J(\mathbf{u}) + \epsilon, \tag{4.1}$$

where ϵ denotes the error.

Let N_{tr} be the number of training data, $\mathbf{u}_k, k = 1, 2, ..., N_{tr}$, be the N_u-dimensional k^{th} training input vector, and let $J_{o,k}$ be a scalar output response corresponding to $\mathbf{u}_k$. The pair $(\mathbf{u}_k, J_{o,k})$ is referred to as the training sample, and the set S is called a training set of N_{tr} outputs (also referred to as observations); that is, $S = \{(\mathbf{u}_k, J_{o,k}), k = 1, 2, ..., N_{tr}\}$. For applications with the ML-based methods, the training set is normalized to treat all the input and output variables equally for training an ML-based proxy (e.g., see Crone et al. 2006; Guo and Reynolds 2018). Hence, we define $\bar{\mathbf{u}}$ to be the N_u-dimensional normalized vector of design variables (see Eq. 2.22) and $\bar{J}$ as the normalized scalar output (see Eq. 2.23). The training set S is then converted to a normalized training set $\bar{S} = \{(\bar{\mathbf{u}}_k, \bar{J}_{o,k}), k = 1, 2, ..., N_{tr}\}$.

In the LS-SVR method, function estimation on a given vector of normalized input data $\bar{\mathbf{u}}$ is performed as follows (Suykens et al., 2002):

$$\hat{\bar{J}}(\bar{\mathbf{u}}) = \mathbf{w}^T \boldsymbol{\phi}(\bar{\mathbf{u}}) + b, \tag{4.2}$$

where $\hat{\bar{J}}$ is called the predictive function, $\boldsymbol{\phi}(\bar{\mathbf{u}})$ is a mapping function (a vector function) that maps the input data into a higher dimensional feature space, $\mathbf{w}$ is a vector of regression

coefficients (or also referred to as weights) in the nonlinear feature space and can be infinite-dimensional, and b is a scalar constant (also referred to as the bias term).

4.1.2 Training Procedure of LS-SVR

The last two parameters in Eq. 4.2 ($\mathbf{w}$ and b) are obtained over the training set $S = \{(\bar{\mathbf{u}}_k, \bar{J}_{o,k}), k = 1, 2, ...,N_{tr}\}$ by minimizing the following objective or objective function:

$$O(\mathbf{w}, b) = \frac{1}{2}\mathbf{w}^T\mathbf{w} + \frac{1}{2}\gamma \sum_{k=1}^{N_{tr}} [\mathbf{w}^T \boldsymbol{\phi}(\bar{\mathbf{u}}_k) + b - \bar{J}_{o,k}]^2, \tag{4.3}$$

where γ is the regularization term (also referred to as a hyperparameter) that controls the regularization procedure and is related to the signal-to-noise ratio in the measurements. The bigger the value of γ, the higher the signal-to-noise ratio, and the less the necessary reduction in the coefficients. We choose γ using hyperparameter optimization method to be discussed in Section 4.1.3. In the case of noisy data, one avoids overfitting by taking a smaller value. Note that this minimization process is the training process of LS-SVR. The minimization problem given by Eq. 4.3 is equivalent to (see Suykens et al. (2002); Guo and Reynolds (2018) and Cawley and Talbot (2007)):

$$\min_{\mathbf{w},b,\mathbf{e}} O(\mathbf{w}, \mathbf{e}) = \frac{1}{2}\mathbf{w}^T\mathbf{w} + \frac{1}{2}\gamma \sum_{k=1}^{N_{tr}} e_k^2, \tag{4.4}$$

subject to

$$e_k = J_{o,k} - \mathbf{w}^T \boldsymbol{\phi}(\bar{\mathbf{u}}_k) - b. \tag{4.5}$$

where $\mathbf{e}$ is the N_{tr}-dimensional vector of residuals with components $e_k; k = 1, 2, ...,N_{tr}$, between the predicted function and the observation. The problem defined by Eq. 4.4 is not solvable for $\mathbf{w}$ due to the potential infinite-dimensionality of the weight vector $\mathbf{w}$. Instead, one constructs the Lagrange dual function (see Suykens et al. 2002) with the Lagrange multipliers $\alpha_k; k = 1, 2, ...,N_{tr}$ as

$$L(\mathbf{w}, b, e; \boldsymbol{\alpha}) = O(\mathbf{w}, e) - \sum_{k=1}^{N_{tr}} \alpha_k [\mathbf{w}^T \boldsymbol{\phi}(\bar{\mathbf{u}}_k) + b + e_k - \bar{J}_{o,k}]. \tag{4.6}$$

The optimal solution of Eqs. 4.4 and 4.6 is the training procedure of the LS-SVR. This procedure is shown in Appendix B. As a result of this training procedure, we find $\boldsymbol{\alpha}$ and b. Therefore, basically, finding $\boldsymbol{\alpha}$ and b is called the training process. After having found them, we just substitute Eq. B.2a into Eq. 4.2 to obtain

$$\hat{\bar{J}}(\bar{\mathbf{u}}) = \sum_{k=1}^{N_{tr}} \alpha_k \boldsymbol{\phi}(\bar{\mathbf{u}}_k)^T \boldsymbol{\phi}(\bar{\mathbf{u}}) + b, \tag{4.7}$$

where $\boldsymbol{\phi}(\bar{\mathbf{u}}_k)^T \boldsymbol{\phi}(\bar{\mathbf{u}})$ is the dot product of mapping functions $\boldsymbol{\phi}(\bar{\mathbf{u}}_k)$ and $\boldsymbol{\phi}(\bar{\mathbf{u}})$. From Eqs. B.6 and B.7, we can see that computational complexity of training procedure has relationship with the size of the training data as $\mathcal{O}(N_{tr}^3)$ for the Cholesky decomposition. However, some researchers attempted to reduce the computational complexity, such as Gordan et al. (2006). They achieved 30% reduction in the number of elementary operations per classification in SVM in the face classification problem.

Using Mercer's condition (the details are skipped here; see Suykens et al. 2002), the LS-SVR proxy model given by Eq. 4.7 for function estimation can be expressed in terms of a scalar kernel function, $K(\bar{\mathbf{u}}_k, \bar{\mathbf{u}}) = \boldsymbol{\phi}(\bar{\mathbf{u}}_k)^T \boldsymbol{\phi}(\bar{\mathbf{u}})$, as

$$\hat{\bar{J}}(\bar{\mathbf{u}}) = \sum_{k=1}^{N_{tr}} \alpha_k K(\bar{\mathbf{u}}_k, \bar{\mathbf{u}}) + b. \tag{4.8}$$

As can be seen, we need to know the kernel function to be used in Eq. 4.8, as well as to be able to find $\boldsymbol{\alpha}$ with Eq. B.7, and to find b from Eq. B.6, in advance. In the following subsection, we briefly discuss the kernel model selection and its hyperparameters.

4.1.3 Kernel Model Selection And Hyperparameter Optimization

Selection of the kernel model and its hyperparameters is also part of the training procedure. Therefore, this selection process is also called training of the LS-SVR. There are

various types of kernel functions to use for K in Eq. 4.8. Selection of the kernel function is not part of our training procedure, and we choose kernel function manually. For our applications, a radial-basis-function (RBF) kernel is used

$$K(\bar{\mathbf{u}}_k, \bar{\mathbf{u}}) = \exp\left(-\frac{||\bar{\mathbf{u}}_k - \bar{\mathbf{u}}||_2^2}{2\sigma_{bw}^2}\right), \tag{4.9}$$

where σ_{bw} is the bandwidth (or also referred to as the characteristic length-scale). The reason we use RBF is that it is preferred for the cases where we do not know the structure of the response surface to different design variables, as in the case of the NPV for the production-optimization problem considered here, and we want to have a smooth surface to avoid being trapped in small local maxima and overfitting if the training data are noisy (Drucker et al. 1997, Guo and Reynolds 2018). As to the role of the characteristic length-scale in Eq. 4.9, when the length-scale is small, the function is very sensitive to variances in the input variables and thus can potentially produce precise predictions, but with a risk of overfitting. When the characteristic length is large, the function changes slowly, so predictions become less precise but more robust.

Those hyperparameters (γ and σ_{bw}^2) can be assumed as we do for the kernel function. However, it is better for those hyperparameters to be estimated. So, the question is how to estimate them. Obviously, it is an optimization process, and therefore, we need a score function (it is an objective function to be optimized in the hyperparameter optimization procedure). k-fold Cross-validation is used for evaluating many ML methods such as a decision-tree and XGBoost. It is also used for evaluating the LS-SVR model. The cross-validation iterates through the folds, and at every iteration, with one of the folds, it validated the model that trained with the rest of the folds. We repeat this process until all folds are used as validation. Using an accuracy measure, we score each model and compute the average of all scores. Using this score, we can tune hyperparameters. One way of doing it is to manually change hyperparameters to achieve a higher score (accuracy). However, this is time-consuming, and we do not want the process to be manual since we will use an iterative-

sampling-refinement optimization algorithm which is an automatic process. *Randomized grid search cross-validation* is one of the most popular approaches. In this approach, we start by creating a grid of hyperparameters we want to optimize with values that we want to try for those hyperparameters. Instead of trying all possible combinations of hyperparameters the algorithm randomly chooses a value for each hyperparameter from the grid and evaluates the model for those random hyperparameters. The randomized grid search method is one of the optimization methods used. However, one can also use the simplex optimization method instead of the randomized grid search method on the score generated using k-fold cross-validation. Note that in this research, the simplex gradient is different from the simplex optimization method. Also known as Dantzig's simplex algorithm is an optimization method used for linear programming. For details of this optimization method one can check the books Dongarra and Sullivan (2000) and Stone and Tovey (1991).

We use the LS-SVMLab v1.7 toolbox in Matlab which considers k-fold cross-validation to estimate these parameters. As an optimization algorithm, it uses simplex and grid-search. We use grid-search, and 10-fold cross-validation (De Brabanter et al., 2010).

The construction of the initial LS-SVR-based proxy model to be used in the iterative-sampling-refinement method is shown in **Fig. 4.1**. Once constructed, the initial proxy is used in the iterative-sampling-refinement algorithm to perform production optimization. As the LS-SVR proxy given by Eq. 4.8 is analytic to the design variables, we make accurate and efficient use of the sequential quadratic programming (SQP) method to maximize NPV using the gradient of the LS-SVR proxy model, given by

$$\nabla \hat{\tilde{J}}(\bar{\mathbf{u}}) = \sum_{k=1}^{N_{tr}} \left[\frac{\alpha_k}{\sigma_{bw}^2} K(\bar{\mathbf{u}}_k, \bar{\mathbf{u}}) \cdot (\bar{\mathbf{u}}_k - \bar{\mathbf{u}}) \right], \tag{4.10}$$

which follows by using Eq. 4.9 in Eq. 4.8 and then taking the gradient of the resulting equation.

4.2 GPR Proxy Model

Unlike the LS-SVR and other supervised ML algorithms that learn exact values (therefore, they are deterministic) for every parameter in a function, GPR uses a fully Bayesian approach, which infers a probability distribution over all possible values of the parameters of the function. The uncertainty over the parameters induces uncertainty in the predictive function itself. Thus, it becomes a stochastic process where every finite collection of output functions are distributed by multivariate Gaussian distribution. So rather than calculating the probability distribution of parameters of a specific function, we calculate the probability distribution over all functions that fit the data (Williams and Rasmussen, 2006). Note that the GPR proxy is a nonparametric model.

4.2.1 Review of Basic Theory

First, we define a prior Gaussian distribution over output latent functions of NPV for constructing a GPR based proxy model as follows:

$$p(\mathbf{j}|\mathbf{U}) \sim \mathcal{N}(\boldsymbol{\mu}, \mathbf{K}), \tag{4.11}$$

where $p(\mathbf{j}|\mathbf{U})$ represents the probability density function for the latent function $\mathbf{j}$ given the $N_u \times N_{tr}$ design matrix $\mathbf{U}$ aggregating the N_u-dimensional column vector inputs for all N_{tr} training cases, that is $\mathbf{U} = [\mathbf{u}_1, \mathbf{u}_2, ..., \mathbf{u}_{N_{tr}}]$. Here and throughout, we use the notation $\mathcal{N}(\mathbf{a}, \boldsymbol{\Sigma})$ for the Gaussian distribution with mean $\mathbf{a}$ and covariance matrix $\boldsymbol{\Sigma}$.

In Eq. 4.11, $\mathbf{j}$ should not be confused with a vector-valued function. Rather, it is a vector containing entries of a scalar random function evaluated at different values of the input vector $\mathbf{u}_k, k = 1, 2, ...,N_{tr}$. For our problem, the NPV is treated as a random function (denoted by J). Therefore, $\mathbf{j}$ is an N_{tr}-dimensional vector of latent scalar functions of NPV computed at each input vector $\mathbf{u}_k$ of the design variables; that is, $\mathbf{j} = [J(\mathbf{u}_1), J(\mathbf{u}_1), ..., J(\mathbf{u}_{N_{tr}})]^T$. In Eq. 4.11, $\mathcal{N}(\boldsymbol{\mu}, \mathbf{K})$ represents the Gaussian distribution with mean vector $\boldsymbol{\mu}$, an N_{tr}-dimensional vector, and covariance matrix $\mathbf{K}$, the $N_{tr} \times N_{tr}$ (symmetric positive semidefinite) matrix. In the equations to be given here, we will consider identical and constant

mean for the vector $\mathbf{j}$, that is, $\boldsymbol{\mu} = \mu \mathbf{1}_{N_{tr}}$, where $\mathbf{1}_{N_{tr}}$ is the N_{tr}-dimensional vector having all entries equal to unity. Each entry of $\mathbf{K}$ is the covariance function specifying the covariance between pairs of $J(\mathbf{u}_i)$ and $J(\mathbf{u}_j)$ for $i = 1, 2, ..., N_{tr}$, and $j = 1, 2, ..., N_{tr}$. In GPR, the covariance functions between the pairs are estimated by using a kernel function: $cov[J(\mathbf{u}_i), J(\mathbf{u}_j)] = K(\mathbf{u}_i, \mathbf{u}_j)$, where K is the kernel function. Note that $\mathbf{j}$ here is a noise-free random vector of latent functions.

4.2.2 Training Procedure of GPR

As stated by Williams and Rasmussen (2006), we are usually not primarily interested in drawing random functions from the prior but want to incorporate the knowledge that the training data provide about the function. As mentioned before, we often have the noisy versions of the model function as the sample training set; i.e., $S = \{(\mathbf{u}_k, J_{o,k}), k = 1, 2, ..., N_{tr}\}$. Assuming additive independent identically distributed Gaussian noise ϵ with mean zero and variance σ_n^2 (in the vectorial notation: $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \sigma_n^2 \mathbf{I})$ with zero mean vector and covariance matrix $\sigma_n^2 \mathbf{I}$, where $\mathbf{I}$ is the $N_{tr} \times N_{tr}$ identity matrix), then the relationship between the random vector $\mathbf{j}(\mathbf{u})$ and the observed noisy (targets) data vector $\mathbf{j}_o$ is given by

$$\mathbf{j}_o = \mathbf{j}(\mathbf{u}) + \boldsymbol{\epsilon}, \tag{4.12}$$

and hence, the prior on the noisy observations becomes

$$p(\mathbf{j}_o|\mathbf{U}) \sim \mathcal{N}(\mu \mathbf{1}_{N_{tr}}, \mathbf{K} + \sigma_n^2 \mathbf{I}). \tag{4.13}$$

Then, the prior model given by Eq. 4.11 is updated to the posterior distribution by using the sample training set through Bayes' theorem as

$$p(\mathbf{j}|\mathbf{j}_o, \mathbf{U}) \propto p(\mathbf{j}_o|\mathbf{j}, \mathbf{U}) p(\mathbf{j}|\mathbf{U}), \tag{4.14}$$

and this procedure is training procedure of GPR. Here, the proportionality can be turned into

an equality by dividing through the marginal distribution by $p(\mathbf{j}_o|\mathbf{U}) = \int p(\mathbf{j}_o|\mathbf{j}, \mathbf{U})p(\mathbf{j}|\mathbf{U})\mathrm{d}\mathbf{j}$.

Eq. 4.14 constitutes the basis of the GPR training procedure to construct a GPR-based proxy model. The posterior $p(\mathbf{j}|\mathbf{j}_o, \mathbf{U})$ is a Gaussian distribution given by (von Mises, 1964; Williams and Rasmussen, 2006; Quinonero-Candela et al., 2007):

$$p(\mathbf{j}|\mathbf{j}_o, \mathbf{U}) \sim \mathcal{N}(\mu\mathbf{1}_{N_{tr}} + \mathbf{K}(\mathbf{K} + \sigma_n^2\mathbf{I})^{-1}(\mathbf{j}_o - \mu\mathbf{1}_{N_{tr}}), \mathbf{K} - (\mathbf{K} + \sigma_n^2\mathbf{I})^{-1}\mathbf{K}). \tag{4.15}$$

In GPR, our main objective is to construct a predictive GP-based proxy model, and hence we can consider just the function values at the points $\mathbf{u}$ that concern us, namely the training points $\mathbf{u}_1, \mathbf{u}_2, ..., \mathbf{u}_{N_{tr}}$, and the test points $\mathbf{u}_1^*, \mathbf{u}_2^*, ..., \mathbf{u}_{N_{pr}}^*$ whose $\mathbf{j}(\mathbf{u}^*)$ value we wish to predict. Given the prediction points aggregated in the $N_u \times N_{pr}$ design matrix $\mathbf{U}^* = [\mathbf{u}_1^*, \mathbf{u}_2^*, ..., \mathbf{u}_{N_{pr}}^*]$, where N_{pr} is the number of prediction points, we make predictions of the latent functions at these points: $\mathbf{j}^* = [J(\mathbf{u}_1^*), J(\mathbf{u}_2^*), ..., J(\mathbf{u}_{N_{pr}}^*)]^T$, from the predictive posterior distribution, $p(\mathbf{j}^*|\mathbf{j}_o, \mathbf{U}, \mathbf{U}^*)$. We follow the book of von Mises (1964) on mathematical theory probability and statistics to derive the predictive posterior distribution; section 9.3 in his book.

Let us define an N_t-dimensional random vector $\mathbf{j}^\Sigma$ aggregating the N_{pr}-dimensional vector of predictions $\mathbf{j}^*$ and the N_{tr}-dimensional vector of training points $\mathbf{j}_o$; that is

$$\mathbf{j}^\Sigma = \begin{bmatrix} \mathbf{j}^* \\ \mathbf{j}_o \end{bmatrix}. \tag{4.16}$$

Note that $N_t = N_{pr} + N_{tr}$. The joint distribution of $\mathbf{j}_o$ and $\mathbf{j}^*$ is a Gaussian distribution with mean vector $\mu\mathbf{1}_{N_t}$, where $\mathbf{1}_{N_t}$ is the N_t-dimensional vector, i.e., $p(\mathbf{j}^*, \mathbf{j}_o|\mathbf{U}^*, \mathbf{U}) \sim \mathcal{N}(\boldsymbol{\mu}^\Sigma, \mathbf{K}^\Sigma)$,

$$\boldsymbol{\mu}^\Sigma = \begin{bmatrix} \mu\mathbf{1}_{N_{pr}} \\ \mu\mathbf{1}_{N_{tr}} \end{bmatrix}, \tag{4.17}$$

and $N_t \times N_t$ symmetric covariance matrix $\mathbf{K}^\Sigma$ which can be partitioned as

$$\mathbf{K}^{\Sigma} = \begin{bmatrix} \mathbf{K}_p & \mathbf{K}_{p,o} \\ \mathbf{K}_{o,p} & \mathbf{K}_o \end{bmatrix}, \tag{4.18}$$

where $\mathbf{K}_p$ is the $N_{pr} \times N_{pr}$ covariance matrix for the predicted data, $\mathbf{K}_o$ the $N_{tr} \times N_{tr}$ covariance matrix for noisy training data and is computed from $\mathbf{K}_o = (\mathbf{K} + \sigma_n^2\mathbf{I})$, and $\mathbf{K}_{o,p}$ ($=\mathbf{K}_{p,o}^T$) is the $N_{tr} \times N_{pr}$ cross-covariance matrix describing the covariance between the training data and predicted data.

Using Eqs. 4.16-4.18 and following books by Muirhead (2009) and Eaton (1983) on multivariate Gaussian distribution, it can be shown that the conditional distribution $p(\mathbf{j}^*|\mathbf{j}_o, \mathbf{U}, \mathbf{U}^*)$ is Gaussian with mean and covariance matrix given, respectively, by

$$\boldsymbol{\mu}^* = \mu\mathbf{1}_{N_{pr}} + \mathbf{K}_{p,o}\mathbf{K}_o^{-1}(\mathbf{j}_o - \mu\mathbf{1}_{N_{tr}}), \tag{4.19}$$

and

$$\mathbf{K}^* = \mathbf{K}_p - \mathbf{K}_{p,o}\mathbf{K}_o^{-1}\mathbf{K}_{o,p}. \tag{4.20}$$

When we make predictions at prediction point, $\mathbf{u}^*$, using the predictive model which comes from Gaussian normal distribution with mean given in Eq. 4.19 and covariance given in Eq. 4.20, we want to make point prediction, we do not want to stochastic prediction; that is, prediction of a bunch of points coming from predictive distribution. Therefore, we try to find our point prediction, denoted as $J_{guess}(\mathbf{u}^*)$, that will achieve the smallest loss with true output, denoted as $J_{true}(\mathbf{u}^*)$, but we do not know $J_{true}(\mathbf{u}^*)$. So, we, instead, minimize the *expected loss* with respect to the distribution of predictive models defined as follows:

$$R_{\mathcal{L}}(J_{guess}(\mathbf{u}^*)) = \int \mathcal{L}(J^*, J_{guess}(\mathbf{u}^*))p(J^*|\mathbf{j}_o, \mathbf{u}^*)dJ^*, \tag{4.21}$$

where $J^* = J(\mathbf{u}^*)$ is Gaussian random variable at prediction point $\mathbf{u}^*$; $\mathcal{L}(J^*, J_{guess}(\mathbf{u}^*))$ is a loss function. According to Berger (2013), in general the value of $J_{guess}(\mathbf{u}^*)$ when loss function is absolute loss, $|J^* - J_{guess}(\mathbf{u}^*)|$, is the median of $p(J^*|\mathbf{j}_o, \mathbf{u}^*)$, while loss function is

squared loss, $(J^* - J_{guess}(\mathbf{u}^*))^2$, then it is the mean of this distribution. Since our predictive distribution is Gaussian, the mean and the median is the same. They also state that, for any symmetric loss function and for any symmetric predictive distribution we always get $J_{guess}(\mathbf{u}^*)$ as the mean of the predictive distribution. It means that our predictions, $\mathbf{j}_{guess}(\mathbf{U}^*)$ are given by the Eq. 4.19.

If we define $\bar{\mathbf{j}}^* = \boldsymbol{\mu}^* - \mu \mathbf{1}_{N_{pr}}$ and $\bar{\mathbf{j}}_o = \mathbf{j}_o - \mu \mathbf{1}_{N_{tr}}$, then we can express Eq. 4.19 as

$$\bar{\mathbf{j}}^* = \mathbf{K}_{p,o}\mathbf{K}_o^{-1}\bar{\mathbf{j}}_o, \tag{4.22}$$

which shows that that the mean prediction is a linear combination of observations $\bar{\mathbf{j}}_o$ and this is sometimes referred to as a linear predictor. Derivations of Eqs. 4.19 and 4.20 are given in Appendix C.1. Covariance and mean given by Eqs. 4.19 and 4.20, respectively, are the same as the covariance and mean given in Eq. 4.15 since both of these derivations are based on the derivation of conditional multivariate Gaussian distribution conditioned to another random vector. One can get Eq. 4.15, knowing that the marginal distribution of $\mathbf{j}$ is given by the Eq. 4.11, and thus by replacing the term $\mu \mathbf{1}_{N_{pr}}$ in Eq. 4.19 with $\mu \mathbf{1}_{N_{tr}}$, and replacing the term $\mathbf{K}_p$ in Eq. 4.20 with $\mathbf{K}$.

Eq. 4.22 can also be seen as a linear combination of kernel functions, each one centered on a training point, by writing (see Williams and Rasmussen 2006)

$$\bar{\mathbf{j}}^* = \mathbf{K}_{p,o}\boldsymbol{\alpha}, \tag{4.23}$$

where $\boldsymbol{\alpha}$ is the N_{tr}-dimensional vector and is calculated from

$$\boldsymbol{\alpha} = \mathbf{K}_o^{-1}\bar{\mathbf{j}}_o = (\mathbf{K} + \sigma_n^2\mathbf{I})^{-1}(\mathbf{j}_o - \mu \mathbf{1}_{N_{tr}}). \tag{4.24}$$

If we consider a single prediction point $\mathbf{u}_j^*$ and then, Eq. 4.23 can be written as:

$$\hat{J}(\mathbf{u}_j^*) = \sum_{i=1}^{N_{tr}} \alpha_i K(\mathbf{u}_i, \mathbf{u}_j^*), \text{ for } j = 1, 2, ..., N_{pr}. \tag{4.25}$$

As in the case of the LS-SVR, if we work with the normalized vectors of design variables and the prediction model function denoted by $\hat{\bar{J}}(\bar{\mathbf{u}})$ and express Eq. 4.25 in terms of normalized variables, then the GPR prediction proxy function is then given by

$$\hat{\bar{J}}(\bar{\mathbf{u}}_j^*) = \sum_{i=1}^{N_{tr}} \alpha_i K(\bar{\mathbf{u}}_i, \bar{\mathbf{u}}_j^*). \tag{4.26}$$

Eq. 4.26 is similar to the equation for the LS-SVR prediction (Eq. 4.8), but Eq. 4.26 does not have a bias term b. So, as we can see from Eqs. 4.8 and 4.26, LS-SVR and GPR have similar predictive models. However, in LS-SVR, as mentioned in the previous paragraph, we do not account for the uncertainty in our prediction, whereas the GPR provides fully probabilistic predictive distributions, including estimates of the uncertainty in the predicted values of NPV.

4.2.3 Selection of Kernel Function And Hyperparameter Optimization

As can be seen, we need to choose the kernel model and its hyperparameters to be able to perform predictions. However, it is not easy to specify all aspects of the covariance function. Therefore, to make GPR a more practical tool, it is essential to find a method that solves the model selection problem. The model selection problem is interpreted to include a discrete choice of the form of kernel function and to include finding hyperparameters of the chosen kernel function.

A general form of a kernel or covariance function that can be used in Eq. 4.26 can be expressed as

$$K(\bar{\mathbf{u}}_i, \bar{\mathbf{u}}_j^*, \boldsymbol{\Theta}) = C(\bar{\mathbf{u}}_i, \bar{\mathbf{u}}_j^*, \boldsymbol{\theta}) + \sigma_n^2 \delta_{\bar{\mathbf{u}}_i, \bar{\mathbf{u}}_j^*}, \tag{4.27}$$

where δ is the Kronecker delta, C is the correlation function, σ_n^2 is noise in data (or also referred to as nugget), $\boldsymbol{\theta}$ is a vector containing the hyperparameters of the correlation function. $\boldsymbol{\Theta} = [\boldsymbol{\theta}, \sigma_n^2]^T$ is a vector containing all the hyperparameters of the covariance function. A multitude of possible families of covariance functions exist, including squared exponential, polynomial, Matérn, neural network, etc., see Table 4.2 of Williams and Rasmussen (2006).

Each of these covariance functions has a number of free hyperparameters which need to be determined. For example, the correlation function (C in Eq. 4.27) such as the squared exponential can be parameterized in terms of hyperparameters as (Williams and Rasmussen, 2006)

$$C(\bar{\mathbf{u}}_i, \bar{\mathbf{u}}_j^*, \boldsymbol{\theta}) = \sigma_f^2 \exp\left[-\frac{1}{2}(\bar{\mathbf{u}}_i - \bar{\mathbf{u}}_j^*)^T \mathbf{M}(\bar{\mathbf{u}}_i - \bar{\mathbf{u}}_j^*)\right], \tag{4.28}$$

where $\boldsymbol{\theta} = (\{\mathbf{M}\}, \sigma_f^2)^T$, σ_f^2 is the variance of the (noise free) signal, $\mathbf{M}$ is the $N_u \times N_u$ symmetric distance measure matrix, and $\{\mathbf{M}\}$ denotes the parameters in $\mathbf{M}$. There are various choices for the matrix $\mathbf{M}$. For example, $\mathbf{M} = \text{diag}(\boldsymbol{l})^{-2}$, where $\boldsymbol{l}$ is the N_u-dimensional vector with positive elements (hyperparameters) $l_1, l_2, ..., l_{N_u}$ which play the role of characteristic length-scales; they determine how far you need length-scale to move (along a particular axis) in input space for the function values to become uncorrelated. Note that, each of these length-scales can take the same value. So, choosing a covariance function for a particular application comprises both setting of hyperparameters within a family and comparing across different families. Once we want to find hyperparameters as well as the kernel function, this procedure is also part of the training procedure of GPR. Therefore, the selection of a covariance function and its parameters is also referred to as training of GPR.

There are various model selection methods. However, the following principles cover the most:

- given the data compute probability of the model, *model selection*;
- *estimate the generalization error*;
- *bound the generalization error.*

In Appendix C.2, we discuss the Bayesian view on *model selection.* It basically computes the probability of the model given the data, based on the marginal likelihood. As an application of Bayesian model selection to GPR, as we see from Appendix C.2, the generalized maximum likelihood (in the machine learning literature is also called the "evidence framework" approximation) applied to Gaussian Processes allowing to learn the parameters of the

kernel by maximizing the log marginal likelihood function (Eq. C.36) with respect to the hyperparameters.

In Section 4.1.3, we discussed the cross-validation, which is the *estimation of the generalization error*. Since GPR is a probabilistic model, as a loss function for evaluation of the predictions, we consider cross-validation using the negative log probability loss

$$-\log p(J^*|\mathbf{j}_o, \mathbf{u}^*) = \frac{1}{2}\log(2\pi(\sigma^*)^2) + \frac{(J^* - \mu^*)^2}{2(\sigma^*)^2}, \tag{4.29}$$

where μ^* and $(\sigma^*)^2$ are predictive mean and predictive variance, respectively, at the prediction point $\mathbf{u}^*$. One can easily see that μ^* is just the element of $\boldsymbol{\mu}^*$ given in Eq. 4.19 at the prediction point; and $(\sigma^*)^2$ is the diagonal element of $\mathbf{K}^*$ given in Eq. 4.20 at the prediction point, $\mathbf{u}^*$ plus the variance of noise assumed for predictions, σ_n. We provide formulations of them as

$$\mu^* = \mu + \mathbf{k}_{p,o}\mathbf{K}_o^{-1}(\mathbf{j}_o - \mu\mathbf{1}_{N_{tr}}) \tag{4.30}$$

and

$$(\sigma^*)^2 = K(\mathbf{u}^*, \mathbf{u}^*) - \mathbf{k}_{p,o}\mathbf{K}_o^{-1}\mathbf{k}_{o,p} + \sigma_n^2, \tag{4.31}$$

where $\mathbf{k}_{p,o}$ is $1 \times N_{tr}$ dimensional matrix showing covariance between prediction point ($\mathbf{u}^*$) and observation points, and $\mathbf{k}_{o,p}$ is $N_{tr} \times 1$ dimensional matrix and is just transpose of the matrix $\mathbf{k}_{p,o}$. For further details of using cross-validation as generalization error, see Wahba (1990). However, for the *bound on the generalization error*, one can check section 7.4.2 of the book Williams and Rasmussen (2006).

We use the Matlab toolbox *gprMdl* to perform GPR computations, model selection, etc., (The MathWorks, 2020). This package allows us to estimate the mean of the random latent function, kernel model as well as hyperparameters. It also allows users to use parallel programming in hyperparameter optimization (Bayesian model selection process) decreasing the computational time more than 4 times, depending on the number of logical processors. In our case, we have 12 logical processors, and in this case, the computational time required

for hyperparameter optimization with a maximum number of evaluations of the objective function (which is log marginal likelihood function) is set to 50, and with using parallel programming option was 37 seconds, whereas without using parallel programming it was 150 seconds. The GPR package also gives users freedom of choosing optimizer, stopping criteria of hyperparameter optimization, a maximum number of objective function evaluations, etc. In all our cases, we used the *quasi-newton* method as an optimizer, which is the default option for this package. For further details of the package, please see the user guide of the toolbox (The MathWorks, 2020). The construction of the initial GPR proxy model to be used in the iterative-sampling-refinement optimization method is shown in the flow chart given in Fig. 4.1.

Since we do not fix our kernel function during iterative-sampling-refinement optimization, a numerical gradient of the model is used for the optimization in SQP when we use the GPR proxy model in our applications. There are plenty of kernel functions that can be used in GPR, such as *matern31*, *rational quadratic*, etc. in *gprMdl* toolbox in Matlab (Williams and Rasmussen, 2006).

4.2.4 Uncertainty Assessment of GPR.

As mentioned before, one of the main advantages of the GPR over the LS-SVR method is to have covariance information. The diagonal elements of this covariance matrix show the uncertainty bound of the points (observation points and prediction points), and off-diagonal elements show covariance between those points. Using the diagonal elements of covariance, the standard deviation and confidence interval of those points can be calculated. So, we can construct the confidence interval of our NPV predictions. The Matlab GPR package allows us to get the standard deviation of the GPR prediction model. Putting aside its computational inefficiency as compared to the LS-SVR, the main advantage of a GPR-based proxy model over an LS-SVR-based proxy is that it provides fully probabilistic forecast distributions, including predictions regarding the uncertainty of predictions. In other words, it is a nonparametric stochastic model providing an estimation of the mean and covariance

information on the predictive latent function by accounting for noise in the observed and test data as well as the hyperparameters of the kernel function (see Eqs. 4.19, 4.20 and 4.27). The diagonal elements of the covariance information given by Eq. 4.20 can be used to assess the uncertainty on the predicted value of the NPV and the off-diagonal elements to assess the correlation between the predicted value and the training values. Here, we construct the approximate 95% confidence interval of the optimum NPV predicted by the GPR-based iterative optimization method by using the following equation:

$$\hat{J}_{max} - 1.96\sqrt{\sigma^2_{\hat{J}_{max}}} \leq J_{true} \leq \hat{J}_{max} + 1.96\sqrt{\sigma^2_{\hat{J}_{max}}}, \tag{4.32}$$

where $\hat{J}_{max} = \hat{J}(\mathbf{u}_{opt})$ is the maximum NPV predicted by the GPR proxy model at the optimal vector of design variables $\mathbf{u}_{opt}$, J_{true} is the "true", but the unknown maximum (or optimum) value of the NPV, and $\sigma^2_{\hat{J}_{max}}$ is the variance of the predicted NPV at the optimum point found as a result of GPR-based iterative-sampling-refinement optimization, which can be obtained from Eq. 4.20 (expressed for a single prediction point; that is the optimum vector $\mathbf{u}_{opt}$ yielded $\hat{J}_{max}$) as

$$\sigma^2_{\hat{J}_{max}} = K(\bar{\mathbf{u}}_{opt}, \bar{\mathbf{u}}_{opt}) - \mathbf{k}^T_{opt,o}\mathbf{K}_o^{-1}\mathbf{k}_{opt,o} + \sigma_n^2, \tag{4.33}$$

where

$$\mathbf{K}_o = \mathbf{K}(\bar{\mathbf{U}}, \bar{\mathbf{U}}) + \sigma_n^2\mathbf{I}. \tag{4.34}$$

In Eqs. 4.33 and 4.34, $K(\bar{\mathbf{u}}_{opt}, \bar{\mathbf{u}}_{opt})$ is the signal variance σ_f^2 and $\mathbf{k}_{opt,o}$ denote the N_{tr}-dimensional vector of covariances between $\hat{J}_{opt}$ and the N_{tr} training points, $\mathbf{K}(\bar{\mathbf{U}}, \bar{\mathbf{U}})$ is $N_{tr} \times N_{tr}$ covariance matrix containing the covariances between pairs of training points. For the robust case, $\mathbf{u}_{opt}$ and $\mathbf{U}$ in Eqs. 4.33 and 4.34 contain the model parameter vector $\mathbf{m}$ as well. Hence, the variance given by Eq. 4.33 applies for a single realization of the model vector $\mathbf{m}$. Since, in the robust case, we look at the mean of these NPV values predicted considering N_m realizations of the reservoir model, the overall variance of J_{max} for this case

is taken as the mean of the variances of all realizations of the model parameters.

In the GPR model, accuracy and the variance of the model are different things. When we test our GPR model with the test data, we use the mean of the GPR predictive model. Hence, when we calculate accuracy measures (R^2, MAE, $RMSE$), we consider the response of the mean of the GPR predictive model at test points, and we compare it (checking the data mismatch) with the test data. However, the variance of the GPR model shows the range of predictive models around the mean of these predictive models. The mean of the GPR can match the test data, but it may have high variance at those points. The size of the design variable affects the variance of the GPR model. If one checks any kernel function, there is a term that shows the norm of the difference of the design variables at two points. If the size of the design variable is large, the norm between the two points becomes larger. As the value of the norm becomes larger the covariance between points decreases. If we check the Eq. 4.33, we will see that the variance, $\sigma_{j_{max}}$ will decrease as the covariance between $\bar{\mathbf{u}}_{opt}$ and training data points decreases. Another reason for the high variance of the GPR model is the low value of "characteristic lengths", $\boldsymbol{l}$, and the high value of the variance of (noise-free) signal, σ_f^2 since it decreases the covariances between points. In general, as the covariance between points decreases, the variance of prediction increases.

4.3 Training, Validation and Test of ML Models

Data are sampled using different sampling methods such as random sampling, stratified sampling. We use the Latin Hypercube Sampling (LHS) method. This method is an application of the uniform distribution to higher dimensions. The method became famous due to its generalization ability; that is since we sample all over the response surface, we will have less chance to have an overfitting problem. Each data point has features (we call it design variable), $\mathbf{U}$, and its corresponding outputs, $\mathbf{j}_o$. In our research since our output is obtained from the simulator, after having samples of design variables, for each design variable, we run a simulator for each design variable. Using the flow response of simulator, we calculate NPV using corresponding NPV equation of the production optimization problem

(Eqs. 2.5, 2.36, 2.43, and 2.57). These NPVs are our outputs ($\mathbf{j}_o$) in our data set. Then we split data into training, validation, and test data sets. The training set is used for the training procedure; the validation set is used for checking the performance of the model while training to observe if our model overfits or not. With validation data, we want to have a generalized model. It can be used either manually modifying hyperparameters or using hyperparameter optimization on the loss function defined using the loss of each fold. Then later we check the performance of this generalized model on test data. Usually, training, validation, and test data are split into 2:1:1 fractions. However, one can choose their own fraction. For the case where k-fold cross-validation is performed, we just split our data into training and test set in 2:1 portions. Then training and k-fold cross-validation are performed on the training data set.

The training procedure of the ML models usually ends up being an optimization problem. The objective function to be optimized changes depending on the ML method used. The objective function is called the loss function when it needs to be minimized and the score function when it needs to be maximized. For most of the ML methods, this objective function is mean absolute error (MAE), mean square error (MSE), for especially parametric models, which is also called data mismatch. In Section 4.1.2, in the training procedure of the LS-SVR, we performed such kind of optimization problem, where we found the minimum of the objective function, which was Lagrangian (see Appendix B). Since that objective function was convex, it did not require iterative optimization. For the hyperparameter optimization process of LS-SVR, however, we perform an iterative optimization process (see Section 4.1.3) where we used either simplex or randomized grid search optimization method on the k-fold cross-validated loss function. However, the objective functions for other ML methods such as Neural Networks require an iterative optimization process, not only for hyperparameter optimization but also for the optimization of the model parameters.

In Section 4.2.2, we saw that Bayes process of finding posterior mean and covariance of predictive distribution is a part of the training process of GPR, which is not an optimization problem yet. However, for the next stage, for hyperparameter optimization, we

need to perform optimization with respect to hyperparameters, both in using the Bayesian model selection and cross-validation methods. Note that in hyperparameter optimization using the Bayesian model selection, our loss function is the negative log marginal likelihood function (Eq. C.36). For prediction, we also use predictive distribution. However, for evaluating predictions of model (validation) using k-fold cross-validation, we use the negative log probability of prediction (since validation data are used for prediction) (Eq. 4.29). For point prediction of GPR, we observed that we solve the optimization problem where the loss function is given by Eq. 4.21.

Cross-validation is an important measure to see the trade-off between model fit and its complexity; that is a trade-off between errors coming from the bias of the model and from the variance of the model. As mentioned earlier, when k-fold cross-validation is used, the loss becomes the average of the loss of each fold. And one can plot this average loss such as MAE and MSE with respect to the complexity of a model to find the optimum complexity where the average loss is minimum. Note that the complexity of a model is also hyperparameter. For instance, the type of the kernel model, hyperparameters of the kernel model affects the complexity of the GPR and LS-SVR models. Therefore, for LS-SVR and GPR, k-fold cross-validation can be used as hyperparameter optimization. For more detail about cross-validation as well as statistical analysis methods of models, we refer to the book James et al. (2013).

As training/test data splitting process from the data obtained using LHS sampling method, we prefer using what we call as *uniform data split* instead of random data split method. In *uniform data split*, after having sampled all data using LHS sampling, we include the first two points into the training set and the following one point into the test set, and repeat the process till all data are split into training and test sets. This data split method is not random. In Subsection 5.3.7, we show that the accuracy of an ML model is very sensitive to random data split. As the ML method in that section we used LS-SVR, and our results are based on the *uniform data split*.

4.4 Proxy-Based Iterative Optimization Method

As mentioned before, our objective is to perform accurate and efficient production optimization by using an LS-SVR- or GPR-based proxy for production optimization problems. The basic idea is to build a proxy that acts as a 'curve fit' to the available data (NPV as a function of design variables) so that the results may be predicted without recourse to the use of the expensive compositional reservoir simulator. Our premise is that once built, the proxy will be many orders of magnitude faster than the simulator to perform this optimization. To this end, in this work, we follow the so-called iterative-sampling-refinement optimization method proposed by Guo and Reynolds (2018). Similar and more advanced proxy-based iterative optimization procedures have also been used in other engineering applications; for example, see, Queipo et al. (2005) and Forrester and Keane (2009).

To create a sample set, first, we sample the design variables and then run our simulator with these design variables to compute NPV. In the end, $\{J_i, \mathbf{u_i}\}_{i=1:N_s}$ is our sample set, where N_s is the number of sample points. As mentioned in previous section we use *LHS* as data sampling method (McKay et al., 2000). It is important to provide some details on how we choose the size of the sample set that is split into training and test sets using *uniform data split*, and the initial proxy model to be used in the iterative-sampling-refinement optimization method (see **Fig. 4.2**). As mentioned earlier, we need an initial ML-based (either LS-SVR or GPR in our study) proxy model to be used in the iterative-sampling-refinement optimization method. Our purpose here is not to generate an initial proxy model that predicts with high accuracy the value of NPV output from the reservoir simulator for any input vector of design variables. Instead, we want the prediction by the proxy to be most accurate in the region of the optimum. The iterative-sampling-refinement method enhances the accuracy of an LS-SVR or GPR-based proxy, as a predictor of the NPV generated from the reservoir simulator, in the region of the optimum by repeatedly adding training samples in addition to the initial samples. Hence, all we need is to construct an initial proxy that has an "appropriate degree of accuracy" to start the iterative-sampling-refinement method. We usually start with a sample set consisting of a training set having a size of one or two times the number of design

variables and a test set having a size approximately equal to half of the size of the training set. Using the training set, we construct an LS-SVR or GPR proxy (model). After training, we compute the accuracy of the predictor using the test set. As for accuracy measures, we consider mean-absolute-error (MAE), root-mean-square-error ($RMSE$), and the coefficient of determination (R^2). We then choose the proxy which gives an MAE value less than equal to 0.1 and R^2 to be equal or greater than 0.8 as the initial proxy to be used in the iterative-sampling-refinement optimization method unless otherwise is stated. Since we applied this optimization method at first to the production optimization problem of HnP, we choose our accuracy measures tight. However, for the WAG and well shutoff problems, we try looser accuracy measures than that for the HnP problem. If given training and test sets do not provide a proxy satisfying these measures, we increase the size of the sample by a factor of 1.5 and continue with training and testing until we obtain a proxy that yields $MAE \leq 0.01$ and $R^2 \geq 0.8$ (for HnP problem only). Although we have used this approach, it is worth mentioning that there are other approaches to construct a proxy that has an appropriate degree of accuracy to be used in the iterative-sampling-refinement process (see Forrester et al. 2008 and James et al. 2013).

In iterative-sampling-refinement optimization, we start with a reasonably accurate, the initial LS-SVR or GPR-based proxy model constructed by using (training) data generated from a high-fidelity compositional simulator (CMG-GEM (2016) in our applications) for the HnP and WAG problems and black-oil simulator (CMG-IMEX (2020) in our applications) for the well shutoff problem. As we mentioned before, this initial proxy needs not be highly accurate so that it can be constructed with a reasonably small number of data generated from the simulator.

The iterative-sampling-refinement optimization method can be summarized as follows (also see Fig. 4.2 for a flow chart):

1. Construct an initial LS-SVR- or GPR-based proxy model and take $l = 0$, where l denotes the iteration index, and choose an initial guess, $\bar{\mathbf{u}}_{opt}^{l=0}$.

2. Set $l = l + 1$. Using the proxy model, its analytical gradient, and initial guess for normalized design variable perform optimization using the SQP algorithm to find the vector of optimum design variables $\bar{\mathbf{u}}_{opt}^{l+1}$, and the corresponding $\hat{\bar{J}}(\bar{\mathbf{u}}_{opt}^{l+1})$.

3. Unnormalize $\bar{\mathbf{u}}_{opt}^{l+1}$ to $\mathbf{u}_{opt}^{l+1}$ and use this unnormalized vector of the optimum design variable in the simulator and calculate the true normalized NPV from the simulator, $\bar{J}(\mathbf{u}_{opt})$.

4. Check for convergence criteria, defined on NPV predicted from both the simulator and the proxy model by, respectively,

$$\frac{|\bar{J}(\mathbf{u}_{opt}^{l+1}) - \bar{J}(\mathbf{u}_{opt}^{l})|}{max(\bar{J}(\mathbf{u}_{opt}^{l}), 1)} \leq \epsilon_1, \tag{4.35}$$

and

$$\frac{|\hat{\bar{J}}(\bar{\mathbf{u}}_{opt}^{l+1}) - \hat{\bar{J}}(\bar{\mathbf{u}}_{opt}^{l})|}{max(\hat{\bar{J}}(\bar{\mathbf{u}}_{opt}^{l}), 1)} \leq \epsilon_2, \tag{4.36}$$

where l indicates the iteration number. Unless otherwise stated, we set $\epsilon_1 = 0.0001$ and $\epsilon_2 = 0.001$ in our applications given in this research. If both Eqs. 4.35 and 4.36 are achieved, stop and exit for the optimum design variable and corresponding NPV from the simulator.

5. If not converged, we add $\bar{\mathbf{u}}_{opt}^{l+1}$ and the corresponding NPV from the simulator, $\bar{J}(\mathbf{u}_{opt}^{l+1})$, into our existing training set, and update our proxy model by training it with this new training set. Using the last optimum design variable as an initial guess for the next iteration, i.e, $\bar{\mathbf{u}}_{opt}^{l} = \bar{\mathbf{u}}_{opt}^{l+1}$.

6. Repeat steps 2 to 5 until we achieve convergence.

Before closing this section, it is worth noting that to perform ML-based proxy optimization for the life-cycle production optimization problem, we have developed a code that is coupled with the commercial simulator (CMG-GEM (2016) and CMG-IMEX (2020)). This code is used to generate training sets to be used to construct initial LS-SVR and GPR-based

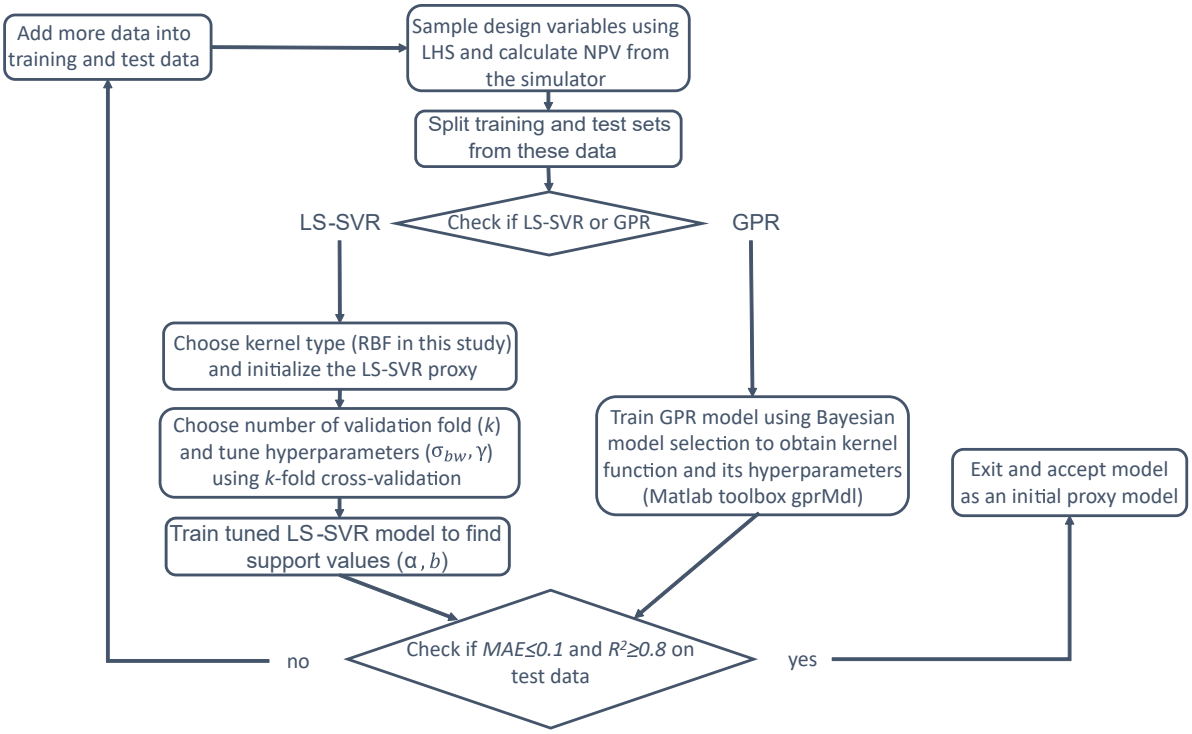


Figure 4.1: Flow chart for constructing the initial LS-SVR- or GPR-based proxy models to be used in the iterative-sampling-refinement optimization method.

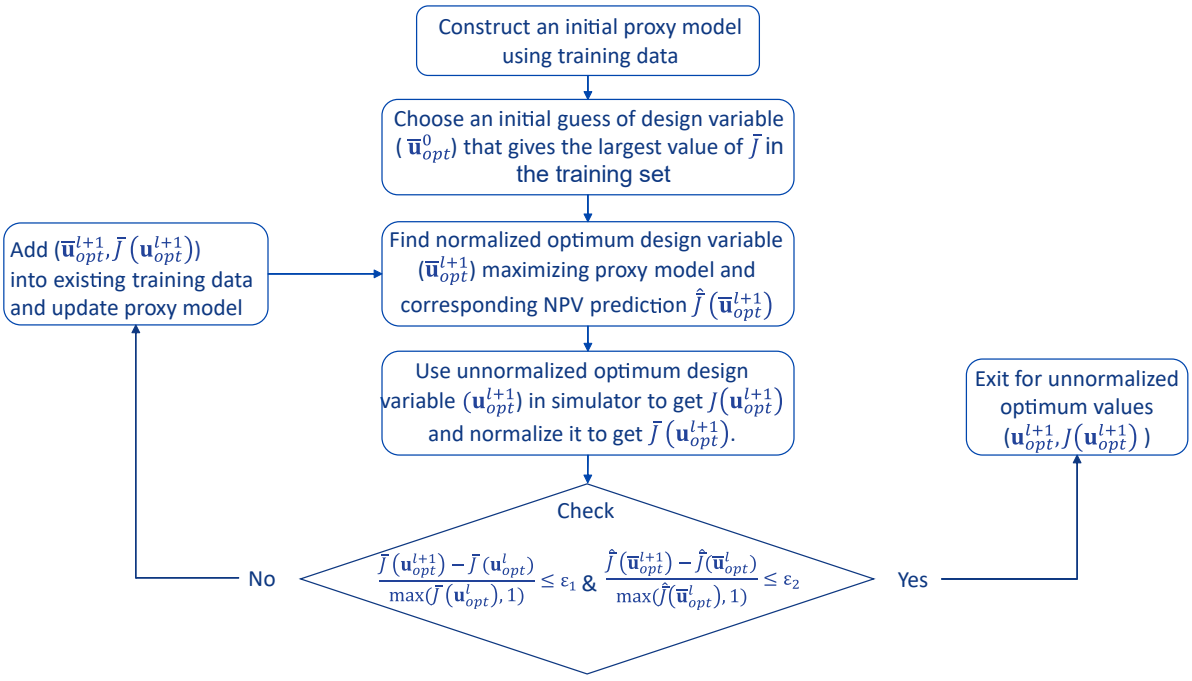


Figure 4.2: Flow chart for the iterative-sampling-refinement optimization method.

proxies and to perform the ML proxy-based iterative-sampling-refinement optimization procedure starting with these initial ML-based proxies constructed. We have used MATLAB programming language to write the codes along with some MATLAB tool packs such as LS-SVMLab v1.7 and gprMdl toolbox and sequential quadratic programming (SQP). As to coupling the MATLAB codes with the commercial simulator, first using a vector of design variables, we generate a SCHEDULE.INC file to be included in our DAT file (DAT file is the file read by the simulator); then we call CMG GEM.exe or IMEX.exe (depending on production optimization problem) file on that DAT file; finally, after a simulation run having finished, we extract the production and injection data into TXT files and read them with MATLAB to calculate our objective function-NPV using the corresponding formulation depending on the production optimization problem (Eqs. 2.5, 2.36, 2.43, 2.57).

CHAPTER 5

PRODUCTION OPTIMIZATION OF THE CO_2 HNP PROBLEM IN UNCONVENTIONAL RESERVOIRS

In this Chapter, first, we talk about the physical process of CO_2 HnP process in unconventional reservoirs. Then we conduct a sensitivity analysis where we check how recovery factor (RF) behaves with respect to design variables we used for optimization as well as some of the important reservoir model parameters. Finally, in the last section, we perform both the approximate gradient and iterative-sampling-refinement optimization methods on different production optimization cases of the HnP problem. For the definition and derivation of the NPV objective function of the HnP problem, see Section 2.1.

5.1 Physics of The HnP Process in Unconventional Reservoirs

As mentioned earlier, our first and the main production optimization problem is the HnP process. To gain some insights, first, we investigate the physics of the HnP process in unconventional reservoirs before going to production optimization of it. First, we start giving the general governing mass transport equation of which is applied for every EOR process. Then we discuss the governing mass transport equation of HnP in unconventional reservoirs by constraining the general governing equation by our assumptions.

5.1.1 Mass Transport Mechanism of HnP Process in Unconventional Reservoirs

To be able to write general governing equation we combine *compositional balances* and *phase conservation equations*. Thus, the mass transport in porous media can be written as follows:

$$\frac{\partial W_{ij}}{\partial t} + \nabla \cdot \mathbf{N}_{ij} - R_{ij} - r_{m,ij} \quad \text{for } i = 1, ..., N_C \quad \text{and } j = 1, ..., N_p, \tag{5.1}$$

where W_{ij} is the mass concentration of component i in phase j per unit bulk volume; $\nabla \mathbf{N}_{ij}$ is the rate of transport of component i in phase j per bulk volume; R_{ij} is the rate of mass generation of i in phase j per bulk volume per unit time due to chemical reactions or physical sources (wells); $r_{m,ij}$ is the rate of mass transfer (m stands for mass) of component i from or into phase j due to phase change (vaporization, condensation or sorption). It should be noted that $\sum_{j=1}^{N_p} r_{m,ij} = 0$ because mass cannot accumulate at volumeless phase interphase (Lake et al., 2014). Inserting the flux ($\mathbf{N}_{ij}$), the accumulation (W_{ij}) and the source terms (R_{ij} and r_{minj}), summing Eq. 5.1 over all the phases and knowing the fact that $\sum_{j=1}^{N_p} r_{m,ij} = 0$, for isothermal flow, the overall conservation equation is written as:

$$\begin{aligned} \frac{\partial}{\partial t}\left\{\phi \sum_{j=1}^{N_p} \rho_j S_j \omega_{ij} + (1-\phi)\rho_s \omega_{is}\right\} + \nabla \cdot \left\{\sum_{j=1}^{N_p} (\rho_j \omega_{ij} \mathbf{u}_j - \phi S_j \mathbf{K}_{ij} \cdot \nabla(\rho_j \omega_{ij}))\right\} \\ = \phi \sum_{j=1}^{N_p} S_j r_{ij} + (1-\phi) r_{is}, \quad \text{for } i = 1, ..., N_C, \end{aligned} \tag{5.2}$$

where ω_{ij} mass fraction of component i in the phase j; ω_{ij} is mass fraction of component i in the solid phase; $\mathbf{u}$ is velocity governing convective flux; r_{ij} is the source term for the component i in phase j; r_{is} is the source term for the component i in the solid phase; and S_j is the saturation of phase j. The first term on the left-hand side (LHS) of Eq. 5.2 represents the accumulation term which is the summation of the accumulation of the component i in all phases and the accumulation of the component i in the solid phase s (rock), which is the sorption term (considers adsorption-desorption). The third term on the LHS is the flux term, which is the summation of advection (advective flux, or convective flux) and dispersion (dispersive flux) terms. Dispersion is described by Fick's law (Fick, 1855) in Eq. 5.2. The term at the right-hand side is the source term due to either reaction (chemical or nuclear) or physical source (well), which is the summation of the source of component i in all phases and solid phase. In Eq. 5.2, $\mathbf{K}_{ij}$ is the hydrodynamic dispersion coefficient, which is a second-order tensor, includes both molecular diffusion and mechanical dispersion. Mechanical dispersion is a result of velocity gradient (fluctuations of the velocity ($\mathbf{u_j}$). For

an isotropic, homogeneous permeable medium, two components of $\mathbf{K}_{ij}$ can be written as

$$(K_{xx})_{ij} = \frac{D_{ij}}{\tau} + \frac{\alpha_{lj} u_{xj}^2 + \alpha_{tj}(u_{yj}^2 + u_{zj}^2)}{\phi S_j |\mathbf{u_j}|} \tag{5.3}$$

and

$$(K_{xy})_{ij} = \frac{(\alpha_{lj} - \alpha_{tj}) u_{xj} u_{yj}}{\phi S_j |\mathbf{u_j}|}, \tag{5.4}$$

where l indicates the longitudinal direction to bulk flow, and t is any direction perpendicular to l. D_{ij} is the effective binary diffusion coefficient of component i in phase j (Bird et al., 2007), α_{lj} and α_{tj} are longitudinal and transverse dispersivity coefficients, and τ is tortuosity. The first term in the RHS of Eq. 5.3 represents dispersion due to molecular diffusion, and the second term in the RHS represents mechanical dispersion which happens due to the velocity gradient. Off diagonal elements of the dispersion coefficient matrix are only for mechanical dispersion.

However, in a nanoscale medium, molecular diffusion becomes an important mass transport mechanism. Many experimental and simulation studies agree upon the importance of molecular diffusion in HnP process; for instance, see; Alfarge et al. (2017); Yu et al. (2014); Jia et al. (2017); Kanfar et al. (2017). In the case of ultra-tight shale-oil formations, the advective term in Eq. 5.2 is not important, and mechanical dispersion also disappears from Eqs. 5.3 and 5.4, and thus, diffusion becomes the main mass transport mechanism. Therefore, ignoring convective flux term from, and ignoring source terms as an additional assumption (since they are negligible in EOR processes we considered in our research; however, they are important in the EOR processes that require chemical reaction or physical source such as polymer injection), Eq. 5.2 can be written with flux due to only molecular diffusion as

$$\frac{\partial}{\partial t}\left\{\phi \sum_{j=1}^{N_p} \rho_j S_j \omega_{ij} + (1-\phi)\rho_s \omega_{is}\right\} + \nabla \cdot \left\{\sum_{j=1}^{N_p} (-\phi S_j \mathbf{K}_{ij} \cdot \nabla(\rho_j \omega_{ij})\right\} = 0, \quad \text{for } i = 1, ..., N_C. \tag{5.5}$$

For the HnP process, we need to achieve miscibility to achieve a higher RF. First,

it is important to note that the ability of a solute to be solved inside solvent is called *solubility*. However, for a liquid-liquid solution, if they are mixed at all proportions (there is no saturation level to reach), they are called *miscible* liquids. Two gases can be miscible as well. As explained above the interactions of different molecules or ions may be energetically preferred or not. If the interaction is not favorable, the solution will end at the point called *saturation* point.

Since we have only one phase in miscible condition, considering $N_p = 1$ (in our case it is oil phase) in Eq. 5.5, we write the equation of mass transport for the HnP process in unconventional reservoirs as

$$\frac{\partial(\phi\rho\omega_i)}{\partial t} + \frac{\partial}{\partial t}\left[(1-\phi)\rho_s\omega_{is}\right] + \nabla\cdot\left[-\phi\mathbf{K}_i\nabla(\rho\omega_i)\right] = 0, \tag{5.6}$$

where the dispersion coefficient tensor $\mathbf{K}_i$ consists of only molecular diffusion coefficient tensor since we ignore velocity terms in Eqs. 5.3 and 5.4 and considering only one phase ($j = 1$). Doing so off-diagonal elements of dispersion coefficients become zero, and diagonal elements are given as follows:

$$(K_l)_i = \frac{(D_l)_i}{\tau} \tag{5.7}$$

Usually, the diffusion tensor is assumed to be isotropic and diagonal ($\mathbf{D}_i = (D_l)_i\mathbf{I}$, where $\mathbf{I}$ is the identity matrix of appropriate size). GEM (2016) models only isotropic diffusion, but it can model anisotropic dispersion.

5.1.2 Phase Equilibrium

In Subsection 5.1.1 governing physics (mass transport mechanism) is generally formulated. To be able to formulate ω_{ij} values in Eqs. 5.1, 5.2 and 5.5, we need to know phase behaviour. We compute those values using phase equilibrium. At the molecular level, we have interaction potential between two molecules besides the thermal movement of each molecule. The interaction force between molecules can be related to this pair's potential energy, and the maximum work that can be done by this work is called free energy or available

energy. This energy is related to *chemical potential*, which is related which is later related with fugacity. For further details regarding intermolecular forces one can check the books by Israelachvili (2011) and Job and Herrmann (2006)

Through flash calculation, a relation can be found between partial fugacity (which is the function of fugacity) and state variables, but we need an equation that relates all state variables, which is the equation of state (EOS). According to Zhu and Reitz (2002), generally cubic EOS can be written as Eq. 5.8.

$$P = \frac{RT}{V-b} - \frac{a}{V^2 + qbV + wb^2}, \tag{5.8}$$

where V, P are volume and pressure of the system, respectively, and R is the universal gas constant, q and w are just constant which depend on the type of EOS used. The most used and experimentally well-matched model is the Peng-Robinson (PR-EOS) model (Peng and Robinson, 1976). For that model $q = 2$, $w = -1$. In our research, we also use the PR-EOS model in WINPROP (2004), which is a phase package of CMG, to calculate PVT properties of reservoir fluid and injected fluid, which is CO_2.

To be able to compute concentration values ω_{ij}, we need to solve equilibrium, and this leads us to chemical potential, to fugacity, eventually to EOS. The solution method is iterative and can be found in many books related to *vapor-liquid equilibrium*. The solution of the compositions of phases allows us to establish phase diagrams for a given mixture. For a three-component system, we use the ternary diagram and for a four-component system tertiary diagram. Actually, when we talk about 2, 3, 4 components they are pseudo-components, since we cannot visualize more than 4 components. The properties of pseudo components can be found by experiments. For example, C_{7+} is a pseudo component.

These diagrams can be used to establish tie-lines, which can be used to find: if given reservoir fluid and injected fluid achieve miscibility or not; if achieves miscibility what is the type of miscibility achieved; if it is *first-contact miscibility (FCM)* or *multi-contact miscibility (MCM)*, vaporized or condensed miscibility or mixed. These can be found using different

methods, but the most famous one is the *key-tie line method*. The method of characteristics is used in the semi-analytical (key-tie line) method as described by Orr et al. (2007).

In FCM, when injected solvent mixes with all proportions with reservoir fluid, a single-phase is formed. FCM can be achieved at very high pressure. A ternary diagram can be used to illustrate FCM (Fig. 5.1). If the line connecting reservoir fluid point and injected fluid in the ternary diagram does not intersect the two-phase region of the phase diagram, that means injected fluid and reservoir fluid will make a single phase, and thus displacement will consist of single-phase that changes in composition from reservoir fluid to the injected fluid through this line. Fig 5.2 illustrates visually how miscible and immiscible flooding occurs (Kantzas et al., 2012).

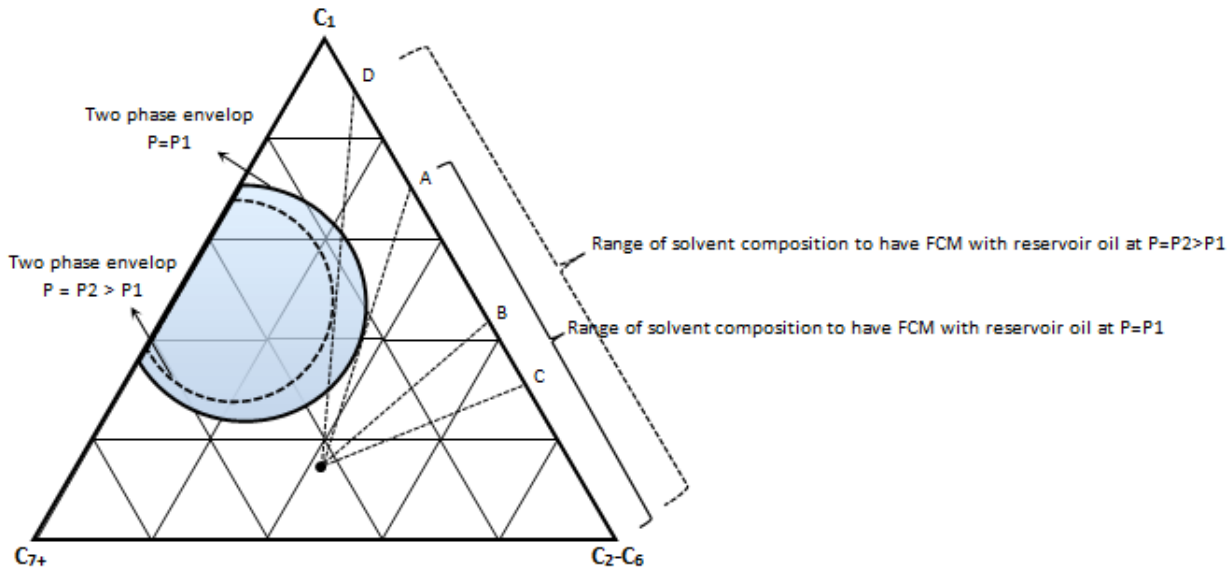


Figure 5.1: First-Contact Miscibility (Kantzas et al., 2012).

However, in MCM, also known as *dynamic miscibility*, there will be a composition alteration between phases (phase that contains all proportions of injected fluid with reservoir fluid, and the phase that is in the other phase than miscible phase). It achieves miscibility after multiple contacts during which interphase mass transfer of components between reservoir fluid and injected fluid will occur. It will result in a mixture that is miscible with either injected fluid or initial reservoir fluid. Therefore this process is categorized as *vaporizing gas drive, condensing, condensing/vaporizing-gas (enriched gas)* and CO_2 *displacement*. To be

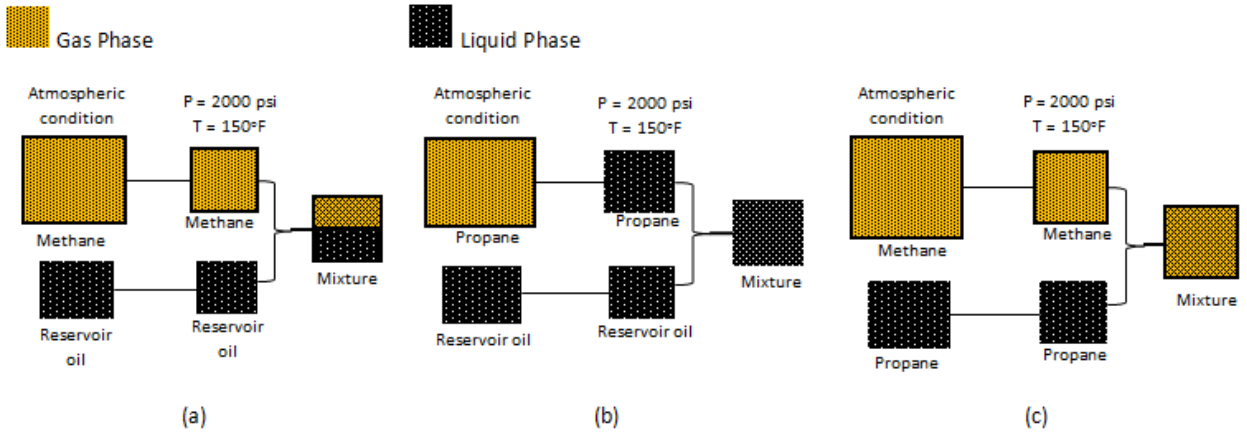


Figure 5.2: Miscible and Immiscible process illustration (Kantzas et al., 2012)

able to understand how MCM works, we can look at only vaporizing gas drive mechanism (Fig 5.3). At the beginning injected fluid S contacts with reservoir fluid O and creates mix-

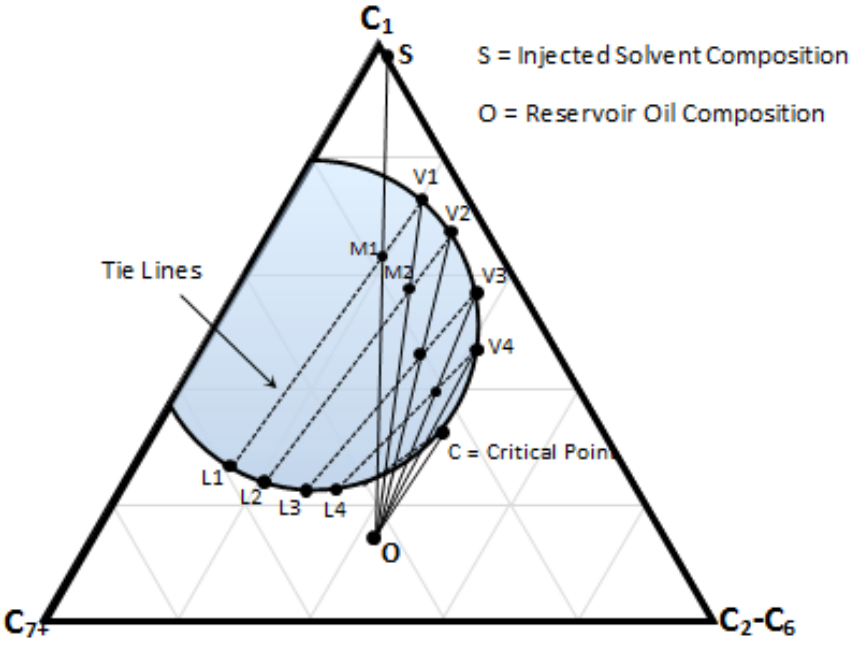


Figure 5.3: Vaporizing gas drive mechanism (Kantzas et al., 2012).

ture M_1. With the tie line method, we can see that this mixture contains gas V_1 and liquid L_1. This vapor is an enriched solvent. Then, this gas phase V_1 moves faster than the liquid phase and contacts with reservoir fluid, and creates a mixture M_2 which contains vapor of enriched solvent (which is enriched more than V_1) V_2 and liquid phase L_2. And this process goes on until when this mixture line will not pass through a two-phase envelope. And

eventually, a continuous transition zone will exist, gas compositions vary from the injected gas to reaching the critical point composition. So miscibility is achieved at the front of the injected gas. (Kantzas et al., 2012).

WINPROP (2004) can be used to calculate MMP or MME at a given pressure, temperature, oil composition, primary and makeup gas compositions. A user can choose any method of the following methods: the cell-to-cell simulation method which uses a gridded ternary diagram, therefore, it can be used only for vaporizing or condensing drive mechanisms; the tie-line method; or the multiple-cell simulation method. The last two methods can be used in vaporizing/condensing mechanisms. In our research, we use the tie-line method.

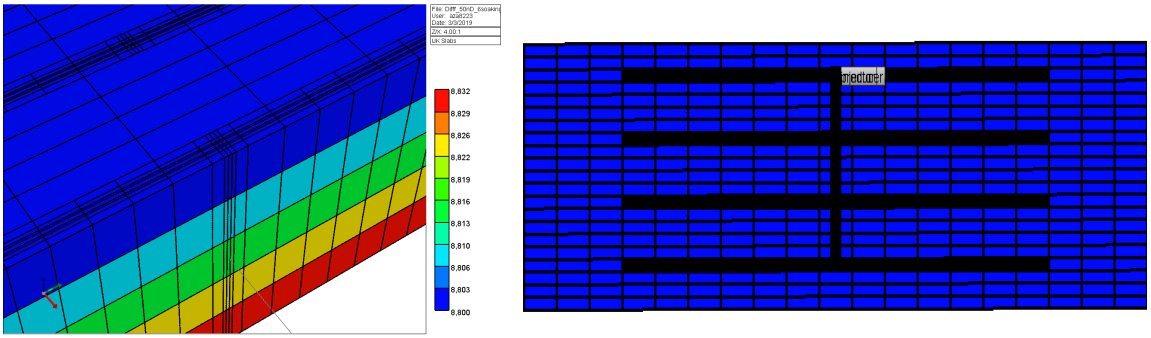
5.2 Sensitivity Study on Design Variables And on Important Reservoir Parameters

Before presenting results of production optimization, we conduct a sensitivity study to investigate the effects of various flow and transport mechanisms and well-control parameters on a miscible HnP process in a multi-fractured horizontal well in an unconventional oil system. For this purpose, we use GEM (2016) which is a general-purpose compositional simulator. It can model many physical phenomena such as geomechanical and Klinkenberg effects, molecular diffusion, gas phase diffusion between natural fractures and matrix, adsorption-desorption, dual-permeability, dual-porosity, etc. Using WINPROP (2004), we model compositional variations, characterize fluid properties, and compute MCMMP and FCMMP (first contact minimum miscibility pressure). After modeling fluid compositions and properties, we use this fluid model as input to GEM (2016). A hydraulic fracture can be modeled by using a local grid refinement. GEM (2016) provides both planar and complex fracture templates, both of which use the LS-LR (logarithmically spaced local grid refinement) method. Natural fractures with "dual-permeability" and "dual-porosity" options can be modeled. Dispersion can also be modeled in GEM (2016). Non-Darcy flow can be modeled for all phases as well.

Another purpose of pursuing this sensitivity study is to be sure how to get stable

model results in GEM (2016) since compositional models can be unstable, especially during the gas injection process due to wild change of pressure, saturation, and concentrations of components, and, therefore to understand numerical simulation by GEM is very important and has to be modified from the default. GEM (2016) is used to build a model, and to calculate composition properties, flash calculations, MCMMP etc., we used WINPROP (2004).

For the sensitivity study, we use the basic fluid and reservoir property data from (Yu et al., 2014). We consider four fracture stages along a horizontal well, where each fracture stage has only one fracture. Geometry of the well/reservoir system is shown in Fig 5.4. The oil composition is given Table 5.1. Yu et al. (2014) states that the oil composition is pertinent to a Bakken shale-oil play.



(a) 3D grid - cross-sectional view **(b)** 2D grid - areal view

Figure 5.4: Reservoir model and grid system used for sensitivity study.

Table 5.2 shows binary interaction coefficients (BICs) between components of reservoir fluid. BICs considers non-ideality of mixture.

Synthetic reservoir parameters are given in Table 5.3. Unless otherwise stated, the reservoir parameters given in Table 5.3 are for our base case. The reservoir is at the depth of 8800 ft. Water-oil contact (WOC) at 8850 ft is specified to initialize the reservoir simulator. Using parameters of the components in WINPROP (2004), important fluid properties are calculated as well as MCMMP and FCMMP. Phase envelope is given in Fig 5.5. Liquid-gas relative permeability curves are given in Fig 5.6. In applications to be given here, hysteresis effects in relative permeability curves and the effect of capillary pressure are ignored.

Table 5.1: Properties of pseudo-components of Bakken oil, Yu et al. (2014).

Component	Molar fraction	Critical pressure (atm)	Critical temperature (K)	Critical volume (L/mol)	Molar weight (g/gmol)	Acentric factor	Parachor coefficient
CO_2	0.0001	72.8	304.2	0.094	44.01	0.225	78
$N_2 - C_1$	0.2203	45.24	189.67	0.0989	16.21	0.0084	76.5
C_-C_4	0.2063	43.49	412.47	0.2039	44.79	0.1481	150.5
$C_5 - C_7$	0.1170	37.69	556.92	0.3324	83.46	0.2486	248.5
$C_8 - C_{12}$	0.2815	31.04	667.52	0.4559	120.52	0.3279	344.9
$C_{13} - C_{19}$	0.0940	19.29	673.76	0.7649	220.54	0.5672	570.1
$C_{20} - C_{30}$	0.0808	15.38	792.4	1.2521	321.52	0.9422	905.7

Table 5.2: Binary interaction parameters for Bakken oil, Yu et al. (2014).

Component	CO_2	$N_2 - C_1$	$C_1 - C_4$	$C_5 - C_7$	$C_8 - C_{12}$	$C_{13} - C_{19}$	$C_{20} - C_{30}$
CO_2	0	0.1013	0.1317	0.1421	0.1501	0.1502	0.1503
$N_2 - C_1$	0.1013	0	0.0130	0.0358	0.0561	0.0976	0.1449
C_-C_4	0.1317	0.0130	0	0.059	0.0160	0.0424	0.0779
$C_5 - C_7$	0.1421	0.0358	0.059	0	0.0025	0.0172	0.0427
$C_8 - C_{12}$	0.1501	0.0561	0.0160	0.0025	0	0.067	0.0251
$C_{13} - C_{19}$	0.1502	0.0976	0.0424	0.0172	0.067	0	0.061
$C_{20} - C_{30}$	0.1503	0.1449	0.0779	0.0427	0.0251	0.061	0

Sensitivity analysis is done for changing the values of various physical and well-control parameters (Table 5.4). Unless otherwise stated, all applications considered matrix permeabilities of 50, 500, and 5000 nD. All cases repeated with and without molecular diffusion of CO_2 in an oil phase.

Above 88 °F and 1070 psi, CO_2 is at a supercritical condition. At a supercritical condition, CO_2 acts like a liquid (its density is close to the liquid density), but its viscosity still is very low like a gas. In our sensitivity analysis, the BHP is not lower than 2000 psi. Therefore, CO_2 is always at a supercritical condition. GEM (2016) considers supercritical CO_2 as a gas. In GEM (2016), we identify a single-phase fluid in each grid as oil or gas using different methods with keyword PHASEID under the *composition* section. PHASEID is an important parameter to be input by a user in GEM (2016) to calculate phase properties at

Table 5.3: Reservoir parameters for CO_2 HnP process.

Parameter	Value	Unit
The model dimensions	340x1300x40	ft
Initial reservoir pressure	8000	psi
Production time	30	years
Reservoir temperature	240	°F
Initial water saturation	0.25	fraction
Total compressibility	1e-6	1/psi
Matrix permeability	50	nD
Matrix porosity	0.08	fraction
Stage spacing	70	ft
Fracture conductivity	50	mD-ft
Fracture half-length	350	ft
Fracture height	40	ft
FCMMP	10,000	psi
MCMMP	4,875	psi

each gridblock, especially when molecular diffusion is considered. In our model, the critical temperature mixing rule (TCMIX) is used as the PHASEID method. According to this mixing rule, if block temperature is greater or equal to the pseudo-critical temperature, the phase is assumed to be gas, otherwise, it is the liquid phase. Fig. 5.7 a, b, and c show distribution of viscosity, whereas Fig. 5.7 d, e, and f show the distributions of the gas phase density during the injection period of the first cycle of the HnP process. The values of viscosity indicate that under a supercritical state, the viscosity of CO_2 would be very close to a typical gas viscosity. However, the density of CO_2 is very close to the density of oil. In Fig 5.8, the reservoir is half cut from the right side and from the top to be able to see the properties near the fracture. When you look at the behavior of gas saturation during each period at one cycle (first cycle here) (Fig 5.8), you can see that in the 6^{th} day of injection period you have high gas saturation and as we inject at the midtime of injection we have higher gas saturation, and at the end of the injection, you will see highest gas saturation in fracture. This may seem unreasonable because, at that pressure, CO_2 is not in the gas phase, but it seems that simulator takes *supercritical* CO_2 as gas. However, in the 2^{nd} month

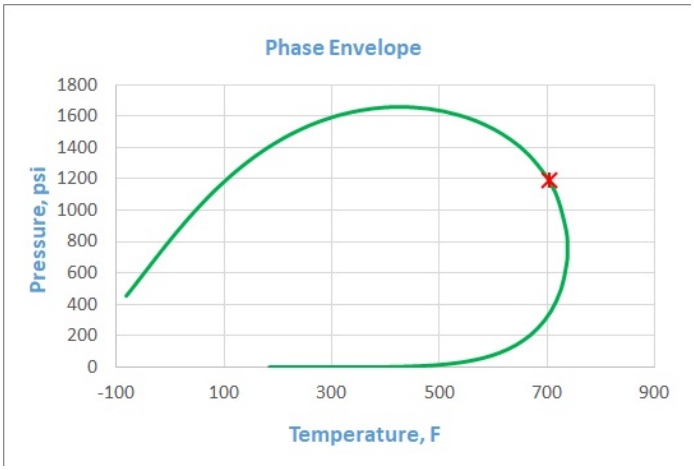


Figure 5.5: Phase envelope of reservoir fluid.

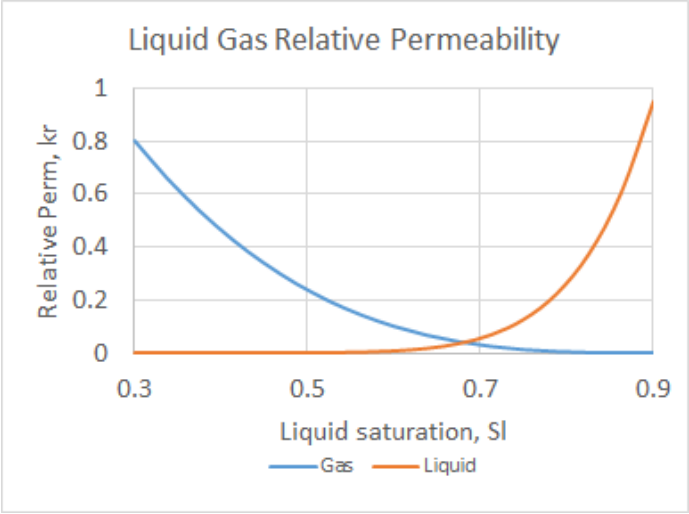
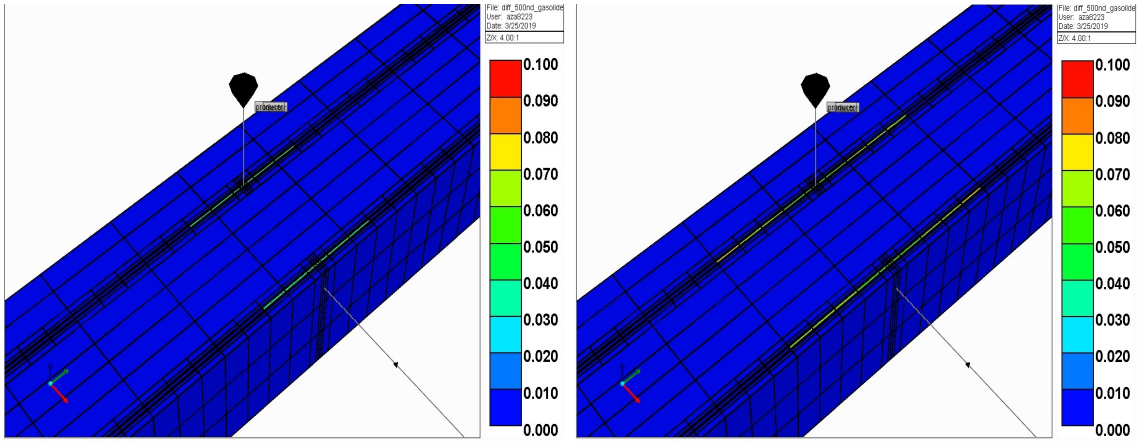


Figure 5.6: Liquid-gas relative permeability.

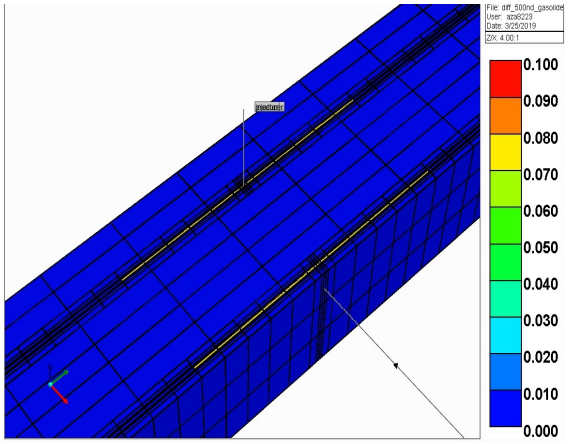
of the soaking period there is almost zero gas saturation.

So, if supercritical CO_2 is considered as the gas phase in GEM (2016), why there is almost zero gas saturation in the fracture in the midtime of the soaking period? This could be possible if there is negligible CO_2 concentration in supercritical CO_2 phase, that is, it is diffused in the oil phase due to molecular diffusion of CO_2 in the oil phase. To be sure about it, the same case was run but ignoring the molecular diffusion of CO_2 (Fig. 5.9). In Fig 5.9, there is always high gas saturation even at the end of the soaking period. It is because of the same reason again-since we do not have a diffusion process, CO_2 components will not diffuse into the oil phase from its supercritical state. Therefore, its concentration will build up in

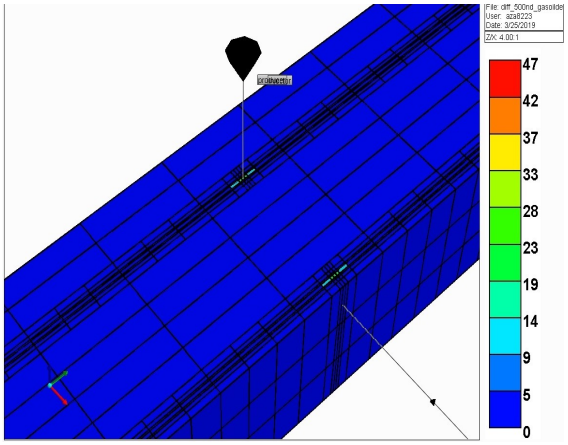


(a) 6^{th} day of injection

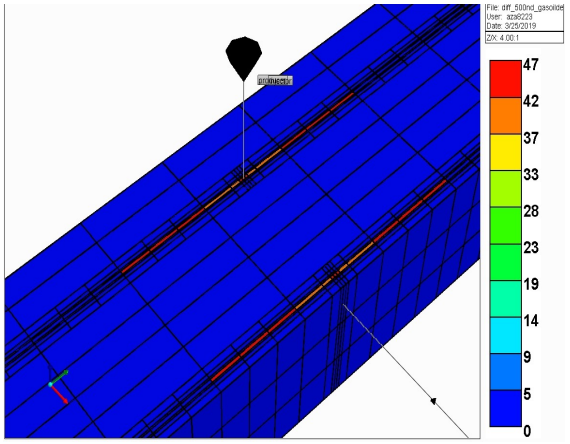
(b) 94^{th} day of injection



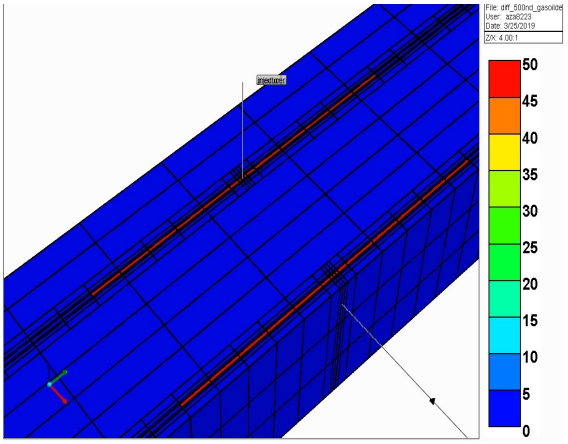
(c) 180^{th} day of injection



(d) 6^{th} day of injection



(e) 94^{th} day of injection



(f) 180^{th} day of injection

Figure 5.7: Viscosity [(a), (b), (c)] and density [(d), (e), (f)] of gas phase during injection at 500 nD permeability considering molecular diffusion of CO_2 in oil phase. Production BHP=2000 psi, $q_{CO_2,i}$=100 MSCF/D, 6-month injection, 3-month soaking, 6-month production.

Table 5.4: Cases considered for sensitivity study for HnP process.

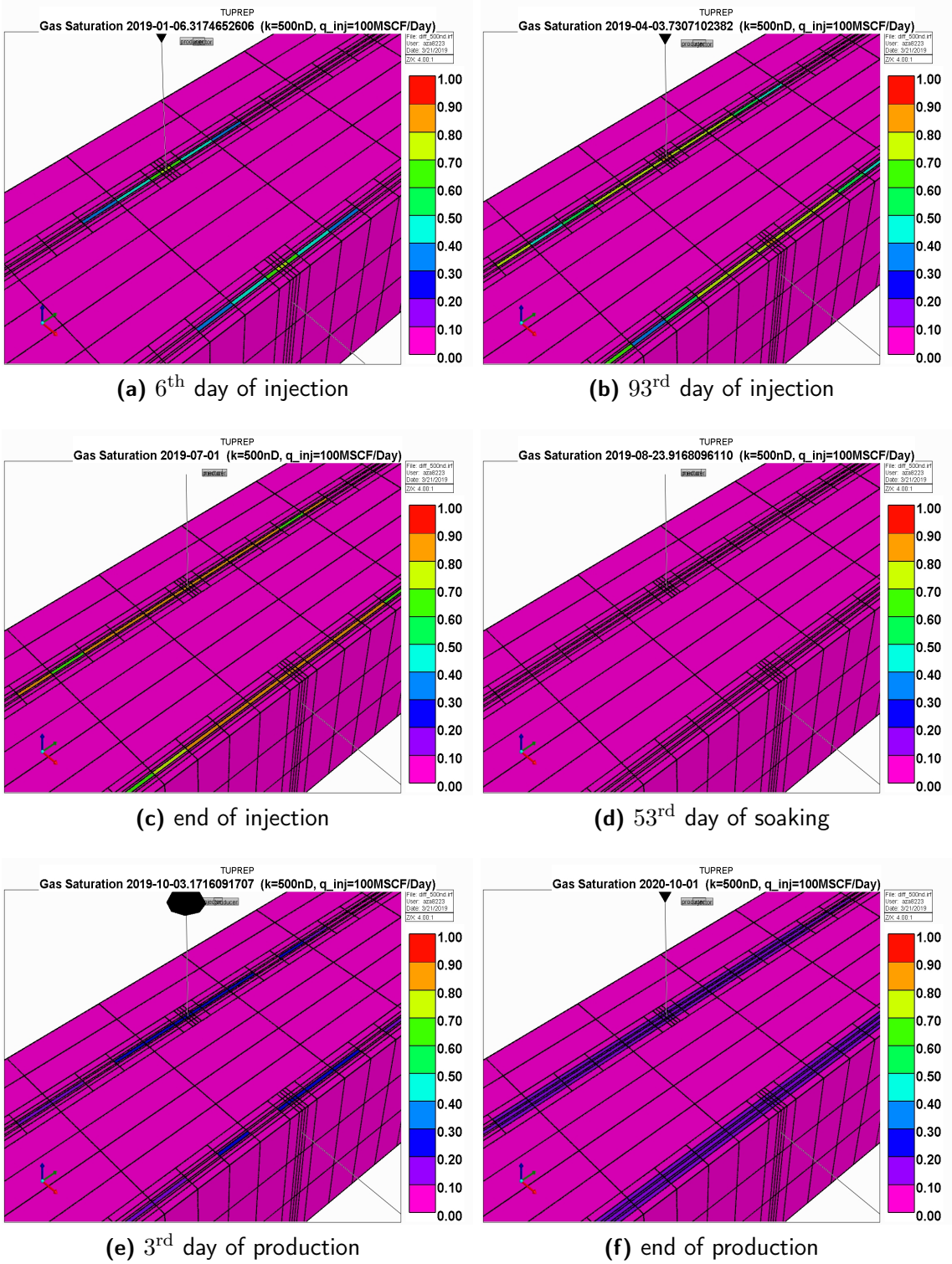
Parameters	Cases		
Injection rate, MSCF/Day	50	100	150
Production BHP, psi	1500	2000	2500
Injection time at each cycle, month	3	6	9
Soaking time at each cycle, month	0	3	6
Production time at each cycle, month	12	12	12
Total production time, month	312	312	312
Molecular diffusion of CO_2 in oil phase	on/off	on/off	on/off
Dual-porosity	off	on/off	off
Dual-permeability	off	on/off	off
Klinkenberg effect	off	on/off	off
Total number of cycles	2	2	2

the fracture, and thus, it will give the highest gas saturation at the end of the soaking period. In both non-diffusion and diffusion cases, the behavior of gas saturation in the production period is approximately the same. Gas saturation decreases because the oil phase, which is liquid at that condition, will go to the fracture. At that pressure ($p_{bh} = 2000$ psi) CO_2 will go back to supercritical CO_2 phase.

The important conclusion from the results of gas saturation, investigated above, is that supercritical CO_2 is handled by GEM (2016) as the gas phase though its density is nearly the same as the oil phase. In GEM (2016), single-phase fluid in each grid as oil or gases is identified by using different methods with a keyword called PHASEID under the composition section. In GEM (2016) user's guide at page 489, it is mentioned that according to Gosset et al. (1986) phase identities will be determined depending on fluid properties using the following criteria using CRIT keyword in front of PHASEID keyword under component section in GEM (2016):

a At supercritical conditions, named gas

b At subcritical conditions, named oil when its molar volume $<$ critical molar volume. To avoid expensive critical point calculations, the EOS critical properties for the mixture computed from mixing rules are assumed to be the true critical values.



(a) 6^{th} day of injection

(b) 93^{rd} day of injection

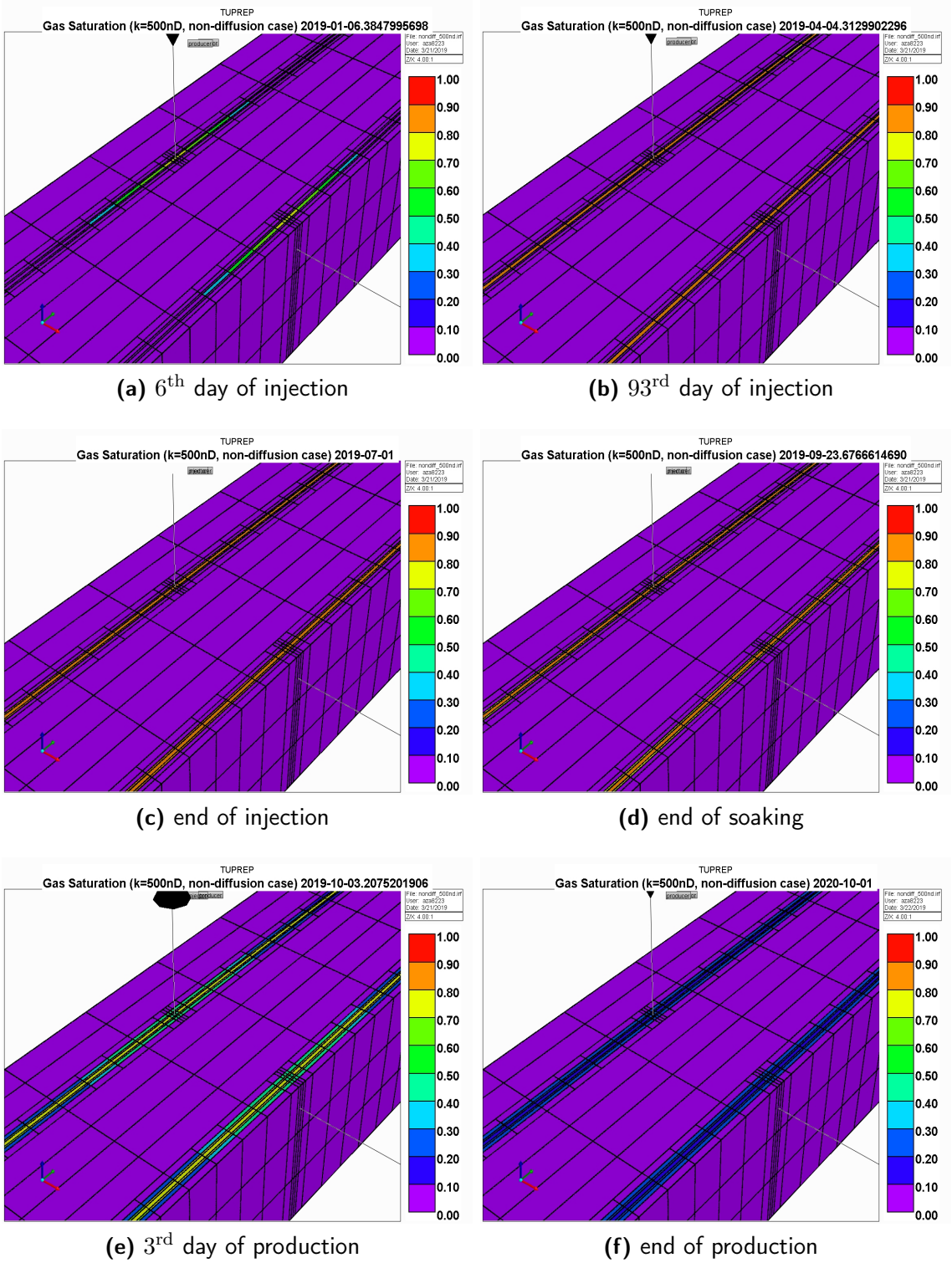
(c) end of injection

(d) 53^{rd} day of soaking

(e) 3^{rd} day of production

(f) end of production

Figure 5.8: Gas saturation at different times at 500 nD permeability considering molecular diffusion of CO_2 in oil phase.



(a) 6^{th} day of injection

(b) 93^{rd} day of injection

(c) end of injection

(d) end of soaking

(e) 3^{rd} day of production

(f) end of production

Figure 5.9: Gas saturation at different times at 500 nD permeability not considering molecular diffusion of CO_2 in oil phase.

PHASEID is an important parameter that must be inputted by the user in GEM (2016) to be able to calculate phase properties at each gridblock, especially when diffusion is considered. However, in all cases in this research, TCMIX (critical temperature mixing rule) is used as the PHASEID method. According to this mixing rule, if block temperature is greater or equal to the pseudo-critical temperature, the phase is assumed to be gas, otherwise, it is the oil phase. Fig 5.4 shows that reservoir fluid is always in the liquid state at that temperature and pressure. For CO_2 240 ^oF is above critical temperature. Therefore, in our results, we see supercritical CO_2 as the gas phase.

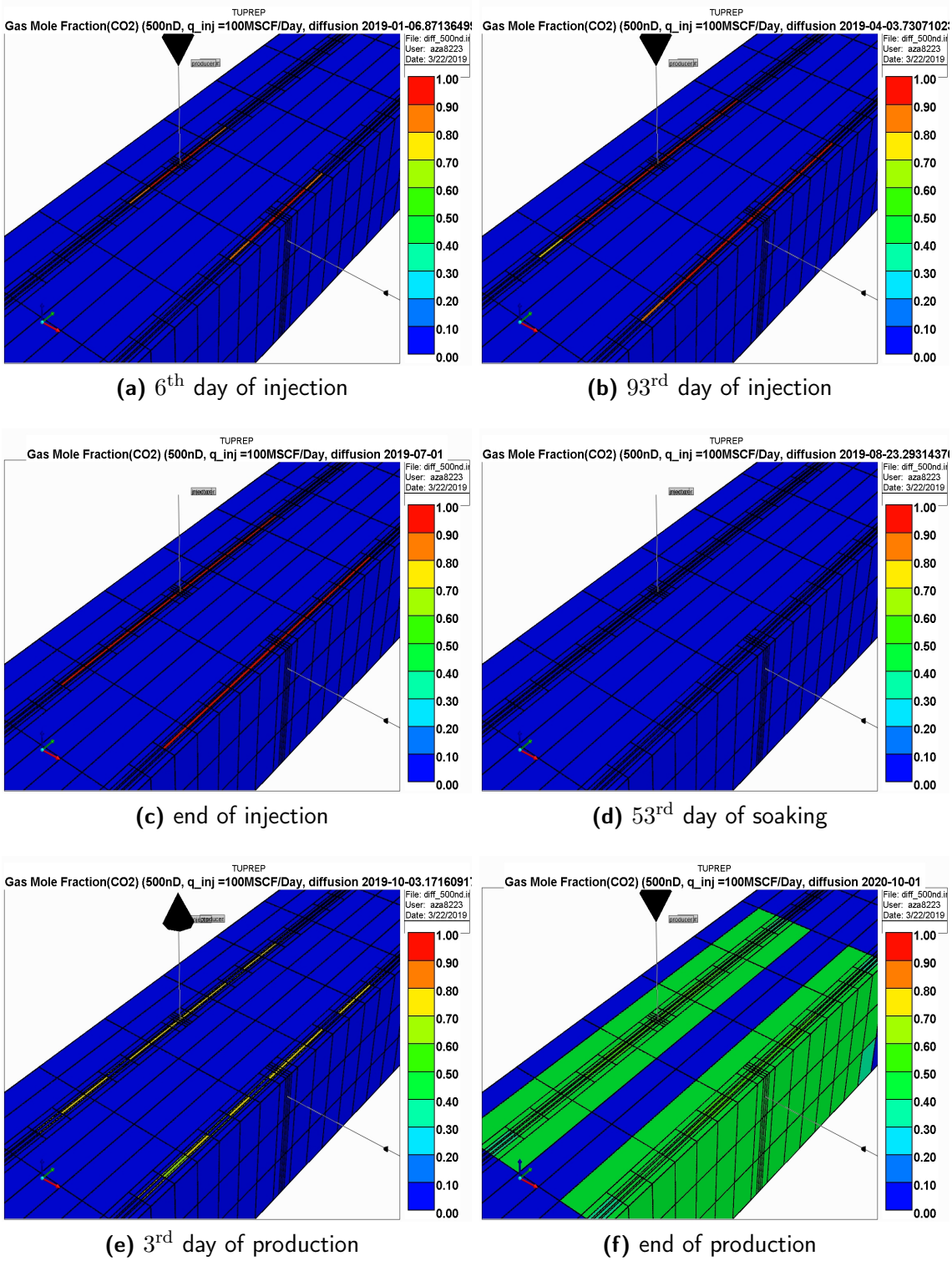
Fig 5.10 presents gas mole fraction of CO_2 component at different times considering molecular diffusion. At the end of production, we see that there is more than 50% gas mole fraction of CO_2 even in the matrix. During injection, CO_2 builds up in the fracture system in a supercritical state (very little amount of it diffuses inside the oil phase in the matrix). In the soaking period, CO_2 components start diffusing into the oil phase into the fracture, therefore, the gas mole fraction of CO_2 becomes zero in the middle of the soaking period. During production, particularly at the end of the production, we see a high gas mole fraction of CO_2 in the matrix because at a lower pressure than MCMMP CO_2 will separate from CO_2-oil mixture. Since at that pressure CO_2 still is at its supercritical state, the simulator will see those CO_2 components in the gas phase. From Fig. 5.10 it can be observed that CO_2 is not the only component in the gas phase (otherwise gas mole fraction of CO_2 would be 1). It is because at low pressure lighter components of reservoir fluid would be near miscible inside supercritical CO_2, and, thus enrich it. However, if we check the same case not considering molecular diffusion (Fig 5.11), we do not see that gas mole fraction will go zero in fracture even at the end of the soaking period, and at the end of the production period of the first cycle, the gas mole fraction of CO_2 will be zero inside the matrix.

To investigate the behavior of oil viscosity, we visualize viscosity at each grid for a reservoir having a matrix permeability of 500 nD, at different times of cycle considering molecular diffusion of CO_2 in the oil phase (Fig. 5.12). Here, we observe that two parameters affect the viscosity of the oil phase simultaneously; namely, pressure at each gridblock and

mixing of CO_2 with oil. At the beginning of the injection, miscibility of oil and CO_2 inside the fracture is achieved, therefore, oil phase viscosity is very low inside the fracture zone near perforation. However, as pressure increases in the gridblocks further from the perforation points and inside the matrix, the oil phase is compressed, and thus viscosity of oil in those gridblocks increases. We can see these effects at the end of the injection period as well. The distribution of viscosity of oil phases depends on the distribution of pressure through gridblocks and how deep CO_2 is diffused inside the oil phase. The further increase of viscosity in some gridblocks in the middle of the soaking period is due to further increase during soaking. Actually, the average reservoir pressure for this case shows its maximum during the soaking period. Fig 5.13 shows how viscosity of reservoir fluid changes with pressure (plot is obtained using WINPROP (2004)). The effect of molecular diffusion proves its effect on viscosity at the end of the soaking period, that is, due to molecular diffusion CO_2 moves from a highly concentrated place, which is the fracture, into its low concentration, which is the oil phase inside the matrix, and therefore, distribution of viscosity around fracture will be even. And, at the end of the production period of the first cycle due to high pressure decrease, and movement of oil from the matrix into fracture will result in evenly distributed low viscosity. If compared, it can be observed that the overall viscosity of the oil phase after one cycle decreases especially around the fracture zone (compare Figs 5.12a and 5.12f).

5.2.1 Effect of Molecular Diffusion in Unconventional Oil Reservoirs

The results of gas mole fraction and gas saturation discussed previously show how molecular diffusion increases the influence area of miscibility. Since it is a low permeability shale oil reservoir, miscibility happens mostly due to molecular diffusion of CO_2 components between supercritical CO_2 state and oil phase. This is because advective flow becomes negligible, and thus mechanical dispersion also becomes negligible. Including molecular diffusion affects bottomhole pressure (Fig 5.14). Since there will be a movement due to molecular diffusion the same amount of injected CO_2 will result in less pressure increase than non-molecular diffusion case.



(a) 6^{th} day of injection

(b) 93^{rd} day of injection

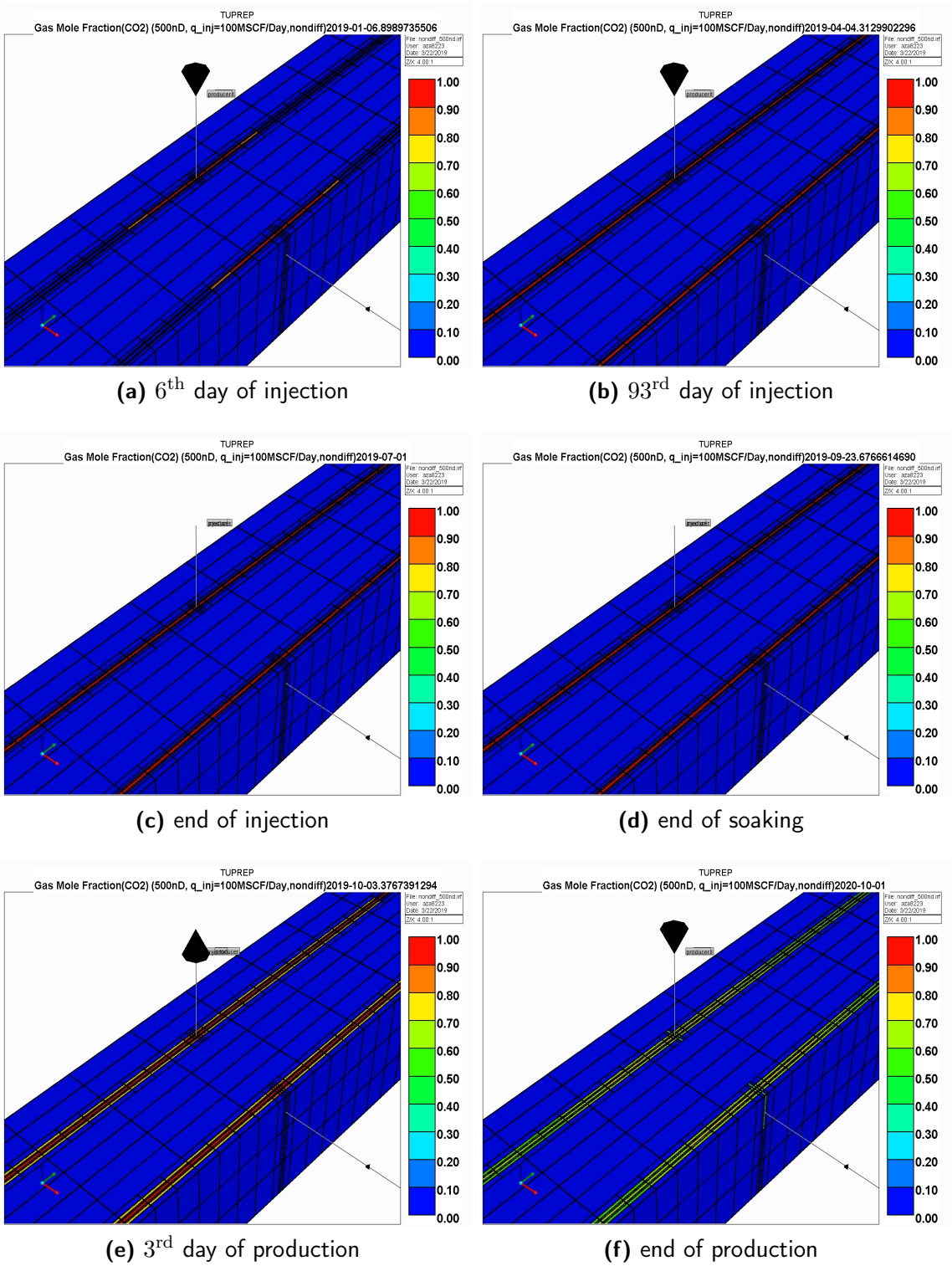
(c) end of injection

(d) 53^{rd} day of soaking

(e) 3^{rd} day of production

(f) end of production

Figure 5.10: Gas mole fraction of CO_2 component at different times at 500 nD permeability considering molecular diffusion of CO_2 in oil phase.



(a) 6^{th} day of injection

(b) 93^{rd} day of injection

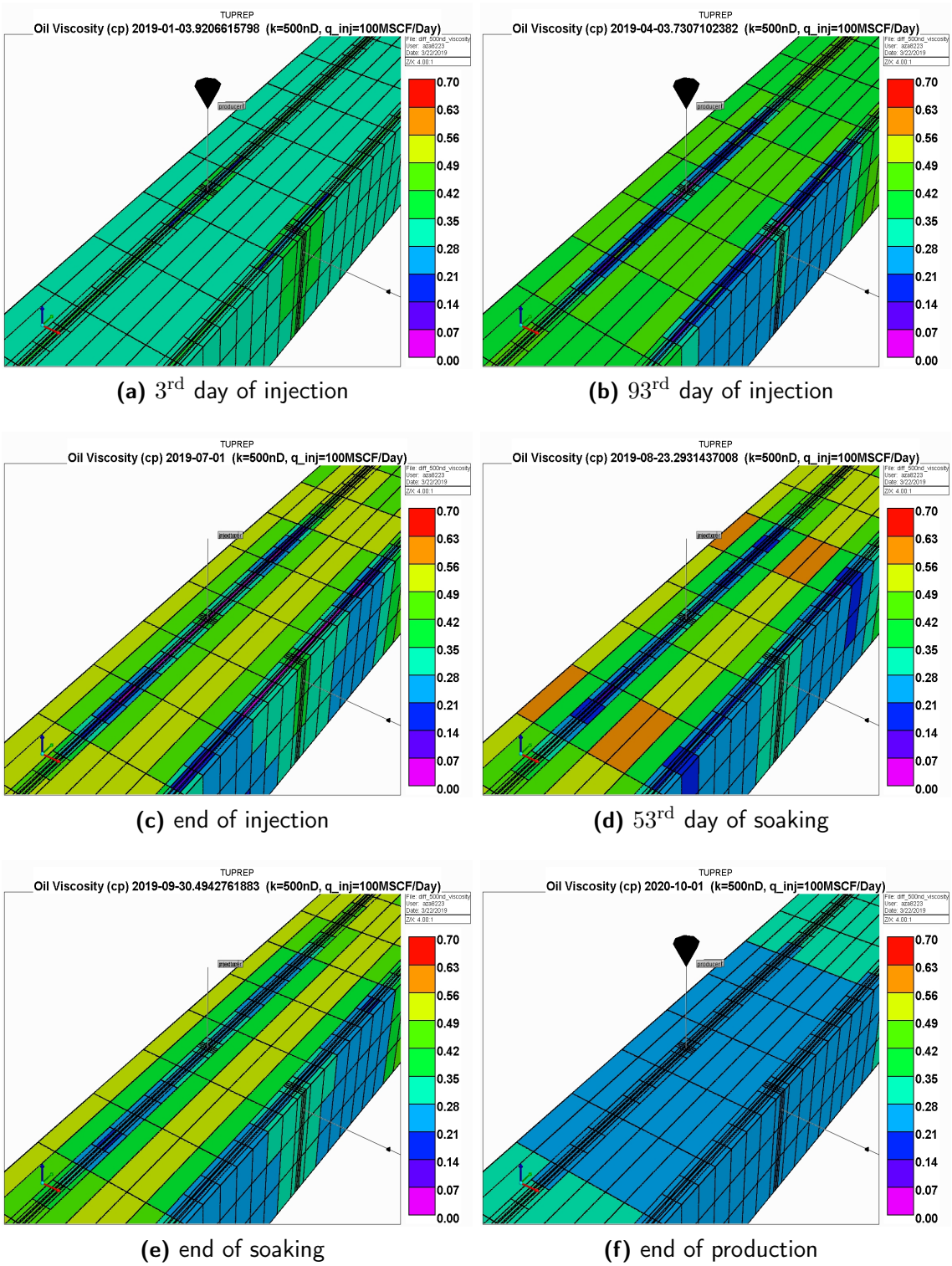
(c) end of injection

(d) end of soaking

(e) 3^{rd} day of production

(f) end of production

Figure 5.11: Gas mole fraction of CO_2 component at different times at 500 nD permeability not considering molecular diffusion of CO_2 in oil phase.



(a) 3^{rd} day of injection

(b) 93^{rd} day of injection

(c) end of injection

(d) 53^{rd} day of soaking

(e) end of soaking

(f) end of production

Figure 5.12: Viscosity of oil phase at different times at 500 nD permeability considering molecular diffusion of CO_2 in oil phase.

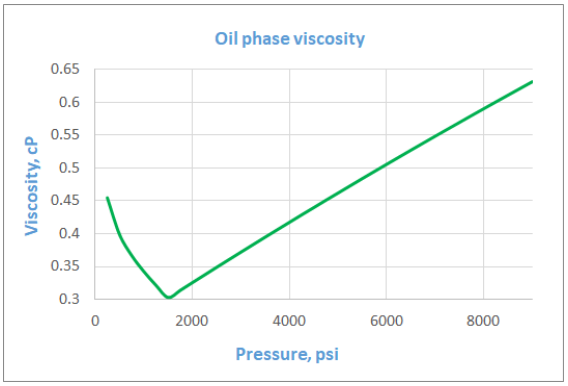


Figure 5.13: Viscosity of oil phase at reservoir temperature.

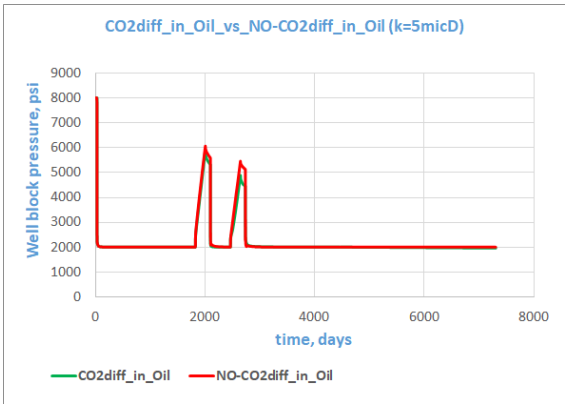
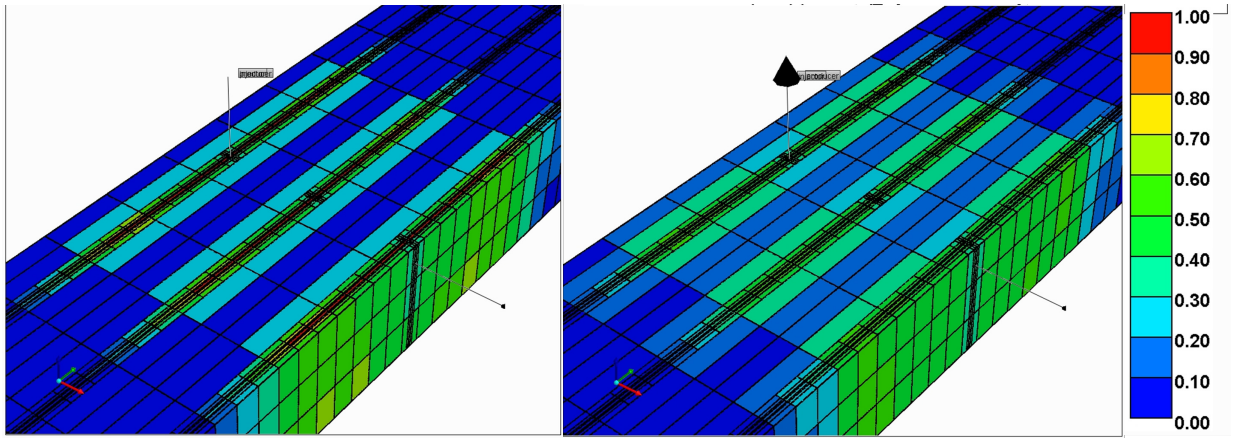
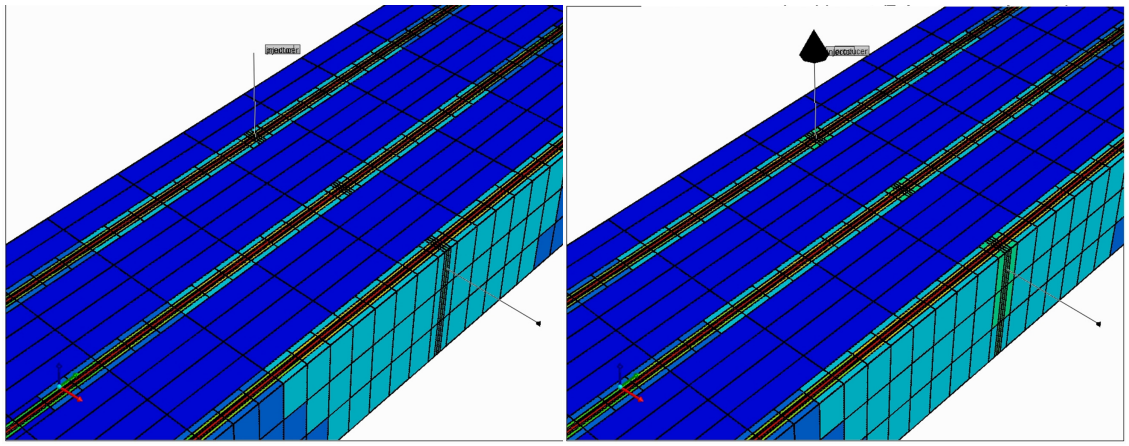


Figure 5.14: Injection BHP in diffusion and non-diffusion case (k=5000 nD; $q_{inj} = 100$ MSCF/Day; $p_{bh} = 2000$ psi.

To show the effect of molecular diffusion, we run the simulator and compare the global concentration of CO_2 (global concentration is the terminology of CMG-GEM) in the system at different periods of the first cycle. The global concentration is defined as the mole fraction of a given component in the entire system. Figs. 5.15 a, b, c, and d show the global concentration during the injection and soaking periods of the first cycle at each grid. We cut our reservoir geometry in half so that we can see what happens around fractures. We performed simulation runs both considering (DIFF) and not considering (NODIFF) molecular diffusion. If we investigate the case considering molecular diffusion, we observe that global concentration of CO_2 near the fracture has increased during the soaking period (Figs. 5.15 a, and b). However, this does not occur for the cases that do not consider molecular diffusion (Figs. 5.15 c, and d). The results of Fig. 5.15 show that molecular diffusion plays a crucial role in the soaking and the injection periods.



(a) 180^{th} day of injection (DIFF) **(b)** 90^{th} day of soaking (DIFF)

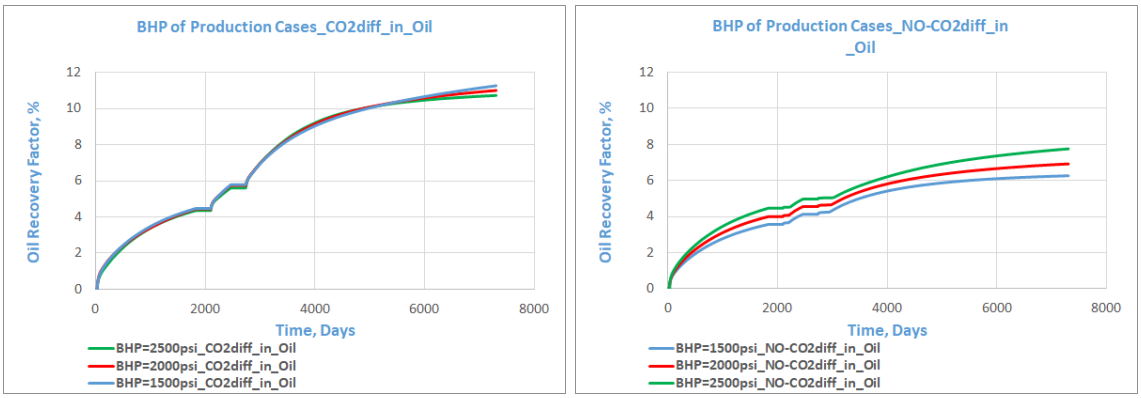


(c) 180^{th} day of injection (NO-DIFF) **(d)** 90^{th} day of soaking (NO-DIFF)

Figure 5.15: Global concentration of CO_2 at each grid in the entire system at 180^{th} day of injection with molecular diffusion (a), 90^{th} day of soaking with molecular diffusion (b), 180^{th} day of injection without molecular diffusion (c), and 90^{th} day of soaking period without molecular diffusion (d).

5.2.2 Sensitivity of BHP on Recovery Factor

Fig. 5.16 shows the behavior of RF at different BHPs. The reservoir permeability is 50 nD, is a single porosity. Flow is multiphase Non-Darcy, injection rate is 100MSCF/Day, soaking time is 3 months, injection time is 6 months. The Klinkenberg effect is not considered. In the case where molecular diffusion is ignored, the lowest BHP yielded the maximum RF. This is reasonable because, at lower BHPs, the pressure difference will be high to produce more. However, in the diffusion case, different BHPs did not make any recognizable difference. We can see that we obtain a similar drawdown even before starting the HnP process. This is because including molecular diffusion changes the behavior of production rate



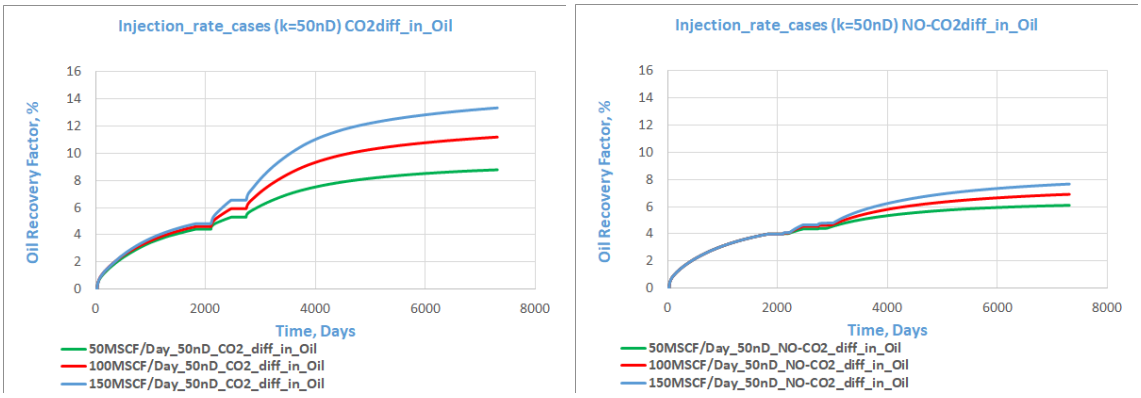
(a) Diffusion **(b)** Non-diffusion

Figure 5.16: Oil recovery factor (RF) at different BHP.

at different BHPs. Since the reservoir is ultralow permeable, the advective term is negligible (Eq. 5.6) as compared to molecular diffusion. Therefore, the magnitude of the pressure difference between BHP and gridblock pressure will not affect mass transport significantly.

5.2.3 Sensitivity of Injection Rate on Recovery Factor

Fig. 5.17 shows the behavior of RF at different injection rates. Reservoir permeability is 50 nD and the reservoir is single porosity. The flow is multiphase Non-Darcy, injection time is 6 months, BHP is 2000 psi, and soaking time is 3 months. The Klinkenberg effect is not considered. As the injection rate increases RF increases, and HnP affects the RF more.



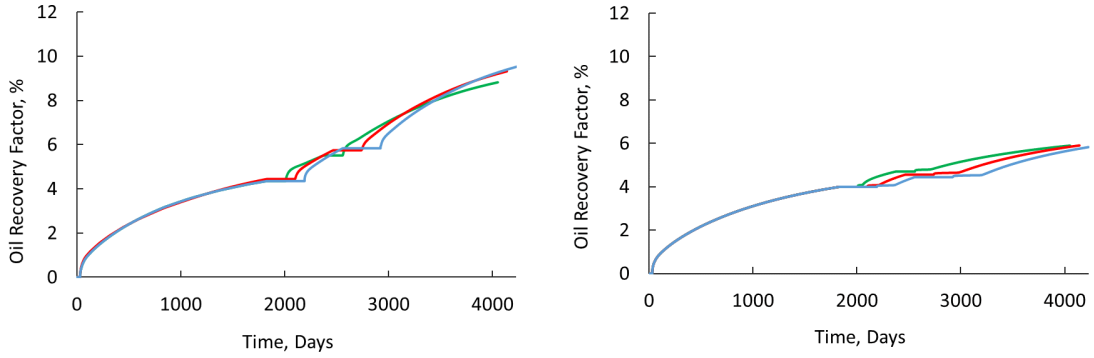
(a) Diffusion **(b)** Non-diffusion

Figure 5.17: Oil recovery factor (RF) at different injection rate cases.

This difference is more in the diffusion case.

5.2.4 Sensitivity of Soaking Time on Recovery Factor

Fig. 5.18 shows a plot of oil RF versus time for three different duration of the soaking period with/without molecular diffusion. Reservoir permeability is 50 nD; production BHP is 2000 psi; duration of injection and production periods are 6 and 9 months, respectively. Different soaking times are considered ranging from a 0 soaking to a 6-month soaking period. To make a fair comparison, the duration of production time after HnP (5 years) was kept unchanged from case to case. However, the total duration of the life-cycle differs from case to case, as can be seen from Fig. 5.18. Increasing soaking time seems not to increase the



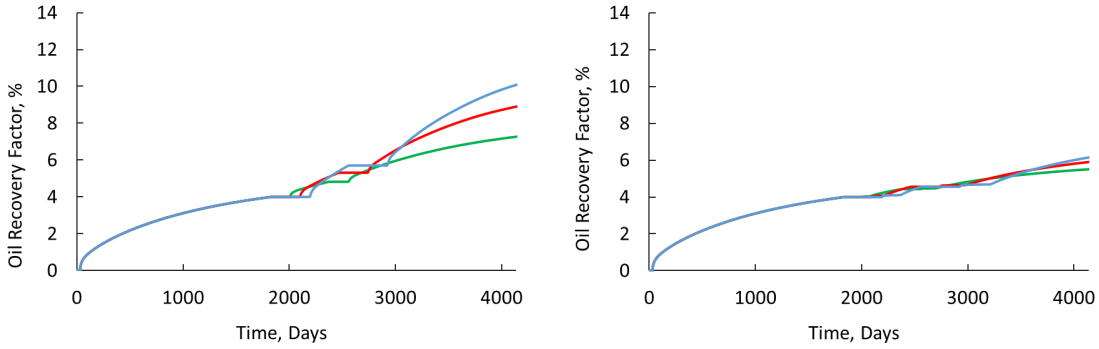
(a) Diffusion **(b)** No-Diffusion

Figure 5.18: Oil recovery factor (RF) vs time at three different duration of soaking period with molecular diffusion (a) and without molecular diffusion (b). Green curve represents no-soaking period; red curve represents 3-months soaking period; blue curve represents 6-months soaking period. k = 50 nD.

RF in the cases where molecular diffusion of CO_2 is not considered. We see from Fig. 5.18b that in fact increasing soaking time decreases the RF slightly. We believe that this occurs because increasing the length of soaking time decreases well-block pressure slightly, which in turn decreases the oil production rate slightly during the production period. However, when molecular diffusion of CO_2 in the oil phase is accounted for in simulation, we observe that increasing soaking period yields more RF, though the increase in RF from 3-month to 6-month soaking is not significant. This is because even though we gain the benefit of molecular diffusion during the soaking period, increasing soaking time results in decreasing well-block pressure slightly.

5.2.5 Sensitivity of Injection Time on Recovery Factor

Fig 5.19 shows oil RF versus time for three different durations of injection period with and without molecular diffusion. Reservoir permeability is 50 nD; the production BHP is 2000 psi; the duration of soaking and production periods in each cycle are 3 and 9 months, respectively. We consider three different durations of the injection period; 3, 6, and 9 months.



(a) Diffusion **(b)** No-Diffusion

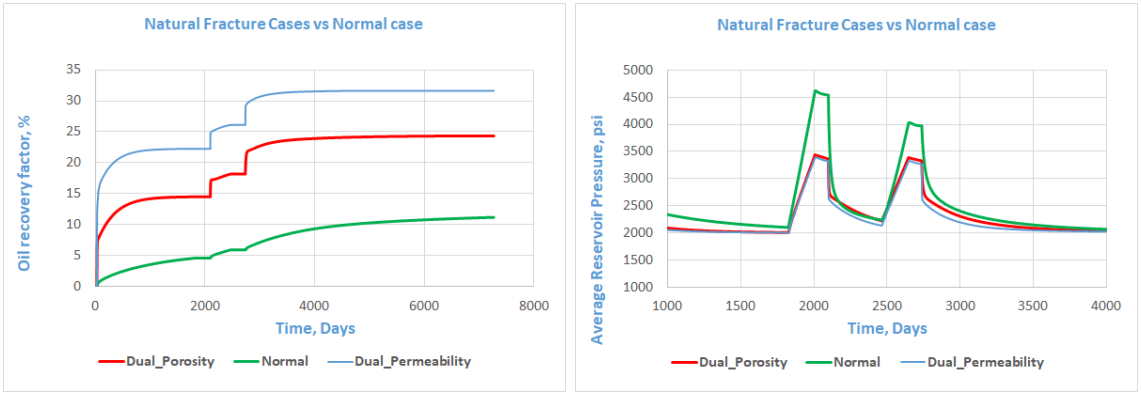
Figure 5.19: Oil recovery factor (RF) vs time at three different duration of injection period with molecular diffusion (a) and without molecular diffusion (b). Green curve represents 3-months injection period; red curve represents 6-months injection period; blue curve represents 9-months injection period. k = 50 nD.

For both cases, with/without molecular diffusion, we observe an increase in RF while the increase is significant in the case of molecular diffusion. This result is expected since injecting a longer period of CO_2 results in a more cumulative volume of gas injected and hence, CO_2 penetrates deeper into the matrix and swells more oil as well as making oil less viscous. A comparison of the effects of the soaking and injection times on oil RF (see Figs. 5.18 and 5.19) shows that RF is more sensitive to the duration of the injection period than that of the soaking period.

5.2.6 Sensitivity of Natural Fractures on Recovery Factor

Fig. 5.20 shows the behaviors of the RF and average reservoir pressure for dual-porosity, dual-permeability, and a homogeneous single-porosity model. The reservoir permeability is 50 nD, BHP is 2000 psi, flow is multiphase Non-Darcy, injection rate is 100MSCF/Day,

soaking time is 3 months, injection time is 6 months, Klinkenberg effect is not considered and molecular diffusion of CO_2 in the oil phase is considered. Before analyzing the plot, it is worth mentioning what is dual-porosity and dual-permeability models and their difference. They both are used to model naturally fractured reservoirs. Fractures are assumed to be orthogonal in the three directions acts as if they are boundaries to matrix elements. In a dual-porosity system and permeabilities of the fractures and matrix are assigned separately (permeability is very high in fractures), but matrix blocks are not connected to each other except the fractures. However, in a dual-permeability system, matrix blocks are connected to each other and thus it provides an alternate path for fluid to flow (GEM, 2016). As



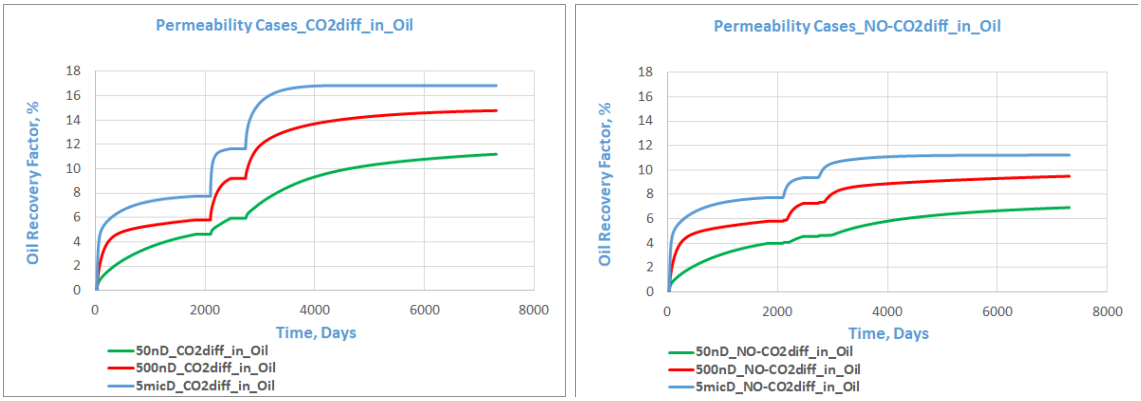
(a) Oil recovery factor **(b)** Average reservoir pressure

Figure 5.20: Effect of including natural fractures on RF (a) and average reservoir pressure (b).

expected in the dual-permeability system, since matrix blocks are also connected (there is transmiscibility between them as well as between them and their fractures), RF is higher than the dual-porosity case. There is a big effect of natural fractures on RF. Having a naturally fractured reservoir plays an important role in the HnP process. HnP is much more effective in naturally fractured reservoirs. Due to ease of flow in natural fracture cases, pressure increases during injection periods are lower than in normal cases. This should be considered when we choose injection rate-we have to inject more in natural fracture cases to achieve high pressure so that we achieve MCM.

5.2.7 Sensitivity of Matrix Permeability on Recovery Factor

Fig. 5.21 shows the behavior of RFs as a function of the matrix permeability for both the molecular diffusion and no molecular diffusion cases. The injection rate is 100 MSCF/Day, the reservoir is single porosity, flow is multiphase Non-Darcy, injection time is 6 months, BHP is 2000 psi, soaking time is 3 months and the Klinkenberg effect is not considered. As expected increasing reservoir permeability increases RF. The effect of the



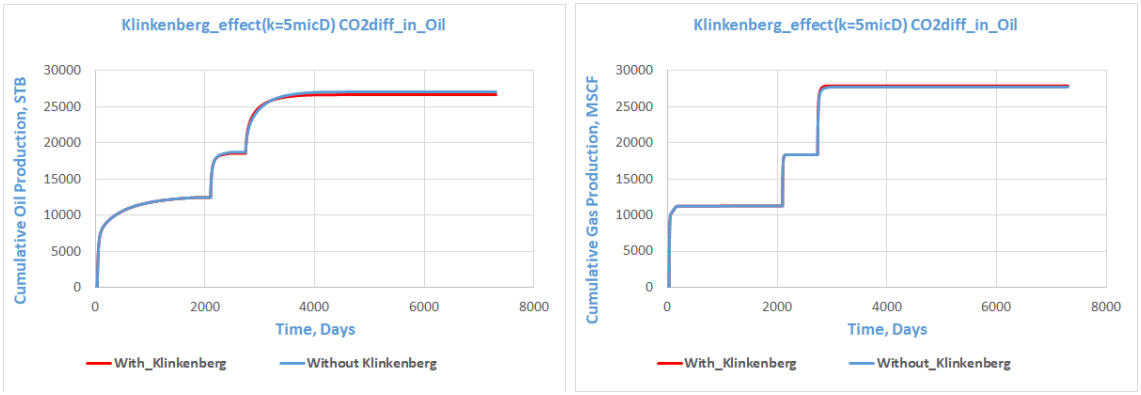
(a) Diffusion **(b)** Non-diffusion

Figure 5.21: Oil recovery factor (RF) as a function of permeability.

matrix permeability is more when we consider molecular diffusion of CO_2 into the oil phase.

5.2.8 Sensitivity of Slippage Flow-Klinkenberg Effect on Recovery Factor

Fig. 5.22 shows cumulative oil and cumulative gas productions for the cases where the Klinkenberg effect is included and not included in the simulation. The reservoir matrix permeability is 5000nD, the reservoir is single porosity, flow is multiphase Non-Darcy, injection rate is 100MSCF/Day, soaking time is 3 months, injection time is 6 months, and molecular diffusion of CO_2 in the oil phase is considered. It must be mentioned that the Klinkenberg effect occurs in the gas flow. Since the supercritical CO_2 state is assumed to be a gas phase in GEM (2016), there is meaning to check this effect also. As expected, due to slippage of gas we get more cumulative gas production and less cumulative oil production. However, this effect is negligible and can be ignored in the HnP process.



(a) Cumulative oil production **(b)** Cumulative gas production

Figure 5.22: Klinkenberg effect on RF.

5.2.9 Selection of Numerical Parameters in GEM.

Here, we report some difficulties that may be encountered when calling the CMG-GEM (2016) compositional simulator for performing CO_2 HnP production optimization and provide some guidelines to circumvent these difficulties. Compositional simulation models can be unstable, especially during a gas injection process due to rapid changes in pressure, saturation, and concentrations of components. Therefore, understanding the compositional modeling of CMG-GEM (2016) is important. Besides, understanding how the numerical parameters of the solver in CMG-GEM (2016) must be modified from their given defaults to achieve convergence is important. We believe that the information we provide here could be useful to the readers who may be interested in modeling the miscible CO_2 HnP process by CMG-GEM (2016). In the numerical section of CMG-GEM (2016), the default values of the solver parameters need to be changed to achieve convergence. When permeability is very low and there is a hydraulic fracture (or local grid refinement), small gridblocks can cause convergence problems. Specifically, when we inject fluid into a small well-block, where the perforation exists, variations in pressure, the global mole fraction of components, and saturation can be very high at a given time step, causing convergence problems. Therefore, it is recommended to increase the maximum number of time steps (MAXSTEPS), linear solver iterations (ITERMAX), and linear solver orthogonalization (NORTH). To handle convergence issues, it is also suggested to decrease the maximum time-step size (DTMAX),

minimum time-step size (DTMIN), and first time-step size after changing a well (DTWELL). After having modified those numerical parameters, one may still have convergence issues. In that case, it is recommended to increase the maximum allowed changes of pressure (PRESS), saturation (SATUR), and global composition (GMOLAR). One can also increase the number of maximum Newton iterations (NEWTONCYC) to achieve convergence, but this causes much more memory use. Another convergence issue which one could face is related to the vapor-liquid equilibrium (VLE) equation. Since we include molecular diffusion, the solution method must be changed to a fully implicit method for all grids in the reservoir by setting an implicit flag to 3. Note that using the fully implicit solution for all grids increases the computational time.

5.2.10 Conclusions of Sensitivity Analysis

1. The density and viscosity of gas phase as computed from GEM (2016) show consistency with the typical viscosity and density behavior of supercritical CO_2.

2. Supercritical CO_2 is handled as the gas phase in GEM (2016).

3. Due to molecular diffusion of CO_2 in the oil phase, we see an increase of gas mole fraction of CO_2 inside the matrix.

4. Mass transport of CO_2 components in ultralow permeable reservoirs is mainly due to molecular diffusion of CO_2 components in the oil phase in which the bulk velocity is so small that the advective transport on flow can be neglected.

5. Even though the overall behavior of oil viscosity shows that it decreases during HnP, it changes differently during each period of the cycle. This is due to the effects of pressure and molecular diffusion of CO_2 in the oil phase on the viscosity of the oil.

6. Increasing the matrix permeability increases RF in the HnP process. And this increment is high when we consider molecular diffusion of CO_2 in the oil phase.

7. Increasing the injection rate increases RF in the HnP process. Effect of injection rate becomes more significant when we consider molecular diffusion of CO_2 in the oil phase.

8. When the injection time increases, RF increases. Effect of injection time is important when molecular diffusion of CO_2 in the oil phase is considered. However, in NPV optimization, this is not the conclusion since we need to consider injection cost as well.

9. Increasing soaking time does not seem to increase RF in the cases where molecular diffusion of CO_2 is not considered. In the cases where we considered molecular diffusion of CO_2 in the oil phase, soaking yields more RF.

10. Effect of BHP for the production period is negligible when molecular diffusion of CO_2 in the oil phase is considered. However, when it is not considered, the lower the BHP the higher RF is.

11. RF is doubled when a naturally fractured reservoir is considered.

12. Since the reservoir fluid is in the oil phase most of the time, considering slippage has no significant effect on RF during the HnP process.

13. Numerical section of GEM (2016) should be modified from its default values to get convergence when we use miscible CO_2 injection, considering hydraulic fracture, and molecular diffusion. A fully implicit method must be used to be able to simulate the cases when molecular diffusion is included.

5.3 Results of Production Optimization of the CO_2 HnP Process in Unconventional Reservoirs

In this section, we provide our results obtained by using synthetic examples to demonstrate the use of GPR- and LS-SVR-based iterative-sampling-refinement optimization methods in comparison with the conventional numerical gradient ascent optimization method. We have to mention that for optimization, we consider a more realistic reservoir model than that considered for the sensitivity analysis presented before, including geomechanical effects,

natural fracture, and full stage of hydraulic fracture instead of considering only one stage. As you will see in the following section, to simulate a realistic reservoir model, we also consider a production period before the HnP process is started. The NPV objective function used is given by Eq. 2.5. The type and number of design variables change depending on the production optimization case considered.

In the first subsection, we introduce rock and fluid properties of the reservoir model considered for the production optimization problem of HnP, and probability distributions of some of the reservoir parameters for the robust optimization. In that subsection, we also introduce the optimization cases. In Subsection 5.3.2 to Subsection 5.3.6, we show the results of the optimization cases. For each production optimization case, we compare optimization methods considered in this research such as GPR-, LS-SVR-based iterative-sampling-refinement optimization methods, simplex, and finite difference optimization methods. We also show the accuracy of the initial proxy models of GPR and LS-SVR using test data. Validation data are split from our training data to be used in the k-fold cross-validation process for hyperparameter optimization. In Subsection 5.3.7, we also investigate the effect of random training/test data split on training accuracy of initial proxy models using the LS-SVR as the initial proxy model. The effect of choice of the lower bound of Δt_p^n on the application and performance of iterative-sampling-refinement optimization method is examined in Subsection 5.3.8. In Subsection 5.3.9, we quantify the uncertainty of the GPR model at the optimum design variable result of each production optimization case.

5.3.1 Reservoir Model and Production Optimization Cases

The fluid composition of the reservoir is taken from Nojabaei et al. (2013), which is Middle Bakken oil. Injected fluid is 100 percent CO_2 with molecular diffusion coefficient 0.0008 cm^2/sec. Properties and mole fractions of reservoir fluid components are given in Table 5.5, while Table 5.6 shows binary interaction coefficients between components. We have used the semi-analytical (key-tie lines) method given in WINPROP (2004) to calculate MMP of reservoir fluid with CO_2. The estimated multi-contact minimum miscibility pres-

sure (MCMMP) is estimated to be 2500 psi, whereas the first contact minimum miscibility pressure (FCMMP) is 11,000 psi at reservoir temperature, $T = 240$ °F. MCM mechanism is vaporizing and condensing combined gas drive. CO_2 is at super-critical condition during the life of the HnP process.

In our applications, the effects of capillary pressure and hysteresis in relative permeability curves on phase behavior in small- or nanometer-scale pores are ignored. Recent studies by Calisgan et al. (2017), Ma and Jamili (2014), Nojabaei et al. (2013), and Firincioglu et al. (2012) have investigated how the capillary pressure affect the phase behavior as well as the field production profiles (e.g., gas rates, oil rates, and gas-oil ratios, etc.) All these studies show that when dealing with the multiphase flow in small or nanometer-scale pores, the capillary pressure can have a large impact on the phase behavior and consequently the field pressure and production profiles. Particularly, the study by Firincioglu et al. (2012) and Nojabaei et al. (2013) show that the capillary pressure in small pores results in lowering bubble point pressure and positively impacts the performance of unconventional liquids-rich reservoirs. We refer the readers to these studies for details on the effect of capillary pressure on the production performance of unconventional liquids-rich reservoirs including the effect of capillary pressure. We should note that none of the studies cited above considered the hysteresis effects on the capillary pressure and relative permeability effects. Such effects may also have a large impact on the CO_2 HnP processes. We believe that investigation of the effects of hysteresis on capillary pressure and relative permeability curves on CO_2 HnP processes in unconventional liquids-rich reservoirs is itself an important topic and calls for a future study.

The reservoir is assumed to be a naturally fractured reservoir with a heterogeneous matrix and natural fracture permeability field. The spherical covariance model is used to include the spatial correlation of the permeability field. We also included the effects of geomechanics using GEM (2016). We have 4 stages each of which has 4 planar hydraulic fractures. In Fig. 5.23, the permeability field of the matrix zone is illustrated. GEM (2016) provides both planar and complex fracture templates, both of which use the LS-LR

Table 5.5: Properties of pseudo-components of Middle Bakken oil, after Nojabaei et al. (2013).

Component	Molar fraction	Critical pressure (atm)	Critical temperature (K)	Critical volume (L/mol)	Molar weight (g/gmol)	Acentric factor	Parachor coefficient
CH_4	0.36736	44.57	186.29	0.0989	16.043	0.0102	74.8
C_2H_6	0.14885	49.13	305.53	0.148	30.07	0.1028	107.7
C_3H_8	0.09334	41.90	369.98	0.203	44.097	0.152	151.9
NC4	0.05751	37.18	421.78	0.255	58.124	0.1894	189.6
C5-C6	0.06406	31.38	486.37	0.336	78.295	0.2684	250.2
C7-C12	0.15854	24.72	585.14	0.549	120.562	0.4291	350.2
C13-C21	0.0733	16.98	792.4	0.948	220.716	0.7203	590
C22-C80	0.03704	12.93	1024.71	2.247	443.518	1.0159	1216.8

Table 5.6: Binary interaction coefficients (BIC) for components of Middle Bakken oil, after Nojabaei et al. (2013).

Component	CO_2	CH_4	C_2H_6	C_3H_8	NC4	C5-C6	C7-C12	C13-C21	C22-C80
CO_2	0	0.105	0.13	0.125	0.115	0	0	0	0
CH_4	0.105	0	0.005	0.0035	0.0035	0.0037	0.0033	0.0033	0.0033
C_2H_6	0.13	0.005	0	0.0031	0.0031	0.0031	0.0026	0.0026	0.0026
C_3H_8	0.125	0.0035	0.0031	0	0	0	0	0	0
NC4	0.115	0.0035	0.0031	0	0	0	0	0	0
C5-C6	0	0.0037	0.0031	0	0	0	0	0	0
C7-C12	0	0.0033	0.0026	0	0	0	0	0	0
C13-C21	0	0.0033	0.0026	0	0	0	0	0	0
C22-C80	0	0.0033	0.0026	0	0	0	0	0	0

(logarithmically spaced local grid refinement) method. We used the planar fracture template in GEM (2016) to model hydraulic fractures. The reservoir system is discretized into 29x17x1 (=493) gridblocks. Only the well gridblocks having hydraulic fractures have been perforated.

Although we consider uncertainty in the reservoir model for robust optimization, some parameters and properties are assumed to be deterministic such as fracture height, fracture width, and critical and irreducible saturations both in the matrix and natural fracture zones (Table 5.7). However, we also have uncertain parameters coming from different probability distributions such as matrix permeability (k_M) and natural fracture permeability (k_f), hydraulic fracture half-length of each stage (x_f), hydraulic fracture permeability (k_{hf}), and secondary permeability (k_{SRV}) at each stage (Table 5.8). We also included relative permeability endpoints $k_{ro,end}$ and $k_{rg,end}$ and exponents (n_o, n_g) both in natural fracture zone and

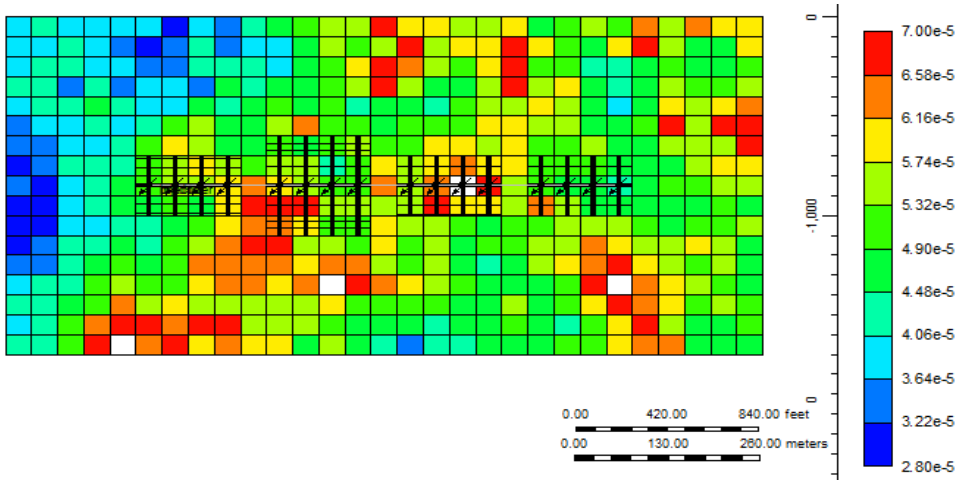


Figure 5.23: Areal grid system, hydraulic-fracture distribution, and permeability field of the matrix zone (color bar represents matrix permeability in mD).

matrix zone, as well as geomechanical parameters (E, ν, α) as uncertain parameters.

Initially, we produce the reservoir for 450 days with the producing bottomhole pressure (BHP) schedule shown in Fig. 5.24. At the end of initial production at 450 days, the recovery factor (RF) is 4.1%. Then we apply an HnP process consisting of 5 cycles for a total duration of 3000 days. In this reservoir model case, the pressure distribution of the reservoir is, no longer, uniform everywhere. Economical values are provided in Table 5.9.

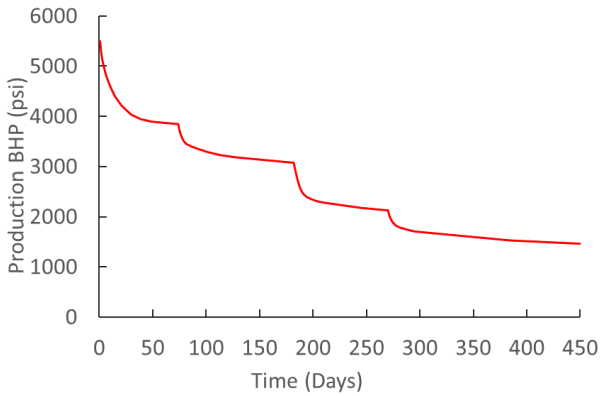


Figure 5.24: Initial production BHP history before HnP process starts.

Unless otherwise stated, our design variables are injection rate, production BHP, injection time fraction, production time fraction, and length of each cycle. We consider the five cycles HnP process for most of the optimization cases. However, we also have one production optimization case where we considered 25 cycles, and cycle lengths are not in

Table 5.7: Input values of deterministic reservoir parameters used in the synthetic reservoir model.

Parameter	**Value**	**Unit**
Dimension of the model	3,480x1,700x50	ft^3
Depth	11,000	ft
Initial reservoir pressure	7,800	psi
Reservoir temperature	240	^oF
Shape factor	2	ft^2
$S_{or,m}$, residual oil saturation, in matrix zone	0.2	fraction
$S_{gc,m}$, critical gas saturation, in matrix zone	0.05	fraction
$S_{or,f}$, residual oil saturation, in natural fracture zone	0.05	fraction
$S_{gc,f}$, critical gas saturation, in natural fracture zone	0	fraction
Compressibility of matrix zone	1e-6	1/psi
Compressibility of natural fracture zone	1e-5	1/psi
Matrix porosity	0.04	fraction
Hydraulic fracture porosity	0.4	fraction
σ_h, minimum horizontal stress	6,500	psi
σ_H, maximum horizontal stress	7,500	psi
σ_v, vertical stress	10,500	psi
Fracture spacing	100	ft
Fracture stage spacing	200	ft
Fracture height	50	ft
Fracture width	0.004	ft

the set of design variables, named as *full-25 optimization case* (Table 5.11). Since the total life is fixed to 3000 days, for that case each cycle length is fixed and equal to 120 days. In that specific optimization case, the number of design variables is 100. The values of bound constraints used for the design variables are given in Table 5.10.

First, we consider the deterministic cases for which a reservoir model is taken randomly as one of the realizations drawn from the stochastic reservoir model where different model parameters may assume different probability distributions (Table 5.8).

Three different deterministic optimization cases are considered (Table 5.11). Here our intention is to investigate if including the duration of production and injection periods in each cycle and cycle length in addition to well control production BHP and injection rate

Table 5.8: Stochastic (or uncertain) reservoir parameters and their distributional properties.

Parameter	Distribution	mean (min)	variance (max)	unit
k_M, matrix zone permeability	log-normal	5.00E-5	1.00E-10	mD
k_f, natural fracture zone permeability	log-normal	1.00E-2	4.00E-6	mD
k_{hf}, hydraulic fracture permeability (each stage i.i.d)	log-normal	5,000	90,000	mD
k_{SRV}, SRV permeability zone permeability (each stage i.i.d)	log-normal	1	4.00E-2	mD
x_f, fracture half-length (each stage i.i.d)	normal	150	400	ft
$k_{ro,end}$ and $k_{rg,end}$, end point relative permeability (in fracture and matrix zone; i.i.d)	normal	0.5	0.04	fraction
n_o and n_g (in fracture and matrix zone; i.i.d)	normal	3	1	fraction
E (Young modulus)	uniform	(3.50E+06)	(7.00E+06)	psi
ν (Poisson ratio)	uniform	(0.25)	(0.40)	fraction
α (Biots coefficient)	uniform	(0.5)	(0.85)	fraction

Table 5.9: Economical constants used for the NPV function (Eq. 2.5).

Constants	Value	Unit
b	0.1	fraction
r_o	63	\$/STB
$c_{CO_2,p}$	0.35	\$/MSCF
$c_{CO_2,i}$	1.5	\$/MSCF

in the design vector increases our maximum NPV. The simple optimization case refers to the optimization where we fix each cycle length and duration of injection and production periods in each cycle. The full optimization case refers to the optimization where only each cycle length is fixed, and the cycle optimization case is the more general case of optimization including the cycle length as a design variable as well (see Table 5.11). For the robust optimization, we considered the most general case, which is the cycle optimization case (Table 5.11).

All optimization cases are compared with the *base case* for which we do not perform any optimization. In the base case, we fix our design variables at reasonable values that

Table 5.10: Bound constraints for the design variables (HnP problem).

Design variables	low	up	unit
p_{bh}^n	1500	2400	psi
$q_{CO_2,i}^n$	40	250	MSCF/Day
$\widehat{\Delta t_p^n}$	0.3	1	fraction
Δt^n	35	1500	Days

make sense. Injection and production time fractions are fixed at 0.33 and 0.58, respectively, and are the same as in the simple optimization case. Also, we fixed the injection rate to its upper bound and production BHP to its lower bound as shown in Table 5.11. The reason why we consider a base case and compare our maximum NPV results with it is to investigate if we can achieve a better NPV by performing optimization. The NPV for the base case is 11.40 million USD.

5.3.2 Simple Optimization Case

In this case, we fix injection and production time fractions as 0.33 and 0.58. These values are chosen arbitrarily. However, a longer production time fraction than an injection time fraction is usually preferred. We built the initial proxy models with 40 training samples and tested them with 18 test samples. The initial LS-SVR and GPR models built with these sizes of training and test sets are accurate enough to start the iterative-sampling-refinement optimization method because our criteria for selecting an initial proxy, i.e., $MAE \leq 0.1$ and $R^2 \geq 0.8$, are satisfied (see Fig. 5.25a and 5.25b).

Using the trained LS-SVR and GPR proxy models as initial models in the iterative-sampling-refinement optimization, we found the optimum design variables and the corresponding maximum NPV. Fig. 5.26 is a comparison of the NPVs computed by the LS-SVR- and GPR-based iterative-sampling-refinement optimization methods and simplex optimization methods as a function of iteration. Iteration number 0 corresponds to the initial guess of the design variables used, and for consistency, we have used the same initial guess of design variables for all optimization methods. *NPV Simulator* in Fig. 5.26 refers to the NPV ob-

Table 5.11: Cases considered for the optimization applications

Cases	**Design variables**	**Fixed variables**	**Number of design variables**
Base case		Δt^n=600 days; $\widehat{\Delta t}_i^n$=0.33; $\widehat{\Delta t}_p^n$=0.58; $q^n_{CO_2,i}$=250 MSCF/D;p^n_{bh}=1500 psi	0
Simple optimization case	$q^n_{CO_2,i}$; p^n_{bh}	Δt^n=600 days; $\widehat{\Delta t}_i^n$=0.33; $\widehat{\Delta t}_p^n$=0.58	10
Full optimization case	$q^n_{CO_2,i}$; p^n_{bh}; $\widehat{\Delta t}_i^n$; $\widehat{\Delta t}_p^n$	Δt^n=600 days	20
Cycle optimization case	$q^n_{CO_2,i}$; p^n_{bh}; $\widehat{\Delta t}_i^n$; $\widehat{\Delta t}_p^n$; Δt^n		25
Full-25 optimization case	$q^n_{CO_2,i}$; p^n_{bh}; $\widehat{\Delta t}_i^n$; $\widehat{\Delta t}_p^n$	Δt^n=120 days	100
Robust optimization case	$q^n_{CO_2,i}$; p^n_{bh}; $\widehat{\Delta t}_i^n$; $\widehat{\Delta t}_p^n$; Δt^n		25

tained from the simulator at the optimum design variables computed by the SQP algorithm using the LS-SVR or GPR proxy model. As can be seen, GPR performed better over the LS-SVR in this case with converging at fewer iterations and finding a slightly higher NPV than that from LS-SVR. We think this is because GPR has more freedom over the model selection as explained in the GPR section previously. However, we should note that the computational time for constructing a GPR-based proxy model is 10 times more than that for constructing an LS-SVR-based proxy model. The maximum NPV values obtained by the LS-SVR and GPR optimization methods are 11.45 and 11.46 million USD, respectively. Note that both these values are higher than that of the base case. It is worth noting from Fig. 5.26 that the simplex optimization method converged to an optimum value within 17 iterations by satisfying the maximum number of numerical gradient calculations, which was set to 5 (*Backtracking Approach* 2). At the 17^{th} iteration, the NPV estimated by the simplex method is 11.44 million USD. Optimum design variables are given in Fig. 5.27. For

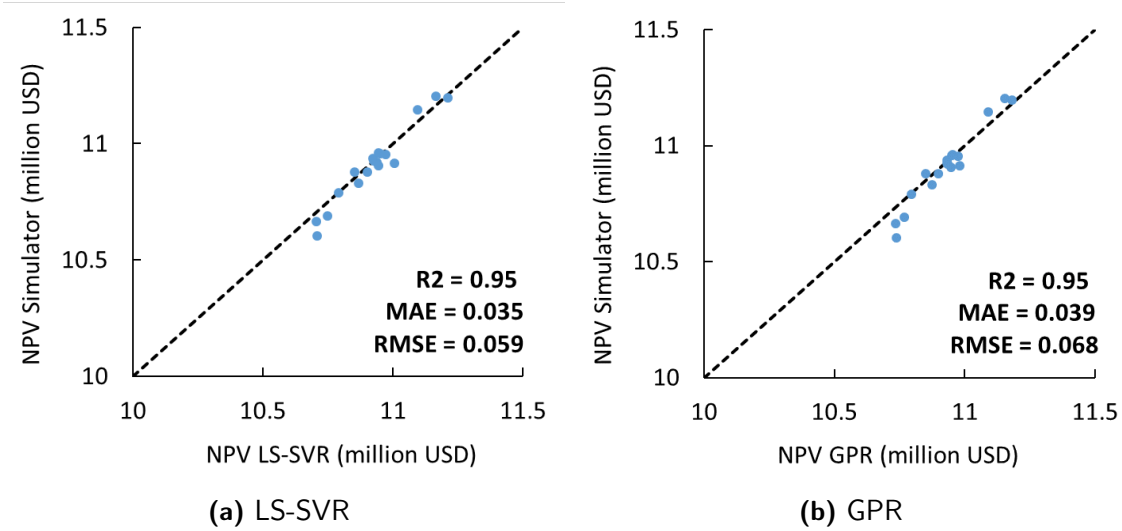
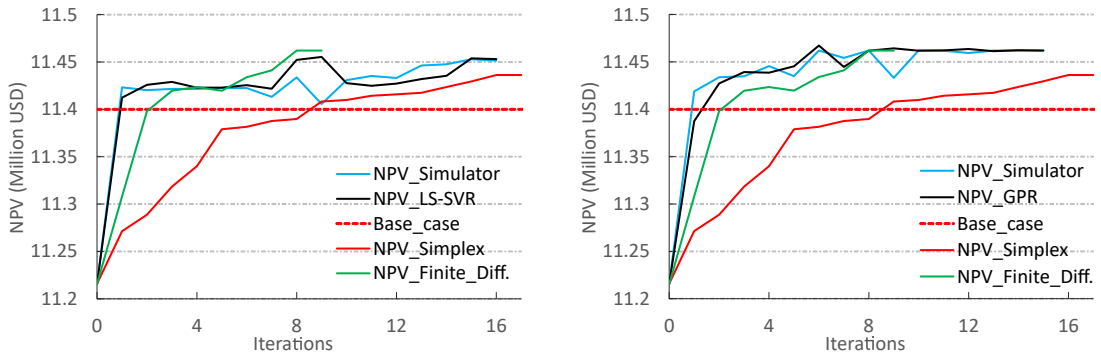


Figure 5.25: Testing of the LS-SVR (a) and GPR (b) proxy models for the simple optimization case; 40 training and 18 test samples.

the optimum production BHP plots, we put zero value for the injection and soaking periods (depicted by dark blue color in Fig. 5.27). Optimum production BHPs are the same and at their lower bound in both LS-SVR and GPR optimization methods. Another point to note from the results in Fig. 5.26 is that the up and downs observed in the proxy-based NPV responses. At each iteration of the SQP method, we update our prediction model by adding a new NPV computed from the simulator corresponding to the design vector determined by SQP into the existing training set and then training the NPV proxy model with a new set of training data. Therefore, at each iteration, the NPV proxy model changes its response surface. Using this new model, the SQP optimization algorithm may drive the NPV proxy function to a newly updated design vector that may change its location in the input (design) space abruptly, resulting in the zig-zag shape of NPVs computed by the proxy-based models. However, from Fig. 5.26, we note that the magnitude of this up and downs in NPV is not significant if one considers the resolution of the vertical scale in Fig. 5.26 and that the oscillations damp out as we continuously update the NPV proxy model.

The results of Table 5.12 show that the simplex optimization method requires approximately six times more reservoir simulation runs than an LS-SVR-based or GPR-based optimization method does. Running a CMG-GEM simulation model requires about 8 min-



(a) LS-SVR **(b)** GPR

Figure 5.26: NPV vs. iterations obtained using the LS-SVR, GPR, simplex and finite-difference optimization methods for the simple optimization case compared with base case.

utes while running an LS-SVR-based proxy model or a GPR-based proxy model requires less than 1 second. The training procedure to build an LS-SVR proxy model for this example is finished in 1 second, while building a GPR proxy is finished in 10 seconds, which are still far negligible compared with the computational costs of simulation runs by CMG-GEM. The computation times in Table 5.12 are for an Intel(R) Xeon(R) CPU E5-1650v2@3.50Hz 3.50 GHz, 12 cores, x64-based processor, where we performed our computations. The times given in minutes in Table 5.12 represent the total computational time spent for running simulation models and constructing an initial proxy model and updating the proxy models during each iteration of the iterative-sampling-refinement optimization method. As the time spent for training an LS-SVR proxy model and running it is less than 1 second, the 592-minute computational time given in Table 5.12 for LS-SVR is equal to the total computational time spent for running 74 simulation models by the LS-SVR-based iterative-sampling-refinement optimization method. We also used a gradient-based method where the gradient of the NPV is computed by using a forward-finite-difference with respect to design parameters to maximize NPV. We used a perturbation size of 0.05 when computing the finite-difference gradient of the NPV with respect to each normalized design variable and an initial step size of $\beta_0 = 1$ in the backtracking algorithm. *Backtracking Approach* 2 was used for convergence with $\epsilon_J = 10^{-4}$ and $\epsilon_u = 10^{-3}$ (see Section 3.4). The finite-difference method required 9

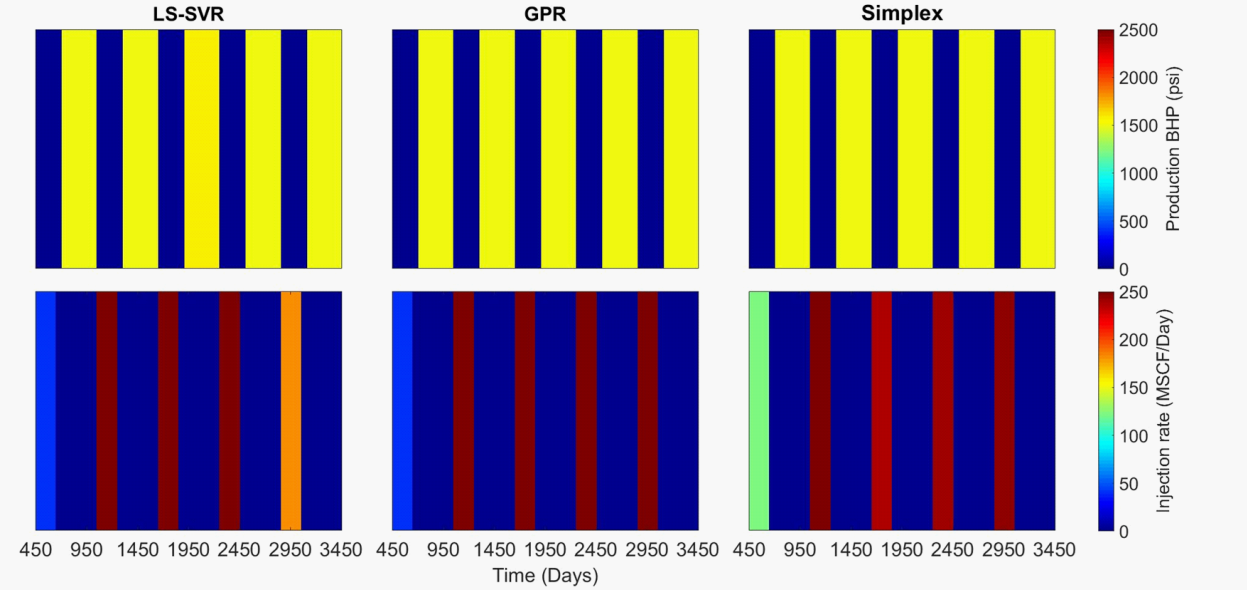


Figure 5.27: Comparison of optimum design variables (upper figures are production BHPs and lower figures are for injection rates) for different optimization methods; the simple optimization case.

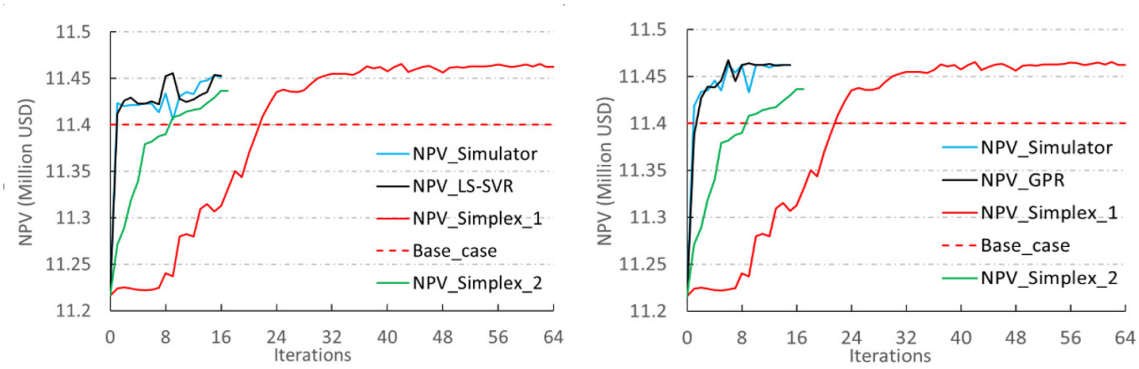
iterations to converge (Fig. 5.26), which required 115 simulation runs (Table 5.12).

Table 5.12: Reservoir simulation runs and computational times required for different optimization methods; the simple optimization case.

Optimization methods	Number of iterations	Number of simulations	Computational time (mins)	Maximum NPV (million USD)
LS-SVR	16	74	592	11.45
GPR	15	73	587	11.46
Simplex_1	64	897	7,176	11.46
Simplex_2	17	403	3,224	11.44
FD	9	115	920	11.46
Base case	-	-	-	11.40

For this optimization case, we also tried *Backtracking Approach* 1 for comparison reason (see Section 3.4). In Fig. 5.28, we showed optimization results of both *Backtracking Approach* 1 (labeled as Simplex_1) and *Backtracking Approach* 2 (labeled as Simplex_2), and compared them each other as well as with LS-SVR-, GPR-based iterative-sampling-refinement optimization results. In *Backtracking Approach* 1, it takes 64 iterations (897 simulation runs) to converge. The reason it took more iterations is that for the iterations where we have inaccurate gradients, we do not re-estimate the gradient unlike in *Backtracking*

Approach 2. This causes the iteration algorithm to go to the point where NPV is lower than the previous iteration even after performing a maximum number of step-size cuts. The zigzag shape of the Simplex_1 plot also confirms what we stated in the previous sentence. However with *Backtracking Approach* 1 we achieve a higher NPV (11.46 million USD) (Table 5.12).



(a) LS-SVR **(b)** GPR

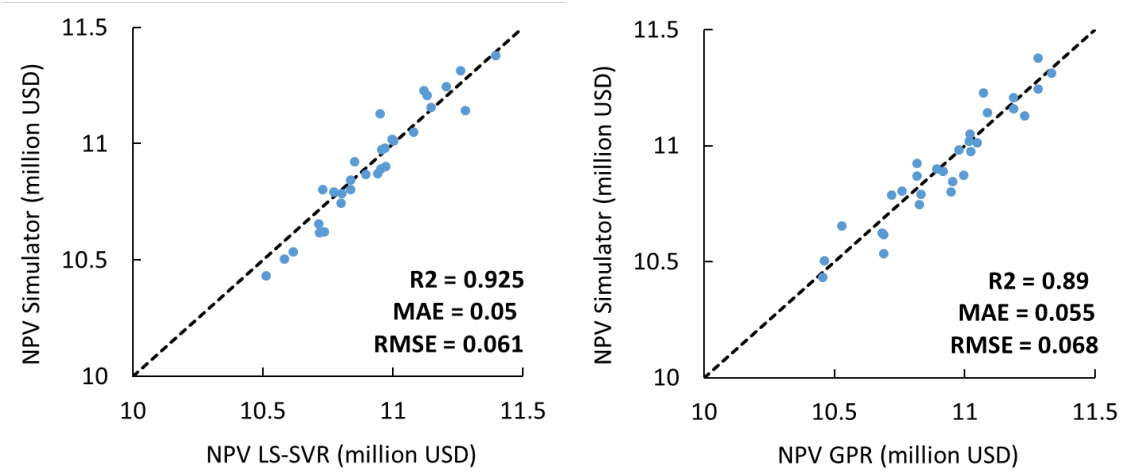
Figure 5.28: NPV vs. iterations obtained using the LS-SVR, GPR, simplex, with both *Backtracking Approach* 1 and *Backtracking Approach* 2, optimization methods for the simple optimization case compared with base case.

The results also indicate that using a gradient-based on finite-difference in the simple optimization case is more efficient than using the simplex method if the size of the design (optimization) variables is small as in this case. However, the finite-difference method is still not as computationally efficient as the LS-SVR- and GPR- based iterative-refinement optimization methods which are approximately 1.5 times computationally more efficient than the finite-difference method.

5.3.3 Full Optimization Case

In this case, we consider the duration of the injection and production periods in each cycle as design variables by fixing cycle length to 600 days so that the design variables are q^n_{i,CO_2}, p^n_{bh}, Δt^n_i, Δt^n_p for $n = 1, ..., N_c = 5$. The total number of design variables is 20. For constructing the initial LS-SVR and GPR proxy models, we used 60 training samples. Then, these models were tested using 30 samples (Fig. 5.29). The accuracy of both initially trained models is sufficient to consider them as the initial proxy models to perform the

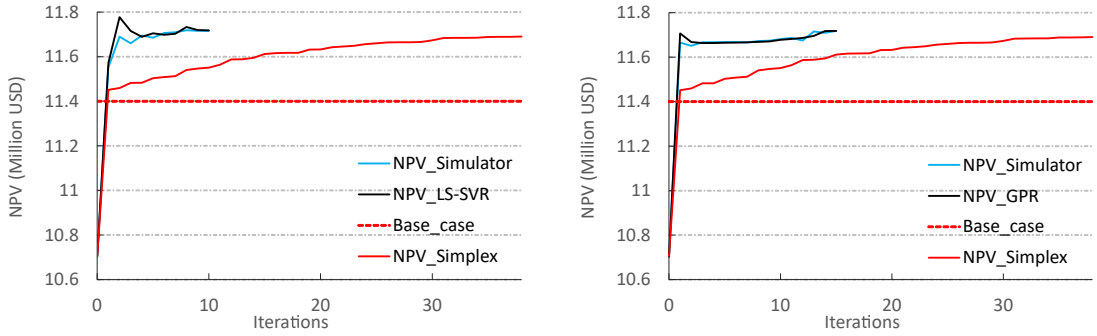
iterative-sampling-refinement optimization procedure based on our initial proxy selection method discussed previously.



(a) LS-SVR **(b)** GPR

Figure 5.29: Testing of the LS-SVR (a) and GPR (b) proxy models for the full optimization case; 60 training samples and 30 test samples.

The plots of NPV vs iterations are given for each method in Fig. 5.30. For this case, unlike the simple optimization case, the GPR converged with more iterations than LS-SVR. The maximum values of NPV obtained by the LS-SVR and GPR methods are 11.71 million USD and 11.72 million USD, respectively.



(a) LS-SVR **(b)** GPR

Figure 5.30: NPV vs. iterations obtained using the LS-SVR, GPR and simplex optimization methods for the full optimization case compared with base case.

As can be seen from Fig. 5.31, each optimization method gives slightly different values of the optimal design variables. From the comparison of the maximum NPVs of

the simple and full optimization cases, we can observe that with including time fractions of injection and production period we achieve a higher maximum value of NPV, about a 2.3% gain in NPV. The number of iterations required for the convergence has increased as compared to the simple optimization case. For this case, GPR took 15 iterations, whereas LS-SVR took 10 iterations. The simplex method took again the largest iteration number to convergence; 38 iterations by satisfying the convergence criterion of the maximum number of gradient calculations, resulting in the maximum NPV equal to 11.70 million USD. It is also interesting to note that for the full optimization case, our results indicate that the duration of the soaking period disappears in each cycle.

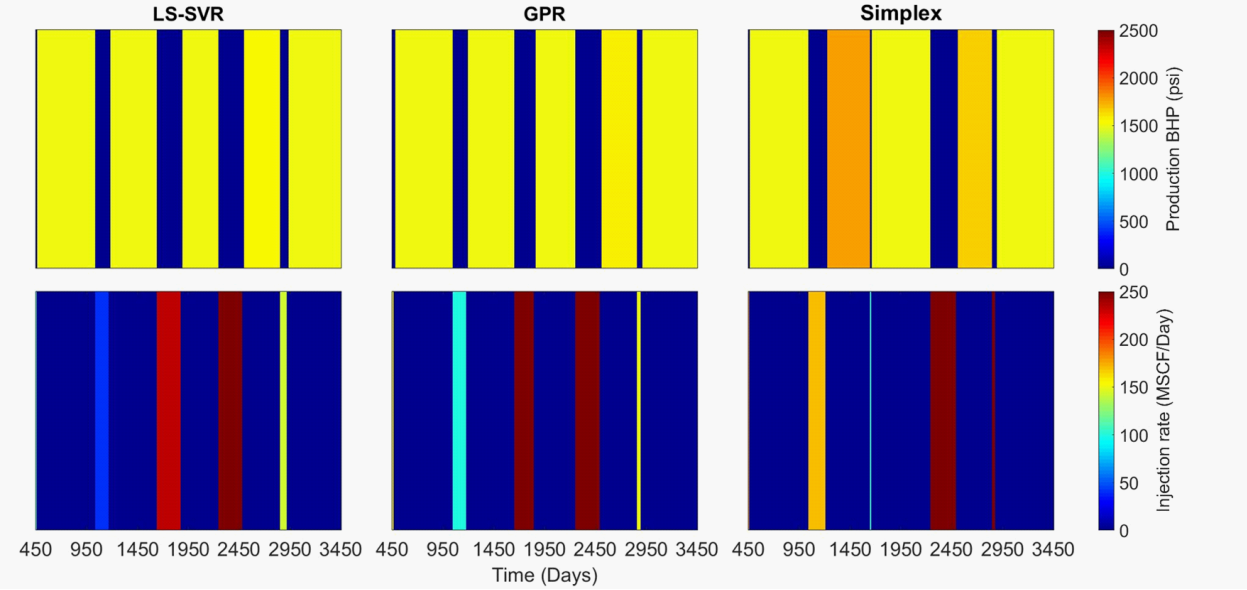


Figure 5.31: Comparison of optimum design variables (upper figures are production BHPs and lower figures are for injection rates) for different optimization methods; the full optimization case.

Our results given in Table 5.13 show that the proxy-based iterative-sampling-refinement optimization method is about 8 times more efficient than the conventional simplex optimization method. In this full optimization case, the LS-SVR optimization method is also more efficient than the GPR optimization method. For this case, in which $N_{tr} = 60$, the required computational times for constructing an LS-SVR and GPR proxy were 1.5 seconds and 15 seconds, respectively.

Table 5.13: Simulation runs and computational times required for different optimization methods; the full optimization case.

Optimization methods	Number of iterations	Number of simulations	Computational time (mins)	Maximum NPV (million USD)
LS-SVR	10	100	800	11.71
GPR	15	105	844	11.72
Simplex	38	873	6,984	11.70
Base case	-	-	-	11.40

5.3.4 Cycle Optimization Case

Now, we add the cycle length as a design variable into a set of our design variables for optimization. The total number of design variables for this case is 25. Here, we also demonstrate the impact of the size of training samples on the accuracy of the proxy models and iterative-sampling-refinement optimization methods. The models trained with 60 training samples were tested with 30 samples, and the models trained with 90 training samples were tested with 40 samples. The results shown in Fig. 5.32 indicate that 90 training samples are sufficient to obtain an accurate proxy model for both LS-SVR and GPR for the cycle optimization case. As expected, training accuracy in return affects optimization results of NPV (Fig. 5.33), and including cycle length as a design variable in addition to other design variables makes the response surface of the NPV function rougher (or more non-linear). Therefore, required training data for achieving a reasonable accuracy increases more as compared to the other deterministic optimization cases. As can be seen from Figs. 5.33a and 5.33b, for the case of LS-SVR, large spikes occur between NPV predicted by the simulator and the LS-SVR proxy at some iteration points, and it takes more iterations as the training size increased from 60 to 90. We believe this is related to the discrepancy between the response surfaces of the "true" NPV function (computed by the simulator) and of the LS-SVR proxy. During iterative-sampling-refinement optimization, we always add new training points into our original training set at each iteration. By adding more training data points, we try to match the true response surface better. When we iteratively train our model with different sets of training data points, the response surface of the predicted model

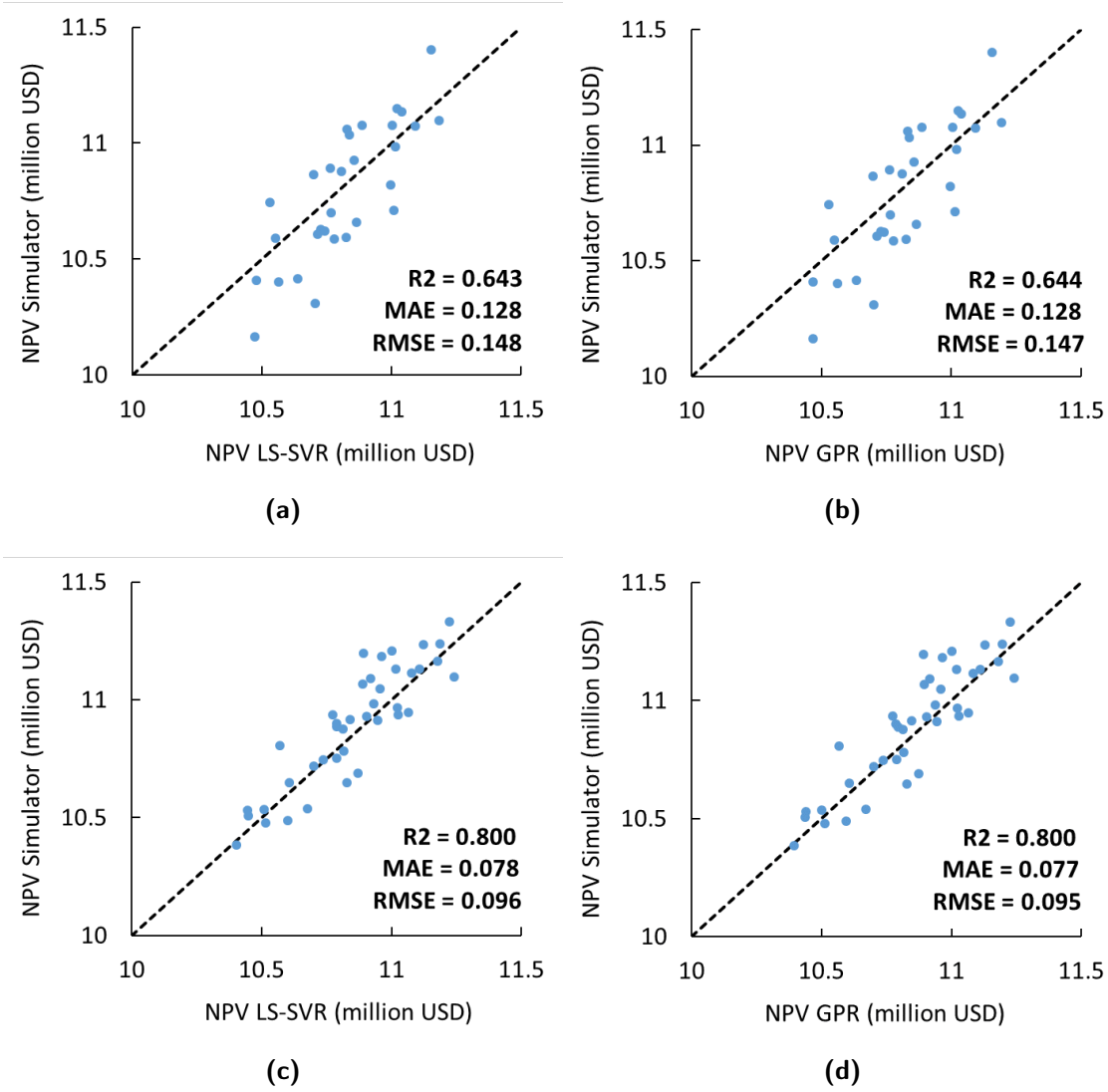


Figure 5.32: Testing of trained models; (a) LS-SVR model trained with 60 samples and tested with 30 samples (b) GPR model trained with 60 samples and tested with 30 samples (c) LS-SVR model trained with 90 samples and tested with 40 samples (d) GPR model trained with 90 samples and tested with 40 samples; the cycle optimization case.

changes its shape. This may be causing the optimum point to be found by the predictive model to change its location in feature (or design) space abruptly, particularly at the earlier iterations. This in turn results in the LS-SVR proxy converging to the optimal point with more iterations. The maximum NPV obtained by the simplex optimization method shows a slightly smaller value than both of the iterative-sampling-refinement optimization methods for the LS-SVR and GPR proxy models constructed with 90 training samples, as shown in Figs. 5.33a and 5.33c. The reason for that is because at the points close to optimum the

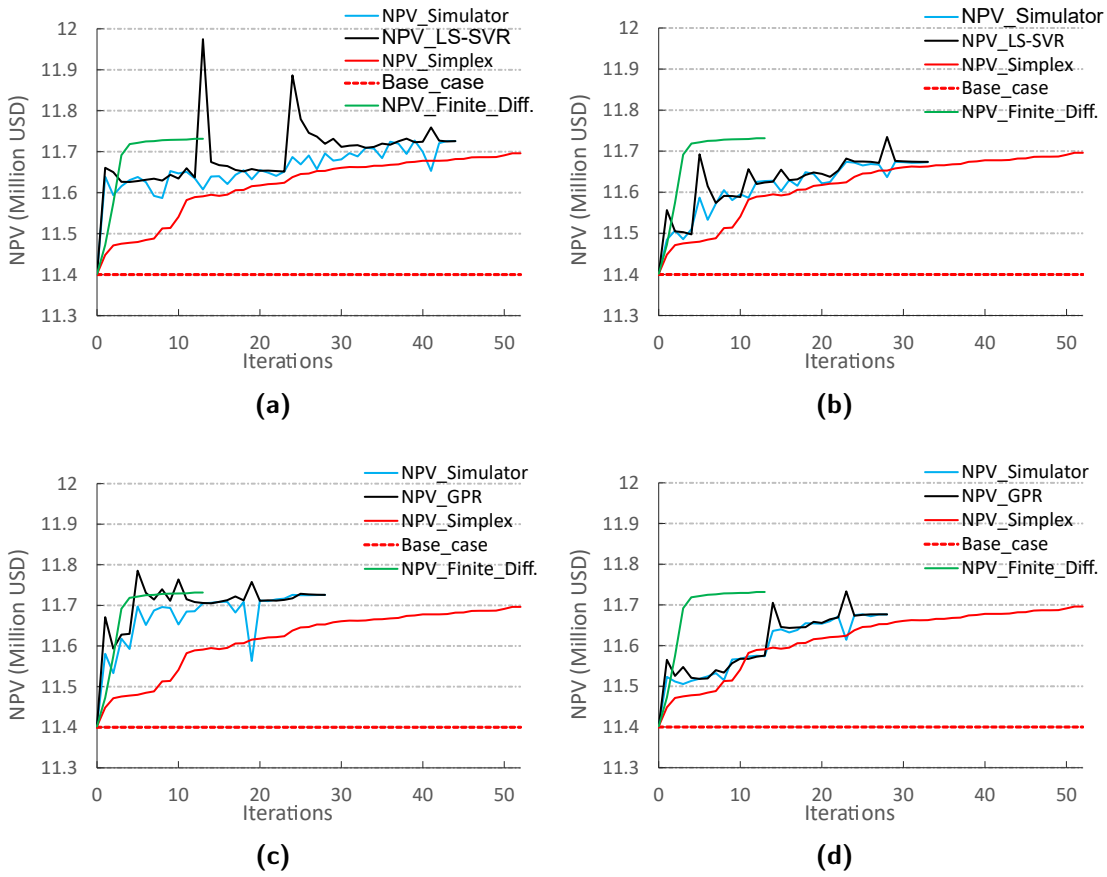


Figure 5.33: NPV vs. iterations obtained using the LS-SVR, GPR, simplex and finite-difference optimization methods; (a) LS-SVR with 90 training samples (b), LS-SVR with 60 training samples (c) GPR with 90 training samples, (d) GPR with 60 training samples; the cycle optimization case compared with the base case.

simplex method may fail to estimate plausible numerical gradient direction (it can estimate descend direction instead of ascent direction), which results in failing to obtain higher NPV for the next iteration of the optimization process. In this case, we re-estimate the numerical gradient again, and it costs simulation runs. To avoid being stuck at re-estimation of the gradient too many times, we put a threshold that if we are required to re-estimate numerical gradient more than 5 times (see *Backtracking Approach* 2), it means we are very close to optimum, and we stop our optimization process at that iteration and accept optimum values. In such situations, we may obtain slightly a lower value of maximum NPV than the iterative-sampling-refinement optimization method. This result also shows the benefit of using the iterative-sampling-refinement optimization method over the conventional simplex

optimization method. Clearly, the initial models trained with 90 training samples achieve higher values of maximum NPV. The results shown in Fig. 5.33 and Fig. 5.34 show that the LS-SVR, GPR, and simplex methods give different optimum design variables even though each method yields essentially the same value of the maximum NPV. This is possible because the response surface of NPV is multi-modal, and we can have local maxima of NPV close to each other with different sets of optimum design variables. The number of simulation runs (and computational times) is compared in Table 5.14. When we used 90 samples for training, the computational times required to construct an LS-SVR and a GPR model were 2 seconds and 20 seconds, respectively. Clearly, the results of Table 5.14 show that the LS-SVR

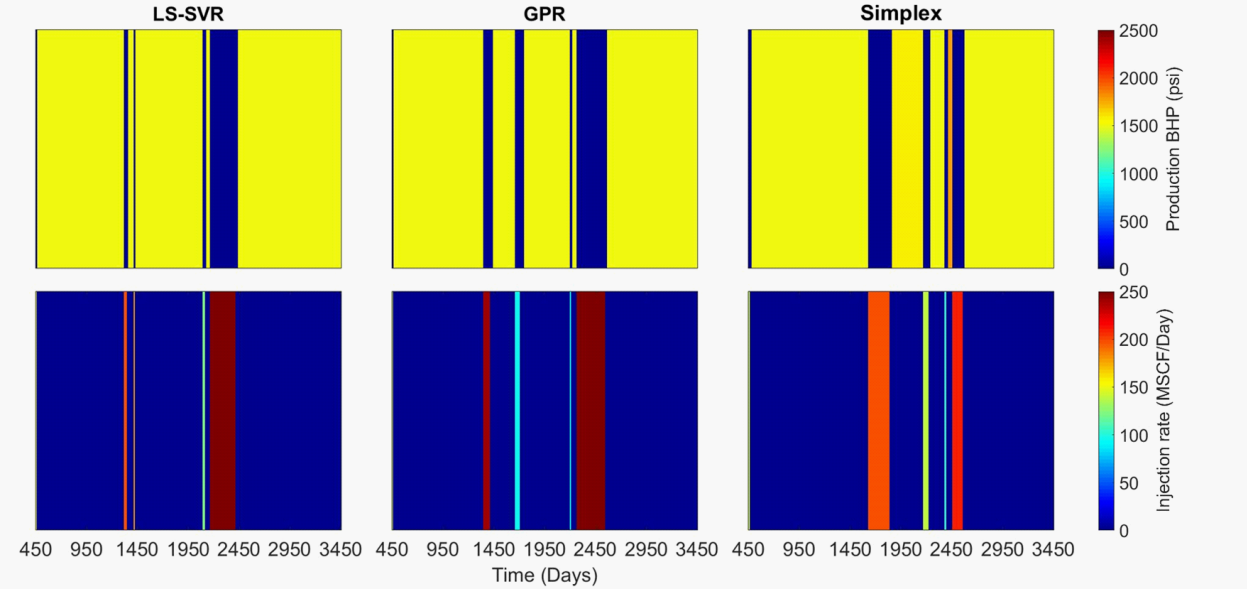


Figure 5.34: Comparison of optimum design variables (upper figures are production BHPs and lower figures are for injection rates) for different optimization methods; the cycle optimization case (where we used a proxy model trained with 90 training samples).

and GPR methods are 6 times more computationally efficient in performing the production optimization for the cycle optimization case. Here, we also used a gradient-based forward-finite-difference method to maximize NPV. For this purpose, we used the forward difference to calculate the numerical gradient with the perturbation size of 0.05 of each normalized design variable and the initial step size of $\beta_0 = 1$ in the backtracking algorithm. We used *Backtracking Approach* 2 for convergence with $\epsilon_J = 10^{-4}$ and $\epsilon_u = 10^{-3}$. Fig. 5.33 shows that the finite-difference-gradient method required 13 iterations to converge. The number of

Table 5.14: Simulation runs and computational times required for different optimization methods; the cycle optimization case (where we used a proxy model trained with 90 training samples).

Optimization methods	Number of iterations	Number of simulations	Computational time (mins)	Maximum NPV (million USD)
LS-SVR	44	174	1,393	11.73
GPR	28	158	1,273	11.73
Simplex	52	975	7,800	11.70
FD	13	384	3,072	11.73
Base case	-	-	-	11.40

simulation runs required and maximum NPV found in the finite-difference-gradient method are given in Table 5.14. The results indicate that using the finite-difference-based gradient method for maximizing the NPV for the cycle optimization case considered here is more efficient than using the simplex-based gradient method. However, the LS-SVR- and GPR-based iterative-refinement optimization methods are more than 2 times computationally more efficient than the finite-difference-gradient method. The results given in Fig. 5.34 and Table 5.15 show that the duration of the optimum soaking period is almost negligible as compared to the duration of optimum injection and production periods.

Table 5.15: Fraction of the optimum soaking [duration of the soaking period / cycle length] for each cycle for three different optimization methods; the cycle optimization case (where we used a proxy model trained with 90 training samples).

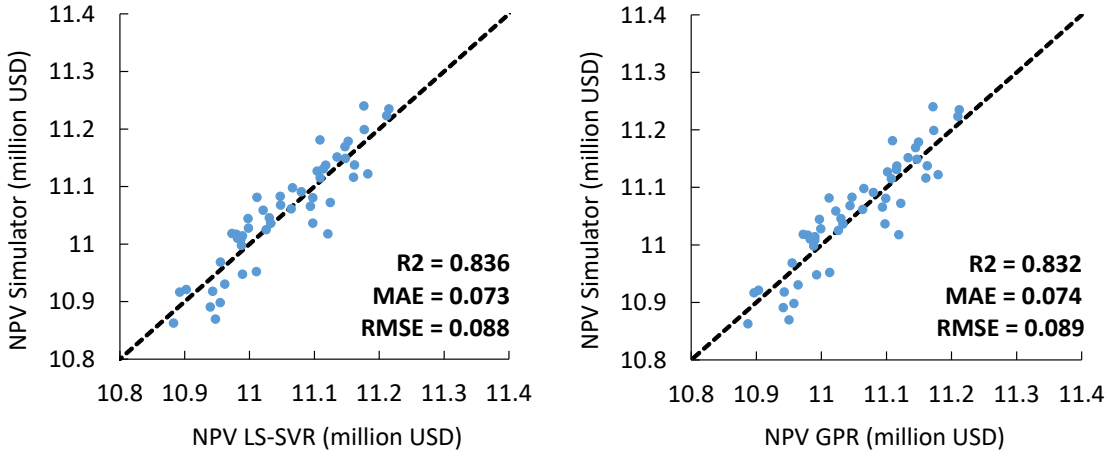
	Duration of soaking period (days)/Cycle length (days)			
Cycle	**LS-SVR**	**GPR**	**Simplex**	**Finite-difference**
1	0/892	0/928	0/1,218	0/1500
2	0/64	17/282	0/534	0/676
3	0/681	38/541	0/181	0/35
4	0/35	0/35	4/35	0/35
5	0/1,330	0/1,214	0/1,032	0/754

The maximum values of NPV achieved in this optimization case for both LS-SVR and GPR is 11.73 million USD, whereas for simplex and finite-difference methods they are 11.70 and 11.73 million USD, respectively. In this complex reservoir model, including cycle length did not improve maximum NPV over the full optimization case. There is just a slight

increase from 11.71 million USD to 11.73 million USD, 0.2% gain in NPV. We believe that it is not correct to deduce a general conclusion such that treating cycle length as a design variable will provide no significant increase in NPV over the full optimization strategy where the cycle length is fixed and that it is a reservoir model specific. An application of the cycle optimization case for a different reservoir model (the reservoir model considered in Almasov et al. 2020a) not shown here provided an 8% gain in NPV over the full optimization case.

5.3.5 Full-25 Optimization Case

In this case, we want to investigate two things: how the iterative-sampling-refinement optimization algorithm works when we increase the number of design variables to 100; how increasing the number of cycles affects maximum NPV results. For this purpose we consider an optimization case where we have 25 cycles, length of each is fixed to 120 days, thus the total life is 25×120 Days=3000 Days. Both LS-SVR and GPR are trained with 100 training samples and tested with 50 test samples. Fig. 5.35 shows that both of the models are accurate enough to perform iterative-sampling-refinement optimization. For simplex optimization,

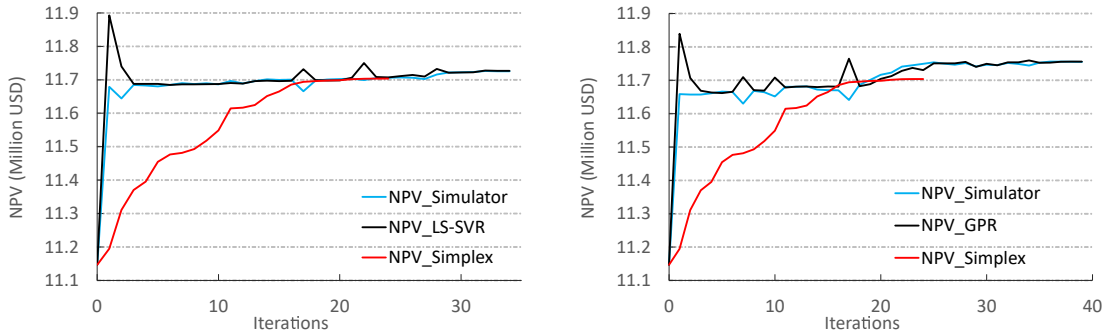


(a) LS-SVR **(b)** GPR

Figure 5.35: Testing of the LS-SVR (a) and GPR (b) proxy models for the full-25 optimization case; 100 training samples and 50 test samples.

we again use *Backtracking Approach* 2 with $N_p = 20$. As you can see from Fig. 5.36, simplex optimization converged in 24 iterations reaching the maximum number of gradient

estimations. Therefore, the maximum NPV obtained from the simplex method is lower than both LS-SVR- and GPR-based iterative-sampling-refinement optimization methods.



(a) LS-SVR **(b)** GPR

Figure 5.36: NPV vs. iterations obtained using the LS-SVR, GPR and simplex optimization methods for the full-25 optimization case.

Our results given in Table 5.16 show that the proxy-based iterative-sampling-refinement optimization method is about 4 times more efficient than the conventional simplex optimization method. In this full-25 optimization case, the LS-SVR optimization method is also more efficient than the GPR optimization method, but GPR finds higher maximum NPV than LS-SVR does. For this case, in which $N_{tr} = 100$, the required computational times for constructing an LS-SVR and GPR proxy were 4 seconds and 25 seconds, respectively. When

Table 5.16: Simulation runs and computational times required for different optimization methods; the full-25 optimization case.

Optimization methods	**Number of iterations**	**Number of simulations**	**Computational time (mins)**	**Maximum NPV (million USD)**
LS-SVR	34	184	1,474	11.72
GPR	39	189	1,528	11.75
Simplex	24	706	5,648	11.70
Base case	-	-	-	11.40

we compare maximum NPV results of full optimization case, where we have 5 cycles, with those of full-25 optimization case, where we have 25 cycles (Tables 5.13 and 5.16), we do not see a significant difference. This sensitivity study of NPV to the number of cycles shows that increasing number of cycle from 5 to 25 did not affect our optimization results.

5.3.6 Robust Optimization Case

As a final optimization, we consider a robust optimization where we maximize the expectation of NPV over the given realizations of plausible reservoir-simulation models to represent the geological uncertainty (see Eq. 3.6). In our specific example here, the ensemble size is $N_e = 20$. In other words, 20 plausible reservoir-simulation models represent the geological uncertainty. In robust production optimization, we include the vector of uncertain model parameters into our input vector (or feature space) in addition to the vector of design variables. So, the feature space for robust production optimization is then $[\mathbf{u}^T, \mathbf{m}^T]^T$. If the number of uncertain model parameters in the vector $\mathbf{m}$ is quite large, the size of the feature space will increase significantly. For example, for our specific reservoir model, we have a total of 2205 gridblocks. If we include all matrix and fracture permeabilities in each gridblock into our feature space, we will additional 4410 ($= 2 \times 2205$) input data. Including the model parameters that have not significant effect on the output (NPV in our case) can lead to an overfitted proxy model being constructed. Hence, reducing the feature space by identifying and then eliminating the model parameters that have not significant effect on the output response is important in machine learning and robust optimization. For this purpose, one can perform a sensitivity analysis. There are various tools to perform sensitivity analysis, such as the principal component analysis (PCA), tornado chart, etc. Here, we have preferred to use a tornado chart for its simplicity in an attempt to identify which of the model parameters for our specific reservoir model may have a significant effect on the NPV. We constructed a tornado chart of NPV with respect to the model parameters (treating them as uniform over all the gridblocks in the simulation model). Here, we used the optimal values of the well controls (injection rate of CO_2 and production BHPs), the duration of injection and production periods, and the length of each cycle estimated from the robust optimization case to be presented later. The mean, low, and high values of each reservoir parameter used to construct the tornado chart shown in Fig. 5.37 are given in Table 5.17. The coding used for fracture stages in Table 5.17 and Fig. 5.37 is as follows: stage 1 refers to the left-most fracture stage shown in Fig. 5.23, while stage 4 refers to the right-most fracture

stage shown in Fig. 5.23. As can be seen from Fig. 5.37, the matrix permeability and SRV region permeabilities, as well as geomechanical properties, do not have a large impact on NPV.

Table 5.17: Reference, low, and high values of the uncertain reservoir parameters.

Parameter	Reference value	low value	high value	unit
k_M (matrix permeability)	4.62E-05	2.31 E-05	6.93E-05	mD
k_f (natural fracture permeability)	8.34E-03	4.17E-03	12.51E-03	mD
$k_{hf,1}$ (hf permeability of hf stage 1)	5,806	2,903	8,709	mD
$k_{hf,2}$ (hf permeability of hf stage 2)	4,628	2,314	6,942	mD
$k_{hf,3}$ (hf permeability of hf stage 3)	5,233	2,616	7850	mD
$k_{hf,4}$ (hf permeability of hf stage 4)	4,575	2,288	6,864	mD
$k_{SRV,1}$ (secondary permeability of hf stage 1)	1.361	0.681	2.043	mD
$k_{SRV,2}$ (secondary permeability of hf stage 2)	0.825	0.413	1.238	mD
$k_{SRV,3}$ (secondary permeability of hf stage 3)	0.752	0.376	1.129	mD
$k_{SRV,4}$ (secondary permeability of hf stage 4)	1.047	0.523	1.570	mD
$x_{f,1}$ (fracture half length of hf stage 1)	200	100	300	ft
$x_{f,2}$ (fracture half length of hf stage 2)	200	100	300	ft
$x_{f,3}$ (fracture half length of hf stage 3)	200	100	300	ft
$x_{f,4}$ (fracture half length of hf stage 4)	200	100	300	ft
$k_{ro,end,m}$ (relative oil permeability end point in matrix zone)	0.433	0.216	0.649	fraction
$k_{ro,end,f}$ (relative oil permeability end point in fracture zone)	0.276	0.138	0.414	fraction
$k_{rg,end,m}$ (relative gas permeability end point in matrix zone)	0.6	0.3	0.9	fraction
$k_{rg,end,f}$ (relative gas permeability end point in fracture zone)	0.632	0.316	0.948	fraction
n_o, m (saturation exponent of oil in fracture zone)	3.183	1.592	4.775	fraction
n_o, f (saturation exponent of oil in matrix zone)	3.455	1.727	5.182	fraction
n_g, m (saturation exponent of gas in matrix zone)	2.523	1.262	3.785	fraction
n_g, f (saturation exponent of gas in fracture zone)	2.151	1.075	3.227	fraction
E (Young modulus)	7E+06	3.50E+06	10.5E+06	psi
ν (Poisson ratio)	0.4	0.2	0.6	fraction
α (Biot's coefficient)	0.650	0.325	0.975	fraction

In the robust optimization case, for dimensionality reduction to avoid overfitting, we assumed the reservoir permeability (matrix and fracture) field to be homogeneous because in unconventional reservoirs the effect of permeability of each grid individually is not as important as the overall effect of the permeability change of the reservoir. In other words,

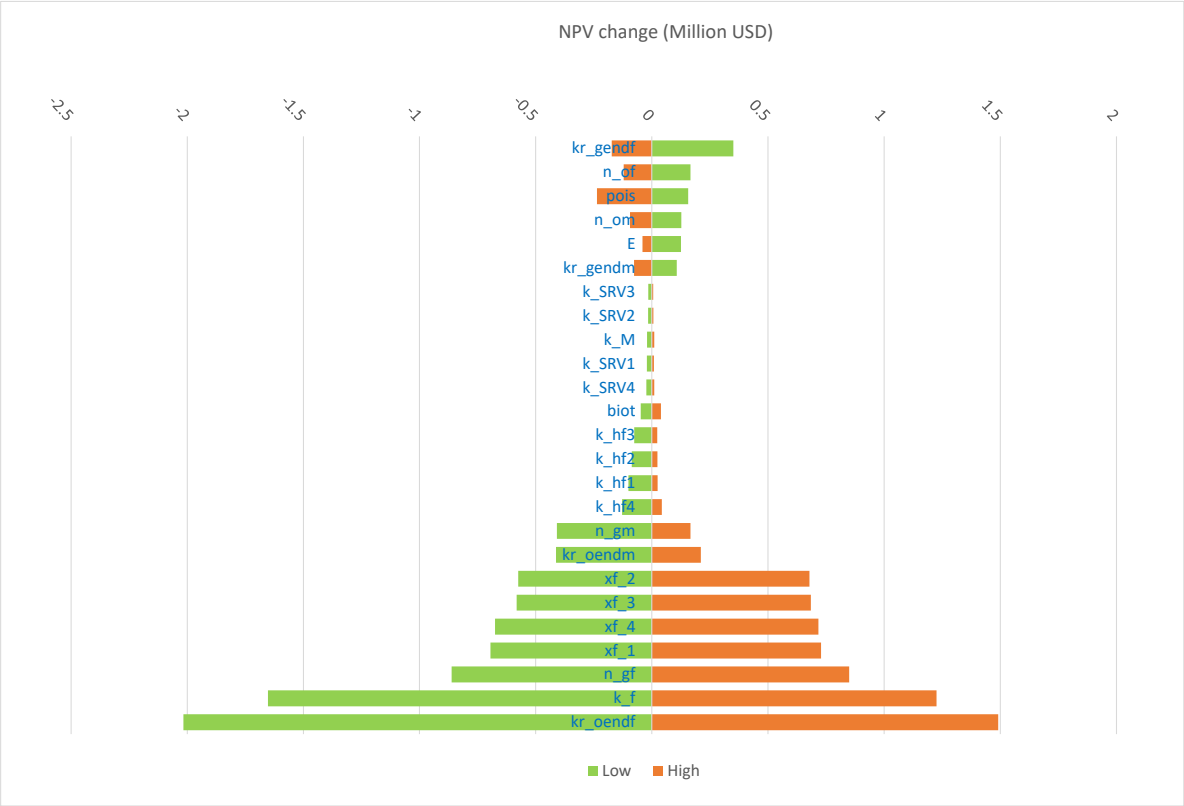


Figure 5.37: Tornado chart for NPV with respect to reservoir model parameters.

we use a homogeneous matrix and natural fracture permeability field by ignoring the spatial correlation and generate matrix and fracture permeabilities from their uncorrelated log-normal distributions with specified mean and variance given in Table 5.8. Then, we assign the same permeability withdrawn from the distribution to each gridblock in the simulation model. So, for our case here, with this dimensionality reduction, we consider a vector of uncertain model parameters including single matrix and fracture permeability of the SRV region at each stage, fracture half-length at each stage, oil relative permeability at both matrix and natural fracture region, etc. Thus, the size of the vector $\mathbf{m}$ is 25: $\mathbf{m} = [k_M, k_f, 4\times k_{hf}, 4 \times k_{SRV}, 4 \times x_f, 2 \times k_{ro,end}, 2 \times k_{rg,end}, 2 \times n_o, 2 \times n_g, E, \nu, \alpha]^T$ and the size of the design vector $\mathbf{u}$ is 25. So, the total size of the feature space is 50. Since the size of our feature space after this dimensionality reduction becomes 50, we did not further reduce the size of the feature space, though the tornado chart of the NPV (Fig. 5.37) suggests that we could do a further reduction in our future space since NPV is not much sensitive to some other

reservoir parameters such as matrix permeability (k_M) and secondary permeability (k_{SRV}) of the hydraulic fractures of each stage. Nevertheless, we construct a proxy model for a future space size of 50 for NPV based on LS-SVR or GPR using training data, $\{[\mathbf{u}_k^T, \mathbf{m}_i^T], J(\mathbf{u}_k, \mathbf{m}_i)\}$ for $k = 1, 2, ..., N_{tr}$ and $i = 1, 2, ..., N_e$.

$N_s = 200$ samples of design parameters generated for $N_e = 20$ reservoir models (i.e., ten samples of design variables are paired with each reservoir model). It provided a sufficiently accurate initial proxy model. So, the input set consists of the 200 suites of paired reservoir models and design variables, whereas the output set is composed of the NPVs predicted by running the reservoir simulator with the 200 suites of paired reservoir models and design variables. 140 samples of this set are used for training, and 60 samples for the test (see Fig. 5.38). Then using the trained LS-SVR or GPR proxy model, we

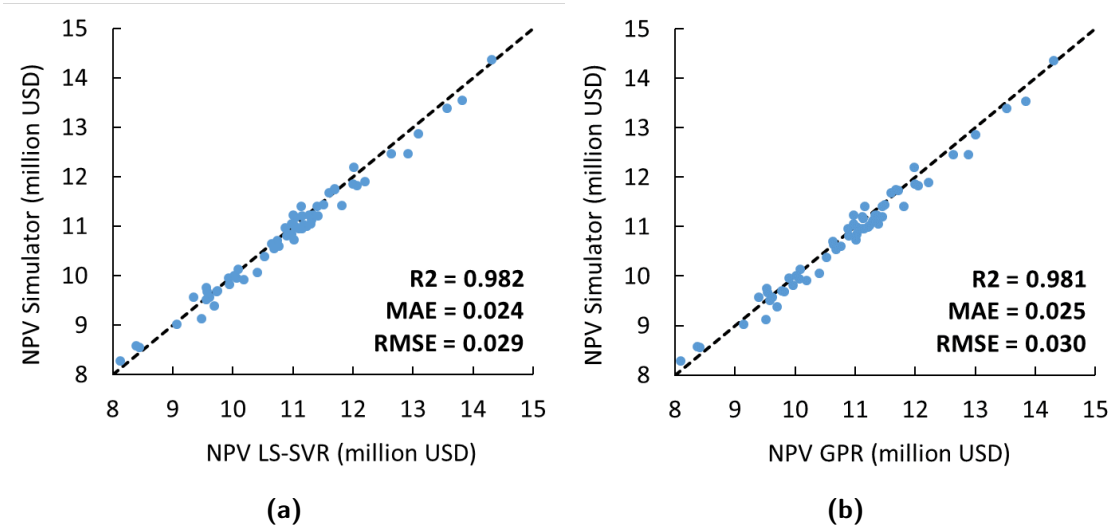


Figure 5.38: Testing of the LS-SVR (a) and GPR (b) models; 140 training samples and 60 test samples.

performed robust optimization based on Eq. 3.6 to maximize the maximum expectation of the NPV over 20 plausible reservoir models. Fig. 5.39 shows that we achieved the maximum NPVs in convergence after 10 iterations for the LS-SVR model and 13 iterations for the GPR model. As the numerical optimization method, as mentioned earlier, we use the ss-cc-StoSAG optimization method, simply called StoSAG here. The number of perturbations (N_p) was chosen to be equal to 2.

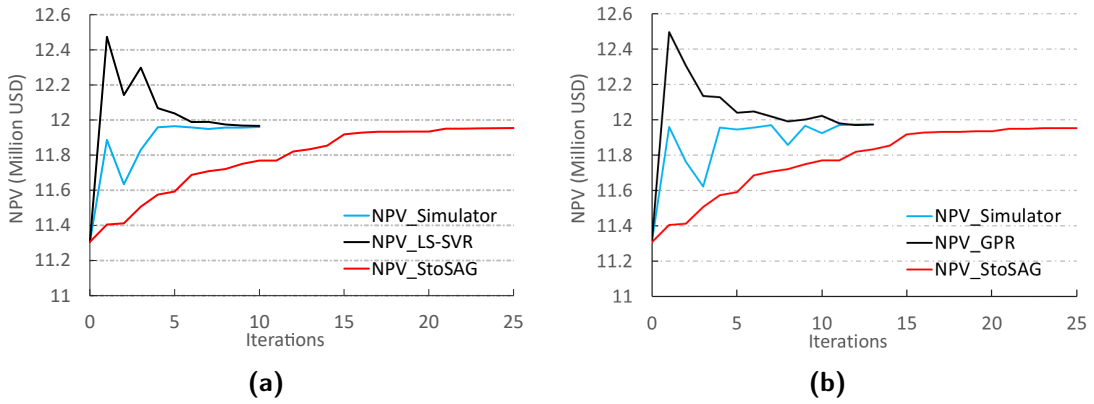


Figure 5.39: Maximum NPV results for LS-SVR (a) and GPR (b) for the robust optimization case compared with StoSAG.

As it was in the cycle optimization case, we obtain a slightly lower maximum value of NPV by the StoSAG optimization method than those by the iterative-sampling-refinement optimization methods. As expected, the optimum design variables are different for different optimization methods. It means each method finds a different optimal solution (Fig. 5.40).

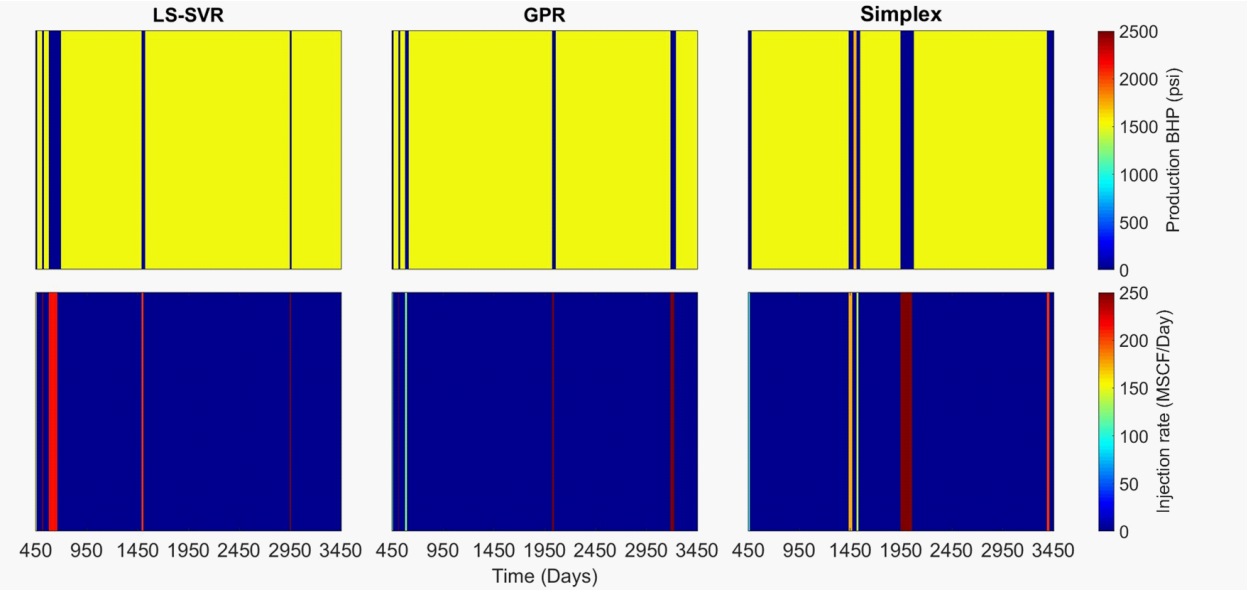


Figure 5.40: Comparison of optimum design variables for different optimization methods; the robust optimization case.

Table 5.18 shows that the optimum soaking fraction (duration of the soaking period/cycle length) is almost zero at each cycle for both models. When we used 140 samples for training, the computational times required to train LS-SVR and GPR models were 4

seconds and 50 seconds, respectively. The number of simulation runs required for LS-SVR is

Table 5.18: Fraction of the optimum soaking [duration of the soaking period / cycle length] for each cycle for different optimization methods; the robust optimization case.

	Duration of soaking period (days)/Cycle length (days)		
Cycle	**LS-SVR**	**GPR**	**StoSAG**
1	0/714	0/35	0/1,018
2	0/35	0/374	0/35
3	0/1,500	0/35	0/412
4	0/716	0/1,058	0/1,500
5	1/35	0/1,497	3/35

higher than GPR for this case, but as expected the StoSAG method requires 6 times more simulation runs (Table 5.19).

Table 5.19: Simulation runs and computational times required for different optimization methods; the robust optimization case.

Optimization methods	**Number of iterations**	**Number of simulations**	**Computational time (mins)**	**Maximum NPV (million USD)**
LS-SVR	10	400	3,201	11.96
GPR	13	460	3,691	11.98
StoSAG	25	2,160	17,280	11.95

The pressure fields for one of the model realizations at the end of HnP by applying the optimal design (optimization) variables generated with LS-SVR, GPR, and StoSAG are shown in Fig. 5.41. The pressure distributions are quite similar for the three methods.

5.3.7 Effect of Random Training/Test Data Split on the Accuracy of Proxy Model

In this section, considering a simple optimization case, we try different randomly split training and test data in the LS-SVR model, and check the model accuracy with R^2, MAE, and MAE accuracy measures. We split data, which are sampled using the LHS, with a ratio of 36/18 training/test split randomly. We have to mention that for the LS-SVR proxy model, 10-fold cross-validation is used for hyperparameter tuning. We create 3 pairs of this kind of randomly split sets. The results of test accuracy are shown in Fig. 5.42. As you

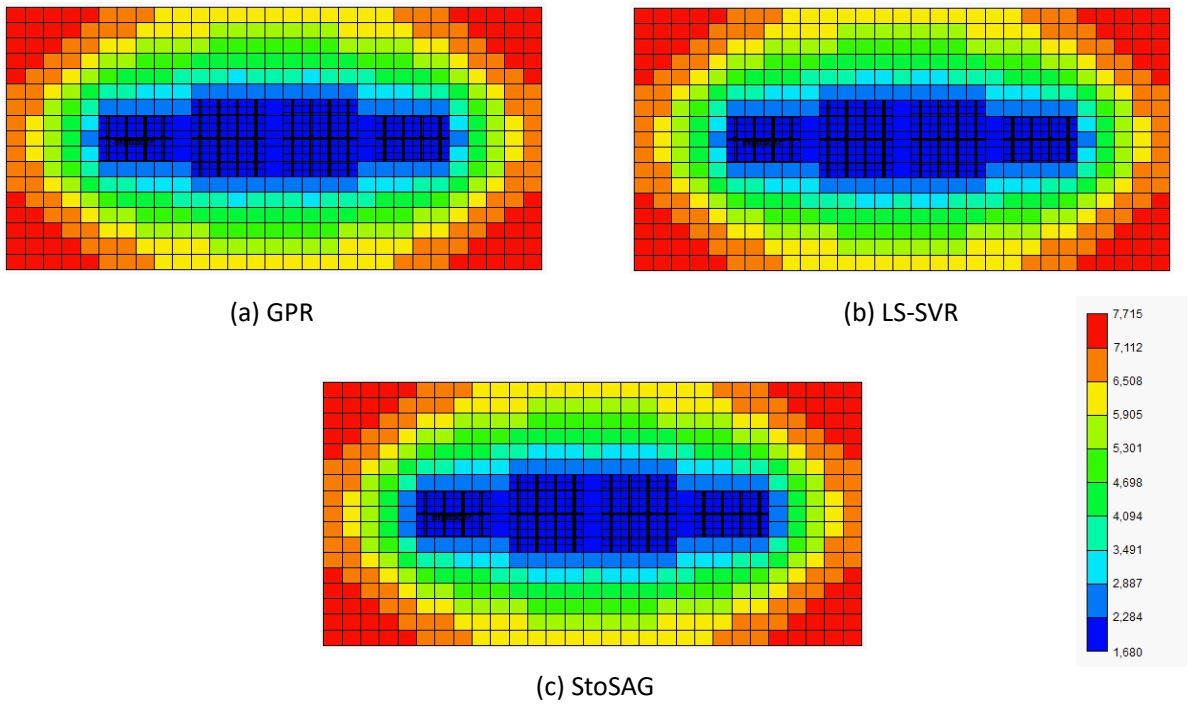
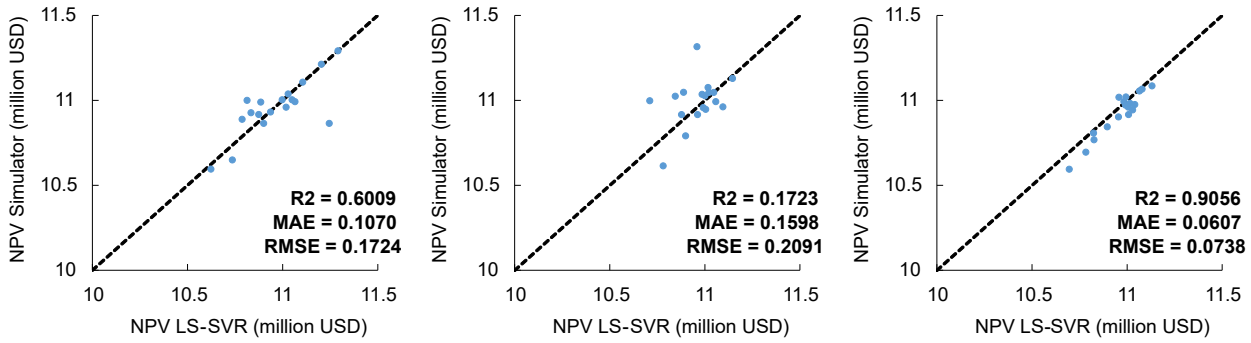


Figure 5.41: Pressure fields for the 1^{st} realization at the end of production life by applying the optimal well controls from L-SVR (a), GPR (b), and StoSAG (c); the robust optimization case. The color bar represents the pressure values.

can see, the accuracy of the LS-SVR proxy model varies depending on which data we choose for training and testing. Therefore, we use a uniform-data-split strategy, as mentioned in Section 4.3 for training our models, both LS-SVR and GPR.

5.3.8 Investigation of Choice of Lower Bound of Production Time Fraction

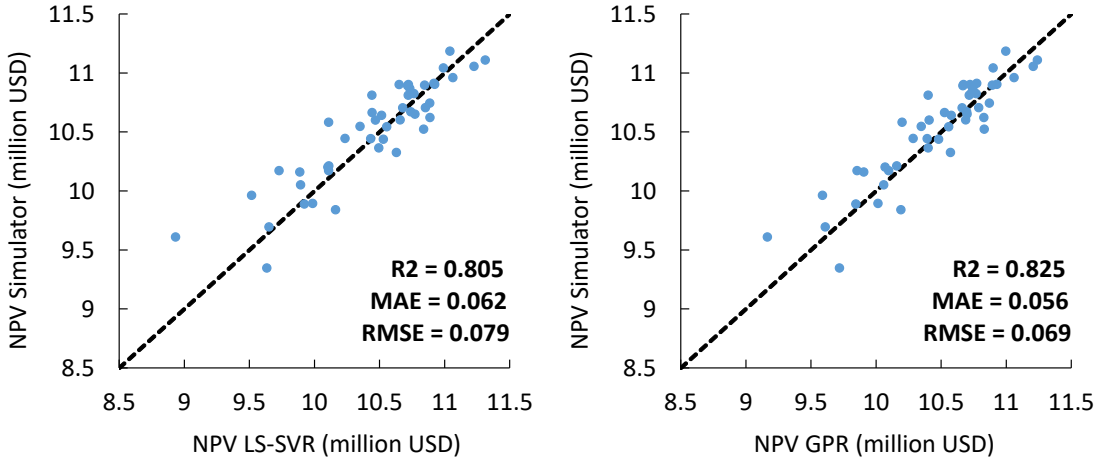
A strategy to sample around the optimum is one of our objectives. One way to achieve this objective is to develop a strategy for the constraints so that we can decrease the computational time. Since our main objective is to decrease the computational time of the optimization process of HnP, choosing this kind of constraint is also one of our main goals in optimization. Therefore, choosing Δt_p^{low} different from zero as a constraint can be applied when we have either a physical or practical reason. Having said that, however, one could set a lower limit for Δt_p^{low} as low as zero, but in this case, the sample space will increase to construct a proxy model. As we show with the specific results below (Fig. 5.43), choosing Δt_p^{low}=0 would not affect the applicability of our optimization methodology. Also, note



(a) set 1 **(b)** set 2 **(c)** set 3

Figure 5.42: Effect of random split on training accuracy. Training/test is 36/18. Simple optimization case.

that having the option Δt_p^{low}=0 means no production period. The models trained with 120 training samples and validated with 48 samples are accurate enough (satisfies our accuracy criteria) as the initial models to be used in the iterative-sampling-refinement method. Our

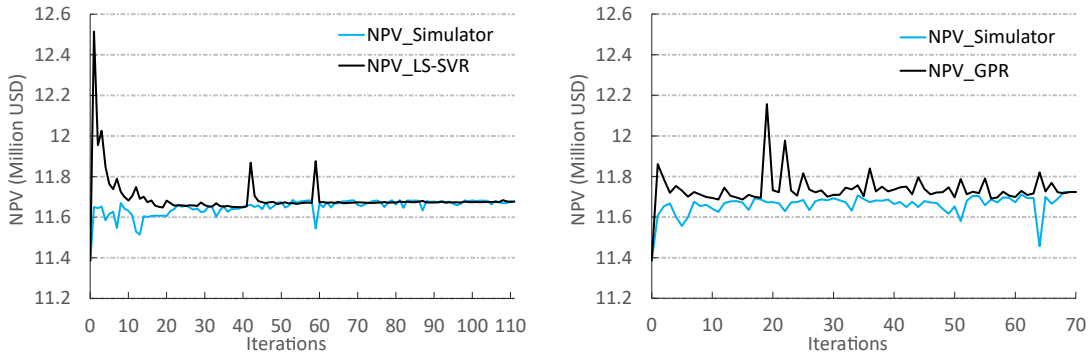


(a) LS-SVR **(b)** GPR

Figure 5.43: Testing of trained models; (a) LS-SVR model trained with 120 samples and tested with 48 samples (b) GPR model trained with 120 samples and tested with 48 samples; the cycle optimization case with Δt_p^{low}=0.

results with Δt_p^{low}=0 show that it takes more computational time and iterations to obtain the optimum with a proxy built with Δt_p^{low}=0 (Fig. 5.44). As shown in Fig. 5.44, the maximum NPVs obtained for LS-SVR and GPR with the iterative-sampling-refinement method are 11.68 million USD and 11.72 million USD, respectively. As can be seen, the LS-SVR yielded

slightly less NPV value compared to the other optimization methods. The LS-SVR required 111 iterations, 279 simulation runs, and 2236 minutes of computational time, while the GPR required 70 iterations, 238 simulation runs, and 1927 minutes of computation time. The maximum NPV values for the LS-SVR and GPR with Δt_p^{low}=0.3 are 11.73 million USD. This NPV value is very similar to those NPV values obtained with Δt_p^{low}=0. However, the number of simulation runs for the LS-SVR and GPR with Δt_p^{low}=0 are 174 and 158, respectively (see Table 5.14), which are much less than those obtained with Δt_p^{low}=0. In



(a) LS-SVR **(b)** GPR

Figure 5.44: NPV vs. iterations obtained using the LS-SVR- and GPR-based iterative-sampling-refinement optimization methods; (a) LS-SVR with 120 training samples, GPR with 120 training samples; the cycle optimization case with Δt_p^{low}=0 compared with the base case.

summary, as can be seen from the results above, we can change the lower bound of the production time fraction and still can apply the iterative-sampling-refinement optimization method to find the optimum.

5.3.9 Uncertainty Assessment of GPR.

In Table 5.20, we present 95% confidence intervals of the maximum NPVs predicted by the GPR for each of the four optimization cases. The results show that as our model (NPVs calculated using a reservoir simulator) becomes more complex by adding more design variables and hence, increasing the size of feature space, the confidence interval of NPV widens. This means that the uncertainty in the maximum NPV predicted increases as we include uncertainty in the geological model as well as increasing the number of design

parameters (increasing the size of feature space). It is worth noting that we determine

Table 5.20: Statistics based on 95% confidence intervals of GPR predictions of maximum NPV for different optimization cases.

Optimization cases	**mean (M USD)**	**std (K USD)**	**lower conf. (M USD)**	**upper conf. (M USD)**	**relative conf. interval (%)**
simple case	11.462	2.880	11.456	11.467	0.050
full case	11.718	4.514	11.709	11.727	0.077
cycle case (trained with 90 data)	11.712	7.892	11.696	11.727	0.134
full-25 case	11.75	34.967	11.686	11.823	0.59
robust	11.972	21.186	11.951	11.993	0.354

the accuracy of an initial GPR proxy model (or any proxy model) on a test (unseen) data set. Its accuracy is quantified by statistical measures such as R^2, MAE, and $RMSE$, for example as illustrated in Fig. 5.38 for the robust optimization case. The accuracy here should not be confused with the predictive variance of the proxy model, which is given by Eq. 4.33, or the variance of the signal (noise-free latent GPR function) given by the first term in the right-hand side of Eq. 4.33. The predictive variance determines the uncertainty (or the confidence interval) in the predicted value of the maximum NPV. It has a complex relationship with the signal variance, the correlation between the pairs of the optimum vector of the design parameters and the training vectors of the design parameters, and noise in observed (simulator) NPV. Our computations show that the variance of the signal, σ_f^2, increases as the size of the design variables (or feature space in general) increases. Although we cannot provide an explicit equation that shows the relationship between the accuracy of the tested proxy model and the predictive variance of the proxy model, they are related to each other by the implicit relationship given by Eq. 4.33. Our results in Table 5.20 show that the variance of the noise-free GPR function is more influential on the predictive variance or equivalently on confidence intervals as the confidence interval widens as the number of design variables (or feature space) increases from the simple optimization case to the robust optimization case. In the 3^{rd} column of the Table 5.20, std represents the standard deviation

of J_{max}; that is the square root of the variance given by Eq. 4.33. Nevertheless, for all optimization cases, the relative percentage confidence interval is less than 0.75%, which indicates an acceptable estimation for NPVs.

CHAPTER 6

PRODUCTION OPTIMIZATION OF THE CO_2 WAG PROBLEM

In this chapter, we perform both approximate gradient optimization and iterative-sampling-refinement optimization on different production optimization cases of the CO_2 water alternating gas injection (WAG) problem. The definition of design variables and derivation of NPV for the WAG problem are given in Section 2.2. Note that the WAG problem, among other optimization problems we consider such as well shutoff and HnP, has the largest number of design variables. The number of design variables changes depending on the optimization strategies considered for the WAG problem. Thus, in this chapter, we also investigate the effect of the number of and the type of the design variables on the production optimization results of the WAG problem. In Section 6.1, first, we discuss the physics of the WAG process in conventional reservoirs, then we explain how the setting of inflow control valves (ICVs) is included mathematically into the simulator (CMG-GEM (2020)). Later, in Section 6.2, we present the optimization results of different optimization cases considered for the WAG problem.

6.1 Physics of WAG Process in Conventional Reservoirs

In Subsection 6.1.1, we briefly discuss mass transport for the WAG process in conventional reservoirs without going into details of derivations since we already showed derivations of general mass transport in Subsection 5.1.1. In Subsection 6.1.2, we show how simulator (we use CMG-GEM (2020)) mathematically handles ICVs.

6.1.1 Mass Transport Mechanism of WAG Process in Conventional Reservoirs

In Subsection 5.1.1, we provided a general formulation of the mass transport (Eq. 5.2). Later, in that subsection, we formulated the mass transport for the miscible CO_2 HnP process in unconventional reservoirs (Eq. 5.6) from the general mass transport equation (Eq.

5.2), considering miscibility and negligible advective term since the reservoir considered for the HnP problem was an ultralow permeable reservoir. However, for the WAG problem, we consider a conventional reservoir. Therefore, for the WAG problem, the advective term in Eq. 5.2 is no longer be negligible. Indeed, in conventional reservoirs, the main mass transport mechanisms are advection and mechanical dispersion terms. The molecular diffusion term can be neglected. Therefore, miscibility for the WAG in conventional reservoirs is achieved by mixing two components in the same phase by mechanical dispersion only since molecular diffusion is negligible. Note that, in the WAG process considered in this study, we alternately inject CO_2 and water. In front, where the injected gas (CO_2) contacts with the reservoir fluid, we achieve miscibility at MMP. However, we cannot achieve miscibility between water and oil.

6.1.2 Mathematical Formulation To Include Inflow Control Valves

In Section 2.2, we defined the design variables, formulated NPV for the general (Eq. 2.36), and simple (Eq. 2.43) cases. In that section, we also discussed how NPV is an implicit function of design variables such as gas injection time fraction, cycle lengths of each well, gas injection rate, water injection rate, and production BHP. However, we did not mention how ICVs are included in the NPV formulation. As we know, in the NPV formulation, we have production flow rate and injection rate terms (see Eqs. 2.36 and 2.43). Those terms are correlated with BHP and pressure at gridblocks intersected by the well. For phase j (phase is gas or water for injectors, and phase is oil, water, or gas for producers), and for the time step n of well k, this correlation is given as follows:

$$q_j^{n,k} = \sum_{l=1} \mathrm{PI}_l^n \lambda_{j,l}^n (p_o^n - p_{wf,l}^n), \tag{6.1}$$

where $q_j^{n,k}$ denotes the flow rate of phase j at the time step n of well k; l denotes the layer l; PI_l^n is the productivity index of the well k at time t_n, at layer l; $\lambda_{j,l}^n$ is the total mobility of phase j in the gridblock of well k at layer l at time t_n; p_o is the oil pressure at that gridblock;

$p^n_{wf,l}$ is the bottom-hole pressure at the perforation of layer l at time t_n and it is function of p_{wf} (BHP), bottom-hole pressure at datum, and specific weight of the wellbore fluid due to gravity. The formulation of productivity index is given as follows:

$$\mathrm{PI}_l = 2\pi\alpha_l k_l h_l \frac{\mathrm{wfrac}_l}{\ln(r_e/r_w) + s_l}, \tag{6.2}$$

where α_l is the completion fraction of the well at gridblock l, and it represents PI multiplier for perforation l; wfrac_l is the fraction of well that is governed by areal geometry at layer l; k_l is the effective permeability in the plane perpendicular to the well direction at layer l; h_l is the thickness of layer l in the well direction; r_e and r_w denote effective radius and wellbore radius, respectively; and s_l is the skin factor of layer l. ICVs are modeled by the PI multipliers, and thus the NPV formulation becomes an implicit function of ICVs (Chen and Reynolds, 2017). Therefore, at each control step of each well and at each layer we have ICV as a design variable.

6.2 Results of Production Optimization of Water Alternating Gas Injection Problem

In this section, we provide our results of GPR- and the LS-SVR-based iterative-sampling-refinement optimization methods on production optimization of the WAG process in comparison with the conventional numerical gradient ascent optimization method. The objective functions for the NPV for the general WAG problem, where gas injection time fractions of injection wells and cycle lengths of production and injection wells are in the set of design variables, and the simple WAG problem, where gas injection time fractions and cycle lengths are no longer design variables, were, previously, given by Eqs. 2.36 and 2.43, respectively.

In the first subsection, first, we introduce reservoir model parameters, and the composition of reservoir fluid considered for the WAG problem. In that subsection, we also introduce the production optimization cases considered for the life-cycle production opti-

mization problem of the WAG process. From Subsections 6.2.2 to 6.2.9, we show the results of the production optimization cases. For each production optimization case, we compare the GPR-, the LS-SVR-based iterative-sampling-refinement optimization methods with the gradient ascent optimization method. We also show the accuracy of the initial proxy models of GPR and the LS-SVR using test data. Besides this comparison, we also compare different optimization cases to investigate the effect of the number and the type of design variables on the production optimization results.

For deterministic optimization cases, the maximum number of simulation runs in StoSAG optimization method is $N_{sim}^{max} = 2000$. However, since for the robust case, StoSAG requires a higher number of simulation runs at each iteration, we choose $N_{sim}^{max} = 6000$ for the robust optimization case; and robust optimization case stop due to a maximum number of simulation runs. Each simulation run requires 1-minute of computational time. Also, we have to note that for the Bayesian model selection process of the training of GPR process, we choose the maximum number of objective function evaluations to be equal to 10. Setting it to a higher number costs a higher computational time. We have tried a different number of objective function evaluations and setting it to a number less than 10 results in the proxy models with poor accuracy. Therefore, based on our observations, 10 seems to be the best number for the maximum number of objective function evaluations. Note that in the GPR process, the objective function that is used in Bayesian model selection to be optimized is the log marginal likelihood function (Eq. C.36).

6.2.1 Reservoir Model and Production Optimization Cases

The fluid composition and the reservoir model are taken from Chen and Reynolds (2017). The injected fluid is 100 percent CO_2. We use the CMG-GEM (2020), a black oil simulator, to simulate the WAG process. There are 7 pseudo components used to describe reservoir fluid. Properties and mole fractions of these components are given in Table 6.1, and BICs of the components are given in Table 6.2. We use the semi-analytical (key-tie lines) method given in WINPROP (2004) to calculate MMP of reservoir fluid with CO_2.

The estimated minimum miscibility pressure (MMP) is 4,150 psi at reservoir temperature, $T = 192.2$ °F.

Table 6.1: Properties of pseudo-components taken after Chen and Reynolds (2017).

Component	Molar fraction	Critical pressure (atm)	Critical temperature (K)	Acentric factor	Molar weight (g/gmol)	Critical volume (L/mol)	Parachor coefficient
CO_2	0	72.8	304	0.225	44	0.094	78
N_2	0.00007	33.5	126	0.04	28	0.0895	41
CH_4	0.0787792	45.4	191	0.008	16	0.099	77
C2 to iC4	0.0246498	41.2	370	0.147	45.3	0.208	150
iC5 to C6	0.199058	33.2	474	0.241	76.1	0.317	232
C7 to C18	0.487665	23.5	643	0.476	149	0.566	416
C19 to C43	0.209778	12.7	779	1.04	378	1.35	884

Table 6.2: Binary interaction coefficients (BIC) pseudo-components taken after Chen and Reynolds (2017).

Component	CO_2	N_2	CH_4	C2 to iC4	iC5 to C6	C7 to C18	C19 to C43
CO_2	0	-0.020	0.103	0.132	0.132	0.150	0.150
N_2	-0.020	0	0.031	0.080	0.102	0.120	0.120
CH_4	0.103	0.031	0	0.009	0.022	0.048	0.104
C2 to iC4	0.132	0.080	0.009	0	0.003	0.016	0.056
iC5 to C6	0.132	0.102	0.022	0.003	0	0.005	0.034
C7 to C18	0.150	0.120	0.048	0.016	0.005	0	0.012
C19 to C43	0.150	0.120	0.104	0.056	0.034	0.012	0

The reservoir model studied in this research is a three-layer, square-shaped, channelized reservoir. We have 4 injectors and 9 producers. All layers have a heterogeneous permeability distribution. They all have the same heterogeneous distribution, but the permeability field of layer 2 is equal to 0.6 times of permeability field of layer 1, and the permeability field of layer 3 is equal to 0.3 times of permeability field of layer 1. For all layers, the vertical permeability field is 0.1 times the horizontal permeability field. The porosity field of all layers is homogeneous and equal to 0.2. The initial reservoir pressure is 4,500 psi. All other reservoir parameters, as well as rock and fluid properties, are provided in Table 6.3. The reservoir is modeled using $25 \times 25 \times 3$ grids in CMG-GEM (2020). The horizontal permeability field of the first layer is shown in Fig. 6.1, as well as the areal gridding. The economical values are

Table 6.3: Input values of deterministic reservoir parameters used in the synthetic reservoir model.

Parameter	Value	Unit
Dimension of the model	2,500x2,500x90	ft^3
Depth	9,800	ft
Initial reservoir pressure	4,500	psi
Reservoir temperature	192.2	°F
S_{or}, residual oil saturation	0.3	fraction
S_{gc}, critical gas saturation	0.05	fraction
S_{wir}, irreducible water saturation	0.2	fraction
Rock compressibility at 3,300 psi	6.1e-5	1/psi
Water compressibility at 14.7 psi	3.3e-6	1/psi
Water viscosity at 14.7 psi	0.7	cP
Porosity	0.2	fraction

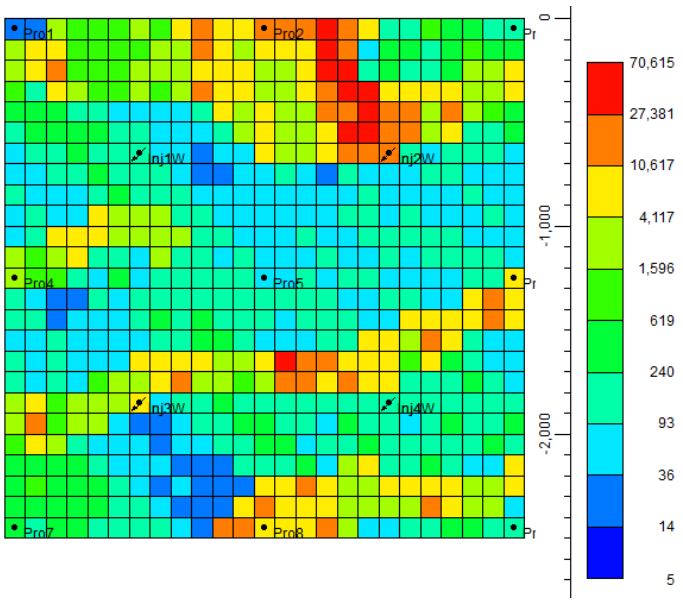


Figure 6.1: Areal grid system and horizontal permeability field of first layer (color bar represents permeability in mD).

provided in Table 6.4.

Depending on the number of injection cycles considered, we treat BHPs as design variables or fixed, ICVs as design variables or fixed, cycle lengths of injection and production wells as design variables or fixed, and/or gas injection time fractions as design variables or fixed. Table 6.5 shows the cases considered. For the cases where we fix BHPs, we fix their values to 1000 psi. We fix ICVs to their upper 1 for the cases in which ICVs are no longer design variables. For the cases where cycle lengths and gas injection time fractions are fixed,

Table 6.4: Economical constants used for the NPV function (Eqs. 2.36 and 2.43).

Constants	**Value**	**Unit**
b	0.1	fraction
r_o	60	\$/STB
c_{wp}	5	\$/STB
c_{wi}	5	\$/STB
c_{gi}	1.5	\$/MSCF

we divide cycle lengths uniformly and we fix gas injection time fraction to 0.5. Therefore, we use Eq. 2.43 as an objective function for the production optimization. Note that for all production optimization cases of the WAG problem considered in this work, $N_c = 2N_{ic}$. Thus, in Table 6.5, we only give the number of injection cycles, N_{ic}, and also only injection cycle length, Δt^n.

Table 6.5: Cases considered for the optimization applications of the WAG problem.

Cases	N_{ic}	Fixed BHP (psi)	Fixed ICV	Fixed Δt^n (Days)	Fixed $\widehat{\Delta t}_g^n$	Number of design variables
Case-8-highBHP	8	NA	1	360	0.5	208
Case-2-highBHP	2	NA	1	1440	0.5	52
Case-8-constBHP	8	1000	1	360	0.5	64
Case-8-lowBHP	8	NA	1	360	0.5	208
Case-8-rate	8	NA	1	360	0.5	208
Case-8-lowBHP-ICV	8	NA	NA	360	0.5	832
Case-8-lowBHP-ICV-robust	8	NA	NA	360	0.5	832
Case-8-lowBHP-cycle	8	NA	1	NA	NA	416

The case names are described in Table 6.5 as follows: the first extension after the *Case* indicates the number of injection cycles; the second extension gives information whether bounds of production BHPs are high (2500 to 4500 psi) or low (500 to 1000 psi) or constant (fixed to 1000 psi); the third extension indicates whether we include ICV as design variable or not; finally, the last extension implies if optimization case considers cycle lengths and gas injection time fractions as design variables. For instance, *Case-8-lowBHP-cycle* implies the

case where we have 8 injection cycles (which means we have 16 production cycles), bounds of production BHPs are low (500 to 1000 psi), cycle lengths, as well as gas injection time fractions, are design variables and ICVs are all fixed to 1. *Case-8-rate* is the only case where instead of production BHP we use production flow rate as a design variable. The order of the design variables inside the design vector is consistent with the ordering given in Eq. 2.38. The number of design variables for the cases where ICVs, cycle lengths, and gas injection time fractions are fixed, is given by Eq. 2.45; for the cases where only cycle lengths and gas injection time fractions are fixed, it is given by Eq. 2.46; for the cases where only ICVs are fixed, it is given by Eq. 2.37; for the cases where ICVs, cycle lengths, and gas injection time fractions are all design variables (not fixed), it is given by Eq. 2.41; and finally for *Case-8-constBHP*, where production BHPs are fixed as well as ICVs, cycle lengths, and gas injection time fractions, the number of design variables is calculated as follows:

$$N_u = N_t \cdot N_I. \tag{6.3}$$

The bound constraints used for the design variables are given in Table 6.6. Note that for all cases these bounds are the same. However, lower and upper bounds of production BHPs can change depending on the case considered as explained in the previous paragraph. For *Case-8-rate*, upper and lower bounds for production flow rates are 0 and 1000 STB/D.

Table 6.6: Bound constraints for the design variables for $n = 1 : N_c$ for production wells and $n = 1 : N_{ic}$ for injection wells, $m = 1 : N_I$ and $k = 1 : N_P$ (WAG problem).

Design variables	lower bound	upper bound	unit
$q_{g,i}^{n,m}$	0	20	MMScf/D
$q_{w,i}^{n,m}$	0	4,000	STB/Day
$\widehat{\Delta t}_g^{n,m}$	0	1	fraction
$\Delta t^{n,m}$	36	720	Days
$\Delta t^{n,k}$	36	720	Days
ICV	0	1	fraction

For the StoSAG optimization of the simple optimization cases of the WAG problem,

where cycle lengths and gas injection time fractions are not design variables (see the objective function defined by Eq. 2.43), the lag distance in time (also known as correlation length) is 720 days for the covariance used in StoSAG gradient approximation calculation. We use a spherical covariance model (see Eq. 3.5) as mentioned in Section 3.1 and Section 3.2. The variance is equal to 0.01. For the deterministic optimization cases, N_p changes its value between 10 and 15 depending on the cases while for the robust optimization case it is equal to 2 (see Eq. 3.12). However, for the cases where the cycle lengths and gas injection time fractions are also design variables, we simply use a diagonal covariance with variance equal to 0.01 since the control time steps are not known anymore (they are also part of optimization).

In the next subsections, we will see optimization results of different optimization cases for the WAG problem. The reasons for considering the cases given in Table 6.5 will be obvious as we examine the results. For all of the production optimization cases, we applied the iterative-sampling-refinement optimization method as well as the StoSAG optimization method and compared their results. Also, we investigate the effect of training size on the optimization results of the iterative-sampling-refinement optimization method for different optimization cases of the WAG problem. Note that for all the optimization cases (robust and deterministic optimization cases) considered below, the StoSAG optimization method is converged because it reached the maximum number of simulation runs. Considering that for the WAG problem each simulation run takes 1 minute, for deterministic optimization cases, computational times for StoSAG optimization are 2000 minutes and for robust optimization case, it takes 6000 minutes.

In Subsection 6.2.9, we also summarize the computational times and maximum NPVs obtained using LS-SVR- and GPR-based iterative-sampling-refinement optimization methods as well as StoSAG optimization method for all optimization cases considered for the WAG problem.

6.2.2 Case-8-highBHP

This case has 208 design variables (Table 6.5). To train the initial LS-SVR and GPR

proxy models we use 1000 training data points and 500 testing data points. For the LS-SVR we use 100-fold cross-validation for the hyperparameter optimization process. For the initial GPR model, the Bayesian model selection determined the rational quadratic kernel function. The accuracies of the initial proxy models on test data are shown in Fig. 6.2. The results

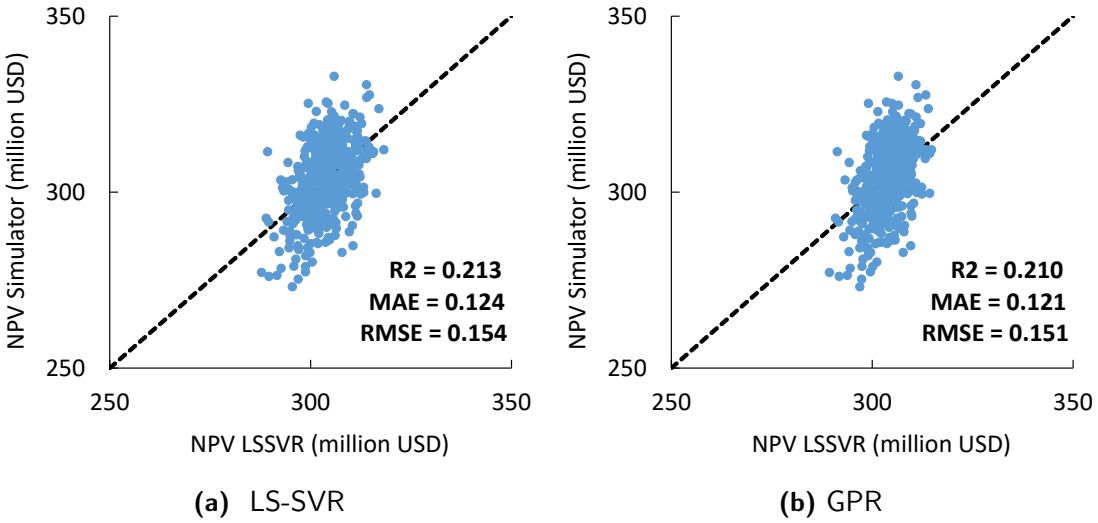
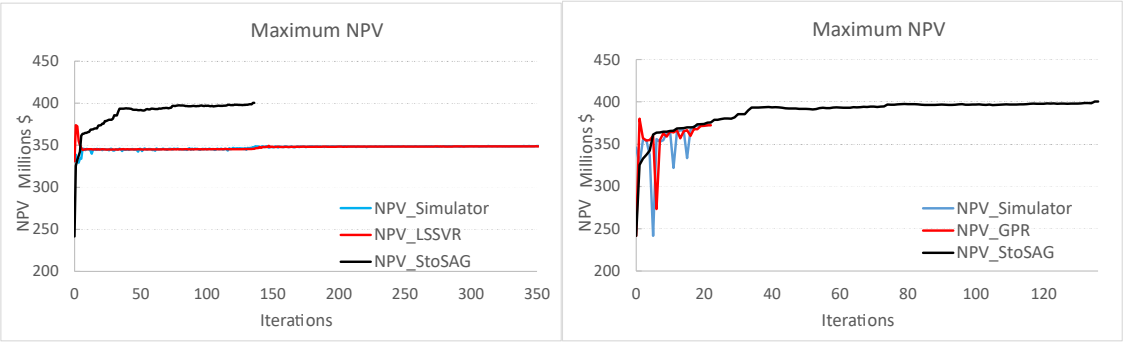


Figure 6.2: Test of the LS-SVR and GPR proxy models for Case-8-highBHP.

show that we do not have an accurate GPR and the LS-SVR proxy models even with a large amount of data. We do not want to choose a data set larger than this since it will not be efficient, especially when considering that GPR requires a large computational time for training with a large data set. There should be other reasons for the accuracy to be low. In the next cases, we investigate this. Even though the accuracy is very low, we would like to see if the optimization results of the iterative-sampling-refinement optimization method change significantly with these initial models. The computational times for the training of the LS-SVR and GPR proxy models for the size of the design variable of 208 and the training size of 1000 are 15 and 920 seconds, respectively. We have to mention that at each iteration of the iterative-sampling-refinement optimization method, we train the proxy model. Therefore, at each iteration, we spent computational time for training the proxy model. Actually, at each iteration, the number of training data is one data more than the size of the training data at the previous iteration. However, computational time approximately is the same at every

iteration because additional computational time due to added one new training point (which is the optimum point from the previous iteration) is negligible. Fig. 6.3 shows optimization results of *Case-8-highBHP* for the iterative-sampling-refinement optimization and StoSAG optimization methods. For StoSAG, the number of perturbations is N_p=10. The results



(a) LS-SVR **(b)** GPR

Figure 6.3: NPV vs. iterations obtained using the LS-SVR and GPR optimization methods for Case-8-highBHP.

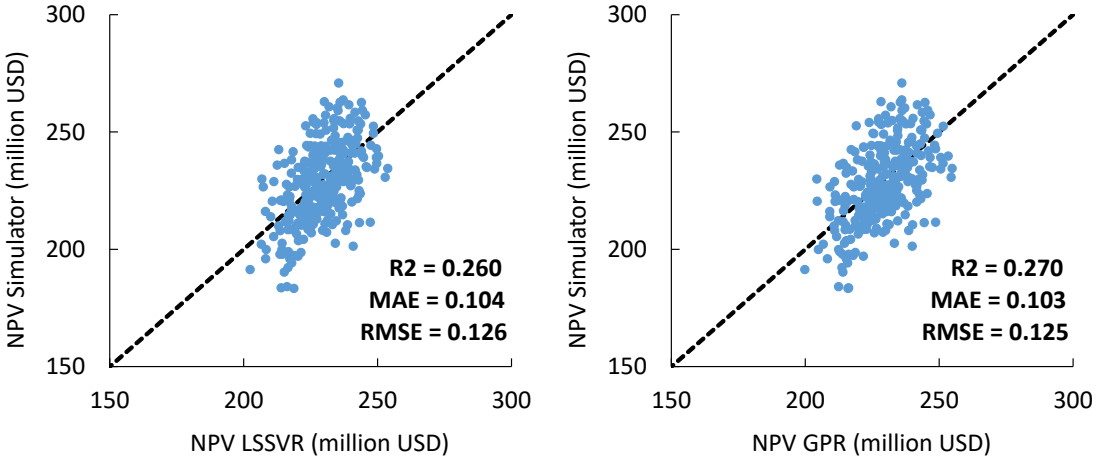
are poor for both the LS-SVR and GPR-based iterative-sampling-refinement optimization methods. The poor initial models give lower maximum NPVs for the proxy-based models than that of the StoSAG optimization method. the LS-SVR required 351 iterations, whereas GPR required 22 iterations to converge.

There could be several reasons for the poor accuracy of the proxy models such as the small training size, a large number of design variables, and/or poorly selected training data set. We think that the training size of 1000 is big enough for the model to be accurate. Therefore, we look at other reasons for this poor accuracy. First, we investigate the effect of the number of design variables on the accuracy of the LS-SVR and GPR proxy models, considering *Case-2-highBHP*, where the number of design variables is 5 times smaller than that of *Case-8-highBHP*.

6.2.3 Case-2-highBHP

In this case, we consider 2 injection cycles only so that we have fewer design variables. This case has 52 design variables (Table 6.5). To train the initial LS-SVR and GPR proxy

models, we use 600 training data points and 300 testing data points. The accuracies of the initial proxy models on the test data are given in Fig. 6.4. The results show that even though

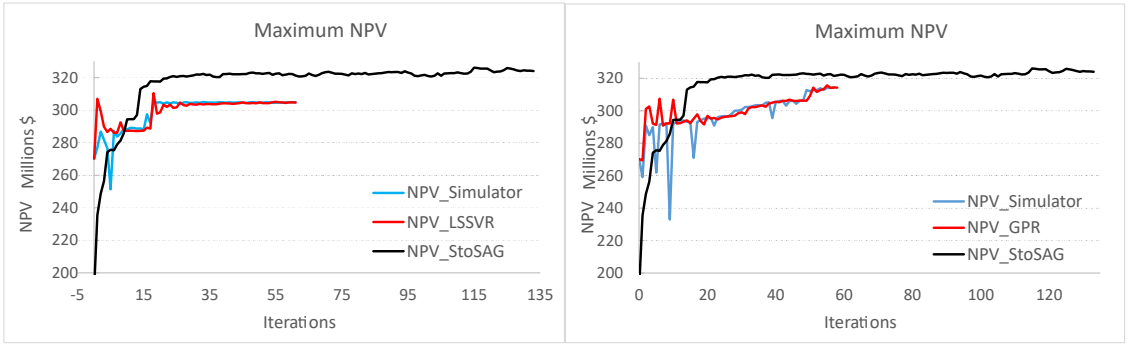


(a) LS-SVR **(b)** GPR

Figure 6.4: Test of the LS-SVR and GPR proxy models for Case-2-highBHP.

the number of design variables is quite small, we still have poor accuracy. For the kernel-based ML models, a training size of 600 for the problem having 52 design variables is more than enough. Computational times for the training of the LS-SVR and GPR proxy models for the size of the design variable of 52, and for the training size of 600 are 6 and 159 seconds, respectively. Before going to investigate other reasons for the poor accuracy of the model, namely poor training and validation data, let us look at the optimization results of this case (Fig. 6.5). For StoSAG, the number of perturbations is N_p=10. The LS-SVR required 61 iterations, whereas GPR required 58 iterations for the iterative-sampling-refinement optimization algorithm to converge. From the optimization results, we observe that again the LS-SVR- and GPR- based iterative-sampling-refinement optimization methods give smaller maximum NPVs than that of StoSAG optimization methods. However, these optimization results are closer to the StoSAG results than it was in the previous case. We also observe that we obtain lower NPV when we consider 2 injection cycles (Fig. 6.5) than that in the case of 8 injection cycles (Fig. 6.3).

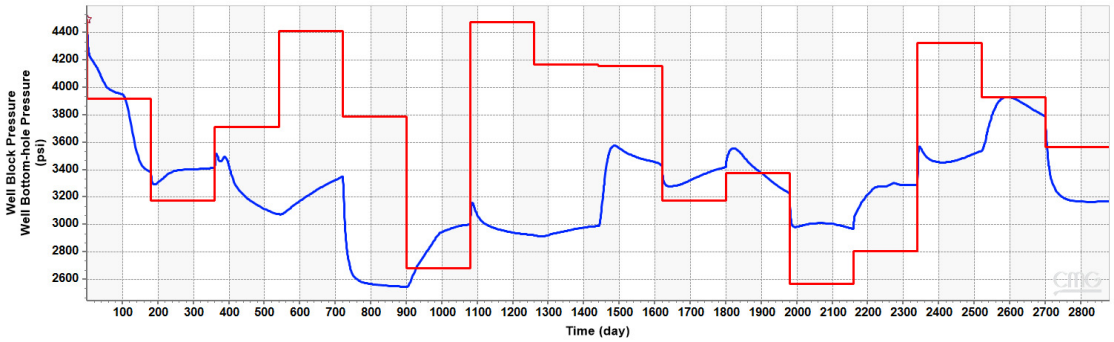
We expected higher accuracies of both the LS-SVR and GPR models and better opti-



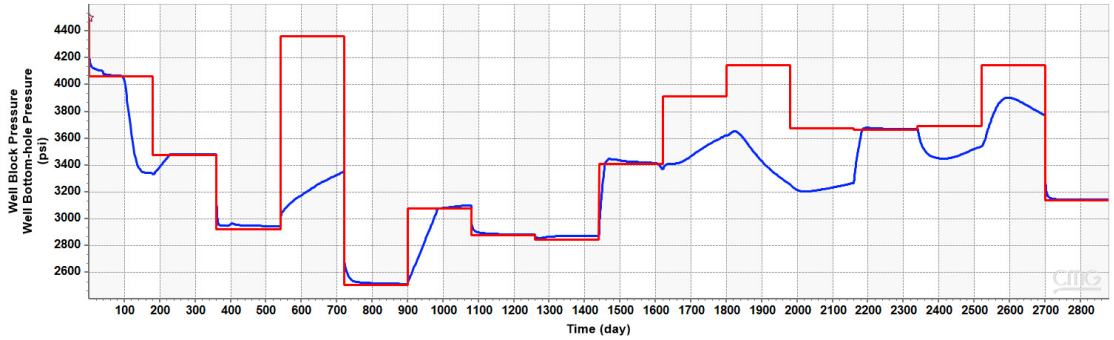
(a) LS-SVR **(b)** GPR

Figure 6.5: NPV vs. iterations obtained using the LS-SVR and GPR optimization methods for Case-2-highBHP.

mization results since we decreased the number of design variables. Therefore, we investigate another reason for poor accuracy which is the poor data set. To investigate whether our data is poor or not, we consider the previous case, *Case-8-highBHP* and check the production BHPs if they really affect the NPV as a design variable or not, that is if they are important design variables of redundant. To investigate it, we check the difference between BHPs and well-block pressures. The reason for doing this is because if BHPs are higher than well-block pressures from the beginning of the control time, their change will not affect NPV since the flow rate will be zero. When we checked our training data, we observed that at most of the data points, producing BHPs are redundant design variables due to the reason mentioned earlier. For the purpose of illustration, in Fig. 6.6, we show producing BHPs and well-block pressures for production wells "Pro1" and "Pro2" of one of the data points at *Case-8-highBHP*. The results show that almost at all control time steps of production wells, BHP is higher than the well-block pressure, which means we do not have any production. Therefore, BHPs become redundant features for the proxy models in the sense that their values do not have an effect on the values of NPV objective function in our data. Note that this does not mean that NPV does not have sensitivity to BHPs. This basically means that NPV values of the data we have for training and testing of the proxy models do not show sensitivity with respect to most of the BHP values in that data. This explains the poor accuracies of the LS-SVR and GPR-based optimizations for *Case-8-highBHP* (Fig. 6.2).



(a) Pro1



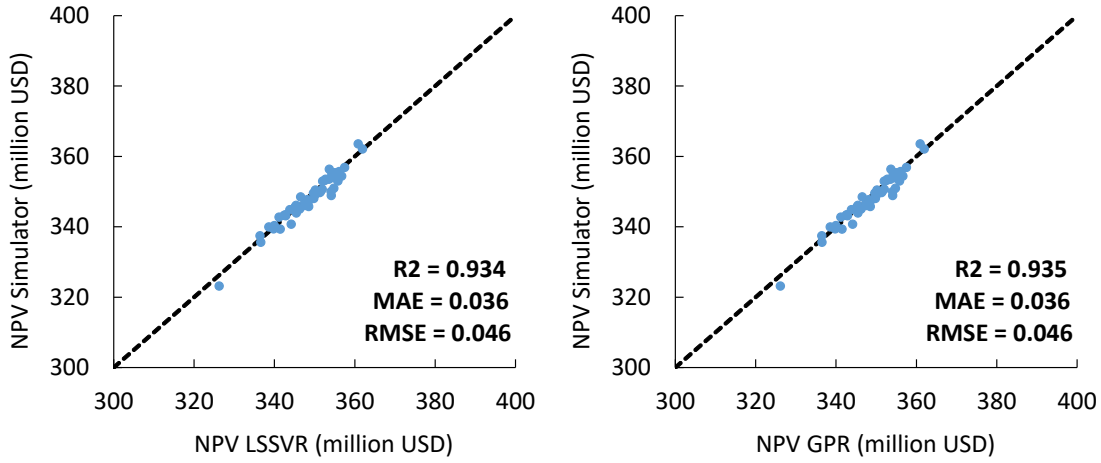
(b) Pro2

Figure 6.6: BHPs (red) and well-block pressures (blue) at different control time steps of production wells "Pro1" and "Pro2" at Case-8-highBHP. This schedule is taken from one of the training data point. We observed the similar behaviour (schedule) at most of the data points and for almost all production wells.

We further confirm this when we consider *Case-8-constBHP* in which BHPs are no longer design variables and fixed to 1000 psi so that we can have production at almost all well-block pressure values. Note that well-block pressures are a function of all design variables.

6.2.4 Case-8-constBHP

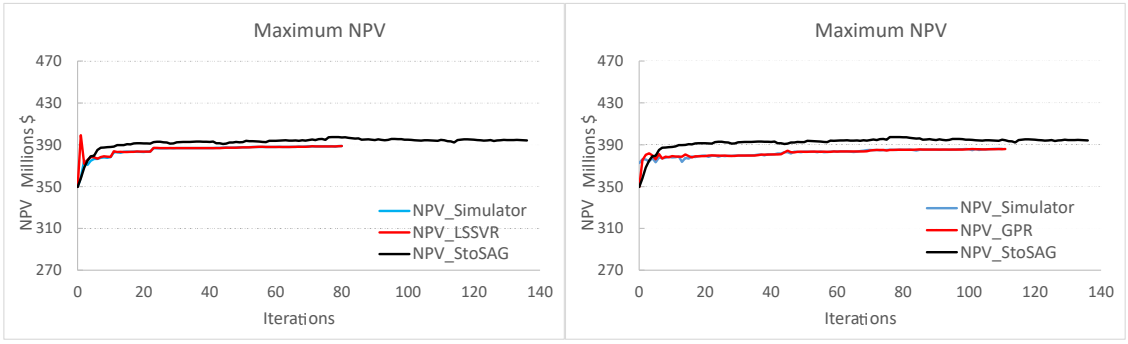
In this case, we consider 8 injection cycles similar to *Case-8-highBHP*, but BHP is constant here. Therefore, the number of design variables for this case is 64 (Table 6.5). To train the initial LS-SVR and GPR proxy models, we use 100 training and 50 testing data points. The accuracies of the initial proxy models on test data are shown in Fig. 6.7. The results show that we have accurate models with 100 training data points which confirms that the poor accuracies in *Case-8-highBHP* and *Case-2-highBHP* were due to poor data resulted from choosing high bounds of BHPs which causes them to be redundant



(a) LS-SVR **(b)** GPR

Figure 6.7: Test of the LS-SVR and GPR proxy models for Case-8-constBHP.

design variables. The computational times for the training of the LS-SVR and GPR proxy models for the size of the design variables of 64 and for the training size of 100 are 5 and 45 seconds, respectively. Fig. 6.8 shows optimization results of *Case-8-constBHP*. For StoSAG, the number of perturbations is N_p=10. The LS-SVR and GPR required 80 and



(a) LS-SVR **(b)** GPR

Figure 6.8: NPV vs. iterations obtained using the LS-SVR and GPR optimization methods for Case-8-constBHP.

111 iterations for the iterative-sampling-refinement optimization algorithm to converge. As expected, the sufficiently accurate initial GPR and LS-SVR proxy models result in good optimization results (Fig. 6.8). Both the LS-SVR- and GPR- based iterative-sampling-refinement optimization methods give approximately the same maximum NPV result, 3.88 ×

10^8 USD, and it is very close to the maximum NPV obtained using StoSAG optimization method, 3.94×10^8 USD.

As we investigated in the previous subsection, poor training data resulted from high bounds of BHPs since BHP at most of the training data points, at almost all of the control time steps is higher than the well-block pressure. By making the BHP constant in this case, we confirmed this. However, this needs further confirmation that without eliminating BHP as a design variable we can still obtain good training data. To do so, we need to consider lower bounds for BHP. In the following section, we consider *Case-8-lowBHP* where lower and higher bounds of BHP are 500 and 1000 psi, respectively.

6.2.5 Case-8-lowBHP

In this case, we consider 8 injection cycles similar to *Case-8-highBHP*, but lower and upper bounds of BHP is lower than *Case-8-highBHP* being 500 and 1000 psi. Therefore, the number of design variables for this case is 208 (Table 6.5). To train the initial LS-SVR and GPR proxy models, we try two different training sizes and compare their accuracy and optimization results: 200 training data, 100 testing data; 320 training data and 160 data. Fig. 6.9 shows the accuracy results of the LS-SVR and GPR models for each training size. As expected, the accuracy improves when the size of the training data increases. The computational times for the training of the LS-SVR and GPR for the size of design variable vector of 208 and for the training size of 200 are 7 and 456 seconds, respectively. However, for a training size of 320, the computational times for training the LS-SVR and GPR models are 9 and 680 seconds, respectively. The computational time of GPR for hyperparameter optimization using the Bayesian model selection depends both on the size of training data as well as the dimension of the design variable vector. In Section 4.2.2 and Section C.2, we discussed how training size affects computational efficiency of the training procedure of GPR and model selection process of GPR. Size of design variable affects computational time as well in operations of vector multiplications and vector-matrix multiplications in Eq. C.36. As we explained (see Section 4.4), the iterative-sampling-refinement optimization method

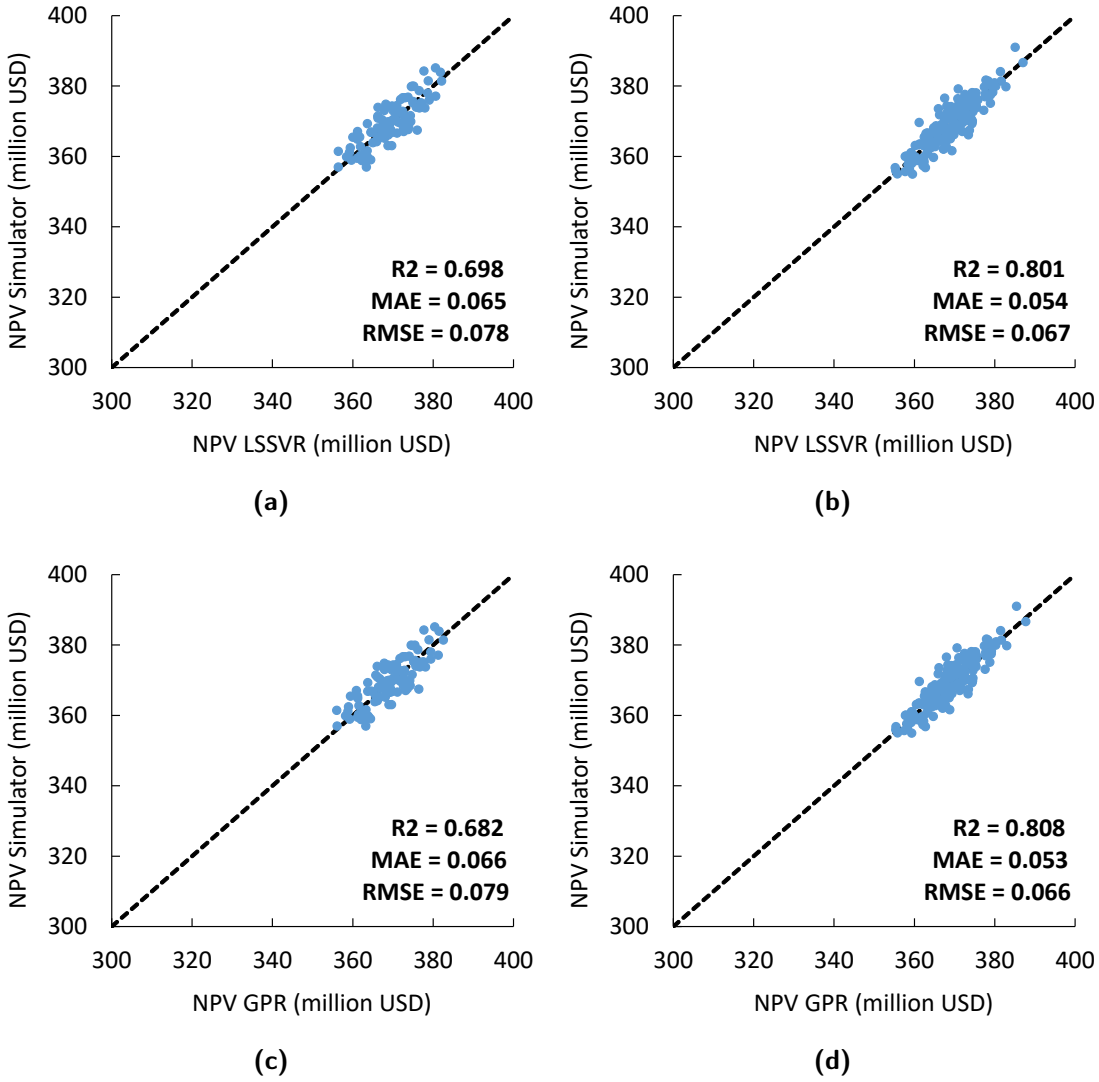


Figure 6.9: Test of the LS-SVR (plots *a*-200 training, 100 test and *b*-320 training, 160 test) and GPR (plots *c*-200 training, 100 test and *d*-320 training, 160 test) proxy models for Case-8-lowBHP.

is an iterative process, and we perform a training procedure at each iteration by adding a data of NPV computed from the simulator. Therefore, in the case of the GPR used as a proxy model, with the Bayesian model selection, the kernel function can be different at each iteration. Our observations showed that, for most of the optimization cases done in the previous subsections, and at most of the iterations, the kernel model was found to be *rational quadratic kernel*. Eq. 6.4 shows the formulation of *rational quadratic kernel*:

$$K(\bar{\mathbf{u}}_i, \bar{\mathbf{u}}_j) = \sigma^2 \left(1 + \frac{||\bar{\mathbf{u}}_i - \bar{\mathbf{u}}_j||^2}{2\alpha l^2}\right)^{-\alpha}, \tag{6.4}$$

where σ^2 is amplitude, l is the lengthscale, and α is the scale-mixture (α>0). By fixing kernel function and optimizing only its hyperparameters, we can decrease computational time. In the previous case (*Case-8-constBHP*), we did not need it since the training of GPR was computationally inexpensive. However, for this case, training GPR without fixing kernel function is very expensive. Therefore, we fix the kernel to the *rational quadratic kernel*. By fixing kernel function, the training process of GPR for the case having design variable with size 208, and training size of 200, takes 26 seconds. However, for the same size of the

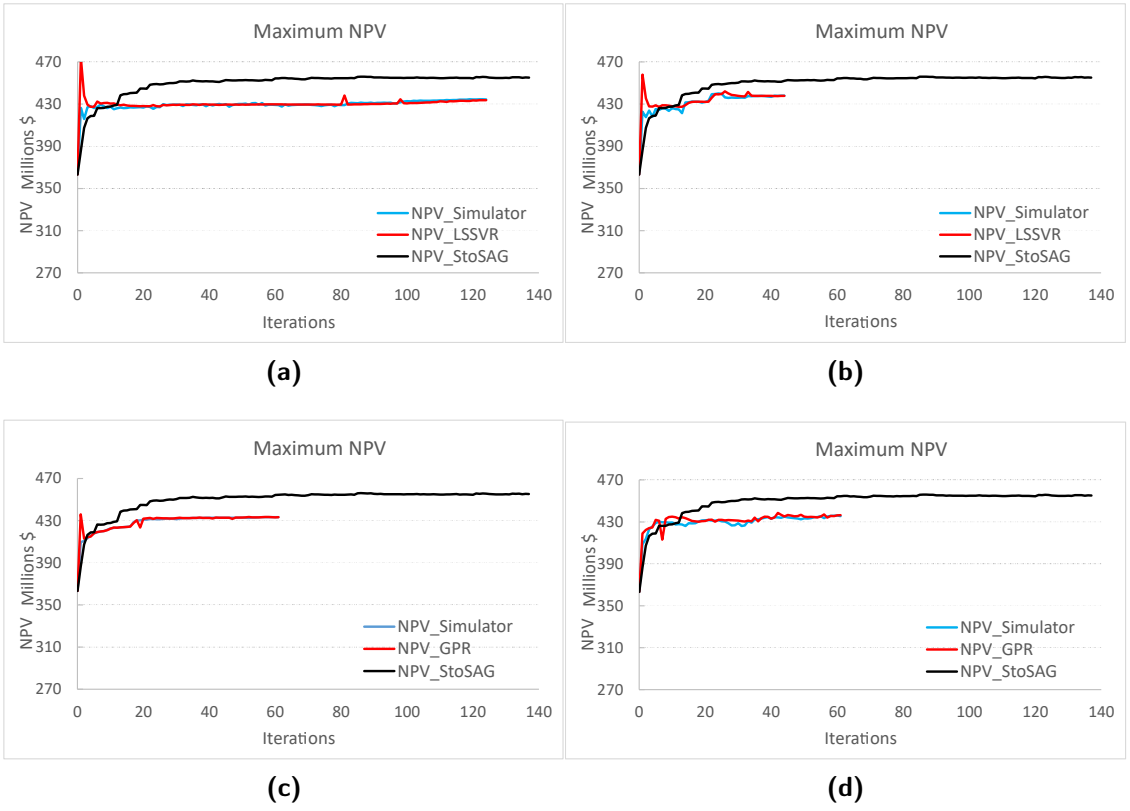


Figure 6.10: NPV vs. iterations obtained using the LS-SVR (plots *a*-200 training, 100 test and *b*-320 training, 160 test) and GPR (plots *c*-200 training, 100 test and *d*-320 training, 160 test) optimization methods for Case-8-lowBHP.

design variable and training size of 320, the computational time for training of GPR is 38 seconds. Computational times for training of GPR fixing kernel function in case of both 200 training data and 320 training data are good enough to start our iterative-sampling-refinement optimization with them. We use these initial models in the iterative-sampling-refinement optimization method to maximize NPV and compare it with StoSAG (Fig. 6.10).

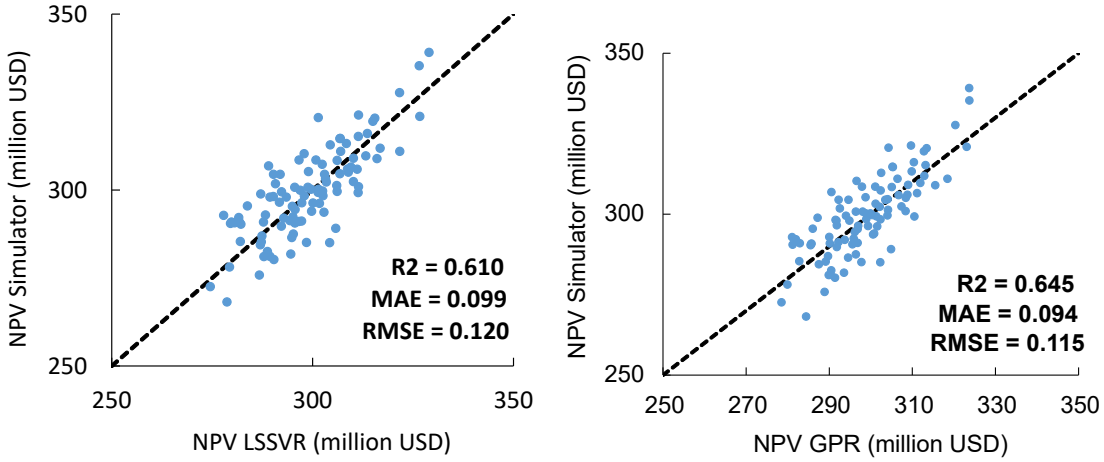
For StoSAG, the number of perturbations is N_p=10.

The LS-SVR model trained with 200 data points required 63 iterations to converge, whereas the LS-SVR model trained with 320 training data converged in 44 iterations. Both the GPR proxy models trained with 200 and 320 training data points required 61 iterations to converge. Maximum NPV results show that starting with the initial proxy with higher accuracy gives us a higher maximum NPV result. With StoSAG we obtain maximum NPV to be equal to 4.55×10^8 USD. The LS-SVR and GPR, trained with 200 data points, as an initial model in optimization result in a maximum of NPV equal to 4.29×10^8 USD and 4.33×10^8, respectively, whereas the optimization result of the iterative-sampling-refinement optimization method with the initial models of both the LS-SVR and GPR trained with 320 training data points is 4.38×10^8, which is 3.7% less than optimization result of StoSAG.

As can be seen, the accuracy and optimization results of the LS-SVR- and GPR-based iterative-sampling-refinement optimization methods, when training data are good, perform much better. And in our problems, the reason for training data to be poor is due to higher lower and upper bounds of BHPs. The reason we started with higher bounds of BHP was due to the fact that we want to achieve miscibility so that we can achieve a higher NPV. However, the optimization results of StoSAG for *Case-8-lowBHP* and *Case-8-highBHP* show that we achieve a higher NPV when bounds of BHP is between 500 and 1000 psi (*Case-8-lowBHP*). Therefore, for the following cases, we set our lower and upper bounds for BHP to 500 and 1000 psi, respectively.

6.2.6 Case-8-rate

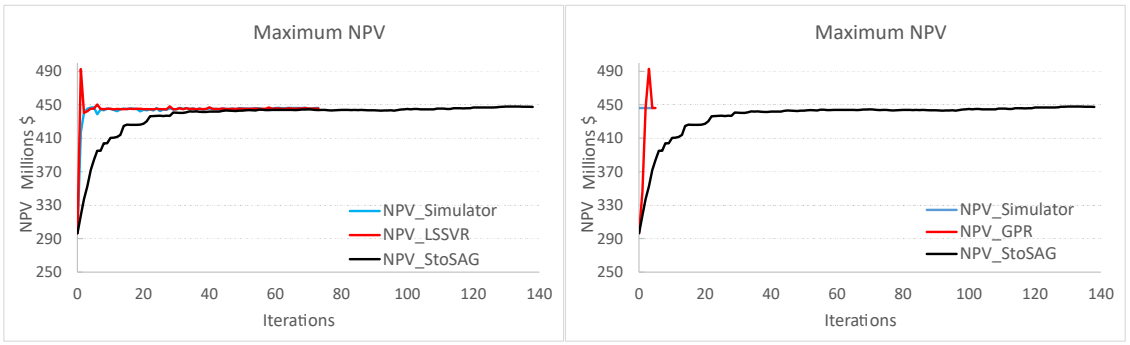
Another way to get good training data instead of considering lower bound for BHP, which can be hard for the facilities in the field, is to use production flow rates instead of BHPs as design variables. Again, in this case, the number of the design variables is 208. To train the initial LS-SVR and GPR proxy models we use 200 training data points and test them with 100 testing data points. Fig. 6.11 shows the accuracy results of the LS-SVR and GPR models. We used the LS-SVR and GPR as the initial proxy model in



(a) LS-SVR **(b)** GPR

Figure 6.11: Test of the LS-SVR and GPR proxy models for Case-8-rate.

the iterative-sampling-refinement optimization method. With both the LS-SVR- and GPR-based iterative-sampling-refinement optimization method, we obtain maximum NPV equal to 4.46×10^8 USD (Fig. 6.12). The LS-SVR requires 73 iterations to converge, whereas GPR requires 5 iterations. For StoSAG optimization we use $N_p = 10$. StoSAG gives us a maximum NPV value of 4.47×10^8 USD. The results show that flow rate works better



(a) LS-SVR **(b)** GPR

Figure 6.12: NPV vs. iterations obtained using the LS-SVR and GPR optimization methods for Case-8-rate.

as a design variable than using BHP in the case of using the iterative-sampling-refinement optimization method. We achieve approximately the same NPV with the iterative-sampling-refinement optimization method as the StoSAG optimization method. However, in *Case-8-lowBHP*, where we used BHP as a design variable, we achieved 3.7% less maximum NPV

value than that in the StoSAG optimization method. Computational times for training of the LS-SVR and GPR were the same as in *Case-8-lowBHP*.

6.2.7 Case-8-lowBHP-ICV

So far, we have looked at the optimization cases in which ICVs were not in the set of design variables. However, ICVs are important design variables, especially when petrophysical properties vary at the perforation of each well. In our reservoir model, the only petro-physical property that varies is the permeability field of the layers. Therefore, in *Case-8-lowBHP-ICV*, we also included ICVs as design variables in addition to production BHPs and water and gas injection rates. Considering that we have a total of 13 wells, and 3 layers and 8 injection wells, 16 half-cycles for each injection wells, and 16 cycles for each production wells, the number of ICVs will be 624. Therefore, the number of design variables is 832. As mentioned in the previous subsection, the lower and upper bounds of BHPs are 500 and 1000 psi, respectively. Since the number of design variables is large, and NPV is sensitive to each of these design variables (this means that we cannot use the dimensionality reduction methods), we need to choose a large training size. Again for this case, we consider two different training sizes; 600 and 800. Fig. 6.13 shows training results of the initial LS-SVR and GPR proxy models trained with these two different training sizes.

For training of the GPR model with the data having training sizes of 600 and 800, for the dimension of the design variable vector of 832, the computational times were 1260 and 2320 seconds, respectively. However, for the LS-SVR, with the data having training sizes of 600 and 800, computational times are 14 and 16 seconds, respectively. However, with fixing the kernel function and only optimizing its hyperparameters, the training process of GPR for the case having a design variable with size 832, and a training size of 600, take 162 seconds. For the same size of the design variable and training size of 800, the computational time for training of GPR is 229 seconds. Note that the GPR model used as the initial model is trained by fixing the kernel model to the *rational quadratic kernel*. Even though we fixed the kernel for this case, training for the GPR model is quite computationally expensive.

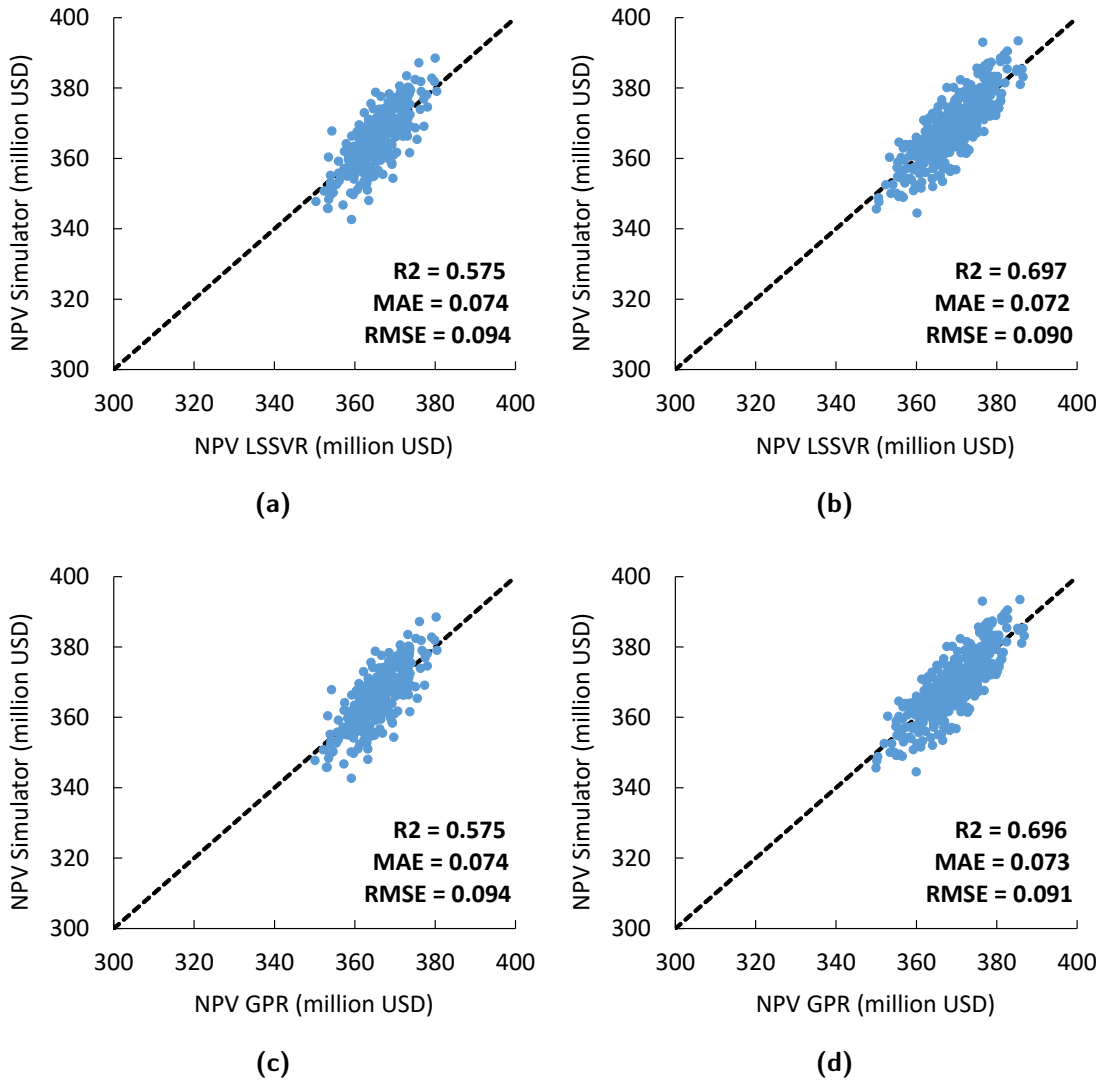
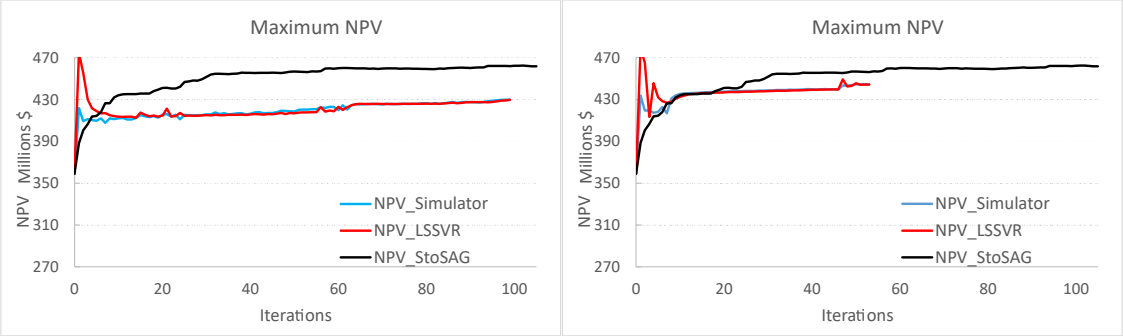


Figure 6.13: Test of the LS-SVR (plots *a*-600 training, 300 test and *b*-800 training, 400 test) and GPR (plots *c*-600 training, 300 test and *d*-800 training, 400 test) proxy models for Case-8-lowBHP-ICV.

Therefore, we show the optimization results when only the LS-SVR trained with 600 and 800 training data points is used as the initial models for the iterative-sampling-refinement optimization method (Fig. 6.14). The results show that increasing training size, which means starting the iterative-sampling-refinement optimization method with a model with sufficient accuracy, yields a higher value of maximum NPV (Fig. 6.14). For StoSAG, the number of perturbations is N_p=15 this time since the number of design variables is higher. The results show that again, increasing training size for training of the initial proxy model (the LS-SVR here) improves the results of optimization. For the case of using the LS-SVR



(a) **(b)**

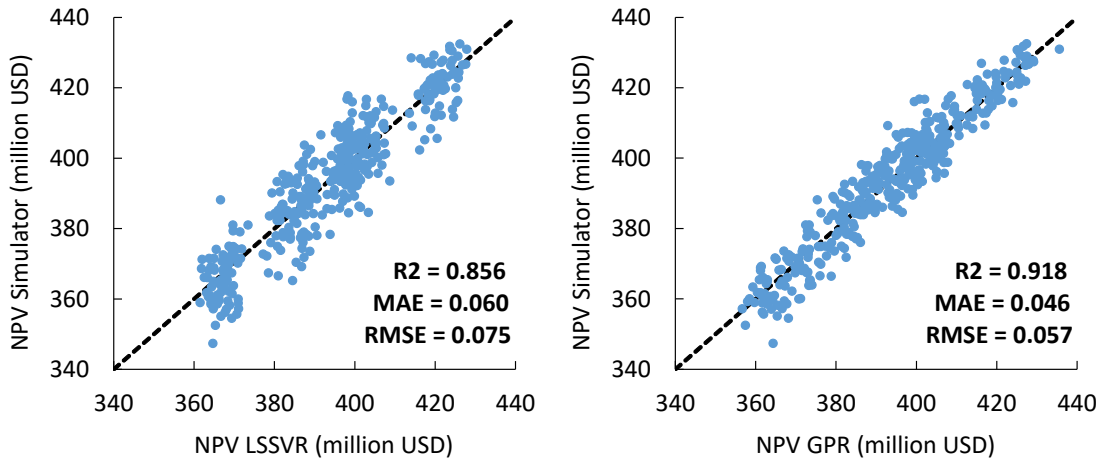
Figure 6.14: NPV vs. iterations obtained using the LS-SVR trained with 600 training data, and tested with 300 test data (*a*) and the LS-SVR trained with 800 training data, and tested with 400 test data (*b*) for Case-8-lowBHP-ICV.

trained with 600 training data points as the initial model for the optimization, we obtain 4.29×10^8 USD converging after 100 iterations, whereas for the case of the LS-SVR trained with 800 training data points, the maximum NPV is 4.44×10^8 USD converging after 53 iterations. However, the StoSAG for this case yields 4.61×10^8 USD which is 3.6% higher than the optimization result of the iterative-sampling-refinement optimization method with the LS-SVR trained with 800 training data points. Comparing Fig. 6.10 and Fig. 6.14, we observe that including ICVs as the design variable increases the value of maximum NPV from 4.55×10^8 USD to 4.61×10^8 USD.

6.2.8 Case-8-lowBHP-ICV-robust

So far, we have considered the deterministic optimization cases where we perform optimization for a single realization of the reservoir model. In this case, we consider 15 plausible realizations of the uncertain reservoir model parameters. In our case, our uncertain reservoir parameters are the permeability of each grid. Our objective function is the expectation of the NPV over reservoir model parameters, where NPV is calculated using Eq. 2.43. The dimension of the feature space is equal to the summation of the dimension of the design variable and the number of reservoir parameters; $832 + 25 \times 25 \times 3 = 2707$. However, considering the fact that in our reservoir model, the permeability field of the second and the third layers is 0.6 and 0.3 times the permeability field of the first layer, respectively,

including the permeability field of only the first layer into the feature space to train the LS-SVR and GPR is enough. Then, the dimension of the feature space is 1457. We use 800 training data points and 400 test data points. The accuracies of both the LS-SVR and



(a) LS-SVR **(b)** GPR

Figure 6.15: Test of the LS-SVR and GPR proxy models for Case-8-lowBHP-ICV-robust.

GPR are good as can be seen from Fig. 6.15. Since GPR for this high dimensional problem requires very high computational time for the Bayesian model selection even when we fix the kernel model (296 seconds), we show the optimization results when only the LS-SVR model is used for iterative-sampling-refinement optimization (Fig. 6.16). The LS-SVR required 20 seconds for the training process. For StoSAG, in robust optimization case, we choose $N_p = 2$, and $N_{sim}^{max} = 6000$, as mentioned earlier. The LS-SVR-based iterative-sampling-refinement

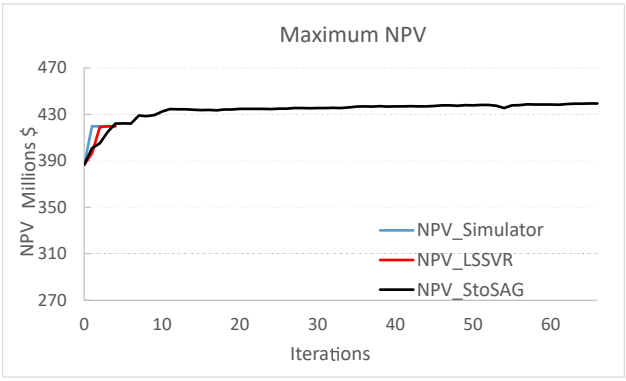


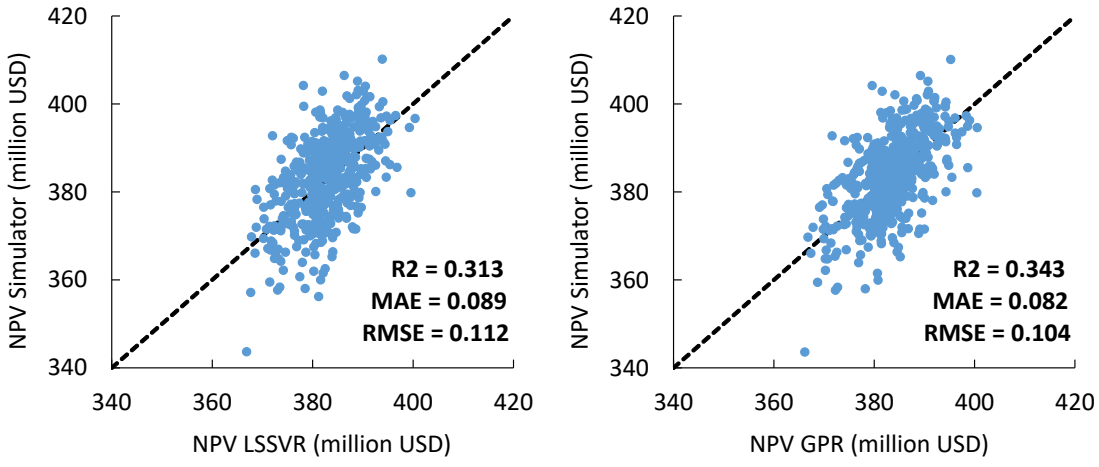
Figure 6.16: NPV vs. iterations obtained using the LS-SVR trained with 800 training data, and tested with 400 test data for Case-8-lowBHP-ICV-robust.

optimization method converged in 4 iterations to a maximum NPV value of 4.20×10^8 USD. However, the maximum value of NPV obtained from the StoSAG optimization method is 4.39×10^8 USD, which is 4.3% higher.

In all of the optimization cases conducted so far, we fixed the cycle lengths as well as gas injection time fractions. However, in the following case, we consider the deterministic optimization problem where we also optimize the cycle lengths and gas injection time fractions.

6.2.9 Case-8-lowBHP-cycle

In this deterministic case, we include cycle lengths of injection and production wells and gas injection time fractions into the set of design variables in addition to gas injection rate, water injection rate, and production BHP. Therefore, the number of design variables is 416. Note that the lower and upper bounds of BHPs of this case are 500 and 1000 psi, respectively. Also, note that the objective function for the previous cases was Eq. 2.43. However, for this case, we use Eq. 2.36 as the objective function. We use a training size of 1000 for the training of the LS-SVR and GPR and test them using 500 data points. Note that for training the GPR model, we again use hyperparameter optimization by fixing the kernel to the *rational quadratic kernel*. The accuracies of the LS-SVR and GPR are shown in Fig. 6.17. The



(a) LS-SVR **(b)** GPR

Figure 6.17: Test of the LS-SVR and GPR proxy models for Case-8-lowBHP-cycle.

computational time required for training the LS-SVR model is 10 seconds. However, training the GPR model by fixing kernel function to the *rational quadratic kernel* takes 216 seconds, which is computationally very expensive. Therefore, we show the optimization results when only the LS-SVR is used as the initial proxy model to perform iterative-sampling-refinement optimization (Fig. 6.18). For the StoSAG optimization method, we use $N_p = 15$ for this deterministic optimization case. The maximum NPV obtained by the iterative-sampling-

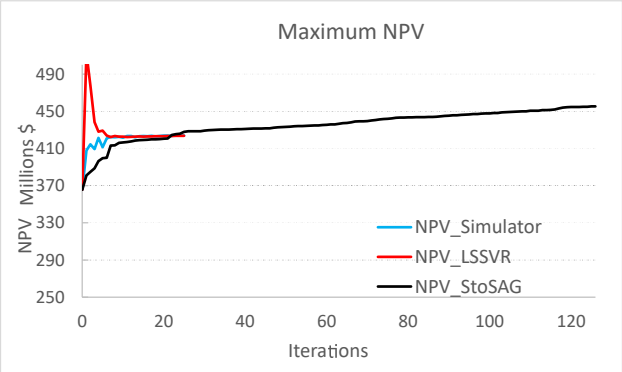


Figure 6.18: NPV vs. iterations obtained using the LS-SVR trained with 800 training data, and tested with 400 test data for Case-8-lowBHP-cycle.

refinement optimization method is 4.24×10^8 USD converging after 25 iterations, whereas the StoSAG method yields a maximum NPV value of 4.55×10^8 USD, which is 6.8% higher. The reason for the low accuracy of the LS-SVR model with the large training size can be because of poor training data as it was in *Case-8-highBHP*. We believe, for this case, poor training data is caused by the production cycles because NPV does not increase for some of the values of production cycle lengths. For instance, if, for a given production well, after 60 days, the well-block pressure gets lower than BHP, NPV will get the same value for cycle length values more than 60 since we will not produce. This results in poor training data, similar to the problem when we had high bounds for BHP. Our suggestion to prevent this problem is to use fixed cycle lengths for production wells and only include cycle lengths of injection wells and gas injection time fractions besides other design variables, such as gas and water injection rates, and production BHPs.

This case also shows us that including cycle lengths did not improve NPV for this reservoir model and well configuration.

Table 6.7 shows the summary of the computational efficiency of optimization cases considered for the WAG problem for iterative-sampling-refinement optimization and StoSAG optimization methods. In the column "Optimization method" the number in the parenthesis

Table 6.7: Number of simulations required for optimization in the WAG problem.

Cases	**Optimization method**	**Number of iterations**	**Computational time (mins)**	**NPV, million $**
Case-8-constBHP	StoSAG	136	2000	394
Case-8-constBHP	LS-SVR (100)	80	237	386
Case-8-constBHP	GPR (100)	111	344	388
Case-8-lowBHP	StoSAG	137	2000	455
Case-8-lowBHP	LS-SVR (320)	44	530	438
Case-8-lowBHP	GPR-fix (320)	61	579	438
Case-8-rate	StoSAG	138	2000	447
Case-8-rate	LS-SVR (200)	73	381	447
Case-8-rate	GPR-fix (200)	5	307	447
Case-8-lowBHP-ICV	StoSAG	105	2000	461
Case-8-lowBHP-ICV	LS-SVR (800)	53	1267	444
Case-8-lowBHP-ICV	GPR-fix (800)	109	1725	410
Case-8-lowBHP-ICV-Robust	StoSAG	66	6000	439
Case-8-lowBHP-ICV-Robust	LS-SVR (800)	4	1261	420
Case-8-lowBHP-ICV-Robust	GPR-fix (800)	28	1758	408
Case-8-lowBHP-cycle	StoSAG	126	2000	455
Case-8-lowBHP-cycle	LS-SVR (1000)	25	1529	423
Case-8-lowBHP-cycle	GPR-fix (1000)	80	1868	432

shows the size of training data used to train the initial LS-SVR and GPR proxy models. GPR-fix indicates a GPR model trained with kernel function fixed. However, we did not put the computational efficiency results for the cases *Case-8-highBHP* and *Case-2-highBHP* because for those cases we could not have good training data to be able to build accurate enough initial LS-SVR and GPR proxy models. Therefore, it is not fair to compare the computational efficiency of the LS-SVR- and GPR-based iterative-sampling-refinement optimization methods with the StoSAG optimization method for these two optimization cases. We should note that Table 6.7 also shows the computational results of GPR-based iterative-

sampling-refinement optimization of cases *Case-8-lowBHP-ICV*, *Case-8-lowBHP-cycle*, and *Case-8-lowBHP-ICV-Robust* for which we stated in the previous sections that training of GPR was computationally very expensive and we did not show the optimization results of them. However, in Table 6.7 we see maximum NPV results as well as computational times of those cases for the GPR-based iterative-sampling-refinement optimization method.

CHAPTER 7

PRODUCTION OPTIMIZATION OF THE WELL SHUTOFF PROBLEM

In this Chapter, we perform both approximate gradient optimization and iterative-sampling-refinement optimization on different production optimization cases of the waterflooding process including well shutoff option in a conventional oil reservoir. A definition of design variables and derivation of NPV to include the well shutoff option keeping NPV continuous with respect to these design variables are given in Section 2.3. We also investigate the importance of the well shutoff option for the realistic production optimization problems that the companies may consider. Therefore, in addition to the production optimization cases for the application of iterative-sampling-refinement optimization, we also consider optimization cases to see the importance of shutoff time as well as the sensitivity of the optimization process to the number of cycles of production wells.

7.1 Results of Production Optimization of Well Shutoff Problem in Conventional Oil Reservoir

In this section, we provide our results obtained by using synthetic examples to demonstrate the use of GPR- and LS-SVR-based iterative-sampling-refinement optimization methods on production optimization including well shutoff in comparison with the conventional numerical gradient ascent optimization method. For the application of the well shutoff option, as mentioned previously, we consider the waterflooding process in a conventional oil reservoir. Our objective function for this problem was, previously, given by Eq. 2.57.

In the following subsection, we introduce rock and fluid properties of the reservoir model considered for the production optimization problem of well shutoff problem as well as the optimization cases considered for the well shutoff problem. In Subsections 7.1.2 to 7.1.11, we show the results of the production optimization cases. For each of the production optimization cases, we compare optimization methods considered in this research such

as GPR-, LS-SVR-based iterative-sampling-refinement optimization methods, simplex, and finite difference optimization methods. We also show the accuracy of initial proxy models of GPR and LS-SVR using test data.

7.1.1 Reservoir Model and Production Optimization Cases

The fluid composition of the reservoir, as in the HnP problem, is taken from Nojabaei et al. (2013), which is Middle Bakken oil. The injected fluid is water. Properties and mole fractions of the reservoir fluid components are given in Table 7.1. It should be noted that we use the CMG-IMEX (2020), a black oil simulator, to simulate the waterflooding for the optimization problem considered here. The composition of oil given in Table 7.1 is used to generate black oil properties via CMG-WINPROP (2004). The PVT properties of the injected fluid (water) and reservoir fluid (oil and gas) are provided in Table 7.2. Fig. 7.1 shows the relative permeability curves of oil, water, and gas.

Table 7.1: Properties of pseudo-components of Middle Bakken oil, after Nojabaei et al. (2013).

Component	Molar fraction	Critical pressure (atm)	Critical temperature (K)	Critical volume (L/mol)	Molar weight (g/gmol)	Acentric factor	Parachor coefficient
CH_4	0.36736	44.57	186.29	0.0989	16.043	0.0102	74.8
C_2H_6	0.14885	49.13	305.53	0.148	30.07	0.1028	107.7
C_3H_8	0.09334	41.90	369.98	0.203	44.097	0.152	151.9
NC4	0.05751	37.18	421.78	0.255	58.124	0.1894	189.6
C5-C6	0.06406	31.38	486.37	0.336	78.295	0.2684	250.2
C7-C12	0.15854	24.72	585.14	0.549	120.562	0.4291	350.2
C13-C21	0.0733	16.98	792.4	0.948	220.716	0.7203	590
C22-C80	0.03704	12.93	1024.71	2.247	443.518	1.0159	1216.8

In the optimization cases considered here, we used waterflooding with 2 injectors and 4 producers in a conventional oil reservoir with the heterogeneous vertically-anisotropic permeability field. All wells are vertical. The reservoir parameters and the dimension of the reservoir are given in Table 7.3. The reservoir system is discretized into 29x17x1 (=493) gridblocks.

The reservoir permeability field comes from a log-normal distribution with a spherical covariance. The long and short distances of the ellipsoid of spherical covariance are

Table 7.2: PVT table of water and reservoir fluid.

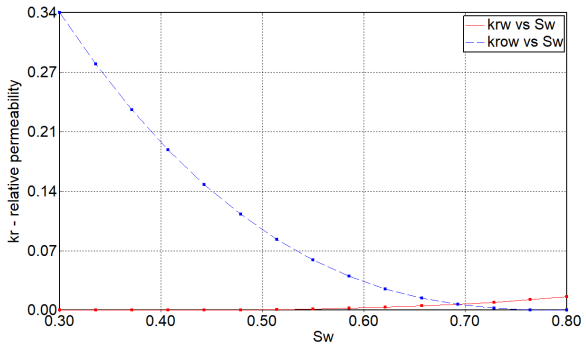
Description	Value	Unit
Water density	62.3	lb/ft^3
Oil density	49.1	lb/ft^3
Gas density	0.06	lb/ft^3
Water Formation volume factor (WFVF)	1	fraction
Water compressibility	1e-6	1/psi
Reference pressure for WFVF	2000	psi
Water viscosity	1	cP
Solution gas oil ratio at initial reservoir pressure	1985	ft^3/bbl
Oil FVF at initial reservoir pressure	2.15	fraction
Gas FVF at initial reservoir pressure	0.00073	bbl/ft^3
Oil viscosity at initial reservoir pressure	0.17	cp
Gas viscosity at initial reservoir pressure	0.05	cp

Table 7.3: Input values of reservoir parameters used in the synthetic reservoir model.

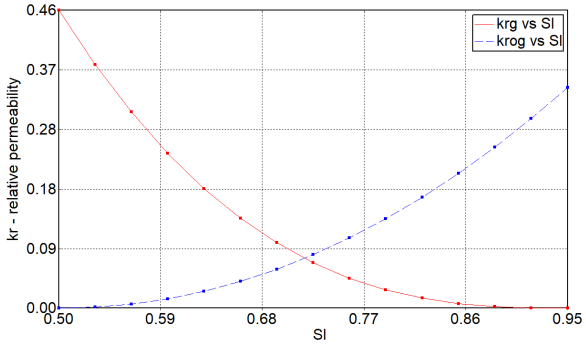
Parameter	Value	Unit
Dimension of the model	3,480x1,700x50	ft^3
Depth	9,000	ft
Initial reservoir pressure	4,200	psi
Reservoir temperature	240	^oF
S_{wir}, irreducible water saturation	0.3	fraction
S_{or}, residual oil saturation	0.2	fraction
S_{gc}, critical gas saturation	0.05	fraction
Rock compressibility	1e-6	1/psi
Porosity	0.2	fraction

L_{max}=1200 ft and L_{min}=1000 ft, respectively. The variance and the mean of this log-normal distribution are 0.2 and 0.01, respectively. In Fig. 7.2, we see the permeability field as well as the areal gridding used in CMG-IMEX (2020). The economical values are provided in Table 7.4. The daily cost of production wells is the same for all wells and is equal to 230 \$/D. The values of bound constraints used for the design variables are given in Table 7.5.

The number of control time steps used for the injection wells ($N_{i,c}$) is 20 for all the optimization cases. Table 7.6 shows the optimization cases considered. Note that we also consider optimization cases where we fix BHPs at all production wells at their lower bounds. The case names are described in Table 7.6 in such a way that one can easily have



(a)



(b)

Figure 7.1: Relative permeability curves of oil and water (a), and gas and liquid (b). Here k_{rw}, k_{row} are water and oil relative permeability respectively; k_{rg}, k_{rog} are gas and liquid relative permeability; S_w, S_l are water and liquid saturation, respectively

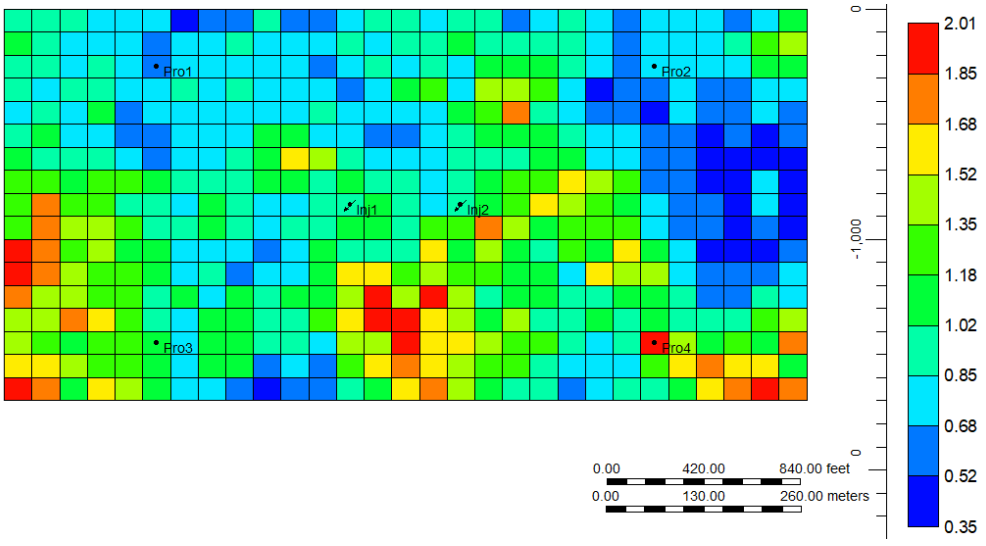


Figure 7.2: Areal grid system and permeability field (color bar represents permeability in mD).

the information about cases from their names without needing to visit Table 7.6: *Case-1* indicates 10 cycle cases, *Case-2* shows 1 cycle cases, whereas *Case-3* indicates 5 cycle case;

Table 7.4: Economical constants used for the NPV function (Eq. 2.57).

Constants	**Value**	**Unit**
b	0.1	fraction
r_o	45	\$/STB
c_{wp}	5	\$/STB
c_{wi}	5	\$/STB
cp_j	230	\$/D

Table 7.5: Bound constraints for the design variables (well shutoff problem).

Design variables	lower bound	upper bound	unit
p_{bh}^n	750	2500	psi
$q_{w,i}^n$	1	10	STB/Day
$\widetilde{\Delta t}_p^n$	0	1	fraction
Δt^n	30	1000	Days

the latter extension of the name code shows the total life in days. For example, the cases with *3000* mean that the total life considered for the case is 3000 days; if the case name has a further extension of *-BHP*, then it is for a constant BHP fixed at its bound of 750 psi in our cases. We also have the simplest case named as the *Case-2-7000-BHP-TP*. In this specific case, the extension, *TP* indicates that we fix all the production time fractions at their upper bound of unity, which means that we do not shut off the production wells. The reason to consider this case will be clear to the reader later.

For all cases, the order of the design variables inside the design vector is consistent with the ordering given in Eq. 2.60.

Case-2-7000-BHP-TP is designed to see the importance of shutoff. For this specific case, the design variable vector consists of only well-controls of the injection wells. Therefore, the number of design variables is calculated regardless of how many cycles are considered for the production wells as follows:

$$N_u = N_I N_{i,c}. \tag{7.1}$$

For *Case-1-3000*, *Case-1-7000*, *Case-1-3000-BHP*, and *Case-1-7000-BHP*, we use

Table 7.6: Cases considered for the optimization applications of well shutoff problem.

Cases	N_c	**Fixed BHP (psi)**	**Total life (Days)**	**Number of design variables**
Case-1-3000	10	NA	3000	160
Case-1-7000	10	NA	7000	160
Case-1-3000-BHP	10	750	3000	120
Case-1-7000-BHP	10	750	7000	120
Case-2-3000	1	NA	3000	48
Case-2-7000	1	NA	7000	48
Case-2-3000-BHP	1	750	3000	44
Case-2-7000-BHP	1	750	7000	44
Case-2-7000-BHP-TP	1	750	7000	40
Case-3-3000	5	NA	3000	100

$N_p = 20$ in the simplex-based gradient ascent optimization method. For other cases, we take $N_p = 10$. The reason for choosing different number of perturbations is due to changing the size of design variable vector for each case. However, only for the cases *Case-2-3000* and *Case-2-7000*, we also consider the finite-difference method to compare their results with the simplex method.

In the following subsections, we present the optimization results for the different cases and their analyses. We also check the well-block pressure of production wells and the pressure distribution of the reservoir at the end of the waterflooding with the optimum design variables. Only for the cases *Case-1-3000* and *Case-2-3000*, we also perform the iterative-sampling-refinement optimization procedure in addition to the simplex optimization methods to compare efficiency and accuracy of two different optimization methods.

7.1.2 Case-1-3000

As mentioned above, the number of design variables for this case is 160. The number of perturbations is chosen to be 20 for the simplex gradient estimation. The initial guess is $\bar{\mathbf{u}}_0 = 0.5 \cdot \mathbf{1}_{N_u}$. However, to satisfy linear equality constraints (Eq. 2.64), we modify the values of cycle lengths in the initial design vector. Fig. 7.3 shows the NPV values generated as a function of iterations of the simplex optimization method. The maximum value of NPV

obtained is 9.9 million $. Fig. 7.4 shows the optimum design variables for Case-1-3000. Even

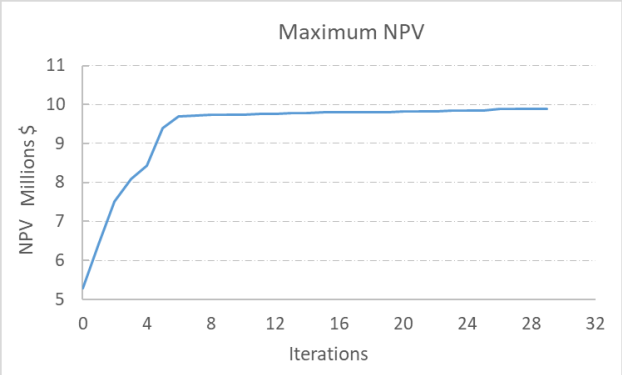


Figure 7.3: NPV vs. iterations obtained by using the simplex optimization method for Case-1-3000.

though we started with 10 possible shutoff initial time periods for each of the production wells, we see that in most cases, the production wells make less than 10 shutoffs.

Since, in this case, we have a relatively high number of design variables, it is highly possible to converge different local optimums with different initial guesses due to the increase of the number of local optimum points of the NPV surface as well as the low accuracy of the gradient approximation. Therefore, we considered different initial guesses for this case and repeated the optimization procedure. The initial guess for the normalized design variable is the same as above except here we replaced values of production time fractions with 1 (instead of 0.5), which means that initially, we do not have any shutoff periods. Fig. 7.5 shows the NPV values generated as a function of iterations of the simplex optimization method for this initial guess. The maximum value of NPV obtained is 10.6 million $, which is 700 thousand $ higher than that of the Case-1-3000 with the first initial guess. However, it required more iterations to converge. Fig. 7.6 shows the optimum design variables for Case-1-3000 with the second initial guess. Even though we started with the initial guess where all production time fractions of all the production wells are equal to 1, which means we do not have any shutoffs initially, the optimization result shows that wells PW2 and PW4 have a single, very short shutoff period and PW1 has two short shutoff periods. Comparing Figs. 7.4 and 7.6, we see that optimum BHP values do not fluctuate for the optimization results of Case-1-3000 with the second initial guess.

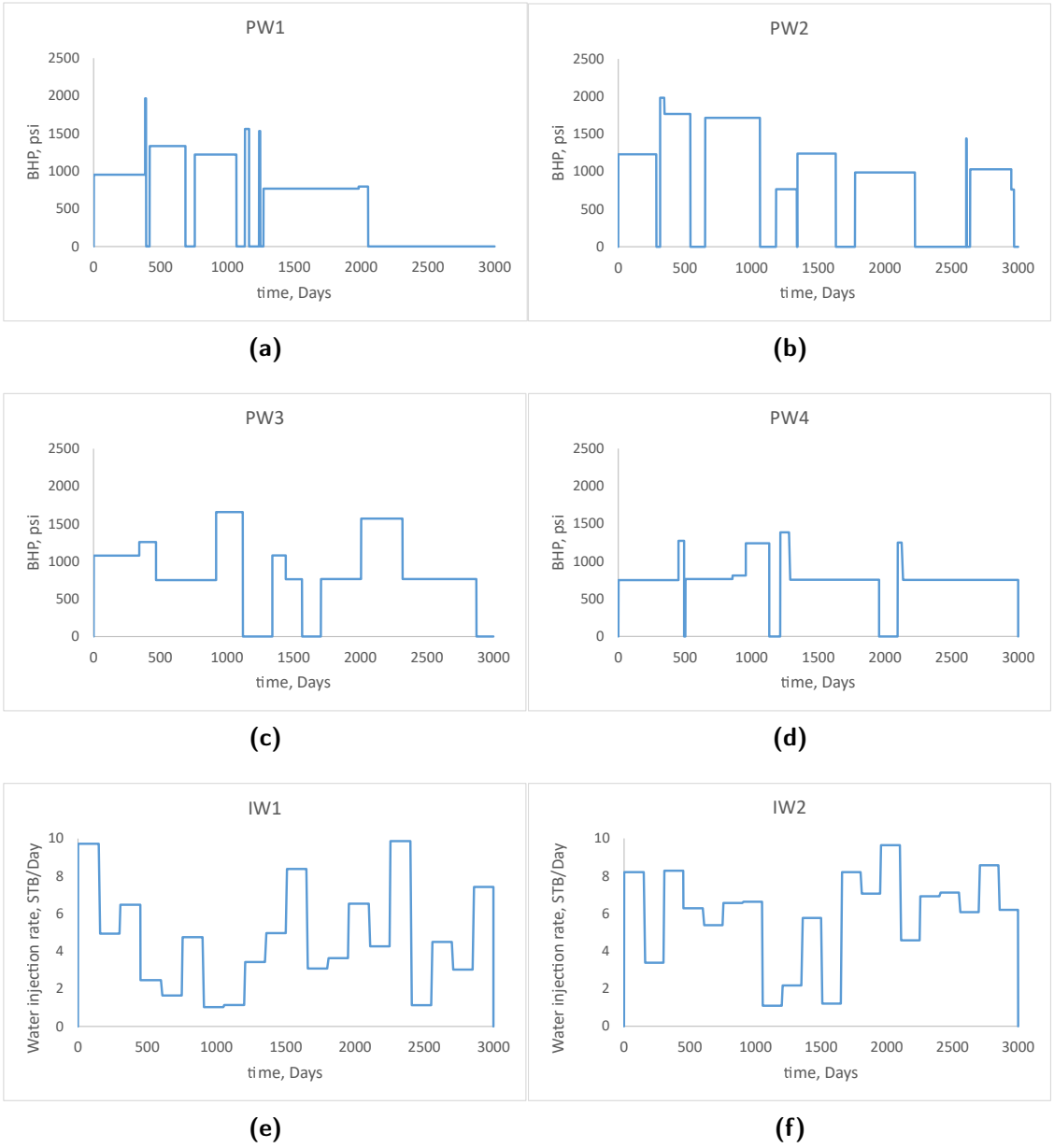


Figure 7.4: Optimum well controls for the simplex optimization method for Case-1-3000.

As mentioned before, for this case, we also perform the iterative-sampling-refinement optimization method. We considered two ML models for this optimization method: the LS-SVR and GPR. To check the effect of the training size on optimization results, we trained and tested models with 3 different training and test sizes. Fig. 7.7 shows training results of the LS-SVR and GPR trained with different sizes of training data. We have to mention that the hyperparameters of the LS-SVR and GPR models are obtained by hyperparameter optimiza-

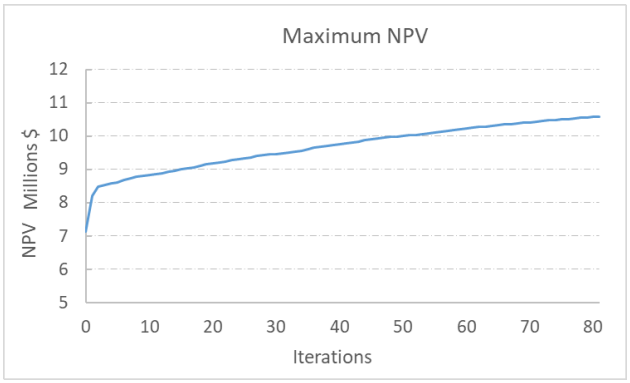


Figure 7.5: NPV vs. iterations obtained by using the simplex optimization method for Case-1-3000.

tion. As we increase the training size, the accuracy of both the LS-SVR- and GPR- based proxy models increases as well. Then, using these models as an initial model, we performed iterative-sampling-refinement optimization. Fig. 7.8 shows the NPV vs iterations obtained by the LS-SVR- and GPR-based iterative-sampling-refinement optimization methods. For the LS-SVR-based iterative-sampling-refinement optimization method, as the accuracy of the initial proxy increases, it converges to higher maximum NPV values. However, it is interesting to note that the number of iteration required for convergence increases as the accuracy of the initial proxy increases. The reason for this is because the model covers more points, and thus it has more local optimums. As it has more local optimums, it is possible that the method slightly converges to a different local optimum at each iteration. However, this kind of behavior of LS-SVR helps us to find a better local optimum at each iteration. We can see from Fig. 7.8 that for the models, trained with a higher number of training data points, after a certain number of iterations, it converges to a higher optimum and keeps increasing. From the least accurate to the most accurate initial LS-SVR-based proxy models, the maximum values of the NPV achieved after the optimization are 10.81, 11.17, and 11.74 million $. For the GPR-based iterative-sampling-refinement optimization method we observe that as the accuracy of the initial proxy model increases, the maximum NPV obtained after the optimization increases. However, the relationship between the accuracy of the initial GPR-based proxy and the number of iterations required for the convergence is not linear and different from that of the LS-SVR. From the least accurate to the most

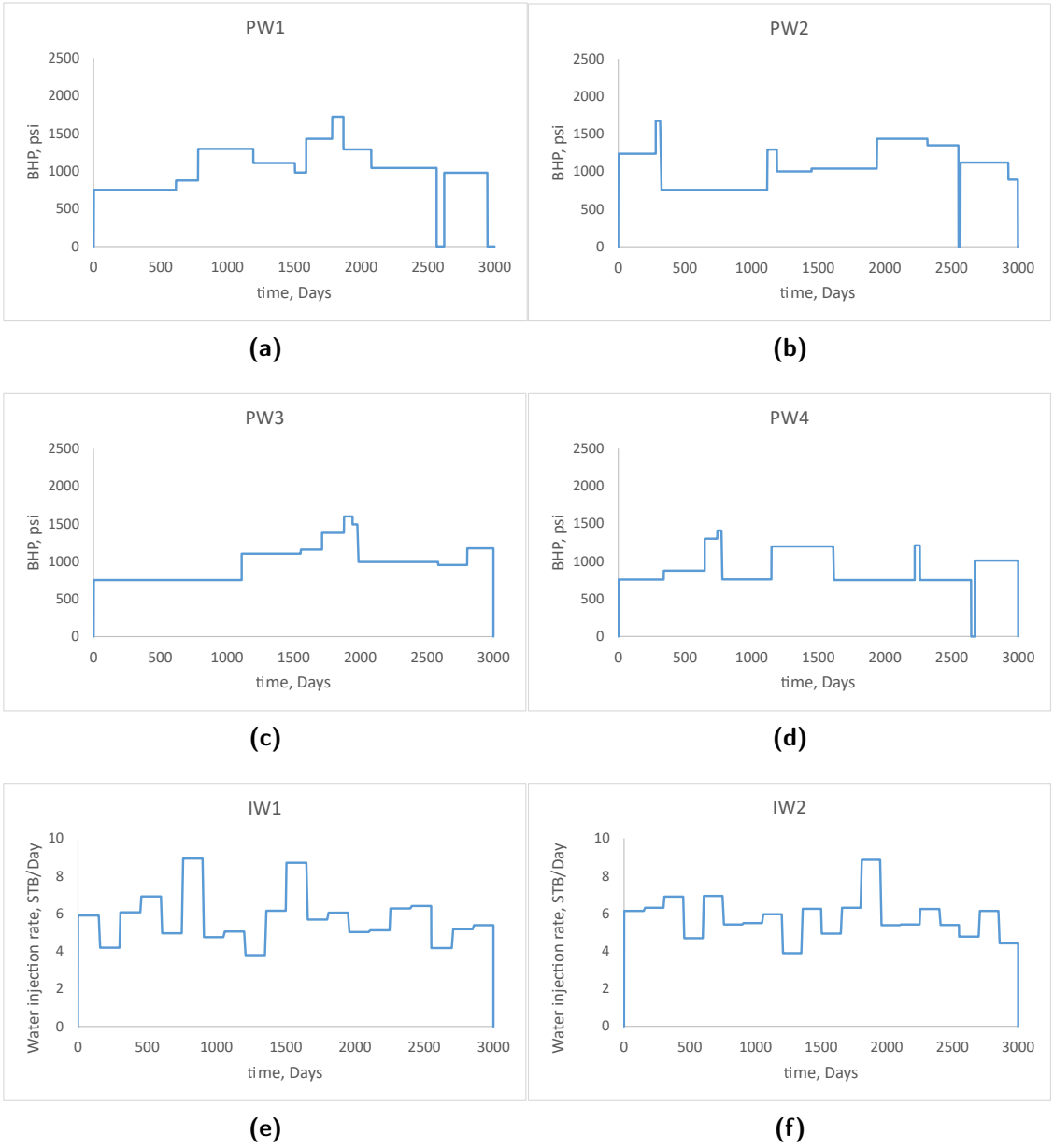


Figure 7.6: Optimum well controls for the simplex optimization method for Case-1-3000 for second initial guess.

accurate initial GPR-based proxy models, the maximum values of the NPV achieved after the optimization are 10.38, 10.85, and 11.19 million $. So the LS-SVR with 140 training data points gave the highest NPV which is 11.74 million $. This NPV value is higher than the NPV value obtained with the simplex optimization method using the first and second initial guesses, which are 1.84 million $ and 1.14 million $, respectively. In Fig. 7.9, we see the optimum design variables obtained using LS-SVR-based iterative-sampling-refinement

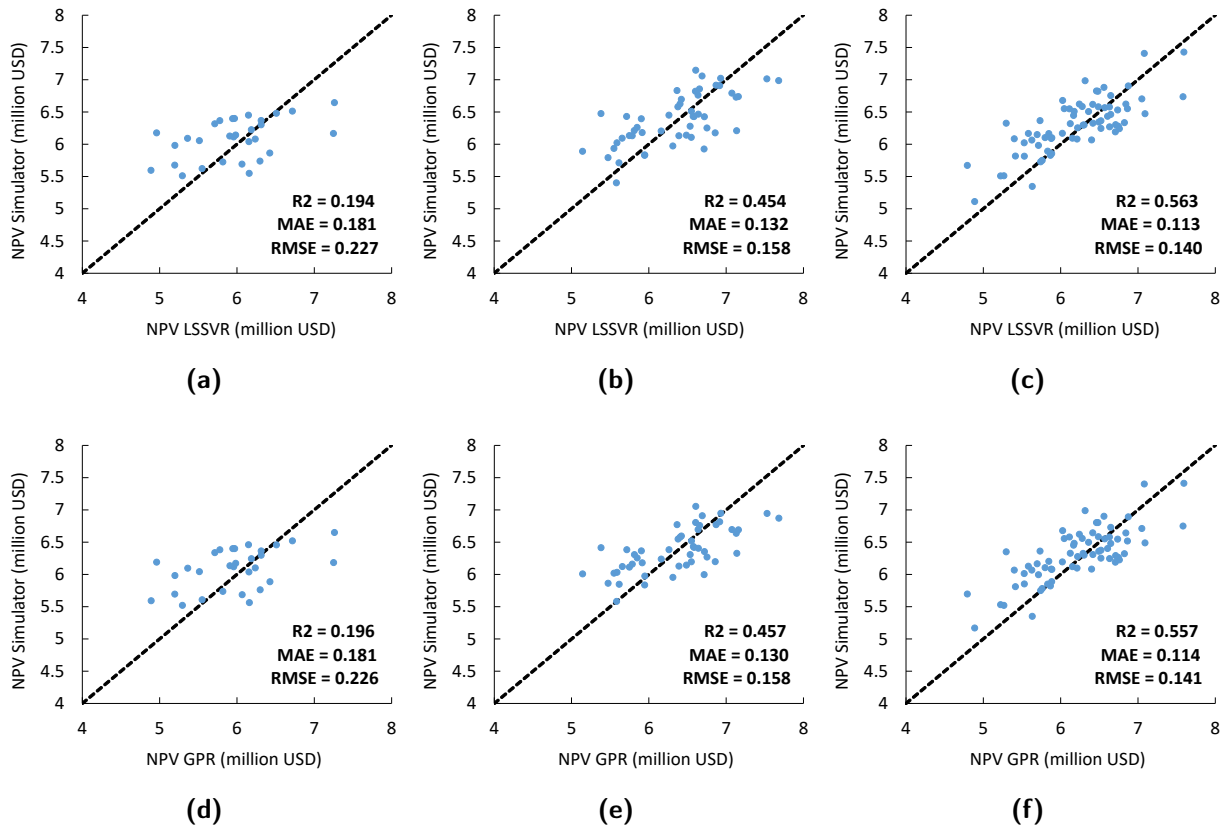


Figure 7.7: Test of the LS-SVR (plots *a*-70 training, 30 test, *b*-100 training, 50 test and *c*-140 training, 70 test) and GPR (plots *d*-70 training, 30 test, *e*-100 training, 50 test and *f*-140 training, 70 test) proxy models for Case-1-3000.

trained with 140 training data points. The results of production wells show that when we used the LS-SVR-based iterative-sampling-refinement optimization method, we obtain almost all the BHPs at their lower bound, 750 psi, and we have very few numbers of shutoff periods, and each of these shut off periods are very short. This result is totally different from the result obtained using the simplex optimization method on this case (Case-1-3000) with the first initial guess (Fig. 7.4), at which we achieved 1.84 million $ less NPV than the NPV obtained based on this optimum design variables. This makes sense if we check the optimum design variables obtained using the simplex optimization method on Case-1-3000 using a second initial guess (Fig. 7.6). These optimum values are similar to those in Fig. 7.9. We can observe that as BHPs approach closer to their lower bounds and as shutoff time diminishes, we achieve higher values of NPV for the case where the total life of production

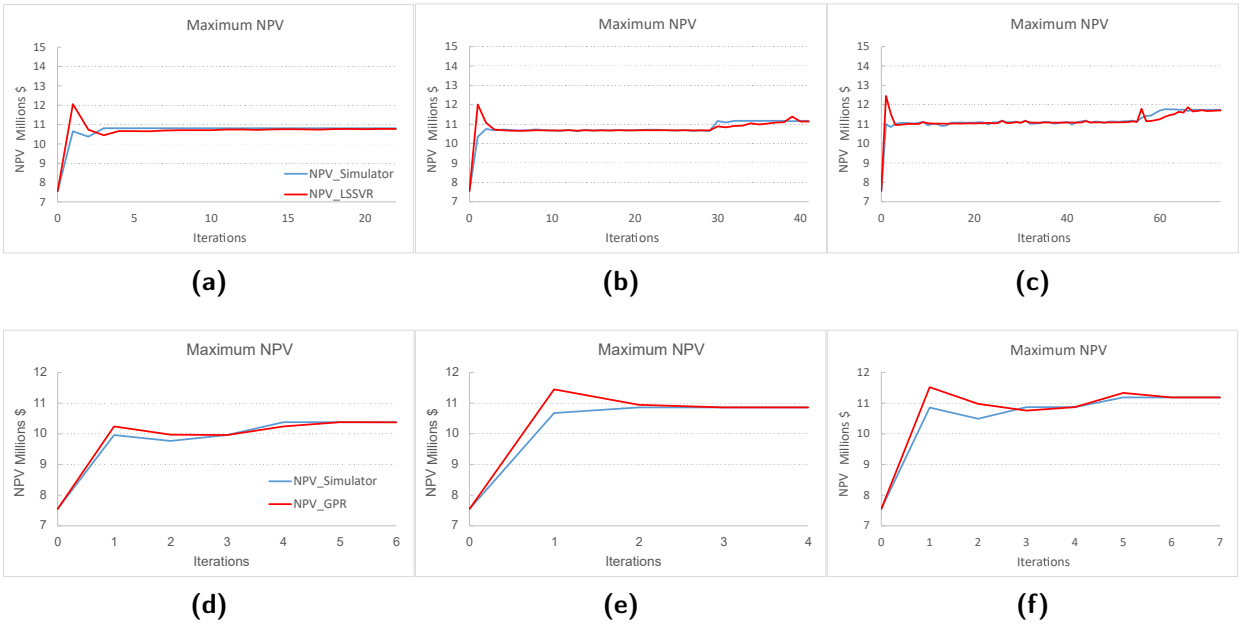


Figure 7.8: NPV vs. iterations obtained using the LS-SVR (plots *a*-70 training, 30 test, *b*-100 training, 50 test and *c*-140 training, 70 test), GPR (plots *d*-70 training, 30 test, *e*-100 training, 50 test and *f*-140 training, 70 test) optimization methods for Case-1-3000.

is 3000 days. This conclusion will make more sense when we look at the optimization result of Case-2-3000 (Fig. 7.20). These results show that with the iterative-sampling-refinement optimization method, we achieved more than 10% higher NPV than that obtained by the simplex method.

We observe that the response surface of the LS-SVR vs iteration shows much more oscillations (see NPV from LS-SVR in Fig. 7.8) compared to the response surface of the real model (see NPV from the simulator in Fig. 7.8), which is referred to as NPV calculated using simulator directly. The oscillations are because the LS-SVR model changes its shape more sharply due to the changes of the hyperparameters at every iteration. Therefore, we think that fixing hyperparameters to reasonable values can decrease the number of iterations required for convergence of the optimization method due to the fewer oscillations of the LS-SVR-based proxy model. The value of the regularization term is highly important due to the fact that as its value gets smaller, we assume higher noise in our real NPV values coming from the simulator. This in turn does not allow the algorithm to converge due to the high relative difference between real NPV (obtained from the simulator) and NPV predicted by

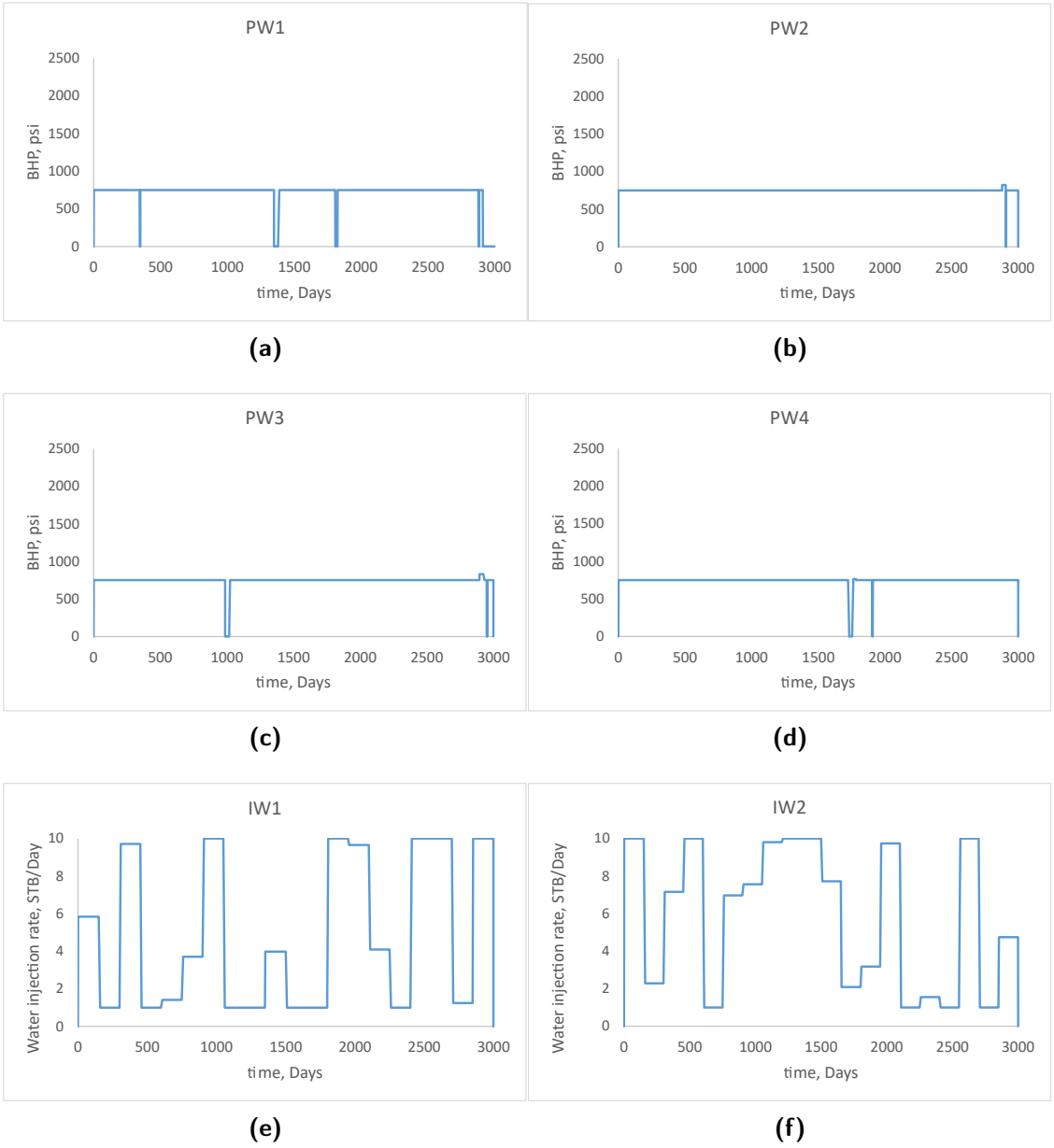


Figure 7.9: Optimum well controls for the LS-SVR-based iterative-sampling-refinement optimization method for Case-1-3000, where initial LS-SVR is trained using 140 training data.

the LS-SVR at a given iteration. We should remember most of the training data points are sampled from all over the response surface, therefore the LS-SVR attempts to match all the training data points fairly equally. When we say fairly equally, it means that it assumes approximately the same data mismatch for each point including the optimum point we found. Therefore, the prediction of NPV at the optimum point could be less or higher than the actual value of NPV obtained from the simulator. If we check Fig 7.8, for almost

at every iteration, the value of NPV obtained from the simulator is exactly the same at each iteration. This means that most probably at every iteration we obtain the same optimum design variables. However, for the LS-SVR model to match that optimum point takes many iterations. This also explains why we need more iterations to converge when we start with a model trained with the higher number of training data points. It is because, as we increase the number of training data points, the proxy model honors all the data points with a similar data mismatch. We observed that the value of the regularization term becomes smaller as we increase the number of training points. This means that the LS-SVR model trained with a higher number of training data assumes a higher data mismatch than the model trained with fewer training data points. Another hyperparameter is the bandwidth (see Section 4.1.3). It affects the smoothness of the response surface of the LS-SVR. However, there is a benefit of the LS-SVR not matching optimum data point immediately (due to the regularization term to be low) and change its surface at each iteration (mainly due to bandwidth hyperparameter) so that we may find a higher optimum point. To support this explanation, we used the same training sizes to train LS-SVR, but at this time with fixed hyperparameters. Fig. 7.10 shows NPV vs iteration of the iterative-sampling-refinement optimization where LS-SVR proxy model was trained using the hyperparameters obtained through hyperparameter optimization and LS-SVR trained using fixed hyperparameters (we call it *LS-SVR-fix*). As a rule of thumb for σ_{bw}^2 we fix it to $N_u/4 = 160/4 = 40$. For the regularization term, γ, we can use anything more than 100, because when we set the regularization term to more than 100 we ignore the data mismatch between real NPV (from the simulator) and NPV prediction with LS-SVR. We also wanted to know what happens if we fix γ only and let σ_{bw}^2 be part of hyperparameter optimization. We called this strategy *LS-SVR-fix-gamma*. In Fig. 7.10, we compare *LS-SVR-fix*, *LS-SVR-fix-gamma*, and for different training sizes. To compare *LS-SVR-fix*, *LS-SVR-fix-gamma* with *LS-SVR* we can compare the Figs. 7.8 and 7.10. These results confirm our explanation above. As we stated, when we fix both hyperparameters we expect the model to not modify the response surface dramatically as it was in the case when the model is trained with the hyperparameters

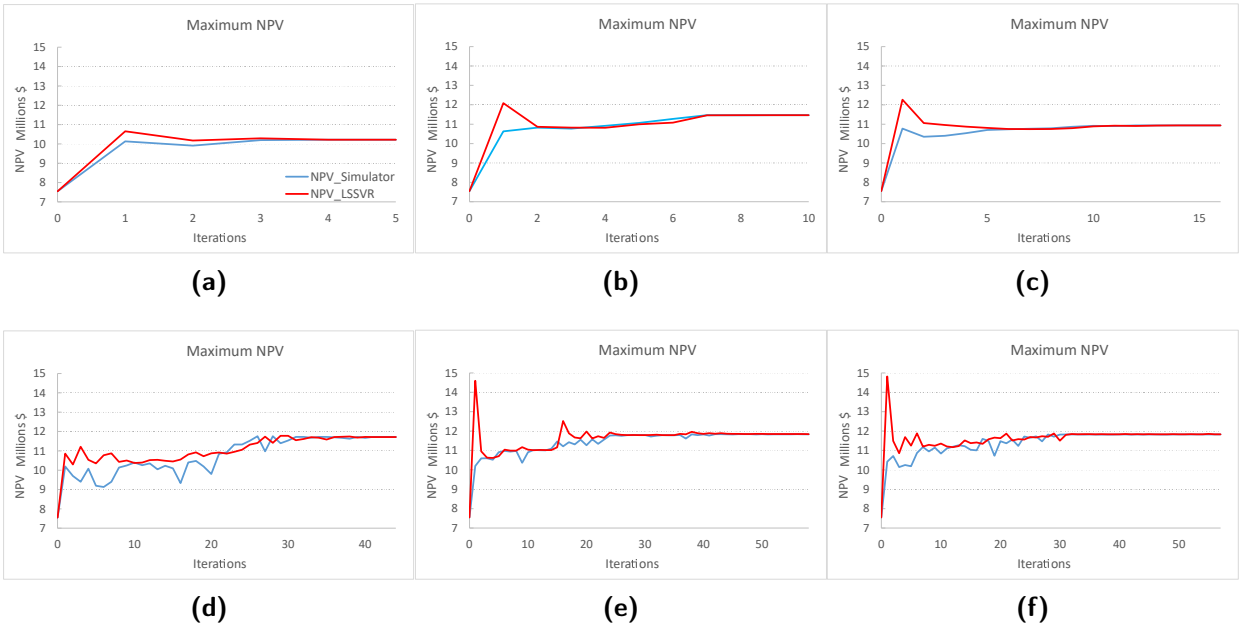


Figure 7.10: NPV vs. iterations obtained using the LS-SVR-fix (plots *a*-70 training, 30 test, *b*-100 training, 50 test and *c*-140 training, 70 test), LS-SVR-fix-gamma (plots *d*-70 training, 30 test, *e*-100 training, 50 test and *f*-140 training, 70 test) optimization methods for Case-1-3000.

obtained from hyperparameter optimization at each iteration. Therefore, it converges faster. However, they converge to the lower NPV values than the NPV obtained when we use the LS-SVR model. When we use the *LS-SVR-fix-gamma* approach, we always converge to higher NPV values than those of LS-SVR and LS-SVR-fix even though it requires more iterations. We should note that we did not compare the training accuracy results of LS-SVR-fix and LS-SVR-fix-gamma since they were approximately the same as the LS-SVR.

7.1.3 Case-1-7000.

In this case, we extend the life of the production life-cycle to 7000 days. Again this case has 160 design variables. Therefore, the number of perturbations is chosen to be 20. We use the same normalized initial guess as in Case-1-3000. Fig. 7.11 shows the NPV as a function of iterations, generated by the simplex optimization method. As can be seen, the maximum NPV obtained at the end of 28 iterations is 9.7 million $. Fig. 7.12 shows the optimum design variables obtained by maximizing the NPV for Case-1-7000. Again similar to Case-1-3000, the production wells have a number of shutoffs of less than 10. NPV of this

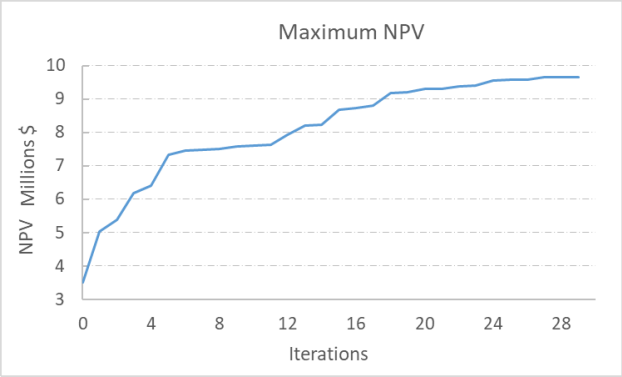


Figure 7.11: NPV vs. iterations obtained by using the simplex optimization method for Case-1-7000.

case is a little bit less than Case-1-3000. To investigate this we performed optimization for Case-2-7000-BHP-TP to check if we can get higher NPV when we force all production wells to produce all the life of production in the subsection Case-2-7000-BHP-TP. In our opinion, it is due to the injection cost of the injection wells. However, we will discuss this more in that subsection based on the result we obtained for Case-2-7000-BHP-TP.

7.1.4 Case-1-3000-BHP.

Here the BHPs of all production wells are fixed at each cycle over the control times steps and hence are not treated as design variables in optimization. Therefore, this case has 120 design variables which consist of all production well shutoff time intervals and injection rates at injection wells. The number of perturbations is chosen to be 20 for this case as well. The initial guess of the design vector is $\bar{\mathbf{u}}_0 = 0.5 \cdot \mathbf{1}_{N_u}$. However, to satisfy linear equality constraints (Eq. 2.64), we modify cycle length values in the initial design vector. Fig. 7.13 shows the NPV values obtained at each iteration of the simplex optimization method. The maximum NPV obtained for this case is 11 million \$, which is 1 million \$ more than the Case-1-3000. Although we would expect to obtain a lower NPV for this case as compared to the case where we treated production BHPs as optimization parameters in addition to shutoff time intervals at the producers, we think that this may have occurred due to the inaccurate computation of the stochastic simplex gradient for the case where BHPs are treated as optimization parameters. As is known, the reduction in the number of design variables is

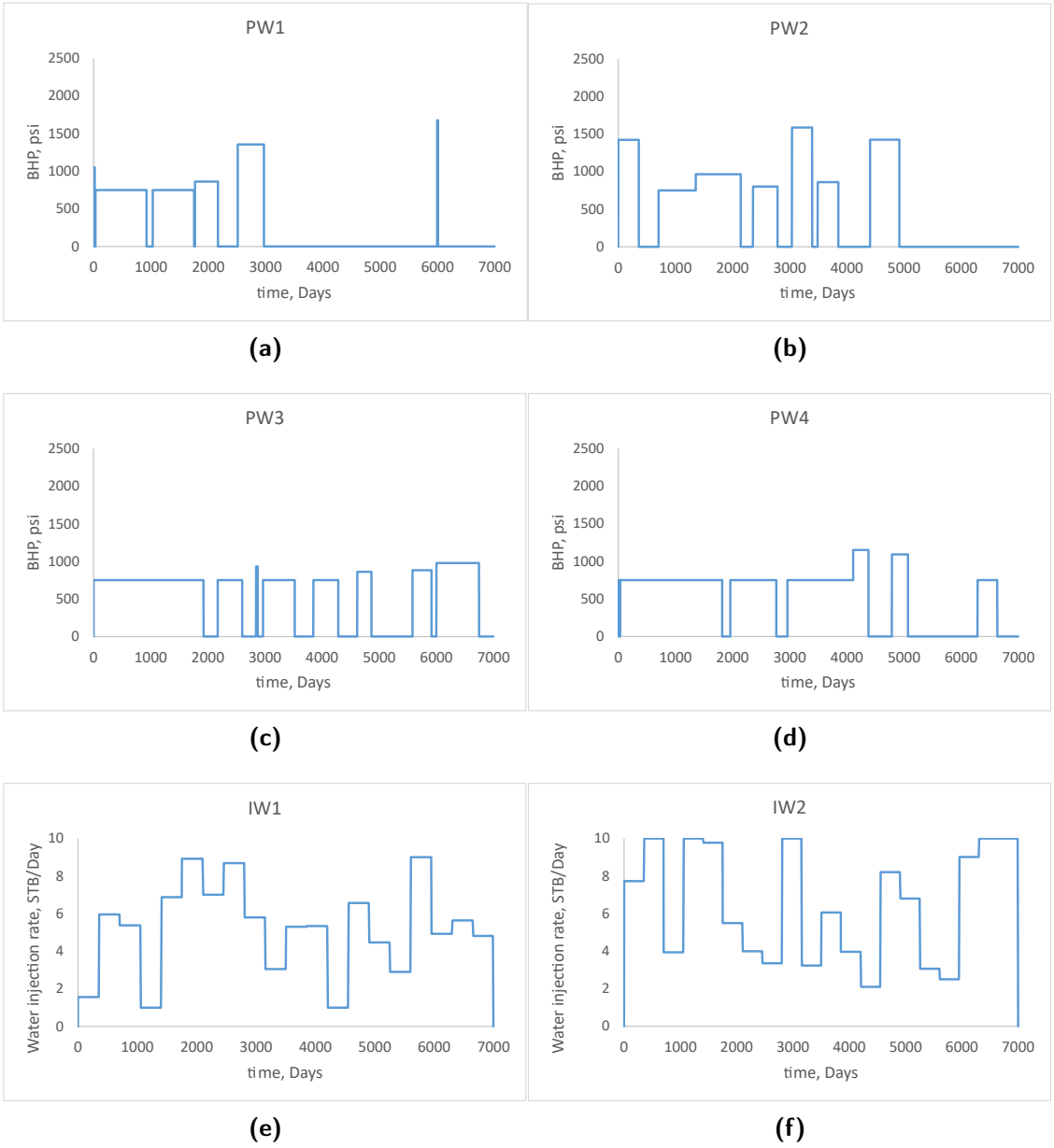


Figure 7.12: Optimum design variables for the simplex optimization method for Case-1-7000.

quite important when using stochastic approximate gradient-based optimization methods because the quality of the gradient may deteriorate for increasing numbers of optimization parameters. For the iterations close to convergence we changed the initial step size from 1 to 0.25, and doing that further increased the NPV from 10 million $ in Case-1-3000 (Fig. 7.14). However, we stopped it before it converged due to the computational time. If we compare Fig. 7.14 with Fig. 7.3 we see that how NPV keeps increasing when we decrease

the initial backtracking step size. It goes to a higher local optimum with a very small number of iterations.

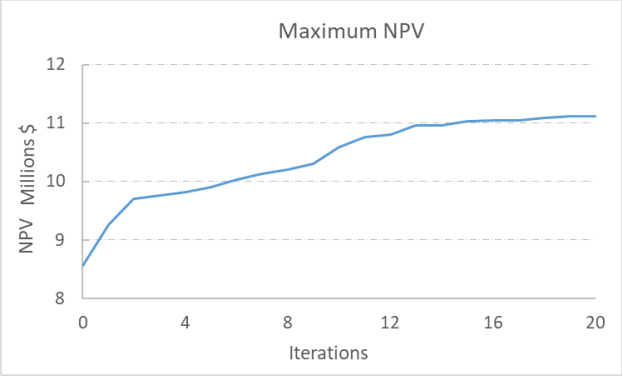


Figure 7.13: NPV vs. iterations obtained by using the simplex optimization method for Case-1-3000-BHP.

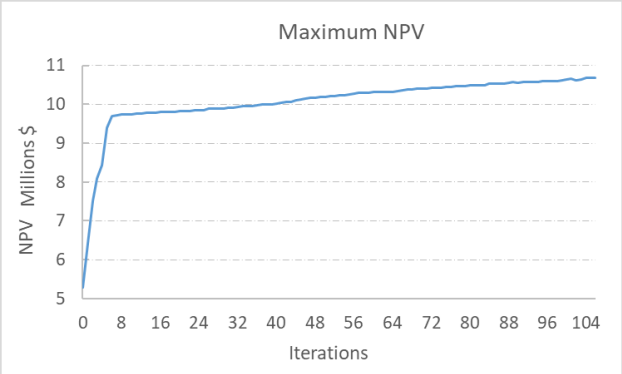


Figure 7.14: NPV vs. iterations obtained by using the simplex optimization method for Case-1-3000 modifying initial backtracking step size to 0.25.

Fig. 7.15 shows optimum design variables of the Case-1-3000-BHP. Here, we observe similar results as to the number shutoff intervals to the Case-1-3000. In most cases, the production wells have a total number of shutoff times or intervals less than 10. We also inspect oil production rates at the maximum value of NPV achieved (Fig. 7.16). Multiple shutoffs of the production wells results in increasing reservoir pressure. We can observe how flow rate increases after every shutoff from the oil rate plots of each production well (Fig. 7.16).

7.1.5 Case-1-7000-BHP

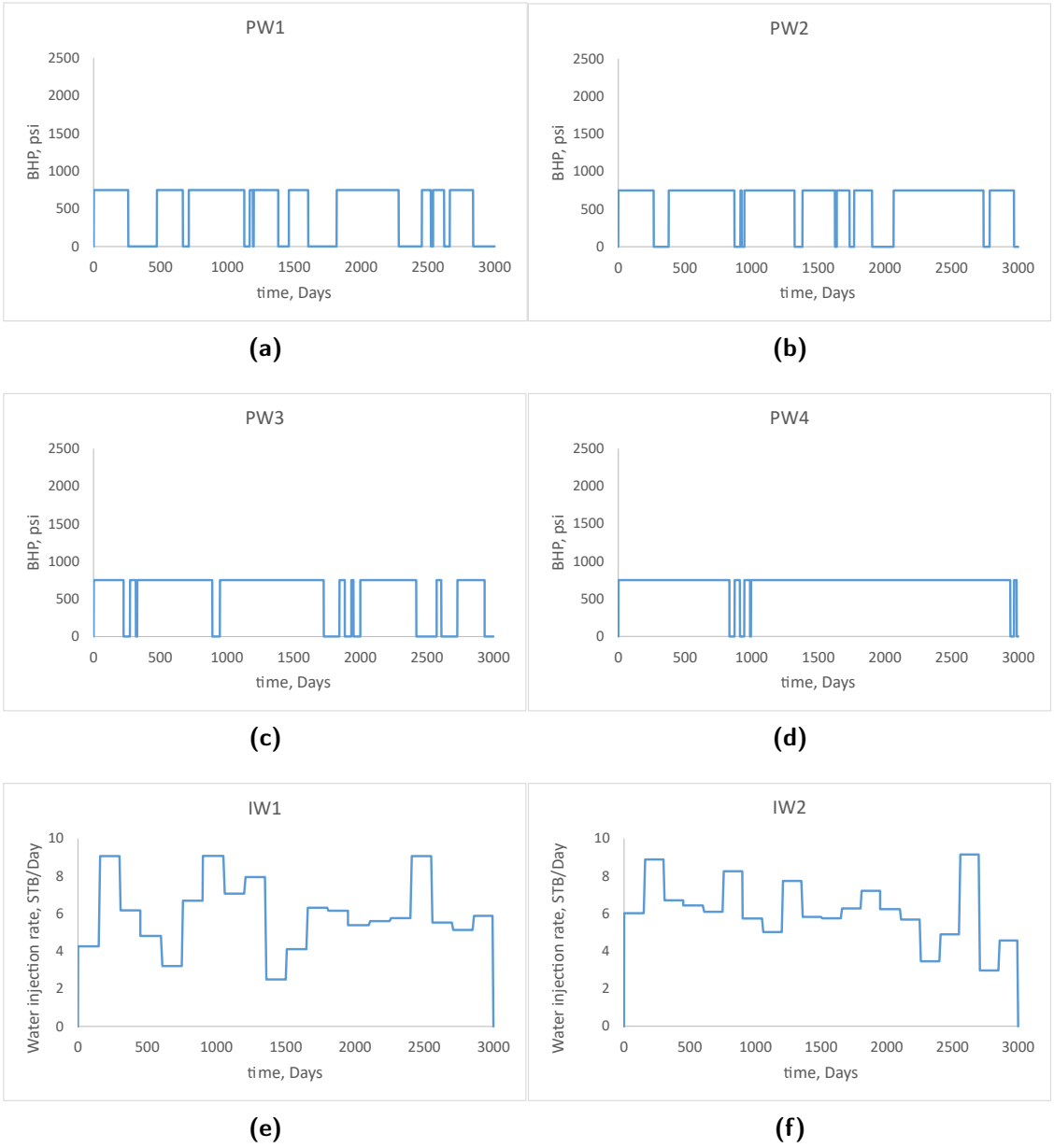


Figure 7.15: Optimum design variables for the simplex optimization method for Case-1-3000-BHP.

Now, we extend the life-cycle to 7000 days. Fig. 7.17 shows the NPV as a function of each iteration of the simplex optimization method. As can be seen, the maximum NPV obtained at the end of 21 iterations is 10.7 million $, which is about 1 million $ more than that obtained for Case-1-7000. This is again because we have fewer design variables compared to Case-1-7000 and most likely we have a more accurate stochastic simplex gradient computed for Case-1-7000-BHP which drives the algorithm to a higher NPV. Fig. 7.18 shows optimum

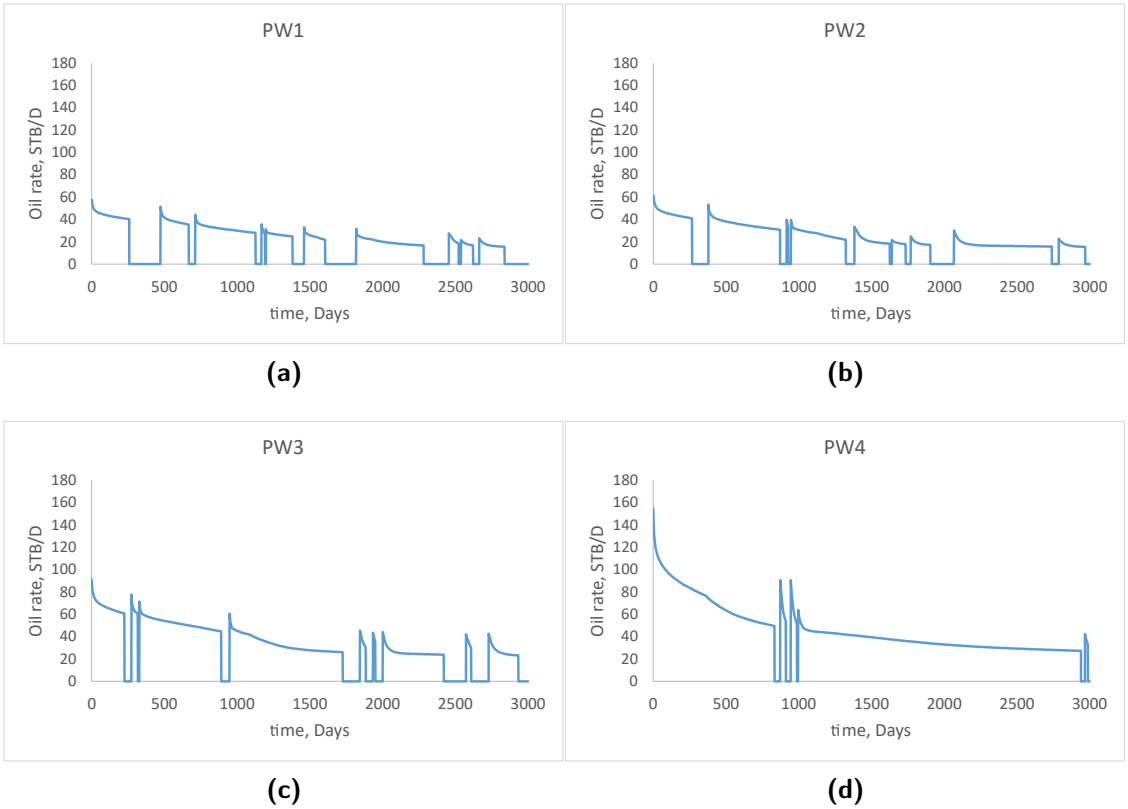


Figure 7.16: Oil rate history at production wells at maximum of NPV achieved at 11 million USD (see Fig. 7.13) at optimum point for Case-1-3000-BHP.

design variables obtained at the maximum value of NPV achieved for Case-1-7000-BHP.

7.1.6 Case-2-3000

This case has 48 design variables (Table 7.6). Recall that in this case, we consider the entire total life-cycle of 3000 days as a single cycle, and hence we have one shutoff time period to be optimized by maximizing the NPV given by Eq. 1. As the number of design variables is small as compared to the previous cases, we also consider computing gradient by the method of finite-difference in comparison with gradient computed by the stochastic simplex method to investigate the effect of gradient computations by two different methods on the optimization algorithm. The number of perturbations is chosen to be 10 for the simplex gradient estimation. The initial guess for the design vector is $\bar{\mathbf{u}}_0 = 0.5 \cdot \mathbf{1}_{N_u}$. Fig. 7.19 shows the NPV values obtained at each iteration of the simplex method and the finite-difference method. The maximum NPV values obtained for the simplex and finite-

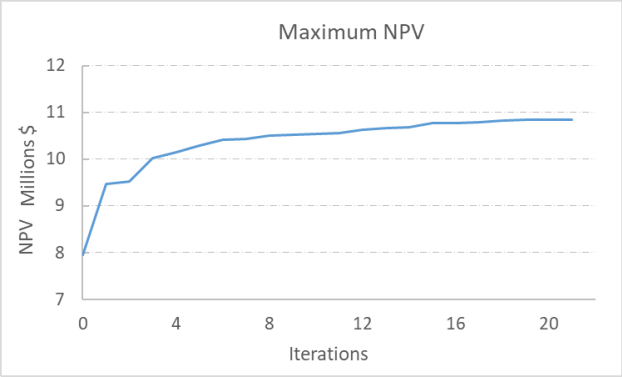


Figure 7.17: NPV vs. iterations obtained by using the simplex optimization method for Case-1-7000-BHP.

difference optimization methods are 11.8 and 11.9 million $, respectively. Even though the maximum NPV values for both optimization methods are close to each other, the optimum design variables obtained for the two methods are slightly different (Fig. 7.20). For this case, the optimum production time fractions computed by the finite-difference method for all production wells are 1. This means the production wells do not shut off for the entire life-cycle of 3000 days. Clearly, the optimum injection rates obtained for the two methods are totally different.

For this case, we also perform the iterative-sampling-refinement optimization method as we did for Case-1-3000. We considered two ML models for this optimization method: the LS-SVR and GPR. To check the effect of the training size on optimization results, we trained and tested models with 2 different training and test sizes. Fig. 7.21 shows training results of the LS-SVR and GPR trained with different sizes of training data. We must mention that hyperparameters of the LS-SVR and GPR models are obtained from hyperparameter optimization. As we increase the training size accuracy of both the LS-SVR- and GPR-based proxy model increases as well. Fig. 7.22 shows NPV results obtained by the LS-SVR- and GPR-based iterative-sampling-refinement optimization methods. For both LS-SVR and GPR-based iterative-sampling-refinement optimization methods as the accuracy of the initial proxy does not change the results of the NPV that much. However, the number of iteration requires for convergence increases as the accuracy of the initial proxy increases. NPVs achieved for both models were the same and equal to 11.9 million $, which is exactly the

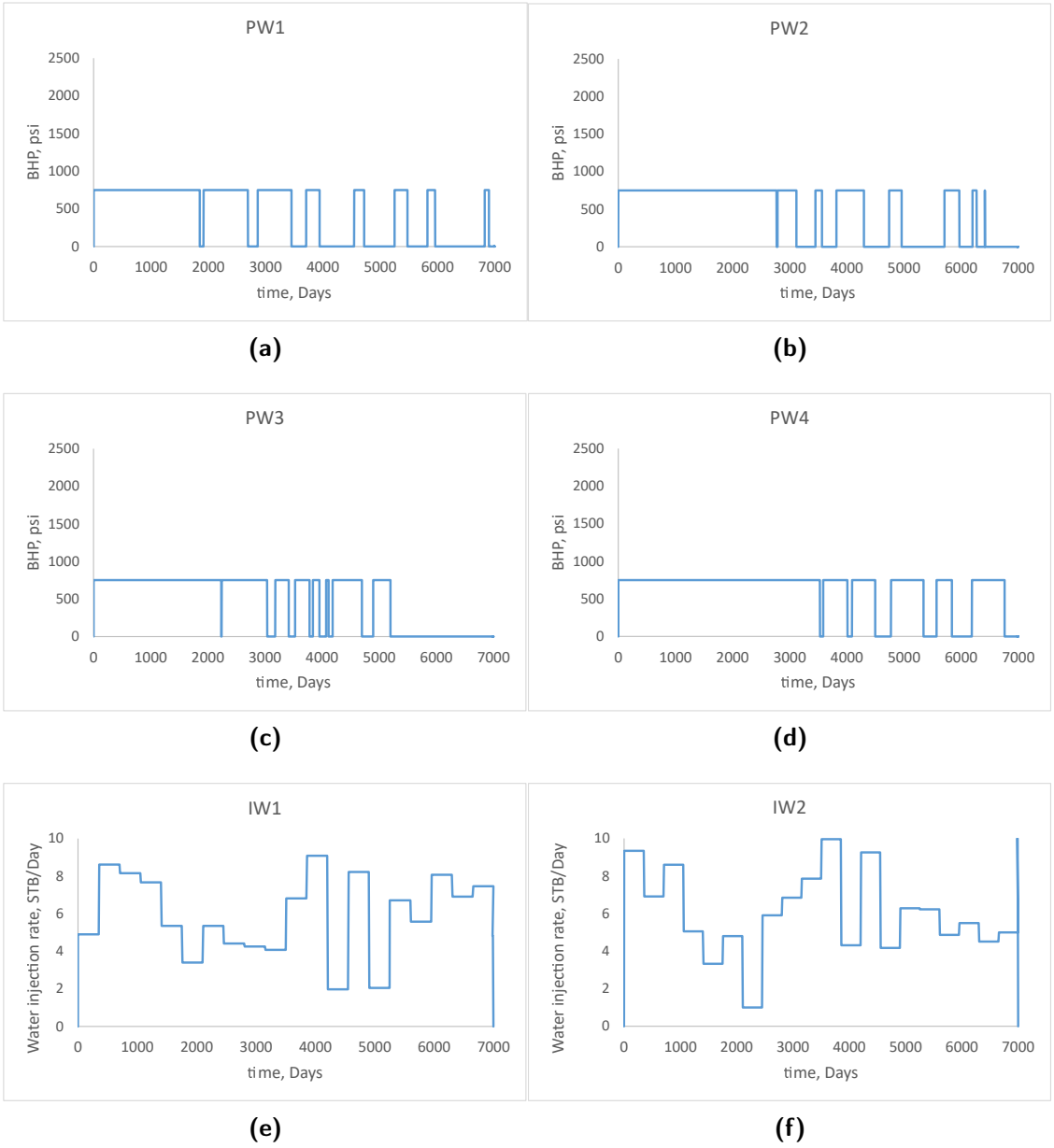


Figure 7.18: Optimum design variables for the simplex optimization method for Case-1-7000-BHP.

same maximum NPV obtained for this case using finite-difference optimization method, and 100 thousand $ more than that obtained using the simplex optimization method. We show only the optimum water injection rates obtained using the LS-SVR- and GPR-based iterative-sampling-refinement optimization method (Fig. 7.23) since other optimum design variables are the same as the results obtained using the FD optimization method (Fig. 7.20).

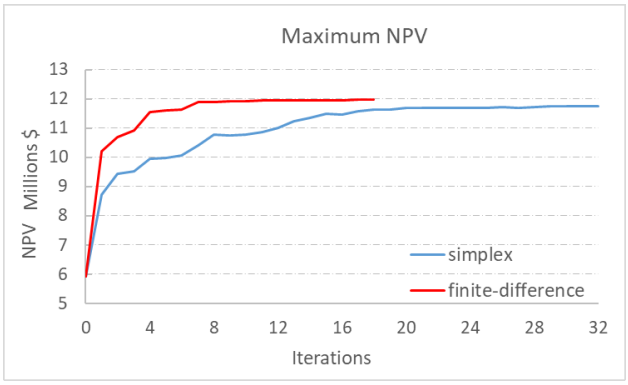


Figure 7.19: NPV vs. iterations obtained by using the simplex and finite-difference optimization methods for Case-2-3000.

7.1.7 Case-2-7000

Here, we increase the total life-cycle of the production to 7000 days. The number of perturbations used is again 10 for the simplex gradient estimation and the initial guess of the vector of the design variables is $\bar{\mathbf{u}}_0 = 0.5 \cdot \mathbf{1}_{N_u}$. Fig. 7.24 shows the NPV versus iteration for both the simplex and the finite-difference methods. The maximum NPV values of the simplex and finite-difference optimization methods are 10.9 and 11 million \$, respectively. Similar to Case-2-3000, we observe differences between the optimal design variables for the two different methods (Fig. 7.25). Both optimization methods show that production wells shut off prior to the end of the production life. The reason for the case where the life-cycle is 3000 days achieves a slightly higher NPV than the case where the life-cycle is 7000 days may be due to the increased cost of the injection wells with the increasing production-life. Besides the reservoir volume is not large enough for the production life-cycle of 7000 days. In Fig. 7.26, we see the oil production rate history of each production well at the maximized value of NPV by the finite-difference optimization method. Since production well PW4 is in the productive region, where the permeability is higher than other regions (Fig. 7.2), its production rate is higher than those of the other production wells and hence continues to produce without shutoff its shutoff until the end of the life of the production.

7.1.8 Case-2-3000-BHP

Here, the production life-cycle is 3000 days, and there is only one shutoff time interval.

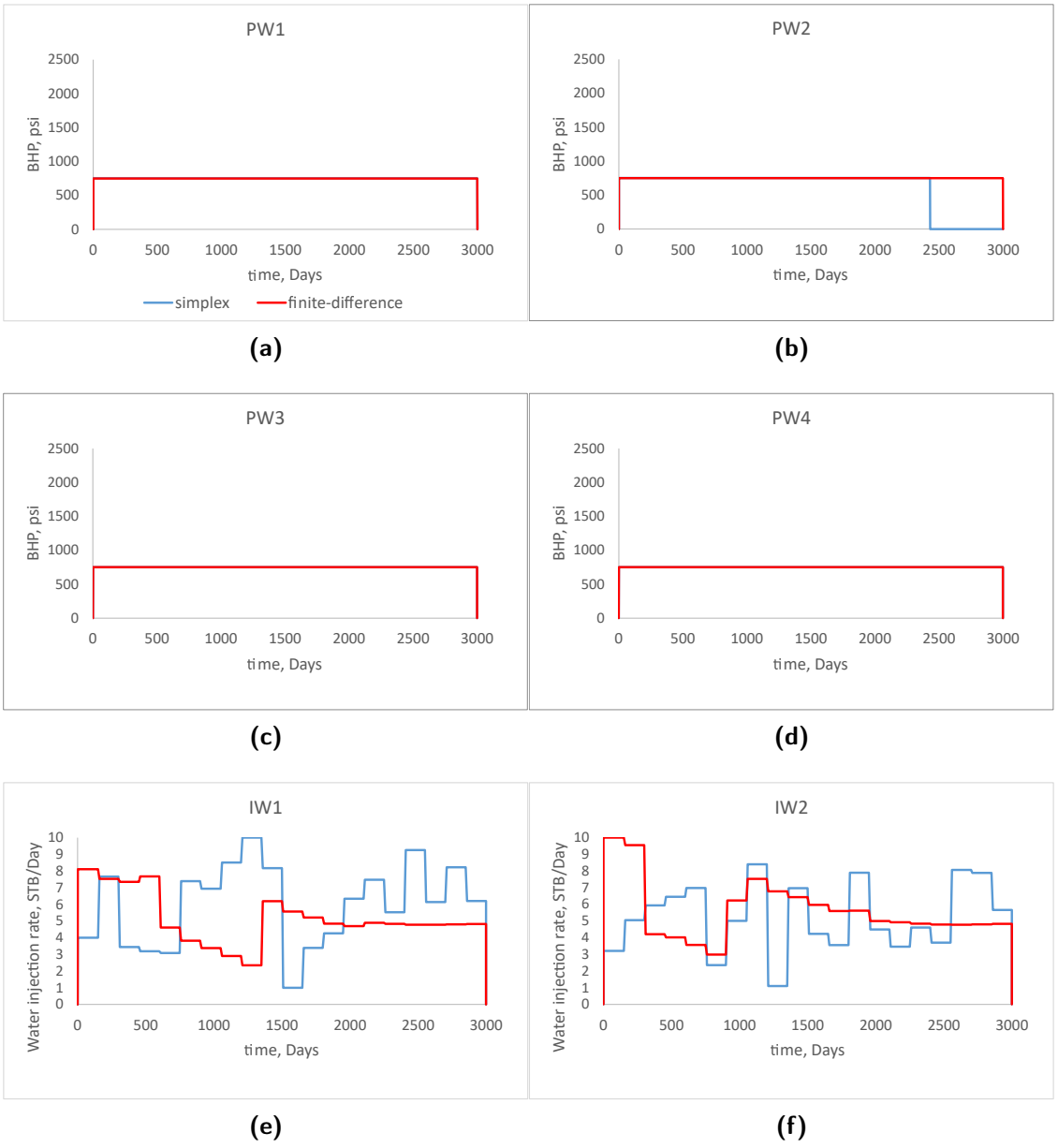


Figure 7.20: Optimum design variables for the simplex and finite-difference optimization methods for Case-2-3000.

Besides, we fix BHPs of all production wells. Therefore, the number of design variables is 44, which is equal to all design variables except production BHPs. The number of perturbations for the simplex method is chosen to be 10 for this case. The initial guess of the vector of design variables is $\bar{\mathbf{u}}_0 = 0.5 \cdot \mathbf{1}_{N_u}$. Fig. 7.27 shows the results obtained for NPV of the simplex optimization method. The maximum value of NPV obtained is 11.9 million \$, which is 100 thousand \$ more than that for Case-2-3000. As explained earlier, this may be due

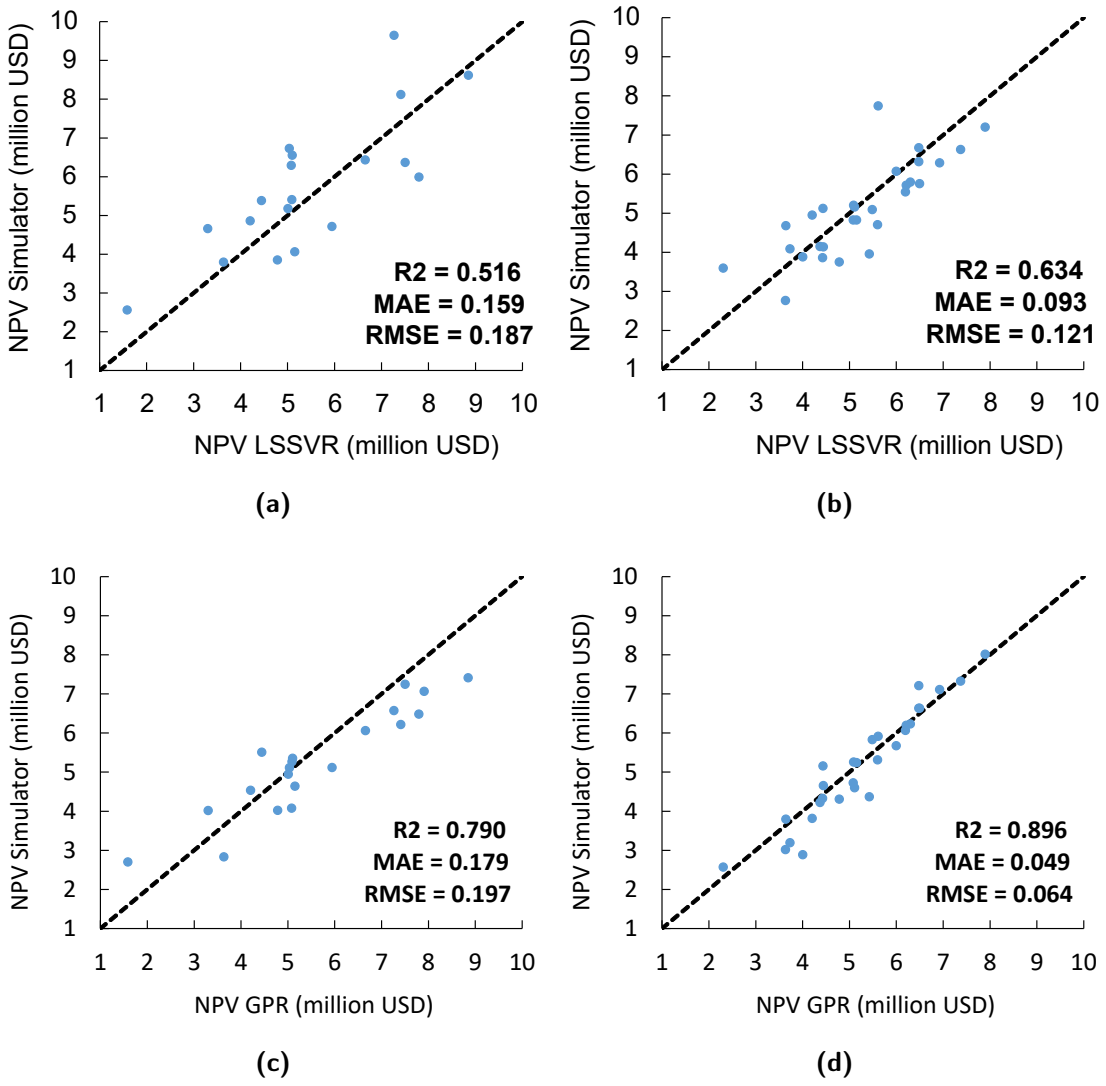


Figure 7.21: Test of the LS-SVR (plots *a*-40 training, 20 test and *b*-60 training, 30 test) and GPR (plots *c*-40 training, 20 test and *d*-60 training, 30 test) proxy models for Case-2-3000.

to more accurate computation of the gradient used in the simplex method when we have fewer design variables than Case-2-3000. Fig. 7.28 shows the optimum design variables obtained for Case-2-3000-BHP. All the production wells shut off at the end of the life of the production.

7.1.9 Case-2-7000-BHP

Here, we extend the total life of production to 7000 days to see if we can observe shutoff times at some of the production wells. We fixed BHPs of all production wells at the

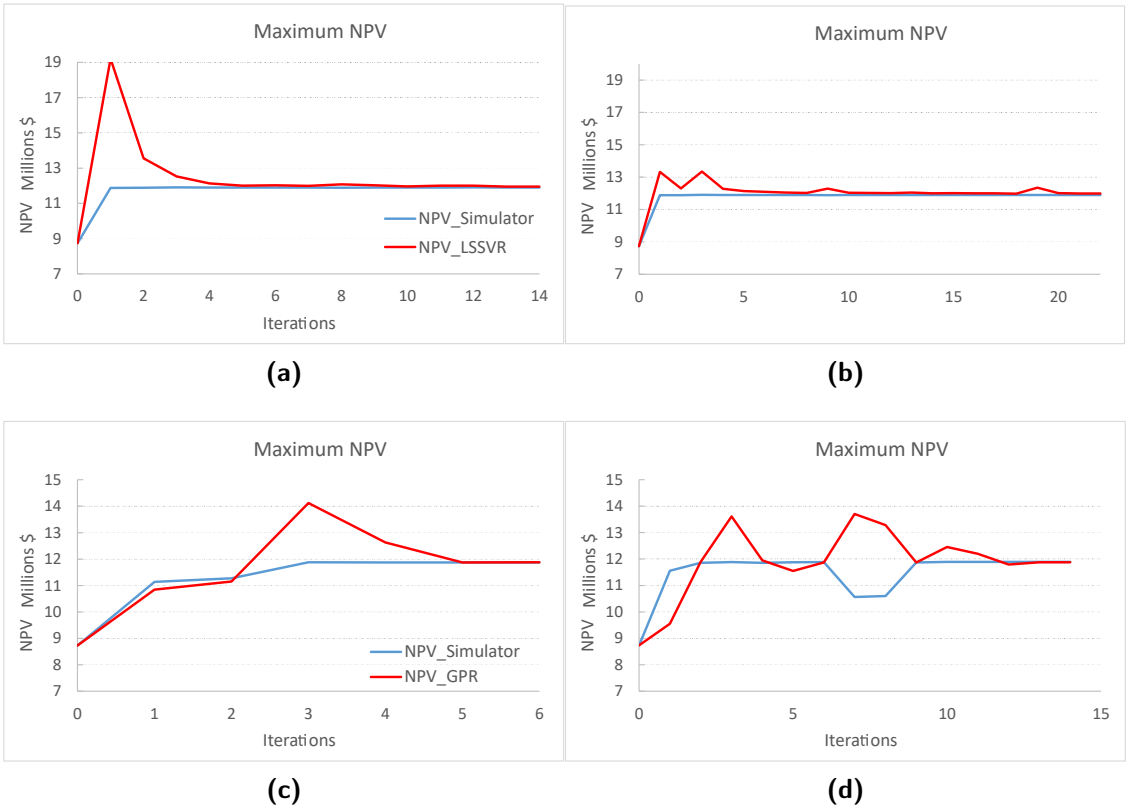
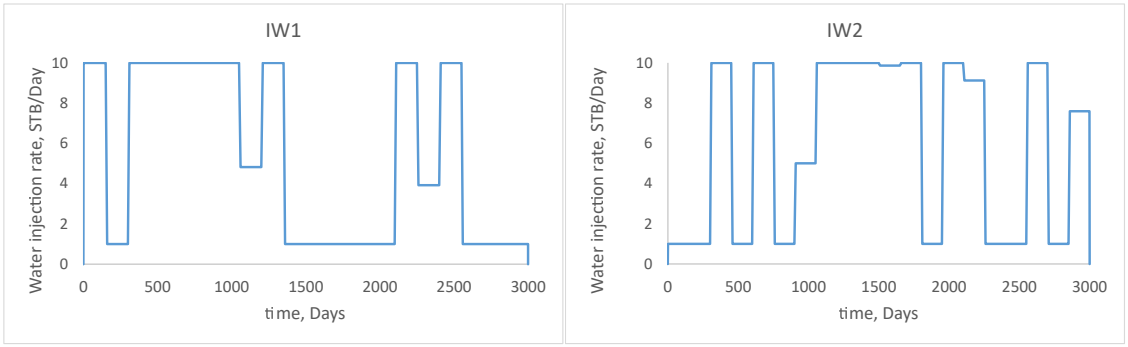


Figure 7.22: NPV vs. iterations obtained using the LS-SVR (plots *a*-40 training, 20 test and *b*-60 training, 30 test) and GPR (plots *c*-40 training, 20 test and *d*-60 training, 30 test) optimization methods for Case-2-3000.

lower bound. The number of design variables is 44. The number of perturbations for the simplex method is 10 for this case as well. The initial guess for the vector of design variables is $\bar{\mathbf{u}}_0 = 0.5 \cdot \mathbf{1}_{N_u}$. Fig. 7.29 shows NPV versus iterations obtained by the simplex optimization method. The maximum NPV obtained is 11 million \$, which is 100 thousand \$ more than that for Case-2-7000. Fig. 7.30 shows optimum design variables of the Case-2-7000-BHP. The production wells PW1 and PW2 shut off earlier than the production wells PW3 and PW4. We also inspect oil production rate histories of each production well to see after what rate it is worth to shut off the well (Fig. 7.31).

7.1.10 Case-2-7000-BHP-TP

In Case-2-7000, we found that the optimum shutoff times obtained by the simplex method showed that production wells do shut off after a certain period of production, while



(a) **(b)**

Figure 7.23: Optimum water injection rates for the LS-SVR-based iterative-sampling-refinement optimization method for Case-2-3000, where initial LS-SVR is trained using 40 training data.

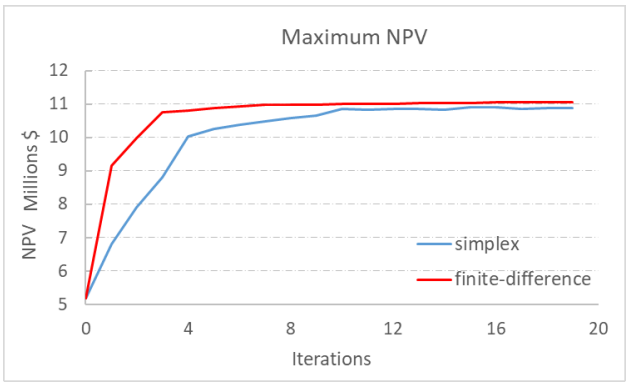


Figure 7.24: NPV vs. iterations obtained by using the simplex and finite-difference optimization methods for Case-2-7000.

the finite-difference method showed that except the production well PW4, which is in the productive area, all other wells shut off. The optimum NPV results of this case showed a lower NPV value than the cases where life is 3000 days (Case-2-3000-BHP and Case-2-3000). Therefore, we want to check if we force all production wells to produce till the end of the life of the production to see if we will get a higher NPV. Therefore, for this case, we fix the production time fraction of each production well to its upper bound, 1, which means that we do not shut off the production wells. In this case, BHPs of the production wells are also fixed to the lower bound, 750 psi. Therefore, the only design variables left are the well controls of injection wells. The number of design variables is 40. The number of perturbations is chosen to be 10 for this case as well. The initial guess is again $\bar{\mathbf{u}}_0 = 0.5 \cdot \mathbf{1}_{N_u}$. Fig. 7.32 shows NPV vs. iterations for the simplex optimization method. The maximum NPV value

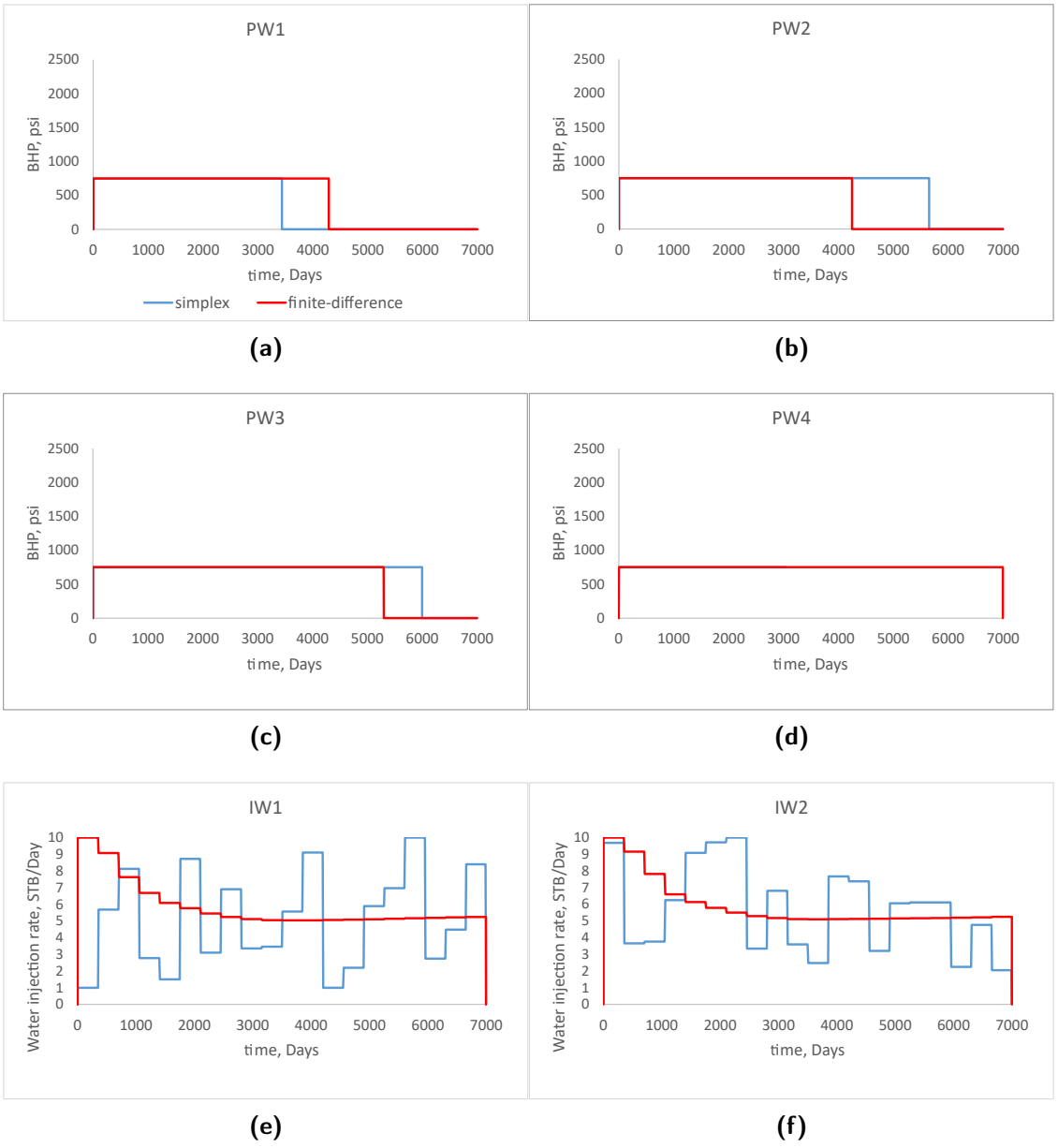


Figure 7.25: Optimum design variables for the simplex and the finite-difference optimization methods for Case-2-7000.

is 10.6 million $, which is 300 thousand $ less than that for Case-2-7000 and 400 thousand $ less than that for Case-2-7000-BHP. This result shows the importance of the shutoff of the production wells; that is, fixing production time fraction, not allowing them to be part of optimization results in lower maximum NPV. Therefore, having the production time fraction be part of the design variables is important. Fig. 7.33 shows optimum design variables of Case-2-7000-BHP-TP, which are only the injection rates at each injection control step.

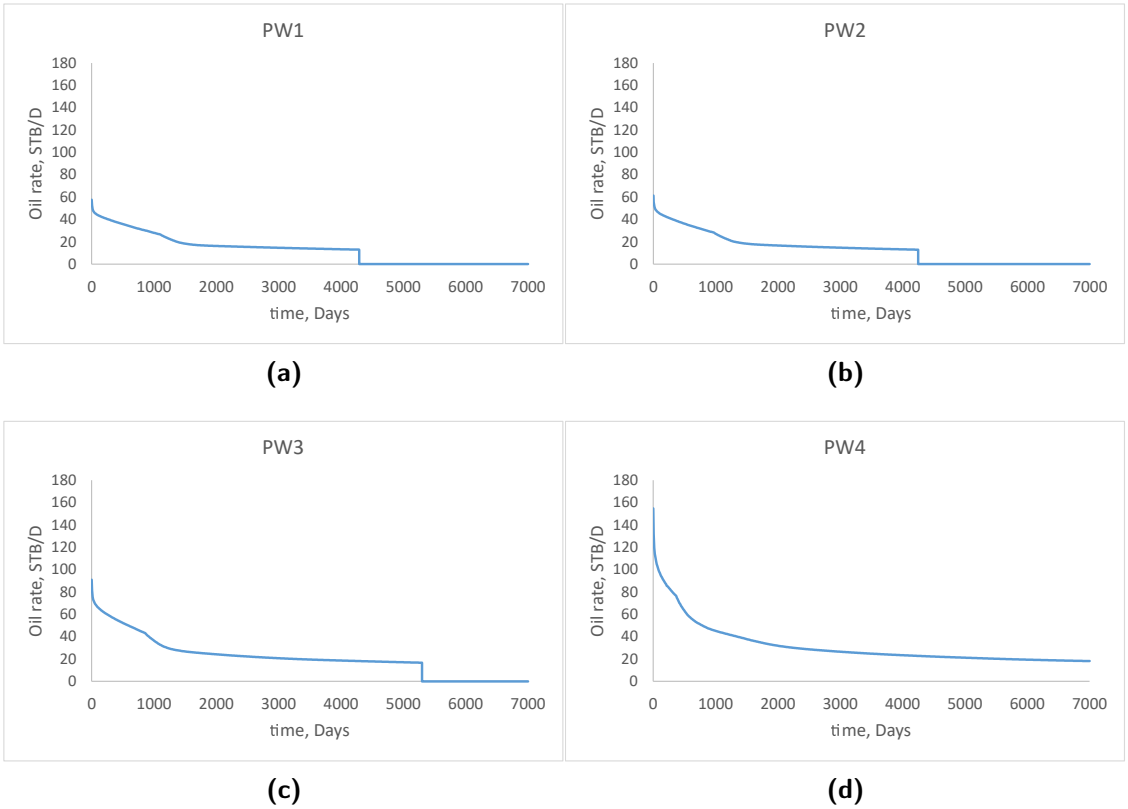


Figure 7.26: Optimum oil production rate histories calculated by using the the finite-difference optimization method for Case-2-7000.

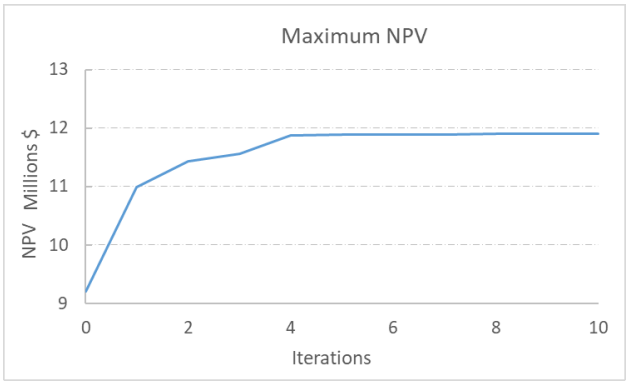


Figure 7.27: NPV vs. iterations obtained by using the simplex optimization method for Case-2-3000-BHP.

7.1.11 Case-3-3000

In this last case, we investigate the effect of the choice of the number of cycles for production wells on the maximum NPV results. In the cases above we looked into 1 cycle cases and 10 cycle cases. The optimization results of *Case-1-3000* (Fig. 7.3) and *Case-2-3000*

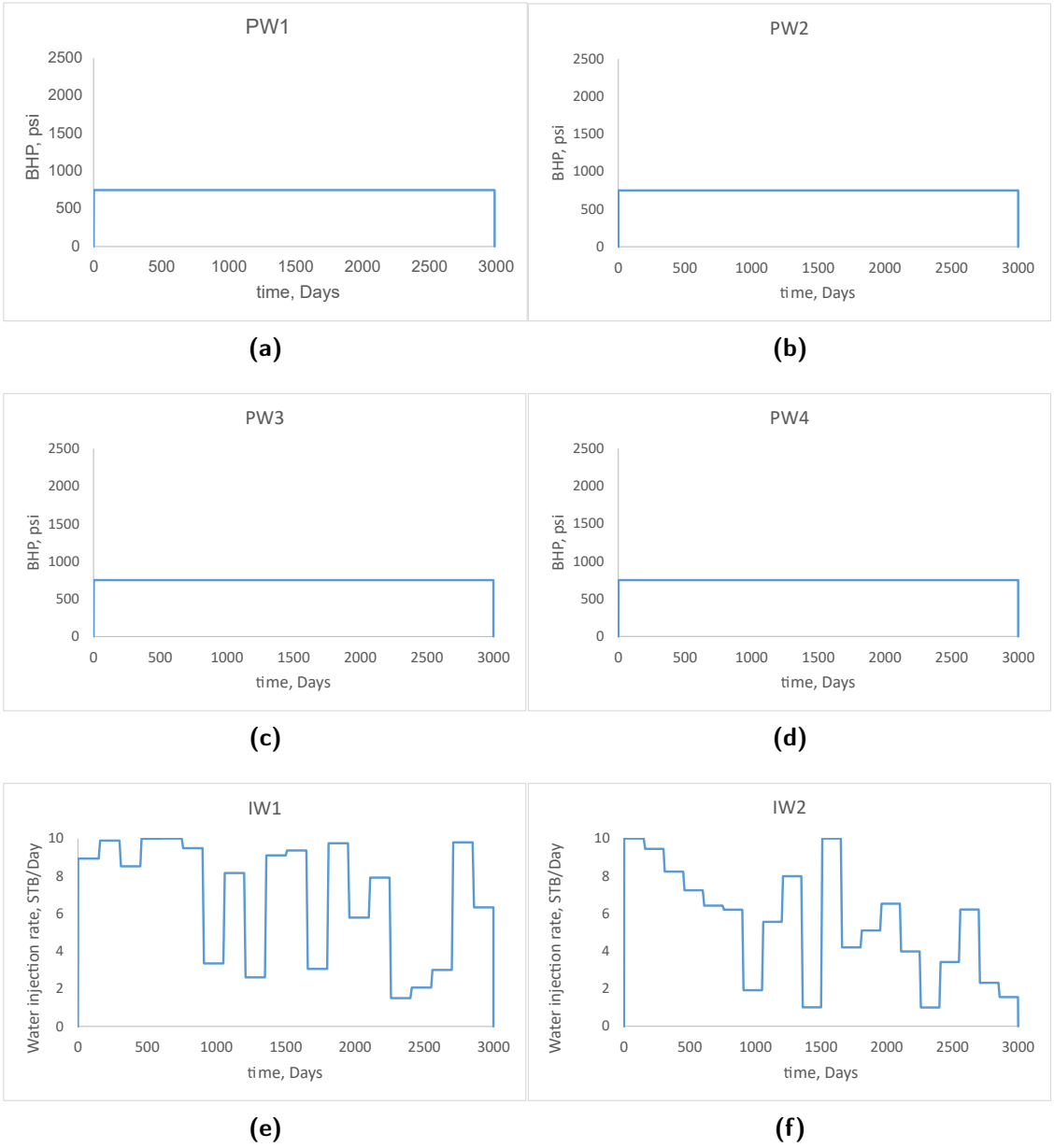


Figure 7.28: Optimum design variables for the simplex optimization method for Case-2-3000-BHP.

(Fig. 7.19) showed that we achieved higher NPV when we considered 1 cycle. In this case, we considered 5 cycles for production wells. The number of control time steps for injection wells is the same as other cases and equal to 20. The life of the production life-cycle is 3000 days. The total number of design variables is 100. The initial guess is again $\bar{\mathbf{u}}_0 = 0.5 \cdot \mathbf{1}_{N_u}$. Fig. 7.34 shows NPV vs. iterations for the simplex optimization method. The maximum NPV value is 11.3 million \$, which is 500 thousand \$ less than that for Case-2-3000 and 1.4

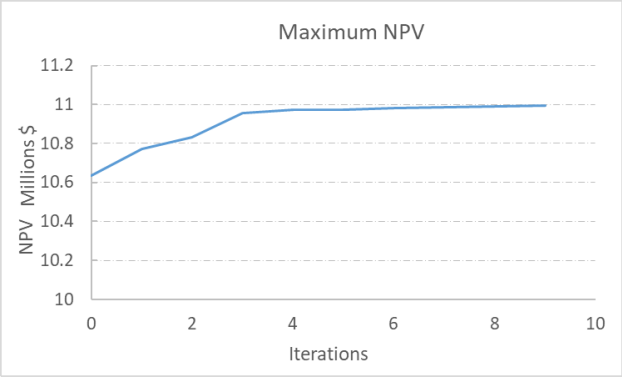


Figure 7.29: NPV vs. iterations obtained by using the simplex optimization method for Case-2-7000-BHP.

million $ higher than that for Case-1-3000. This result shows that, for this reservoir model and for 3000 days of the production period, as we increase the number of cycles for the production wells we achieve less maximum NPV as a result of optimization. The reason is that the number of design variables is large in this optimization case. As we know, the size of the design vector affects the optimization results because the accuracy of the approximate gradient becomes poor as the number of design variables increases. The results also show that the number of iterations required for the optimization algorithm to converge is not a function of the number of cycles for production wells (or it is the same thing saying that it is not a function of the number of design variables). Fig. 7.35 shows optimum design variables of Case-3-3000. All of the production wells have the number of shutoffs less than 5, especially production well PW4 has only 2 shutoffs. This result is consistent with the result of Case-1-3000 (Fig. 7.4), at which we observed that the production well PW4 has the least shutoffs. The optimum BHP results and shutoff periods are very similar to Case-2-3000. Therefore, the maximum NPV obtained for this case is close to that of Case-2-3000, and we achieved a higher maximum NPV than that of Case-1-3000.

Table 7.7 shows the number of simulations required for the optimization of each case as well as maximum NPVs. Even though the FD method required fewer optimization iterations than the simplex method, the number of simulations required for the optimization of FD methods was higher than that of the simplex method for both Case-2-3000 and Case-2-7000. The reason for this difference is that the FD method requires more simulation runs for the

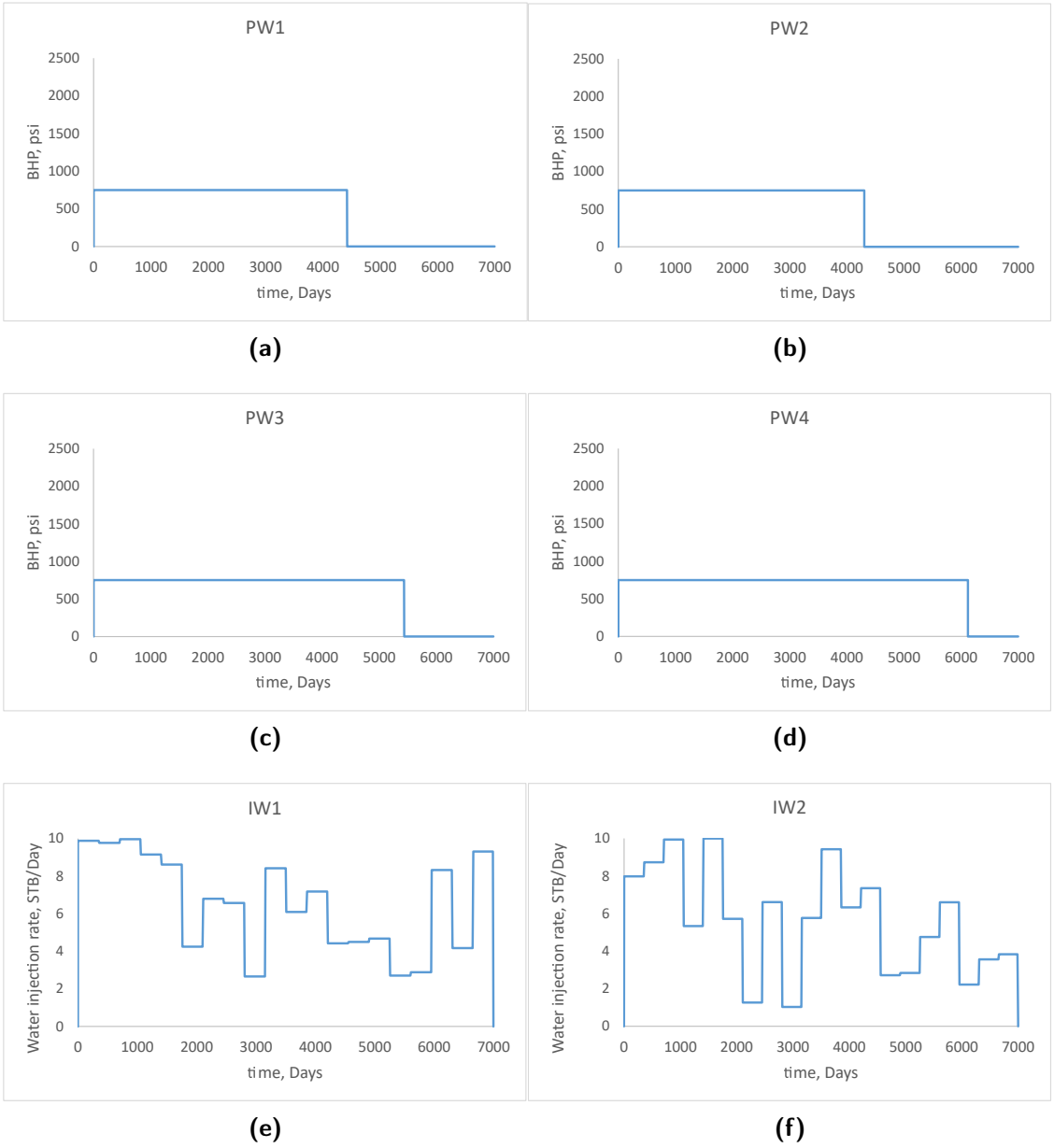


Figure 7.30: Optimum design variables for the simplex optimization method for Case-2-7000-BHP.

calculation of an approximate gradient. When we compare the cases having 10 and 5 cycles with the cases having a single cycle, we see that the number of simulations required for the optimization is much higher for the cases having 10 cycles. However, the maximum NPV found by the cases of a single cycle is slightly higher than the cases of 10 cycles. Therefore, for the type of reservoir and well configuration chosen for these cases, designing the problem with multiple cycles has more disadvantages than advantages over the problem with a single

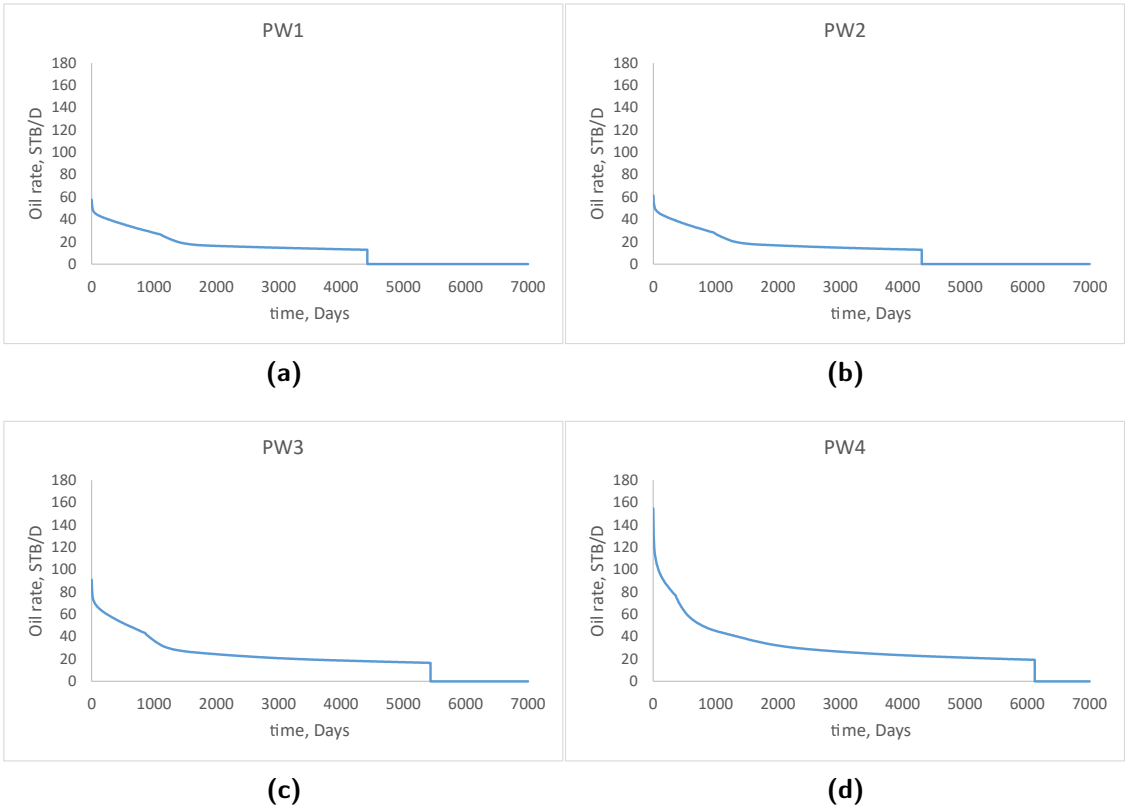


Figure 7.31: Oil production rate histories at the maximum NPV achieved for Case-2-7000-BHP.

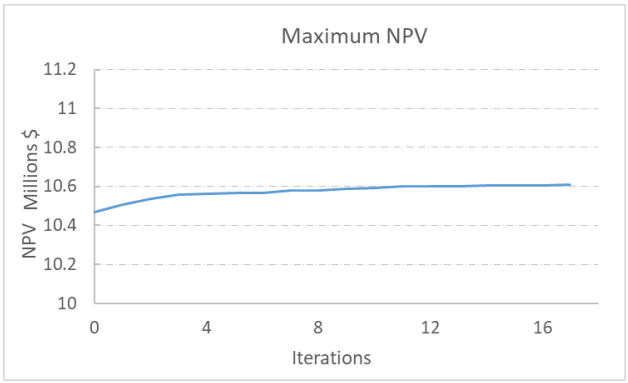
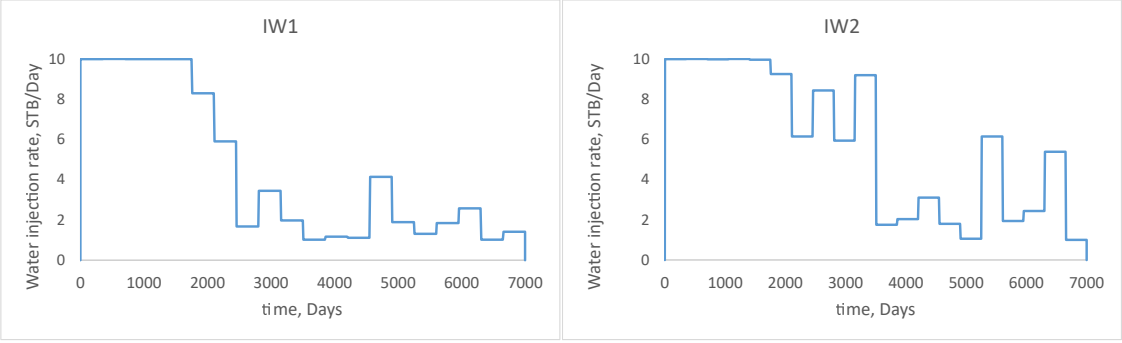


Figure 7.32: NPV vs. iterations obtained by using the simplex optimization method for Case-2-7000-BHP-TP.

cycle.

For the iterative-sampling-refinement optimization methods for Case-1-3000, we chose the best model which was LS-SVR trained with 140 training data points, and we referred to it as LS-SVR(140), and for Case-2-3000 we chose the LS-SVR model trained with 40 training data points and referred to it as LS-SVR(40) in the Table 7.7. As we can see from the results,



(a) **(b)**

Figure 7.33: Optimum design variables for the simplex optimization method for Case-2-7000-BHP-TP.

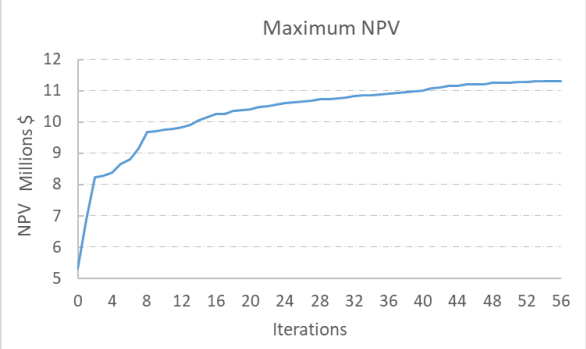


Figure 7.34: NPV vs. iterations obtained by using the simplex optimization method for Case-3-3000.

Table 7.7: Number of simulations required for optimization in the well shutoff problem.

Cases	Optimization method	Number of iterations	Number of simulations	NPV, million $
Case-1-3000	Simplex	29	625	10
Case-1-3000	LS-SVR(140)	73	213	11.74
Case-1-7000	Simplex	29	623	9.7
Case-1-3000-BHP	Simplex	20	426	11
Case-1-7000-BHP	Simplex	21	442	10.7
Case-2-3000	Simplex	32	393	11.8
Case-2-3000	LS-SVR(40)	14	54	11.8
Case-2-7000	Simplex	19	192	10.9
Case-2-3000	FD	18	888	11.9
Case-2-7000	FD	19	979	11
Case-2-3000-BHP	Simplex	10	121	11.9
Case-2-7000-BHP	Simplex	9	117	11
Case-2-7000-BHP-TP	Simplex	17	193	10.6
Case-3-3000	Simplex	56	1,177	11.3

for Case-1-3000 iterative-sampling-refinement optimization is approximately 3 times more computationally efficient and achieved 17% higher NPV. For Case-2-3000, with iterative-

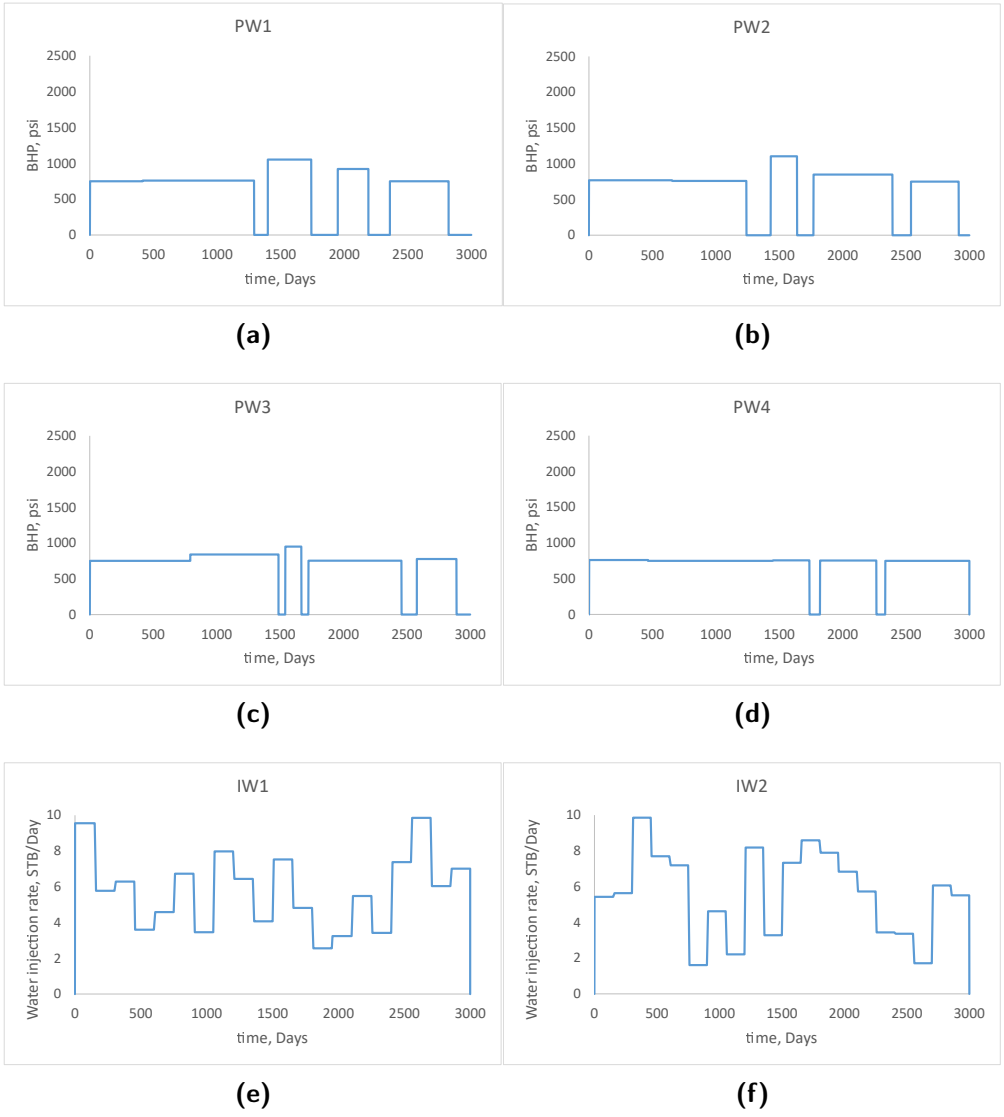


Figure 7.35: Optimum design variables for the simplex optimization method for Case-3-3000.

sampling-refinement optimization, we achieved 7 times more computational efficiency and 0.8% higher NPV than the simplex optimization method.

7.1.12 Pressure Responses

As noted before, Case-1-3000 and Case-2-3000 yield slightly different maximum values of NPV. Case-1-3000, where we considered 10 cycles for each production well, yield a lower local optimum than Case-2-3000. However, these local optimums are not far from each other. We observe similar pressure distribution of the reservoir at the end of the production life at optimum design values obtained for Case-1-3000 and Case-2-3000 (Fig. 7.36).

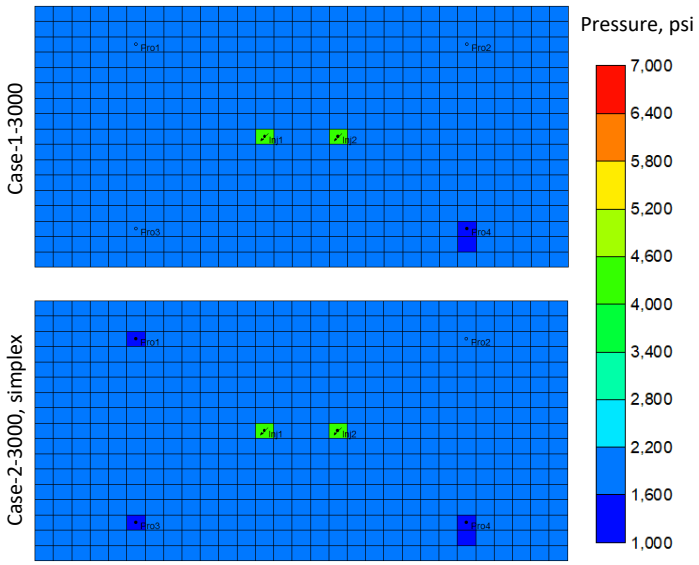


Figure 7.36: Pressure distribution of the reservoir at the end of the production life at optimum design values obtained for Case-1-3000 and Case-2-3000.

We also compared the well-block pressures of each production well vs time at the maximized values of NPV for both cases (Fig. 7.37). Both Case-1-3000 and Case-2-3000

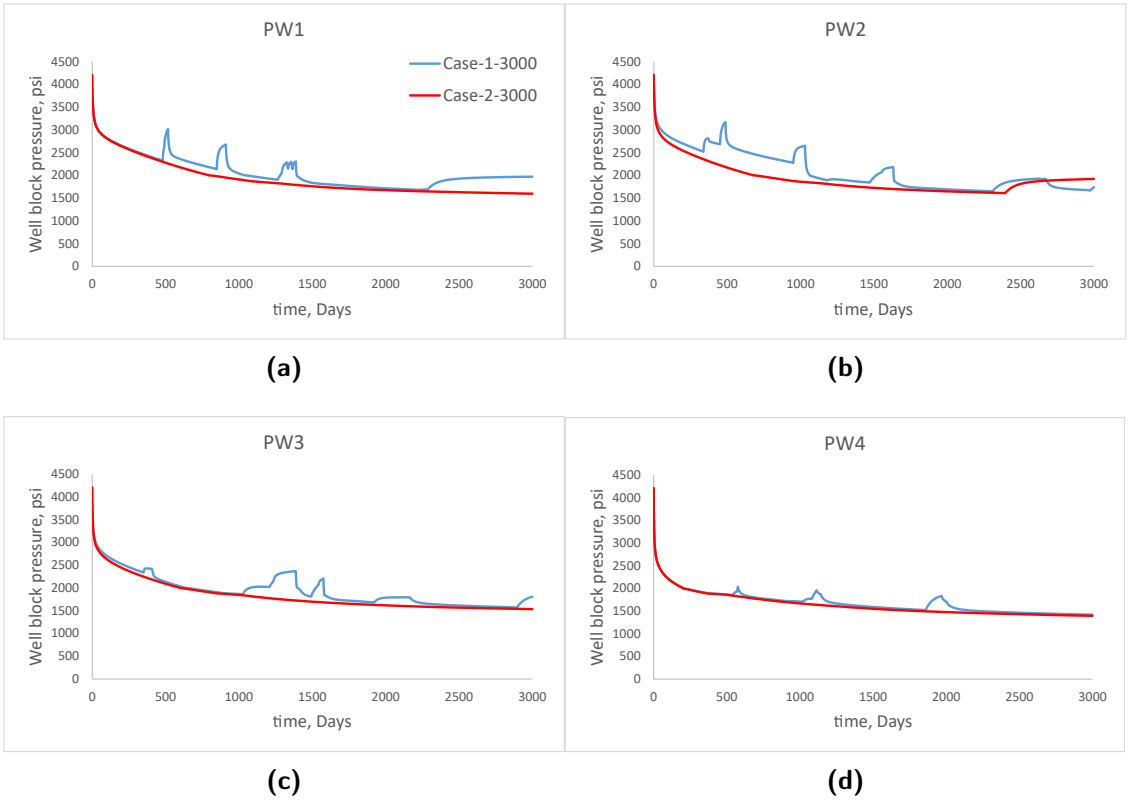


Figure 7.37: well-block pressures of production wells at optimum design variables obtained Case-1-3000 and Case-2-3000.

show similar reservoir pressure responses and well-block pressures at the maximum NPV values.

CHAPTER 8

CONCLUSIONS

In this study, we presented novel applications of the LS-SVR- and GPR-based iterative-sampling-refinement optimization methods to CO_2 HnP process in unconventional reservoirs, CO_2 WAG problem in conventional oil reservoirs (channelized reservoir in our research), and well shutoff problem where we investigated waterflooding process in tight oil reservoir including well shutoff option. Note that only in the well shutoff problem, the NPV formulation includes OPEX for production wells. In this chapter, we summarize our conclusions based on our results for each production optimization case. In Section 8.1, we discuss conclusions of HnP problem; in Section 8.2, for WAG problem; in Section 8.3, for well shutoff problem.

8.1 Conclusions for the Huff-and-Puff Problem

The following conclusions can be drawn from this study which focused on the production optimization of the miscible CO_2 huff-and-puff (HnP) process applied to an unconventional oil formation by using the LS-SVR- and GPR-based iterative-sampling-refinement optimization methods:

- Robust and deterministic life-cycle NPV of a single-well HnP process for a complex reservoir model, including geomechanics, molecular diffusion, natural fractures, etc., can be approximated well using the LS-SVR and GPR proxy models.
- The GPR and LS-SVR proxy models do not provide any superiority over each other in the optimization of NPV, except that GPR provides an assessment of uncertainty in the prediction model and the predicted value of NPV. It was found that as the number of inputs (or features) increases, the uncertainty in the NPV maximized by the GPR increases.

- The size of training data has a significant effect on optimization results. $MAE \leq 0.10$ and $R^2 \geq 0.8$ together are good indicators for choosing an appropriate initial LS-SVR or GPR proxy model to be used in the iterative-sampling-refinement optimization procedure.

- The LS-SVR- and GPR-based iterative-sampling-refinement optimization methods are at least 3 times more computationally efficient than a simplex gradient method using the high fidelity numerical simulator for the cases of both deterministic and robust optimization.

- Including the length of the cycle as a design variable makes NPV much rougher than including other design variables, and thus it requires more training data to achieve more accurate models.

- Including the length of the cycle as a design variable does not improve NPV for the reservoir model cases considered in this paper. However, we believe that this conclusion cannot be generalized and may be dependent on the reservoir model and production history before the HnP process.

8.2 Conclusions for the WAG Problem

- The WAG problem can be well approximated by LS-SVR- and GPR-based proxy models.

- However, in the cases where the ICVs are included as design variables, more training data points are needed to get an accurate LS-SVR or GPR proxy model to be used in the iterative-optimization method.

- Choosing upper and lower bounds of BHPs high can lead to the generation of poor training data which causes poor optimization results of the iterative-sampling-refinement optimization method.

- To solve this issue, we found that either lower down the bounds of production BHPs or choose flow rate as design variables instead of production BHPs. One other alternative is to fix production BHPs (which means BHPs are not design variables anymore).

- Using flow rate as a design variable instead of BHP at production wells improves optimization results of the LS-SVR- and GPR-based iterative-sampling-refinement optimization methods.

- We can optimize cycle lengths of production and injection wells, and gas injection time fractions.

- However, for the reservoir configuration considered in this study, we found that including cycle lengths did not improve maximum NPV

- Training GPR for the cases where a large size of design variable vector and for large training size with the Bayesian model selection requires very high computational time.

- To improve computational efficiency for training GPR, we recommend fixing the kernel function to the *rational quadratic kernel* and perform hyperparameter optimization only on hyperparameters of the kernel function.

- The rational quadratic kernel was found to be the most appealing kernel function found as a result of the Bayesian model selection method in this study.

- In general, for most of the production optimization cases of the WAG problem, the LS-SVR- and GPR-based iterative sampling-refinement optimization methods require less computational time than the StoSAG optimization method.

- However, for some of the production optimization cases of the WAG problem, where the size of the design variable is large and the size of training data is large, the GPR-based iterative-sampling-refinement optimization method was not more efficient. Therefore, for those optimization cases, we recommend one using the LS-SVR-based iterative-sampling-refinement optimization method.

8.3 Conclusions for the Well Shutoff Problem

- The well shutoff optimization problem can be handled by using the gradient-based optimization methods introducing a production time fraction as the design variable for each cycle.
- Having multiple cycles for production wells and making the cycle length unknown in addition to other design variables increase the number of design variables and make optimization problems to find a slightly lower local optimum. We think that this may be a result of inaccurate computation of gradients in the stochastic simplex method.
- However, for the cases having multiple cycles, lowering the initial backtracking step size close to the convergence helps us to get higher local optimums with smaller steps, but this requires additional computational time.
- Fixing production BHPs to their lower bound helps us to find slightly higher local optimums due to less number of design variables.
- For the single-cycle cases, when the life of the production is 3000 days, we do not observe production well shutoff, but this is case-specific.
- However, increasing the life of production to 7000 days for single-cycle cases causes production wells to shut off before the end of the production life.
- Not shutting off production wells for the case where life is 7000 days resulted in 500 thousand $ less NPV than the optimum found for this case, at which all production wells need to shut down before the end of production life.
- The value of the maximum NPV for the cases where the life of production is 3000 days is slightly more than that for the cases where the life of production is 7000 days.
- Reservoir pressure and well block pressure responses of the case with 1 cycle and the case with 10 cycles at the optimum design variables of the cases have similarities, which explains why we obtain close to the maximum NPV values for both cases.

- Finally, without making generalizations for all reservoir types and for all EOR methods, for this specific reservoir type and production scenario we found that using multiple shutoffs and making cycle length as unknown is not beneficial: it results in slightly lower maximum NPV compared to single shutoff cases; it requires more computational time than the single shutoff cases.

- The cases with 10 cycles where the number of design variables is large are sensitive to the initial guess of the simplex optimization method due to the increase in the number of local optimum points and inaccuracy of the gradient approximation with the simplex method.

- For the cases having 10 cycles and 1 cycle, iterative-sampling-refinement optimization is approximately 3 and 7 times more computationally efficient, respectively.

- For the case having 10 cycles, with iterative-sampling-refinement optimization, we obtained 17% higher NPV than that with the simplex optimization method.

- Using fixed hyperparameters for the LS-SVR-based iterative-sampling-refinement optimization method requires fewer iterations than that when we use hyperparameter optimization at each iteration.

8.4 General Conclusions

Based on the optimization results of the HnP, WAG, and well shutoff problems, we can make the following general conclusions:

- LS-SVR- and GPR-based optimization are more efficient than high-fidelity simulator-based optimization.

- LS-SVR and GPR proxies give accurate optimum results provided that initial proxies are sufficiently accurate.

- For the small-scale optimization problems considered in this study where the number of design variables is not large, using the GPR-based iterative-sampling-refinement

optimization is as efficient as using the LS-SVR-based iterative-sampling-refinement optimization.

- For the large-scale optimization problems considered in this study, LS-SVR is preferred over GPR due to its computational efficiency. Training GPR for large design size of design variable vector and for large training size using Bayesian model selection requires very high computational time.

- The uncertainty quantification feature of GPR seems not to be so important for the production optimization problems.

- To improve computational efficiency for training GPR we fix kernel to *rational quadratic kernel* and perform hyperparameter optimization only on hyperparameters of the kernel function.

- The rational quadratic kernel was found to be the most appealing kernel function as a result of Bayesian model selection in this study.

BIBLIOGRAPHY

Al-Khalifa, M. T., A. T. Mishkhes, K. N. Baruah, N. M. Al-Otaibi, et al., Smart well completion utilization to optimize production in mrc well-a case study, in *SPE Saudi Arabia Section Technical Symposium and Exhibition*, Society of Petroleum Engineers, 2013.

Al-Muailu, H., M. Al-Suwailem, S. Aldawsari, et al., Evaluating flow contributions and enhancing the design of smart well completions, in *SPE Middle East Oil and Gas Show and Conference*, Society of Petroleum Engineers, 2013.

Alfarge, D., M. Wei, B. Bai, A. Almansour, et al., Effect of molecular-diffusion mechanisim on co2 huff-n-puff process in shale-oil reservoirs, in *SPE Kingdom of Saudi Arabia Annual Technical Symposium and Exhibition*, Society of Petroleum Engineers. SPE-188003-MS. https://doi.org/10.2118/188003-MS, 2017.

Alharthy, N., T. W. Teklu, H. Kazemi, R. M. Graves, S. B. Hawthorne, J. Braunberger, B. Kurtoglu, et al., Enhanced oil recovery in liquid-rich shale reservoirs: laboratory to field, *SPE Reservoir Evaluation & Engineering*, **21**(01), 137–159. SPE–175,034–PA. https://doi.org/10.2118/175,034–PA, 2018.

Almasov, A., M. Onur, and A. C. Reynolds, Production optimization of the CO_2 huff and puff process in unconventional oil reservoirs using ls-svr and gpr based machine learning proxies. unpublished tuprep research report., in *TUPREP Annual Board Meeting 2020.*, Research Report No. 36, 2020a.

Almasov, A., M. Onur, and A. C. Reynolds, Production optimization of the CO_2 huff-n-puff process in an unconventional reservoir using a machine learning based proxy, in *SPE Improved Oil Recovery Conference*, Society of Petroleum Engineers. SPE-200360-MS. https://doi.org/10.2118/200360-MS, 2020b.

Alsyed, S. and K. Yateem, Testing methodology for smart wells completion toward attaining optimal production rate setting for maximum hydrocarbon recovery, in *IPTC 2013: International Petroleum Technology Conference*, pp. cp–350, European Association of Geoscientists & Engineers, 2013.

Artun, E., T. Ertekin, R. Watson, and M. Al-Wadhahi, Development of universal proxy models for screening and optimization of cyclic pressure pulsing in naturally fractured reservoirs, *Journal of Natural Gas Science and Engineering*, **3**(6), 667–686. IPTC–13,663–MS. https://doi.org/10.3997/2214–4609–pdb.151.iptc13,663, 2011.

Bahagio, D. N. T., *Ensemble Optimization of* CO_2 *WAG EOR*, Master's thesis, Delft University of Technology, the Netherlands. http://resolver.tudelft.nl/uuid:3b5ea1f5-f415-4c5e-8254-1b95b212125b, 2013.

Bender, S., M. Yilmaz, et al., Full-field simulation and optimization study of mature iwag injection in a heavy oil carbonate reservoir, in *SPE Improved Oil Recovery Symposium*, Society of Petroleum Engineers, 2014.

Berger, J. O., *Statistical decision theory and Bayesian analysis*, Springer Science & Business Media, 2013.

Bird, R. B., W. E. Stewart, and E. N. Lightfoot, *Transport phenomena*, John Wiley & Sons, 2007.

Bohacs, K., Q. Passey, M. Rudnicki, W. Esch, and O. Lazar, The spectrum of fine-grained reservoirs from shale gas to shale oil/tight liquids: essential attributes, key controls, practical characterization, in *IPTC 2013: International Petroleum Technology Conference*, pp. cp–350, European Association of Geoscientists & Engineers. https://doi.org/10.3997/2214-4609-pdb.350.iptc16676, 2013.

Bortz, D. M. and C. T. Kelley, The simplex gradient and noisy optimization problems, in *Computational methods for optimal design and control*, pp. 77–90, Springer, 1998.

Calisgan, T., E. Ozkan, T. Firincioglu, H. Sarak, C. Ozgen, et al., Impact of pore confinement on production behavior and gor profiles of unconventional reservoirs, in *SPE Annual Technical Conference and Exhibition*, Society of Petroleum Engineers. SPE-187316-MS. https://doi.org/10.2118/187316-MS, 2017.

Capolei, A., 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*, **125**, 23–37, 2015.

Caruana, R., Multitask learning, *Machine learning*, **28**(1), 41–75, 1997.

Cawley, G. C. and N. L. Talbot, Preventing over-fitting during model selection via bayesian regularisation of the hyper-parameters., *Journal of Machine Learning Research*, **8**(4), 2007.

Chan, K. S., R. Masoudi, H. Karkooti, R. Shaedin, and M. B. Othman, Production integrated smart completion benchmark for field re-development, in *IPTC 2014: International Petroleum Technology Conference*, pp. cp–395, European Association of Geoscientists & Engineers, 2014.

Chang, Y., Z. Bouzarkouna, and D. Devegowda, Multi-objective optimization for rapid and robust optimal oilfield development under geological uncertainty, *Computational Geosciences*, **19**(4), 933–950, 2015.

Chen, B., 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*, **153**, 331–344, 2017.

Chen, B. and A. C. Reynolds, Optimal control of icv's and well operating conditions for the water-alternating-gas injection process, *Journal of Petroleum Science and Engineering*, **149**, 623–640. https://doi.org/10.1016/j.petrol.2016.11.004, 2017.

Chen, B. and A. C. Reynolds, CO_2 water-alternating-gas injection for enhanced oil recovery:

Optimal well controls and half-cycle lengths, *Computers & Chemical Engineering*, **113**, 44–56. https://doi.org/10.1016/j.compchemeng.2018.03.006, 2018.

Chen, B., A. C. Reynolds, et al., Ensemble-based optimization of the wag injection process, in *SPE Reservoir Simulation Symposium*, Society of Petroleum Engineers, 2015.

Chen, C., *Adjoint-gradient-based production optimization with the augmented Lagrangian method*, Ph.D. thesis, University of Tulsa, 2011.

Chen, Y. and D. S. Oliver, Ensemble-based closed-loop optimization applied to brugge field, *SPE Reservoir Evaluation & Engineering*, **13**(01), 56–71, 2010.

Chen, Y., D. S. Oliver, D. Zhang, et al., Efficient ensemble-based closed-loop production optimization, *SPE Journal*, **14**(04), 634–645. SPE–112,873–PA. https://doi.org/10.2118/112,873–PA, 2009.

Christensen, J. R., E. H. Stenby, A. Skauge, et al., Review of wag field experience, in *International petroleum conference and exhibition of Mexico*, Society of Petroleum Engineers. SPE-71203-PA. https://doi.org/10.2118/71203-PA , 1998.

Clark, A. J., Determination of recovery factor in the bakken formation, mountrail county, nd, in *SPE Annual Technical Conference and Exhibition*, Society of Petroleum Engineers. SPE-133719-STU. https://doi.org/10.2118/133719-STU, 2009.

Conn, A. R., K. Scheinberg, and L. N. Vicente, *Introduction to derivative-free optimization*, SIAM, 2009.

Crone, S. F., J. Guajardo, and R. Weber, The impact of preprocessing on support vector regression and neural networks in time series prediction., in *DMIN*, pp. 37–44, https://doi.org/10.1.1.86.6266, 2006.

Custódio, A. L. and L. N. Vicente, Using sampling and simplex derivatives in pattern search methods, *SIAM Journal on Optimization*, **18**(2), 537–555, 2007.

De Brabanter, K., P. Karsmakers, F. Ojeda, C. Alzate, J. De Brabanter, K. Pelckmans, B. De Moor, J. Vandewalle, and J. A. Suykens, *LS-SVMlab toolbox user's guide: version 1.7*, Katholieke Universiteit Leuven, 2010.

Do, S. T. and A. C. Reynolds, Theoretical connections between optimization algorithms based on an approximate gradient, *Computational Geosciences*, **17**(6), 959–973. https://doi.org/DOI:10.1007/s10,596–013–9368–9, 2013.

Dongarra, J. and F. Sullivan, The top 10 algorithms (guest editors' intruduction), *Comput. Sci. Eng*, **2**, 22–23, 2000.

Dossary, F. M., S. A. Dawsari, and R. S. Anazi, Production gain and optimization through the implementation of smart well completion technology in saudi aramco, case study, in *SPE Intelligent Energy International*, OnePetro, 2012.

Drucker, H., C. J. Burges, L. Kaufman, A. J. Smola, and V. Vapnik, Support vector regression machines, in *Advances in neural information processing systems*, pp. 155–161, 1997.

Eaton, M. L., Multivariate statistics: a vector space approach., *JOHN WILEY & SONS, INC., 605 THIRD AVE., NEW YORK, NY 10158, USA, 1983, 512*, 1983.

Feng, T., O. Leeuwenburgh, C. Hewson, and R. Hanea, Joint optimization of field development and water-alternating-gas recovery strategies, in *IOR 2017-19th European Symposium on Improved Oil Recovery*, vol. 2017, pp. 1–12, European Association of Geoscientists & Engineers, 2017.

Fick, A., Ueber diffusion, *Annalen der Physik*, **170**(1), 59–86, 1855.

Firincioglu, T., E. Ozkan, C. Ozgen, et al., Thermodynamics of multiphase flow in unconventional liquids-rich reservoirs, in *SPE Annual Technical Conference and Exhibition*, Society of Petroleum Engineers. SPE-159869-MS. https://doi.org/10.2118/159869-MS, 2012.

Fonseca, R., O. Leeuwenburgh, E. D. Rossa, P. M. Van den Hof, J.-D. Jansen, et al., Ensemble-based multi-objective optimization of on-off control devices under geological uncertainty, in *SPE Reservoir Simulation Symposium*, Society of Petroleum Engineers, 2015a.

Fonseca, R. M., B. Chen, J. D. Jansen, and A. Reynolds, A stochastic simplex approximate gradient (stosag) for optimization under uncertainty, *International Journal for Numerical Methods in Engineering*, **109**(13), 1756–1776. https://doi.org/10.1002/nme.5342, 2017.

Fonseca, R. M., S. Kahrobaei, L. Van Gastel, O. Leeuwenburgh, J. D. Jansen, et al., Quantification of the impact of ensemble size on the quality of an ensemble gradient using principles of hypothesis testing, in *SPE Reservoir Simulation Symposium*, Society of Petroleum Engineers. SPE-173236-MS.https://doi.org/SPE-173236-MS, 2015b.

Fonseca, R. M., O. Leeuwenburgh, E. Della Rossa, P. M. Van den Hof, J. D. Jansen, et al., Ensemble-based multiobjective optimization of on/off control devices under geological uncertainty, *SPE Reservoir Evaluation & Engineering*, **18**(04), 554–563. SPE–173,268–MS. https://doi.org/SPE–173,268–MS, 2015c.

Fonseca, R. M., O. Leeuwenburgh, P. Van den Hof, and J. D. Jansen, Improving the ensemble-optimization method through covariance-matrix adaptation, *SPE Journal*, **20**(01), 155–168. SPE–163,657–PA. https://doi.org/10.2118/163,657–PA, 2015d.

Forouzanfar, F., E. Della Rossa, R. Russo, and A. C. Reynolds, Life-cycle production optimization of an oil field with an adjoint-based gradient approach, *Journal of Petroleum Science and Engineering*, **112**, 351–358, 2013.

Forouzanfar, F., G. Li, A. C. Reynolds, et al., A two-stage well placement optimization method based on adjoint gradient, in *SPE annual technical conference and exhibition*, Society of Petroleum Engineers. SPE-135304-MS. https://doi.org/10.2118/135304-MS, 2010.

Forrester, A., A. Sobester, and A. Keane, *Engineering design via surrogate modelling: a practical guide*, John Wiley & Sons, 2008.

Forrester, A. I. and A. J. Keane, Recent advances in surrogate-based optimization, *Progress in aerospace sciences*, **45**(1-3), 50–79. https://doi.org/10.1016/j.paerosci.2008.11.001, 2009.

Fragoso, A., K. Selvan, R. Aguilera, et al., An investigation on the feasibility of combined refracturing of horizontal wells and huff and puff gas injection for improving oil recovery from shale petroleum reservoirs, in *SPE Improved Oil Recovery Conference*, Society of Petroleum Engineers. SPE-190284-MS. https://doi.org/10.2118/190284-MS, 2018.

Gala, D., M. Sharma, et al., Compositional and geomechanical effects in huff-n-puff gas injection ior in tight oil reservoirs, in *SPE Annual Technical Conference and Exhibition*, Society of Petroleum Engineers. SPE-191488-MS. https://doi.org/10.2118/191488-MS, 2018.

Gamadi, T., J. Sheng, M. Soliman, H. Menouar, M. Watson, H. Emadibaladehi, et al., An experimental study of cyclic co2 injection to improve shale oil recovery, in *SPE improved oil recovery symposium*, Society of Petroleum Engineers. SPE-169142-MS. https://doi.org/10.2118/169142-MS, 2014.

Ganjdanesh, R., W. Yu, M. X. Fiallos Torres, K. Sepehrnoori, E. Kerr, R. Ambrose, et al., Huff-n-puff gas injection for enhanced condensate recovery in eagle ford, in *SPE Annual Technical Conference and Exhibition*, Society of Petroleum Engineers. SPE-195996-MS. https://doi.org/10.2118/195996-MS, 2019.

GEM, C.-C. M. G., Compositional and unconventional reservoir simulator, 2016.

GEM, C.-C. M. G., Compositional and unconventional reservoir simulator, 2020.

Ghaderi, S. M., C. R. Clarkson, Y. Chen, et al., Optimization of wag process for coupled co2 eor-storage in tight oil formations: an experimental design approach, in *SPE Canadian Unconventional Resources Conference*, Society of Petroleum Engineers, 2012.

Glasserman, P. and Y.-C. Ho, *Gradient estimation via perturbation analysis*, vol. 116, Springer Science & Business Media, 1991.

Golub, G. H., L. VAN, and M. C. CF, Baltimore, md, 1989.

Gordan, M., A. Georgakis, O. Tsatos, G. Oltean, and L. Miclea, Computational complexity reduction of the support vector machine classifiers for image analysis tasks through the use of the discrete cosine transform, in *2006 IEEE International Conference on Automation, Quality and Testing, Robotics*, vol. 2, pp. 350–355. 10.1109/AQTR.2006.254,658, 2006.

Gosset, R., G. Heyen, and B. Kalitventzeff, An efficient algorithm to solve cubic equations of state, *Fluid Phase Equilibria*, **25**(1), 51–64, 1986.

Gramacy, R. B., *Surrogates: Gaussian Process Modeling, Design, and Optimization for the Applied Sciences*, CRC Press, 2020.

Guo, Z. and A. C. Reynolds, Robust life-cycle production optimization with a support-vector-regression proxy, *SPE Journal. SPE-191378-PA. https://doi.org/10.2118/191378-PA*, 2018.

Hamdi, H., C. R. Clarkson, A. Esmail, M. Costa Sousa, et al., A bayesian approach for optimizing the huff-n-puff gas injection performance in shale reservoirs under parametric uncertainty: A duvernay shale example, in *SPE Europec featured at 81st EAGE Conference and Exhibition*, Society of PetrolFeum Engineers. SPE-195438-MS. https://doi.org/10.2118/195438-MS, 2019.

Hewson, C., O. Leeuwenburgh, et al., Co2 water-alternating-gas flooding optimization of the chigwell viking i pool in the western canadian sedimentary basin, in *SPE Reservoir Simulation Conference*, Society of Petroleum Engineers. SPE-182597-MS. https://doi.org/10.2118/182597-MS, 2017.

Hoffman, B. T. et al., Huff-n-puff gas injection pilot projects in the eagle ford, in *SPE Canada Unconventional Resources Conference*, Society of Petroleum Engineers. SPE-189816-MS. https://doi.org/10.2118/189816-MS, 2018.

IMEX, C.-C. M. G., Black oil and unconventional reservoir simulator, 2020.

Isebor, O. J., D. Echeverría Ciaurri, L. J. Durlofsky, et al., Generalized field-development optimization with derivative-free procedures, *SPE Journal*, **19**(05), 891–908, 2014.

Israelachvili, J. N., *Intermolecular and surface forces*, Academic press, 2011.

James, G., D. Witten, T. Hastie, and R. Tibshirani, *An Introduction to Statistical Learning*, vol. 112, Springer, 2013.

Jansen, J. D., Adjoint-based optimization of multi-phase flow through porous media–a review, *Computers & Fluids*, **46**(1), 40–51, 2011.

Jansen, J.-D., R. Brouwer, and S. G. Douma, Closed loop reservoir management, in *SPE reservoir simulation symposium*, OnePetro, 2009.

Jarrell, P. M., C. E. Fox, M. H. Stein, and S. L. Webb, *Practical aspects of CO2 flooding*, vol. 22, Society of Petroleum Engineers Richardson, TX, 2002.

Jia, B., J.-S. Tsau, R. Barati, et al., Role of molecular diffusion in heterogeneous shale reservoirs during co2 huff-n-puff, in *SPE Europec featured at 79th EAGE Conference and Exhibition*, Society of Petroleum Engineers, 2017.

Jin, H. and S. A. Sonnenberg, *Source rock potential of the Bakken shales in the Williston Basin, North Dakota and Montana*, Ph.D. thesis, Colorado School of Mines Golden, Colorado, 2014.

Job, G. and F. Herrmann, Chemical potential—a quantity in search of recognition, *European journal of physics*, **27**(2), 353, 2006.

Johns, R. T., L. Bermudez, H. Parakh, et al., Wag optimization for gas floods above the mme, in *SPE Annual Technical Conference and Exhibition*, Society of Petroleum Engineers, 2003.

Joslin, K., A. Abraham, T. Thaker, V. Pathak, A. Kumar, et al., Viability of eor processes in the bakken under geological and economic uncertainty, in *SPE Canada Unconventional Resources Conference*, Society of Petroleum Engineers. SPE-189779-MS. https://doi.org/10.2118/189779-MS, 2018.

Kanfar, M., C. Clarkson, et al., Factors affecting huff-n-puff efficiency in hydraulically-fractured tight reservoirs, in *SPE Unconventional Resources Conference*, Society of Petroleum Engineers, 2017.

Kantzas, A., J. Bryan, and S. Taheri, Fundamentals of fluid flow in porous media, *Pore size distribution*, 2012.

Kelley, C. T., Detection and remediation of stagnation in the nelder–mead algorithm using a sufficient decrease condition, *SIAM journal on optimization*, **10**(1), 43–55, 1999.

Kraaijevanger, J., P. Egberts, J. Valstar, H. Buurman, et al., Optimal waterflood design using the adjoint method, in *SPE Reservoir Simulation Symposium*, Society of Petroleum Engineers, 2007.

Lake, L. W., R. Johns, B. Rossen, and G. Pope, *Fundamentals of Enhanced Oil Recovery*, Society of Petroleum Engineers, Texas, USA, 2014.

Liu, H., Y.-S. Ong, X. Shen, and J. Cai, When gaussian process meets big data: A review of scalable gps, *IEEE Transactions on Neural Networks and Learning Systems. https://doi.org/10.1109/TNNLS.2019.2957109*, 2020.

Liu, Z. and A. C. Reynolds, Gradient-enhanced support vector regression for robust life-cycle production optimization with nonlinear-state constraints, *SPE Journal*, pp. 1–24, 2020.

Lorentzen, R. J., A. Berg, G. Nævdal, E. H. Vefring, et al., A new approach for dynamic optimization of water flooding problems, in *Intelligent Energy Conference and Exhibition*, Society of Petroleum Engineers, 2006.

Lu, R., F. Forouzanfar, A. C. Reynolds, et al., Bi-objective optimization of well placement and controls using stosag, in *SPE Reservoir Simulation Conference*, Society of Petroleum Engineers, 2017.

Lu, R. and A. C. Reynolds, Joint optimization of well locations, types, drilling order, and controls given a set of potential drilling paths, *SPE Journal*, **25**(03), 1285–1306, 2020.

Luenberger, D. G., Y. Ye, et al., *Linear and nonlinear programming*, vol. 2, Springer, 1984.

Ma, Y. and A. Jamili, Using simplified local density/peng-robinson equation of state to study the effects of confinement in shale formations on phase behavior, in *SPE Unconventional Resources Conference*, Society of Petroleum Engineers. SPE-168986-MS. https://doi.org/10.2118/168986-MS, 2014.

McKay, M. D., R. J. Beckman, and W. J. Conover, A comparison of three methods for selecting values of input variables in the analysis of output from a computer code, *Technometrics*, **42**(1), 55–61, https://doi.org/10.1080/00401,706.1979.10489,755, 2000.

Muirhead, R. J., *Aspects of multivariate statistical theory*, vol. 197, John Wiley & Sons, 2009.

Naroso, G., J. P. Hofland, et al., How smart completion can maximize oil production and recovery factor in stacked-marginal reservoirs, in *Abu Dhabi International Petroleum Exhibition and Conference*, Society of Petroleum Engineers, 2010.

Naus, M., N. Dolle, J.-D. Jansen, et al., Optimization of commingled production using infinitely variable inflow control valves, in *SPE Annual Technical Conference and Exhibition*, Society of Petroleum Engineers, 2004.

Nojabaei, B., R. T. Johns, L. Chu, et al., Effect of capillary pressure on phase behavior in tight rocks and shales, *SPE Reservoir Evaluation & Engineering*, **16**(03), 281–289, 2013.

Nwachukwu, A., H. Jeong, A. Sun, M. Pyrcz, L. W. Lake, et al., Machine learning-based optimization of well locations and wag parameters under geologic uncertainty, in *SPE improved oil recovery conference*, Society of Petroleum Engineers, 2018.

Nwaozo, J. E., *Dynamic optimization of a water flood reservoir*, Ph.D. thesis, University of Oklahoma Norman, OK, 2006.

Orozco, D., A. Fragoso, K. Selvan, G. Noble, R. Aguilera, et al., Eagle ford huff 'n'puff gas-injection pilot: Comparison of reservoir-simulation, material balance, and real per-

formance of the pilot well, *SPE Reservoir Evaluation & Engineering. SPE-191575-PA. https://doi.org/10.2118/191575-PA*, 2019.

Orr, F. M. et al., *Theory of gas injection processes*, vol. 5, Tie-Line Publications Copenhagen, 2007.

Pankaj, P., H. Mukisa, I. Solovyeva, H. Xue, et al., Boosting oil recovery in naturally fractured shale using co2 huff-n-puff, in *SPE Argentina Exploration and Production of Unconventional Resources Symposium*, Society of Petroleum Engineers. SPE-191823-MS. https://doi.org/10.2118/191823-MS, 2018.

Pari, M. N. and A. H. Kabir, Viability study of implementing smart/intelligent completion in commingled wells in an australian offshore oil field, in *SPE Digital Energy Conference and Exhibition*, OnePetro, https://doi.org/10.2118/122654-MS, 2009.

Peng, D.-Y. and D. B. Robinson, A new two-constant equation of state, *Industrial & Engineering Chemistry Fundamentals*, **15**(1), 59–64, 1976.

Queipo, N. V., R. T. Haftka, W. Shyy, T. Goel, R. Vaidyanathan, and P. K. Tucker, Surrogate-based analysis and optimization, *Progress in aerospace sciences*, **41**(1), 1–28. doi:10.1016/j.paerosci.2005.02.001, 2005.

Quinonero-Candela, J., C. E. Rasmussen, and C. K. Williams, Approximation methods for gaussian process regression, in *Large-scale kernel machines*, pp. 203–223, MIT Press, 2007.

Raniolo, S., L. Dovera, A. Cominelli, C. Callegaro, and F. Masserano, History match and polymer injection optimization in a mature field using the ensemble kalman filter, in *IOR 2013-17th European Symposium on Improved Oil Recovery*, pp. cp–342, European Association of Geoscientists & Engineers, 2013.

Sarma, P., L. J. Durlofsky, K. Aziz, and W. H. Chen, Efficient real-time reservoir management using adjoint-based optimal control and model updating, *Computational Geosciences*, **10**(1), 3–36, 2006.

Saunders, C., A. Gammerman, and V. Vovk, Ridge regression learning algorithm in dual variables, *15th International Conference on Machine Learning, ICML '98*, 1998.

Seeger, M., Gaussian processes for machine learning, *International journal of neural systems*, **14**(02), 69–106, 2004.

Song, C., D. Yang, et al., Performance evaluation of co2 huff-n-puff processes in tight oil formations, in *SPE Unconventional Resources Conference Canada*, Society of Petroleum Engineers. SPE-167217-MS. https://doi.org/10.2118/167217-MS, 2013.

Stone, R. E. and C. A. Tovey, The simplex and projective scaling algorithms as iteratively reweighted least squares methods, *SIAM review*, **33**(2), 220–237, 1991.

Sudaryanto, B. and Y. C. Yortsos, Optimization of fluid front dynamics in porous media using rate control. i. equal mobility fluids, *Physics of Fluids*, **12**(7), 1656–1670. https://doi.org/10.1063/1.870,417, 2000.

Sun, J., A. Zou, E. Sotelo, and D. Schechter, Numerical simulation of co2 huff-n-puff in complex fracture networks of unconventional liquid reservoirs, *Journal of Natural Gas Science and Engineering*, **31**, 481–492. https://doi.org/10.1016/j.jngse.2016.03.032, 2016.

Suykens, J. A., J. De Brabanter, L. Lukas, and J. Vandewalle, Weighted least squares support vector machines: robustness and sparse approximation, *Neurocomputing*, **48**(1-4), 85–105, 2002.

Suykens, J. A. and J. Vandewalle, Least squares support vector machine classifiers, *Neural processing letters*, **9**(3), 293–300. https://doi.org/10.1023/A:1018628609,742, 1999.

Tabatabaei Nejad, S. A., A. A. V. Aleagha, S. Salari, et al., Estimating optimum well spacing in a middle east onshoreoil field using a genetic algorithm optimization approach, in *SPE Middle East Oil and Gas Show and Conference*, Society of Petroleum Engineers, 2007.

Tavakkolian, M., F. Jalali, M. Emadi, et al., Production optimization using genetic algorithm approach, in *Nigeria Annual International Conference and Exhibition*, Society of Petroleum Engineers, 2004.

The MathWorks, I., Gaussian process regression, 2020.

Todd, H. B., J. G. Evans, et al., Improved oil recovery ior pilot projects in the bakken formation, in *SPE Low Perm Symposium*, Society of Petroleum Engineers. SPE-180270-MS. https://doi.org/10.2118/180270-MS, 2016.

Tovar, F. D., M. A. Barrufet, and D. S. Schechter, Gas injection for eor in organic rich shale. part i: Operational philosophy, in *SPE improved oil recovery conference*, Society of Petroleum Engineers. SPE-190323-MS. https://doi.org/10.2118/190323-MS, 2018a.

Tovar, F. D., M. A. Barrufet, and D. S. Schechter, Gas injection for eor in organic rich shales. part ii: Mechanisms of recovery, in *Unconventional Resources Technology Conference, Houston, Texas, 23-25 July 2018*, pp. 2953–2973, Society of Petroleum Engineers, Society of Exploration Geophysicists, American Association of Petroleum. https://doi.org/10.15530/urtec-2018-2903026, 2018b.

van Essen, G., M. Zandvliet, P. Van den Hof, O. Bosgra, and J.-D. Jansen, Robust waterflooding optimization of multiple geological scenarios, *Spe Journal*, **14**(01), 202–210, 2009.

Vapnik, V., The support vector method of function estimation, in *Nonlinear Modeling*, pp. 55–85. https://doi.org/10.1007/978–1–4615–5703–6–3, Springer, 1998.

Vapnik, V., I. Guyon, and T. Hastie, Support vector machines, *Mach. Learn*, **20**(3), 273–297. https://doi.org/10.1007/978–0–387–30,162–4–415, 1995.

Vetterling, W. T., W. T. Vetterling, W. H. Press, W. H. Press, S. A. Teukolsky, B. P. Flannery, and B. P. Flannery, *Numerical Recipes: Example Book C*, Cambridge University Press, 1992.

von Mises, R., *Mathematical theory of probability and statistics*, Academic Press, 1964.

Wahba, G., *Spline models for observational data*, SIAM, 1990.

Wang, L., W. Yu, et al., Gas huff and puff process in eagle ford shale: Recovery mechanism study and optimization, in *SPE Oklahoma City Oil and Gas Symposium*, Society of Petroleum Engineers. SPE-195185-MS. https://doi.org/10.2118/195185-MS, 2019.

Wang, Y., J. Hou, Z. Song, D. Yuan, J. Zhang, T. Zhao, et al., A case study on simulation of in-situ co2 huff-'n'-puff process, *SPE Reservoir Evaluation & Engineering*, **21**(01), 109–121. CMTC–486,365–MS. https://doi.org/10.7122/486,365–MS, 2018.

Williams, C. K., Prediction with gaussian processes: From linear regression to linear prediction and beyond, in *Learning in graphical models*, pp. 599–621, Springer, 1998.

Williams, C. K. and C. E. Rasmussen, *Gaussian processes for machine learning*, vol. 2, MIT press Cambridge, MA, 2006.

WINPROP, C., Winprop user's manual, *Calgary, Alberta, Canada*, 2004.

Wright, S. and J. Nocedal, Numerical optimization, *Springer Science*, **35**(67-68), 7, 1999.

Yu, W., H. Lashgari, K. Sepehrnoori, et al., Simulation study of co2 huff-n-puff process in bakken tight oil reservoirs, in *SPE Western North American and Rocky Mountain Joint Meeting*, Society of Petroleum Engineers. SPE-169575-MS. https://doi.org/10.2118/169575-MS, 2014.

Yu, W., Y. Zhang, A. Varavei, K. Sepehrnoori, T. Zhang, K. Wu, J. Miao, et al., Compositional simulation of co2 huff'n'puff in eagle ford tight oil reservoirs with co 2 molecular diffusion, nanopore confinement, and complex natural fractures, *SPE Reservoir Evaluation & Engineering. SPE-190325-PA. https://doi.org/10.2118/190325-PA*, 2019.

Yu, Y., L. Li, J. J. Sheng, et al., Further discuss the roles of soaking time and pressure depletion rate in gas huff-n-puff process in fractured liquid-rich shale reservoirs, in *SPE Annual*

Technical Conference and Exhibition, Society of Petroleum Engineers. SPE-181471-MS. https://doi.org/10.2118/181471-MS, 2016.

Zhou, D., M. Yan, W. M. Calvin, et al., Optimization of a mature co2 flood-from continuous injection to wag, in *SPE Improved Oil Recovery Symposium*, Society of Petroleum Engineers, 2012.

Zhu, G.-S. and R. D. Reitz, A model for high-pressure vaporization of droplets of complex liquid mixtures using continuous thermodynamics, *International Journal of Heat and Mass Transfer*, **45**(3), 495–507, 2002.

APPENDIX A

COMPARISON OF THE NPV WITH CONTROL TIME STEP AND SIMULATION TIME STEP

To check if both NPV equations (Eqs. 2.5 and 2.10) yield the same result, let us look at the following example case (Table A.1). We will not get numeric values in comparison

Table A.1: Design variables and time step values in example case.

N_c	2
Δt^n for n=1,2	600
$\widehat{\Delta t}_p^n$ for n=1,2	0.5
$\widehat{\Delta t}_i^n$ for n=1,2	0.25
N_i^n for n=1,2	3
N_p^n for n=1,2	3

but we want to end up with the same equation using both Eqs. 2.5 and 2.10. As one can calculate the values for the time steps will be:

$\delta\Delta_{i,j}^n = 100$ Days, for $j = 1, 2, 3$ and $n = 1, 2$, and $\delta\Delta_{p,j}^n = 50$ Days, for $j = 1, 2, 3$ and $n = 1, 2$.

First, let us calculate $t_{p,j}^n$ and $t_{i,j}^n$ values for $n = 1, 2$ and $j = 1, 2, 3$:

$$t_{p,1}^1 = \sum_{l=1}^{1} \Delta t^{l-1} + \sum_{k=1}^{1}(\Delta t^l \cdot (1 - \widehat{\Delta t}_p^l) + \delta\Delta t_{p,k}^l) = \Delta t^0 + (\Delta t^1 \cdot (1 - \widehat{\Delta t}_p^1) + \delta\Delta t_{p,1}^1) = 0 + (600 \cdot (1 - 0.5) + 100) = 400 \text{ Days},$$

$$t_{p,2}^1 = \sum_{l=1}^{1} \Delta t^{l-1} + \sum_{k=1}^{2}(\Delta t^l \cdot (1 - \widehat{\Delta t}_p^l) + \delta\Delta t_{p,k}^l) = \Delta t^0 + (\Delta t^1 \cdot (1 - \widehat{\Delta t}_p^1) + \delta\Delta t_{p,1}^1 + \delta\Delta t_{p,2}^1) = 0 + (600 \cdot (1 - 0.5) + 100 + 100) = 500 \text{ Days},$$

$$t_{p,3}^1 = \sum_{l=1}^{1} \Delta t^{l-1} + \sum_{k=1}^{3}(\Delta t^l \cdot (1 - \widehat{\Delta t}_p^l) + \delta\Delta t_{p,k}^l) = \Delta t^0 + (\Delta t^1 \cdot (1 - \widehat{\Delta t}_p^1) + \delta\Delta t_{p,1}^1 + \delta\Delta t_{p,2}^1 + \delta\Delta t_{p,3}^1)$$

$$= 0 + (600 \cdot (1 - 0.5) + 100 + 100 + 100) = 600 \text{ Days},$$

$$t^2_{p,1} = \sum_{l=1}^{2} \Delta t^{l-1} + \sum_{k=1}^{1} (\Delta t^l \cdot (1 - \widehat{\Delta t}^l_p) + \delta \Delta t^l_{p,k}) =$$
$$\Delta t^0 + \Delta t^1 + (\Delta t^2 \cdot (1 - \widehat{\Delta t}^2_p) + \delta \Delta t^2_{p,1}) = 0 + 600 + (600 \cdot (1 - 0.5) + 100) = 1000 \text{ Days},$$

$$t^2_{p,2} = \sum_{l=1}^{2} \Delta t^{l-1} + \sum_{k=1}^{2} (\Delta t^l \cdot (1 - \widehat{\Delta t}^l_p) + \delta \Delta t^l_{p,k}) =$$
$$\Delta t^0 + \Delta t^1 + (\Delta t^2 \cdot (1 - \widehat{\Delta t}^2_p) + \delta \Delta t^2_{p,1} + \delta \Delta t^2_{p,2}) = 0 + 600 + (600 \cdot (1 - 0.5) + 100 + 100) = 1100$$
$$\text{Days},$$

$$t^2_{p,3} = \sum_{l=1}^{2} \Delta t^{l-1} + \sum_{k=1}^{3} (\Delta t^l \cdot (1 - \widehat{\Delta t}^l_p) + \delta \Delta t^l_{p,k}) = \Delta t^0 + \Delta t^1 + (\Delta t^2 \cdot (1 - \widehat{\Delta t}^2_p) +$$
$$\delta \Delta t^2_{p,1} + \delta \Delta t^2_{p,2} + \delta \Delta t^2_{p,3}) = 0 + 600 + (600 \cdot (1 - 0.5) + 100 + 100 + 100) = 1200 \text{ Days}.$$

$$t^1_{i,1} = \sum_{l=1}^{1} \Delta t^{l-1} + \sum_{k=1}^{1} \delta \Delta t^l_{i,k} = \Delta t^0 + \delta \Delta t^1_{i,1} = 0 + 50 = 50 \text{ Days},$$

$$t^1_{i,2} = \sum_{l=1}^{1} \Delta t^{l-1} + \sum_{k=1}^{2} \delta \Delta t^l_{i,k} = \Delta t^0 + \delta \Delta t^1_{i,1} + \delta \Delta t^1_{i,2} = 0 + 50 + 50 = 100 \text{ Days},$$

$$t^1_{i,3} = \sum_{l=1}^{1} \Delta t^{l-1} + \sum_{k=1}^{3} \delta \Delta t^l_{i,k} = \Delta t^0 + \delta \Delta t^1_{i,1} + \delta \Delta t^1_{i,2} + \delta \Delta t^1_{i,3} = 0 + 50 + 50 + 50 = 150$$
$$\text{Days},$$

$$t^2_{i,1} = \sum_{l=1}^{2} \Delta t^{l-1} + \sum_{k=1}^{1} \delta \Delta t^l_{i,k} = \Delta t^0 + \Delta t^1 + \delta \Delta t^2_{i,1} = 0 + 600 + 50 = 650 \text{ Days},$$

$$t^2_{i,2} = \sum_{l=1}^{2} \Delta t^{l-1} + \sum_{k=1}^{2} \delta \Delta t^l_{i,k} = \Delta t^0 + \Delta t^1 + \delta \Delta t^2_{i,1} + \delta \Delta t^2_{i,2} = 0 + 600 + 50 + 50 = 700$$
$$\text{Days},$$

$$t^2_{i,3} = \sum_{l=1}^{2} \Delta t^{l-1} + \sum_{k=1}^{3} \delta \Delta t^l_{i,k} = \Delta t^0 + \Delta t^1 + \delta \Delta t^2_{i,1} + \delta \Delta t^2_{i,2} + \delta \Delta t^2_{i,3} =$$

$$0 + 600 + 50 + 50 + 50 = 750 \text{ Days}.$$

Now, let us write down NPV with the Eq. 2.12

$$J(\mathbf{u}) = \sum_{n=1}^{2} \left[\sum_{j=1}^{N^n_p} \frac{\delta \Delta t^n_{p,j}}{(1+b)^{\frac{t^n_{p,j}}{365}}} (r^n_o \bar{q}^n_{o,j} - c_{CO_2,p} \bar{q}^n_{CO_2,p,j}) - \sum_{j=1}^{N^n_i} \frac{\delta \Delta t^n_{i,j}}{(1+b)^{\frac{t^n_{i,j}}{365}}} (c_{CO_2,i} \bar{q}^n_{CO_2,i,j}) \right] =$$

$$\left[\sum_{j=1}^{N^1_p} \frac{\delta \Delta t^1_{p,j}}{(1+b)^{\frac{t^1_{p,j}}{365}}} (r_o \bar{q}^1_{o,j} - c_{CO_2,p} \bar{q}^1_{CO_2,p,j}) - \sum_{j=1}^{N^1_i} \frac{\delta \Delta t^1_{i,j}}{(1+b)^{\frac{t^1_{i,j}}{365}}} (c_{CO_2,i} \bar{q}^1_{CO_2,i,j}) \right] +$$

$$\left[\sum_{j=1}^{N_p^2}\frac{\delta\Delta t_{p,j}^2}{(1+b)^{\frac{t_{p,j}^2}{365}}}(r_o\bar{q}_{o,j}^2-c_{CO_2,p}\bar{q}_{CO_2,p,j}^2)-\sum_{j=1}^{N_i^2}\frac{\delta\Delta t_{i,j}^2}{(1+b)^{\frac{t_{i,j}^2}{365}}}(c_{CO_2,i}\bar{q}_{CO_2,i,j}^2)\right]=$$

$$\left[\sum_{j=1}^{3}\frac{\delta\Delta t_{p,j}^1}{(1+b)^{\frac{t_{p,j}^1}{365}}}(r_o\bar{q}_{o,j}^1-c_{CO_2,p}\bar{q}_{CO_2,p,j}^1)-\sum_{j=1}^{3}\frac{\delta\Delta t_{i,j}^1}{(1+b)^{\frac{t_{i,j}^1}{365}}}(c_{CO_2,i}\bar{q}_{CO_2,i,j}^1)\right]+$$

$$\left[\sum_{j=1}^{3}\frac{\delta\Delta t_{p,j}^2}{(1+b)^{\frac{t_{p,j}^2}{365}}}(r_o\bar{q}_{o,j}^2-c_{CO_2,p}\bar{q}_{CO_2,p,j}^2)-\sum_{j=1}^{3}\frac{\delta\Delta t_{i,j}^2}{(1+b)^{\frac{t_{i,j}^2}{365}}}(c_{CO_2,i}\bar{q}_{CO_2,i,j}^2)\right]$$

$$=\left(\frac{\delta\Delta t_{p,1}^1}{(1+b)^{\frac{t_{p,1}^1}{365}}}(r_o\bar{q}_{o,1}^1-c_{CO_2,p}\bar{q}_{CO_2,p,1}^1)\right)+\left(\frac{\delta\Delta t_{p,2}^1}{(1+b)^{\frac{t_{p,2}^1}{365}}}(r_o\bar{q}_{o,2}^1-c_{CO_2,p}\bar{q}_{CO_2,p,2}^1)\right)$$

$$+\left(\frac{\delta\Delta t_{p,3}^1}{(1+b)^{\frac{t_{p,3}^1}{365}}}(r_o\bar{q}_{o,3}^1-c_{CO_2,p}\bar{q}_{CO_2,p,3}^1)\right)$$

$$-\left(\frac{\delta\Delta t_{i,1}^1}{(1+b)^{\frac{t_{i,1}^1}{365}}}(c_{CO_2,i}\bar{q}_{CO_2,i,1}^1)\right)-\left(\frac{\delta\Delta t_{i,2}^1}{(1+b)^{\frac{t_{i,2}^1}{365}}}(c_{CO_2,i}\bar{q}_{CO_2,i,2}^1)\right)-\left(\frac{\delta\Delta t_{i,3}^1}{(1+b)^{\frac{t_{i,3}^1}{365}}}(c_{CO_2,i}\bar{q}_{CO_2,i,3}^1)\right)$$

$$+$$

$$\left(\frac{\delta\Delta t_{p,1}^2}{(1+b)^{\frac{t_{p,1}^2}{365}}}(r_o\bar{q}_{o,1}^2-c_{CO_2,p}\bar{q}_{CO_2,p,1}^2)\right)+\left(\frac{\delta\Delta t_{p,2}^2}{(1+b)^{\frac{t_{p,2}^2}{365}}}(r_o\bar{q}_{o,2}^2-c_{CO_2,p}\bar{q}_{CO_2,p,2}^2)\right)$$

$$+\left(\frac{\delta\Delta t_{p,3}^2}{(1+b)^{\frac{t_{p,3}^2}{365}}}(r_o\bar{q}_{o,3}^2-c_{CO_2,p}\bar{q}_{CO_2,p,3}^2)\right)$$

$$-\left(\frac{\delta\Delta t_{i,1}^2}{(1+b)^{\frac{t_{i,1}^2}{365}}}(c_{CO_2,i}\bar{q}_{CO_2,i,1}^2)\right)-\left(\frac{\delta\Delta t_{i,2}^2}{(1+b)^{\frac{t_{i,2}^2}{365}}}(c_{CO_2,i}\bar{q}_{CO_2,i,2}^2)\right)-\left(\frac{\delta\Delta t_{i,3}^2}{(1+b)^{\frac{t_{i,3}^2}{365}}}(c_{CO_2,i}\bar{q}_{CO_2,i,3}^2)\right)$$

$$=\left(\frac{100}{(1+b)^{\frac{400}{365}}}(r_o\bar{q}_{o,1}^1-c_{CO_2,p}\bar{q}_{CO_2,p,1}^1)\right)+\left(\frac{100}{(1+b)^{\frac{500}{365}}}(r_o\bar{q}_{o,2}^1-c_{CO_2,p}\bar{q}_{CO_2,p,2}^1)\right)$$

$$+\left(\frac{100}{(1+b)^{\frac{600}{365}}}(r_o\bar{q}_{o,3}^1-c_{CO_2,p}\bar{q}_{CO_2,p,3}^1)\right)$$

$$-\left(\frac{50}{(1+b)^{\frac{50}{365}}}(c_{CO_2,i}\bar{q}_{CO_2,i,1}^1)\right)-\left(\frac{50}{(1+b)^{\frac{100}{365}}}(c_{CO_2,i}\bar{q}_{CO_2,i,2}^1)\right)-\left(\frac{50}{(1+b)^{\frac{150}{365}}}(c_{CO_2,i}\bar{q}_{CO_2,i,3}^1)\right)+$$

$$\left(\frac{100}{(1+b)^{\frac{1000}{365}}}(r_o\bar{q}^2_{o,1}-c_{CO_2,p}\bar{q}^2_{CO_2,p,1})\right)+\left(\frac{100}{(1+b)^{\frac{1100}{365}}}(r_o\bar{q}^2_{o,2}-c_{CO_2,p}\bar{q}^2_{CO_2,p,2})\right)$$

$$+\left(\frac{100}{(1+b)^{\frac{1200}{365}}}(r_o\bar{q}^2_{o,3}-c_{CO_2,p}\bar{q}^2_{CO_2,p,3})\right)$$

$$-\left(\frac{50}{(1+b)^{\frac{650}{365}}}(c_{CO_2,i}\bar{q}^2_{CO_2,i,1})\right)-\left(\frac{50}{(1+b)^{\frac{700}{365}}}(c_{CO_2,i}\bar{q}^2_{CO_2,i,2})\right)-\left(\frac{50}{(1+b)^{\frac{750}{365}}}(c_{CO_2,i}\bar{q}^2_{CO_2,i,3})\right)$$

Let us rewrite the result in the increasing order of cumulative time:

$$-\left(\frac{50}{(1+b)^{\frac{50}{365}}}(c_{CO_2,i}\bar{q}^1_{CO_2,i,1})\right)-\left(\frac{50}{(1+b)^{\frac{100}{365}}}(c_{CO_2,i}\bar{q}^1_{CO_2,i,2})\right)-\left(\frac{50}{(1+b)^{\frac{150}{365}}}(c_{CO_2,i}\bar{q}^1_{CO_2,i,3})\right)+$$

$$\left(\frac{100}{(1+b)^{\frac{400}{365}}}(r_o\bar{q}^1_{o,1}-c_{CO_2,p}\bar{q}^1_{CO_2,p,1})\right)+\left(\frac{100}{(1+b)^{\frac{500}{365}}}(r_o\bar{q}^1_{o,2}-c_{CO_2,p}\bar{q}^1_{CO_2,p,2})\right)$$

$$+\left(\frac{100}{(1+b)^{\frac{600}{365}}}(r_o\bar{q}^1_{o,3}-c_{CO_2,p}\bar{q}^1_{CO_2,p,3})\right)$$

$$-\left(\frac{50}{(1+b)^{\frac{650}{365}}}(c_{CO_2,i}\bar{q}^2_{CO_2,i,1})\right)-\left(\frac{50}{(1+b)^{\frac{700}{365}}}(c_{CO_2,i}\bar{q}^2_{CO_2,i,2})\right)-\left(\frac{50}{(1+b)^{\frac{750}{365}}}(c_{CO_2,i}\bar{q}^2_{CO_2,i,3})\right)+$$

$$\left(\frac{100}{(1+b)^{\frac{1000}{365}}}(r_o\bar{q}^2_{o,1}-c_{CO_2,p}\bar{q}^2_{CO_2,p,1})\right)+\left(\frac{100}{(1+b)^{\frac{1100}{365}}}(r_o\bar{q}^2_{o,2}-c_{CO_2,p}\bar{q}^2_{CO_2,p,2})\right)$$

$$+\left(\frac{100}{(1+b)^{\frac{1200}{365}}}(r_o\bar{q}^2_{o,3}-c_{CO_2,p}\bar{q}^2_{CO_2,p,3})\right)$$

Now let us perform the same computation using the Eq. 2.10. For this purpose, let us first define a set of the simulation time steps so that it matches with our previous time steps. Except here we also include the soaking period. To do less computation, we include the whole soaking period as one of the simulation time steps, which is 150 Days:

$\Delta t^k \in \{50, 50, 50, 150, 100, 100, 100, 50, 50, 50, 150, 100, 100, 100\}$. Thus, N_s=14.

$$J(\mathbf{u}) = \sum_{k=1}^{N_s} \frac{\Delta t^k}{(1+b)^{\frac{t^k}{365}}}(r_o\bar{q}^k_o - c_{CO_2,p}\bar{q}^k_{CO_2,p} - c_{CO_2,i}\bar{q}^k_{CO_2,i})$$

We re-write this equation to make it resemble the Eq. 2.12 so we can compare them easily:

$$\sum_{k=1}^{N_s}\left[\frac{\Delta t^k}{(1+b)^{\frac{t^k}{365}}}(r_o\bar{q}^k_o - c_{CO_2,p}\bar{q}^k_{CO_2,p}) - \frac{\Delta t^k}{(1+b)^{\frac{t^k}{365}}}(c_{CO_2,i}\bar{q}^k_{CO_2,i})\right]$$

$$= \frac{\Delta t^1}{(1+b)^{\frac{t^1}{365}}}(r_o\bar{q}_o^1 - c_{CO_2,p}\bar{q}_{CO_2,p}^1) - \frac{\Delta t^1}{(1+b)^{\frac{t^1}{365}}}(c_{CO_2,i}\bar{q}_{CO_2,i}^1)+$$

$$\frac{\Delta t^2}{(1+b)^{\frac{t^2}{365}}}(r_o\bar{q}_o^2 - c_{CO_2,p}\bar{q}_{CO_2,p}^2) - \frac{\Delta t^2}{(1+b)^{\frac{t^2}{365}}}(c_{CO_2,i}\bar{q}_{CO_2,i}^2)+$$

$$\frac{\Delta t^3}{(1+b)^{\frac{t^3}{365}}}(r_o\bar{q}_o^3 - c_{CO_2,p}\bar{q}_{CO_2,p}^3) - \frac{\Delta t^3}{(1+b)^{\frac{t^3}{365}}}(c_{CO_2,i}\bar{q}_{CO_2,i}^3)+$$

$$\frac{\Delta t^4}{(1+b)^{\frac{t^4}{365}}}(r_o\bar{q}_o^4 - c_{CO_2,p}\bar{q}_{CO_2,p}^4) - \frac{\Delta t^4}{(1+b)^{\frac{t^4}{365}}}(c_{CO_2,i}\bar{q}_{CO_2,i}^4)+$$

$$\frac{\Delta t^5}{(1+b)^{\frac{t^5}{365}}}(r_o\bar{q}_o^5 - c_{CO_2,p}\bar{q}_{CO_2,p}^5) - \frac{\Delta t^5}{(1+b)^{\frac{t^5}{365}}}(c_{CO_2,i}\bar{q}_{CO_2,i}^5)+$$

$$\frac{\Delta t^6}{(1+b)^{\frac{t^6}{365}}}(r_o\bar{q}_o^6 - c_{CO_2,p}\bar{q}_{CO_2,p}^6) - \frac{\Delta t^6}{(1+b)^{\frac{t^6}{365}}}(c_{CO_2,i}\bar{q}_{CO_2,i}^6)+$$

$$\frac{\Delta t^7}{(1+b)^{\frac{t^7}{365}}}(r_o\bar{q}_o^7 - c_{CO_2,p}\bar{q}_{CO_2,p}^7) - \frac{\Delta t^7}{(1+b)^{\frac{t^7}{365}}}(c_{CO_2,i}\bar{q}_{CO_2,i}^7)+$$

$$\frac{\Delta t^8}{(1+b)^{\frac{t^8}{365}}}(r_o\bar{q}_o^8 - c_{CO_2,p}\bar{q}_{CO_2,p}^8) - \frac{\Delta t^8}{(1+b)^{\frac{t^8}{365}}}(c_{CO_2,i}\bar{q}_{CO_2,i}^8)+$$

$$\frac{\Delta t^9}{(1+b)^{\frac{t^9}{365}}}(r_o\bar{q}_o^9 - c_{CO_2,p}\bar{q}_{CO_2,p}^9) - \frac{\Delta t^9}{(1+b)^{\frac{t^9}{365}}}(c_{CO_2,i}\bar{q}_{CO_2,i}^9)+$$

$$\frac{\Delta t^10}{(1+b)^{\frac{t^10}{365}}}(r_o\bar{q}_o^{10} - c_{CO_2,p}\bar{q}_{CO_2,p}^{10}) - \frac{\Delta t^10}{(1+b)^{\frac{t^10}{365}}}(c_{CO_2,i}\bar{q}_{CO_2,i}^{10})+$$

$$\frac{\Delta t^11}{(1+b)^{\frac{t^11}{365}}}(r_o\bar{q}_o^{11} - c_{CO_2,p}\bar{q}_{CO_2,p}^{11}) - \frac{\Delta t^11}{(1+b)^{\frac{t^11}{365}}}(c_{CO_2,i}\bar{q}_{CO_2,i}^{11})+$$

$$\frac{\Delta t^12}{(1+b)^{\frac{t^12}{365}}}(r_o\bar{q}_o^{12} - c_{CO_2,p}\bar{q}_{CO_2,p}^{12}) - \frac{\Delta t^12}{(1+b)^{\frac{t^12}{365}}}(c_{CO_2,i}\bar{q}_{CO_2,i}^{12})+$$

$$\frac{\Delta t^13}{(1+b)^{\frac{t^13}{365}}}(r_o\bar{q}_o^{13} - c_{CO_2,p}\bar{q}_{CO_2,p}^{13}) - \frac{\Delta t^13}{(1+b)^{\frac{t^13}{365}}}(c_{CO_2,i}\bar{q}_{CO_2,i}^{13})+$$

$$\frac{\Delta t^14}{(1+b)^{\frac{t^14}{365}}}(r_o\bar{q}_o^{14} - c_{CO_2,p}\bar{q}_{CO_2,p}^{14}) - \frac{\Delta t^14}{(1+b)^{\frac{t^14}{365}}}(c_{CO_2,i}\bar{q}_{CO_2,i}^{14})$$

Let us put the values of cumulative simulation time and simulation time step values in, considering also that during production and soaking period injection rate will be zero, and during injection and soaking period production rate will be zero:

$$= \frac{50}{(1+b)^{\frac{50}{365}}}(r_o 0 - c_{CO_2,p}0) - \frac{50}{(1+b)^{\frac{50}{365}}}(c_{CO_2,i}\bar{q}^1_{CO_2,i})+$$

$$\frac{50}{(1+b)^{\frac{100}{365}}}(r_o 0 - c_{CO_2,p}0) - \frac{50}{(1+b)^{\frac{100}{365}}}(c_{CO_2,i}\bar{q}^2_{CO_2,i})+$$

$$\frac{50}{(1+b)^{\frac{150}{365}}}(r_o 0 - c_{CO_2,p}0) - \frac{50}{(1+b)^{\frac{150}{365}}}(c_{CO_2,i}\bar{q}^3_{CO_2,i})+$$

$$\frac{150}{(1+b)^{\frac{300}{365}}}(r_o 0 - c_{CO_2,p}0) - \frac{150}{(1+b)^{\frac{300}{365}}}(c_{CO_2,i}0)+$$

$$\frac{100}{(1+b)^{\frac{400}{365}}}(r_o\bar{q}^5_o - c_{CO_2,p}\bar{q}^5_{CO_2,p}) - \frac{100}{(1+b)^{\frac{400}{365}}}(c_{CO_2,i}0)+$$

$$\frac{100}{(1+b)^{\frac{500}{365}}}(r_o\bar{q}^6_o - c_{CO_2,p}\bar{q}^6_{CO_2,p}) - \frac{100}{(1+b)^{\frac{500}{365}}}(c_{CO_2,i}0)+$$

$$\frac{100}{(1+b)^{\frac{600}{365}}}(r_o\bar{q}^7_o - c_{CO_2,p}\bar{q}^7_{CO_2,p}) - \frac{100}{(1+b)^{\frac{600}{365}}}(c_{CO_2,i}0)+$$

$$\frac{50}{(1+b)^{\frac{650}{365}}}(r_o 0 - c_{CO_2,p}0) - \frac{50}{(1+b)^{\frac{650}{365}}}(c_{CO_2,i}\bar{q}^8_{CO_2,i})+$$

$$\frac{50}{(1+b)^{\frac{700}{365}}}(r_o 0 - c_{CO_2,p}0) - \frac{50}{(1+b)^{\frac{700}{365}}}(c_{CO_2,i}\bar{q}^9_{CO_2,i})+$$

$$\frac{50}{(1+b)^{\frac{750}{365}}}(r_o 0 - c_{CO_2,p}0) - \frac{50}{(1+b)^{\frac{750}{365}}}(c_{CO_2,i}\bar{q}^{10}_{CO_2,i})+$$

$$\frac{150}{(1+b)^{\frac{900}{365}}}(r_o 0 - c_{CO_2,p}0) - \frac{150}{(1+b)^{\frac{900}{365}}}(c_{CO_2,i}0)+$$

$$\frac{100}{(1+b)^{\frac{1000}{365}}}(r_o\bar{q}^{12}_o - c_{CO_2,p}\bar{q}^{12}_{CO_2,p}) - \frac{100}{(1+b)^{\frac{1000}{365}}}(c_{CO_2,i}0)+$$

$$\frac{100}{(1+b)^{\frac{1100}{365}}}(r_o\bar{q}^{13}_o - c_{CO_2,p}\bar{q}^{13}_{CO_2,p}) - \frac{100}{(1+b)^{\frac{1100}{365}}}(c_{CO_2,i}0)+$$

$$\frac{100}{(1+b)^{\frac{1200}{365}}}(r_o\bar{q}^{14}_o - c_{CO_2,p}\bar{q}^{14}_{CO_2,p}) - \frac{100}{(1+b)^{\frac{1200}{365}}}(c_{CO_2,i}0)$$

Let's eliminate the zero terms and see what is left for us:

$$= -\frac{50}{(1+b)^{\frac{50}{365}}}(c_{CO_2,i}\bar{q}^1_{CO_2,i})+$$

$$-\frac{50}{(1+b)^{\frac{100}{365}}}(c_{CO_2,i}\bar{q}^2_{CO_2,i})+$$

$$-\frac{50}{(1+b)^{\frac{150}{365}}}(c_{CO_2,i}\bar{q}^3_{CO_2,i})+$$

$$\frac{100}{(1+b)^{\frac{400}{365}}}(r_o\bar{q}^5_o - c_{CO_2,p}\bar{q}^5_{CO_2,p})+$$

$$\frac{100}{(1+b)^{\frac{500}{365}}}(r_o\bar{q}^6_o - c_{CO_2,p}\bar{q}^6_{CO_2,p})+$$

$$\frac{100}{(1+b)^{\frac{600}{365}}}(r_o\bar{q}^7_o - c_{CO_2,p}\bar{q}^7_{CO_2,p})+$$

$$-\frac{50}{(1+b)^{\frac{650}{365}}}(c_{CO_2,i}\bar{q}^8_{CO_2,i})+$$

$$-\frac{50}{(1+b)^{\frac{700}{365}}}(c_{CO_2,i}\bar{q}^9_{CO_2,i})+$$

$$-\frac{50}{(1+b)^{\frac{750}{365}}}(c_{CO_2,i}\bar{q}^{10}_{CO_2,i})+$$

$$\frac{100}{(1+b)^{\frac{1000}{365}}}(r_o\bar{q}^{12}_o - c_{CO_2,p}\bar{q}^{12}_{CO_2,p})+$$

$$\frac{100}{(1+b)^{\frac{1100}{365}}}(r_o\bar{q}^{13}_o - c_{CO_2,p}\bar{q}^{13}_{CO_2,p})+$$

$$\frac{100}{(1+b)^{\frac{1200}{365}}}(r_o\bar{q}^{14}_o - c_{CO_2,p}\bar{q}^{14}_{CO_2,p})$$

This is the same result as the one where we used time steps (control time steps) instead of simulation time steps.

APPENDIX B

TRAINING PROCEDURE OF LS-SVR

Training procedure of LS-SVR is the solution of the following optimization problem:

$$L(\mathbf{w}, b, e; \boldsymbol{\alpha}) = O(\mathbf{w}, e) - \sum_{k=1}^{N_{tr}} \alpha_k [\mathbf{w}^T \boldsymbol{\phi}(\bar{\mathbf{u}}_k) + b + e_k - \bar{J}_{o,k}]. \tag{B.1}$$

The optimal solution of Eq. B.1 is obtained by solving $\nabla L = 0$, where ∇ is the gradient vector having partial derivatives with respect to $\mathbf{w}, \alpha_k, e_k$, for $k = 1, 2, ...,N_{tr}$, and b, which yields the following linear system of equations (Cawley and Talbot, 2007):

$$\begin{cases} \nabla_{\mathbf{w}} L = 0 \rightarrow \mathbf{w} = \sum_{k=1}^{N_{tr}} \alpha_k \boldsymbol{\phi}(\bar{\mathbf{u}}_k), & \text{(B.2a)} \\ \dfrac{\partial L}{\partial b} = 0 \rightarrow \sum_{k=1}^{N_{tr}} \alpha_k = 0, & \text{(B.2b)} \\ \dfrac{\partial L}{\partial e_k} = 0 \rightarrow \alpha_k = \gamma e_k, \;\; k = 1 : N_{tr}, & \text{(B.2c)} \\ \dfrac{\partial L}{\partial \alpha_k} = 0 \rightarrow \mathbf{w}^T \boldsymbol{\phi}(\bar{\mathbf{u}}_k) + b + e_k - J_{o,k}, \;\; k = 1 : N_{tr}. & \text{(B.2d)} \end{cases}$$

Substituting Eq. B.2a into Eq. B.2d gives us

$$J_{o,k} = \sum_{l=1}^{N_{tr}} \alpha_k \boldsymbol{\phi}(\bar{\mathbf{u}}_l)^T \boldsymbol{\phi}(\bar{\mathbf{u}}_k) + b + e_k. \tag{B.3}$$

Deriving $e_k \dfrac{\alpha_k}{\gamma}$ from Eq. B.2c, and putting it in Eq. B.3, we can write it in a vector-matrix multiplication form as

$$\mathbf{j}_o = \left(\boldsymbol{\Omega} + \frac{1}{\gamma} \mathbf{I} \right) \alpha + \mathbf{1}_{N_{tr}} b, \tag{B.4}$$

where $\boldsymbol{\Omega}$ is $N_{tr} \times N_{tr}$ matrix with entries of $\Omega_{k,l} = \boldsymbol{\phi}(\bar{\mathbf{u}}_l)^T \boldsymbol{\phi}(\bar{\mathbf{u}}_k)$; $\mathbf{j}_o = [J_{o,1}, J_{o,2}, \ldots, J_{o,N_{tr}}]^T$; and $\boldsymbol{\alpha} = [\alpha_1, \alpha_2, \ldots, \alpha_{N_{tr}}]^T$. We can further combine Eq. B.2b and B.4 to obtain both

equations in a single vector-matrix multiplication form

$$\begin{bmatrix} 0 & \mathbf{1}_{N_{tr}}^T \\ \mathbf{1}_{N_{tr}} & \boldsymbol{\Omega} + \frac{1}{\gamma}\mathbf{I} \end{bmatrix} \begin{bmatrix} b \\ \boldsymbol{\alpha} \end{bmatrix} = \begin{bmatrix} 0 \\ \mathbf{j}_o \end{bmatrix}. \tag{B.5}$$

By multiplying Eq. B.4 by $\mathbf{1}_{N_{tr}}^T \left(\boldsymbol{\Omega} + \frac{1}{\gamma}\mathbf{I}\right)^{-1}$, and knowing that $\mathbf{1}_{N_{tr}}^T \boldsymbol{\alpha} = 0$, we can find b as

$$b = \frac{\mathbf{1}_{N_{tr}}^T \left(\boldsymbol{\Omega} + \frac{1}{\gamma}\mathbf{I}\right)^{-1} \mathbf{j}_o}{\mathbf{1}_{N_{tr}}^T \left(\boldsymbol{\Omega} + \frac{1}{\gamma}\mathbf{I}\right)^{-1} \mathbf{1}_{N_{tr}}}. \tag{B.6}$$

Similarly, we can solve Eq. B.2b for $\boldsymbol{\alpha}$

$$\boldsymbol{\alpha} = \left(\boldsymbol{\Omega} + \frac{1}{\gamma}\mathbf{I}\right)^{-1} \mathbf{j}_o - b \left(\boldsymbol{\Omega} + \frac{1}{\gamma}\mathbf{I}\right)^{-1} \mathbf{1}_{N_{tr}}. \tag{B.7}$$

Actually, one can find both b and $\boldsymbol{\alpha}$, simultaneously, by solving Eq. B.2c. From Eqs. B.6 and B.7, we can see that computational complexity of training procedure has relationship with the size of the data as $\mathcal{O}(N_{tr}^3)$ for the Cholesky decomposition. As can be seen from Eqs. B.6 and B.7, to be able to calculate b and $\boldsymbol{\alpha}$, we need to know entries of the matrix $\boldsymbol{\Omega}$, which are kernel function evaluated between two data points, and γ. Kernel function has unknown parameter (or parameters). In our case, where radial basis function is used as kernel function (Eq. 4.9), that unknown parameter is σ_{bw}^2. These two unknown parameters (σ_{bw}^2 and γ) can be found using hyperparameter optimization method, which is discussed in Section 4.1.3.

APPENDIX C

DERIVATIONS OF GPR

In this appendix, we provide some essential derivations of GPR. In the first section, derivations of conditional mean and conditional covariance of prediction points given observations are provided. In the second section, we discuss the details of the Bayesian model selection process in GPR.

C.1 Derivations Of Mean And Covariance Of Conditional Multivariate Gaussian Distribution Of Prediction Points Given Observations

It is a special property of multivariate Gaussian distribution that conditioning on observed values of a subset of variables leads to conditional distribution for the unobserved variables. This conditional distribution is also Gaussian. We will see that the expectation of conditional distribution is linear in observations, but the covariance matrix does not depend on the observed values. Using this property, we can derive the distribution of the predictions for the GPR model since it is basically a conditional distribution conditioned to observed data.

A combination of prediction points and observation points in one random vector (where each point is a random variable) is given as

$$\mathbf{j}^{\Sigma} = \begin{bmatrix} \mathbf{j}^{*} \\ \mathbf{j}_{o} \end{bmatrix}. \tag{C.8}$$

with mean

$$\boldsymbol{\mu}^{\Sigma} = \begin{bmatrix} \boldsymbol{\mu}_{p} \\ \boldsymbol{\mu} \end{bmatrix}, \tag{C.9}$$

where $\boldsymbol{\mu}_p$ and $\boldsymbol{\mu}$ are mean of predictive points ($\mathbf{j}^*$) and mean of observed points ($\mathbf{j}_o$), respectively; and $N_t \times N_t$ symmetric covariance matrix $\mathbf{K}^{\Sigma}$, where $N_t = N_{tr} + N_{pr}$, as

$$\mathbf{K}^{\Sigma} = cov\left(\begin{bmatrix}\mathbf{j}^* \\ \mathbf{j}_o\end{bmatrix}, \begin{bmatrix}\mathbf{j}^* \\ \mathbf{j}_o\end{bmatrix}\right) = \begin{bmatrix}\mathbf{K}_p & \mathbf{K}_{p,o} \\ \mathbf{K}_{o,p} & \mathbf{K}_o\end{bmatrix}, \tag{C.10}$$

where $\mathbf{K}_p$ is the $N_{pr} \times N_{pr}$ covariance matrix of $\mathbf{j}^*$, $\mathbf{K}_o$ the $N_{tr} \times N_{tr}$ covariance matrix of $\mathbf{j}_o$, and $\mathbf{K}_{o,p}$ ($=\mathbf{K}_{p,o}^T$) is the $N_{tr} \times N_{pr}$ cross-covariance matrix of $\mathbf{j}_o$ and $\mathbf{j}^*$.

If the matrix $\mathbf{K}^{\Sigma}$ is positive definite, we can write the joint multivariate Gaussian distribution as

$$p(\mathbf{j}^*, \mathbf{j}_o) = \frac{1}{(2\pi)^{N_t/2}\sqrt{det\mathbf{K}^{\Sigma}}} \exp\left[-\frac{1}{2}\left\{\begin{bmatrix}\mathbf{j}^* - \boldsymbol{\mu}_p \\ \mathbf{j}_o - \boldsymbol{\mu}\end{bmatrix}^T (\mathbf{K}^{\Sigma})^{-1} \begin{bmatrix}\mathbf{j}^* - \boldsymbol{\mu}_p \\ \mathbf{j}_o - \boldsymbol{\mu}\end{bmatrix}\right\}\right], \tag{C.11}$$

where the marginal PDF of N_{tr} dimensional $\mathbf{j}_o$ random vector (also known as marginal likelihood or evidence) is

$$p(\mathbf{j}_o) = \frac{1}{(2\pi)^{N_{tr}/2}\sqrt{det\mathbf{K}_o}} \exp\left[-\frac{1}{2}\left\{[\mathbf{j}_o - \boldsymbol{\mu}]^T \mathbf{K}_o^{-1} [\mathbf{j}_o - \boldsymbol{\mu}]\right\}\right]. \tag{C.12}$$

Then, we can write an equation for conditional PDF of $\mathbf{j}^*$ given $\mathbf{j}_o$ as

$$p(\mathbf{j}^*|\mathbf{j}_o) = \frac{p(\mathbf{j}^*, \mathbf{j}_o)}{p(\mathbf{j}_o)} \tag{C.13}$$

To simplify the calculations for further derivations, we assume that $\boldsymbol{\mu} = \boldsymbol{\mu}_p = 0$ and they will be re-inserted back to our results. Substituting Eqs.C.11 and C.12 into Eq. C.13 we get

$$\begin{aligned} p(\mathbf{j}^*|\mathbf{j}_o) =& c \exp\left\{-\frac{1}{2}\begin{bmatrix}\mathbf{j}^* \\ \mathbf{j}_o\end{bmatrix}^T (\mathbf{K}^{\Sigma})^{-1} \begin{bmatrix}\mathbf{j}^* \\ \mathbf{j}_o\end{bmatrix} + \frac{1}{2}\mathbf{j}_o^T \mathbf{K}_o^{-1} \mathbf{j}_o\right\}, \\ =& c \exp\left\{-\frac{1}{2} Q(\mathbf{j}^*, \mathbf{j}_o)\right\} \end{aligned} \tag{C.14}$$

where c is constant we get as

$$c = (2\pi)^{(N_{tr}-N_t)/2} \frac{\sqrt{det\mathbf{K}_o}}{\sqrt{det\mathbf{K}^{\Sigma}}} \tag{C.15}$$

Using the matrix inversion lemmas (Vetterling et al., 1992), we can write the inverse of the matrix $\mathbf{K}^{\Sigma}$ as follows:

$$\begin{aligned}(\mathbf{K}^{\Sigma})^{-1} &= \begin{bmatrix} \mathbf{K}_p & \mathbf{K}_{p,o} \\ \mathbf{K}_{o,p} & \mathbf{K}_o \end{bmatrix}^{-1} = \begin{bmatrix} \mathbf{A}_{11} & \mathbf{A}_{12} \\ \mathbf{A}_{21} & \mathbf{A}_{22} \end{bmatrix} \\ &= \begin{bmatrix} [\mathbf{K}_p - \mathbf{K}_{p,o}\mathbf{K}_o^{-1}\mathbf{K}_{o,p}]^{-1}, & -[\mathbf{K}_p - \mathbf{K}_{p,o}\mathbf{K}_o^{-1}\mathbf{K}_{o,p}]^{-1}\mathbf{K}_{p,o}\mathbf{K}_o^{-1} \\ -[\mathbf{K}_o - \mathbf{K}_{o,p}\mathbf{K}_p^{-1}\mathbf{K}_{p,o}]^{-1}\mathbf{K}_{o,p}\mathbf{K}_p^{-1}, & [\mathbf{K}_o - \mathbf{K}_{o,p}\mathbf{K}_p^{-1}\mathbf{K}_{p,o}]^{-1} \end{bmatrix}\end{aligned} \tag{C.16}$$

where, $\mathbf{A}_{11}, \mathbf{A}_{12}, \mathbf{A}_{21}, \mathbf{A}_{22}$ are defined as above to make our further derivation look tidy. Using Eq. C.16 in Eq. C.14, we can expand the term $Q(\mathbf{j}^*, \mathbf{j}_o)$ as

$$\begin{aligned}Q(\mathbf{j}_o, \mathbf{j}^*) =& (\mathbf{j}^*)^T\mathbf{A}_{11}\mathbf{j}^* + (\mathbf{j}^*)^T\mathbf{A}_{12}\mathbf{j}_o + \mathbf{j}_o^T\mathbf{A}_{21}\mathbf{j}^* + \mathbf{j}_o^T\mathbf{A}_{22}\mathbf{j}_o - \mathbf{j}_o^T\mathbf{K}_o^{-1}\mathbf{j}_o \\ =& ((\mathbf{j}^*)^T - \mathbf{j}_o^T\mathbf{B}^T)\mathbf{A}_{11}(\mathbf{j}^* - \mathbf{B}\mathbf{j}_o) + \tilde{Q}(\mathbf{j}_o) \\ =& (\mathbf{j}^*)^T\mathbf{A}_{11}\mathbf{j}^* - (\mathbf{j}^*)^T\mathbf{A}_{11}\mathbf{B}\mathbf{j}_o - \mathbf{j}_o^T\mathbf{B}^T\mathbf{A}_{11}\mathbf{j}^* + \mathbf{j}_o^T\mathbf{B}^T\mathbf{A}_{11}\mathbf{B}\mathbf{j}_o + \tilde{Q}(\mathbf{j}_o)\end{aligned} \quad , \tag{C.17}$$

where matrix $\mathbf{B}$ is defined, which is unknown, as mathematical trick. Quadratic function, $\tilde{Q}(\mathbf{j}_o) = \mathbf{j}_o^T\mathbf{K}_o^{-1}\mathbf{j}_o$, can be excluded as a multiplier of exponent since summation at the exponent is multiplication. What we are left with is just the following equality from Eq. C.17

$$\begin{aligned}&(\mathbf{j}^*)^T\mathbf{A}_{11}\mathbf{j}^* + (\mathbf{j}^*)^T\mathbf{A}_{12}\mathbf{j}_o + \mathbf{j}_o^T\mathbf{A}_{21}\mathbf{j}^* + \mathbf{j}_o^T\mathbf{A}_{22}\mathbf{j}_o = \\ &(\mathbf{j}^*)^T\mathbf{A}_{11}\mathbf{j}^* - (\mathbf{j}^*)^T\mathbf{A}_{11}\mathbf{B}\mathbf{j}_o - \mathbf{j}_o^T\mathbf{B}^T\mathbf{A}_{11}\mathbf{j}^* + \mathbf{j}_o^T\mathbf{B}^T\mathbf{A}_{11}\mathbf{B}\mathbf{j}_o\end{aligned} \tag{C.18}$$

From the equality, we see that $-\mathbf{A}_{11}\mathbf{B} = \mathbf{A}_{12}$. Solving this for $\mathbf{B}$

$$\begin{aligned}\mathbf{B} =& -\mathbf{A}_{11}^{-1}\mathbf{A}_{12} = -([\mathbf{K}_p - \mathbf{K}_{p,o}\mathbf{K}_o^{-1}\mathbf{K}_{o,p}]^{-1})^{-1} - [\mathbf{K}_p - \mathbf{K}_{p,o}\mathbf{K}_o^{-1}\mathbf{K}_{o,p}]^{-1}\mathbf{K}_{p,o}\mathbf{K}_o^{-1} \\ =& \mathbf{K}_{p,o}\mathbf{K}_o^{-1}.\end{aligned} \tag{C.19}$$

From Eq. C.17, which represents the conditional PDF of prediction points, $\mathbf{j}^*$, given observations, $\mathbf{j}_o$, it is obvious that $\mathbf{A}_{11}$ is the inverse of covariance matrix of the prediction points given observations, $cov(\mathbf{j}^*, \mathbf{j}^*|\mathbf{j}_o)$, which is denoted in our research as $\mathbf{K}^*$. Then, from Eq.C.16, knowing $\mathbf{A}_{11}$, we can write

$$\mathbf{K}^* = \mathbf{K}_p - \mathbf{K}_{p,o}\mathbf{K}_o^{-1}\mathbf{K}_{o,p}. \tag{C.20}$$

Also, from Eq. C.17, the term $\mathbf{B}\mathbf{j}_o$ is the expectation of prediction points given observations, $E[\mathbf{j}^*|\mathbf{j}_o]$, which is denoted in our research as $\boldsymbol{\mu}^*$. Then, knowing $\mathbf{B}$ from Eq. C.19, we write

$$\boldsymbol{\mu}^* = \mathbf{K}_{p,o}\mathbf{K}_o^{-1}\mathbf{j}_o. \tag{C.21}$$

However, reinstalling the means of observations, $\boldsymbol{\mu}$, and predictions, $\boldsymbol{\mu}_p$, we can rewrite Eq. C.21 as

$$\boldsymbol{\mu}^* = \boldsymbol{\mu}_p + \mathbf{K}_{p,o}\mathbf{K}_o^{-1}(\mathbf{j}_o - \boldsymbol{\mu}). \tag{C.22}$$

In our research, $\boldsymbol{\mu}_p = \mu\mathbf{1}_{N_{pr}}$, and $\boldsymbol{\mu} = \mu\mathbf{1}_{N_{tr}}$, where μ is just a constant to be determined through hyperparameter optimization (von Mises, 1964; Muirhead, 2009; Eaton, 1983).

One can also get the same results for conditional covariance of predictions given observed data (Eq. C.20), and for conditional expectation of predictions given observed data (Eq. C.22), using the Bayes' rule to get PDF for joint distribution of function space at observed points ($\mathbf{j}(\mathbf{U}_o)$, where $\mathbf{U}_o$ is the matrix of observed points, each column of which is one observation point; for simplicity we will denote it as $\mathbf{j}$) and function space at prediction points ($\mathbf{j}^* = \mathbf{j}(\mathbf{U}^*)$) given observed data points ($\mathbf{j}_o$), $p(\mathbf{j}, \mathbf{j}^*|\mathbf{j}_o)$, and integrating it with respect

to $\mathbf{j}$ to get $p(\mathbf{j}^*|\mathbf{j}_o)$:

$$p(\mathbf{j}, \mathbf{j}^*|\mathbf{j}_o) = \frac{p(\mathbf{j}_o|\mathbf{j})p(\mathbf{j}, \mathbf{j}^*)}{p(\mathbf{j}_o)}, \tag{C.23}$$

$$p(\mathbf{j}^*|\mathbf{j}_o) = \int p(\mathbf{j}, \mathbf{j}^*|\mathbf{j}_o)d\mathbf{j} = \frac{1}{p(\mathbf{j}_o)} \int p(\mathbf{j}_o|\mathbf{j})p(\mathbf{j}, \mathbf{j}^*)d\mathbf{j}, \tag{C.24}$$

where $p(\mathbf{j}, \mathbf{j}^*)$ and $p(\mathbf{j}_o|\mathbf{j})$ are given as follows:

$$p(\mathbf{j}, \mathbf{j}^*) = \mathcal{N}\left(\begin{bmatrix} \boldsymbol{\mu} \\ \boldsymbol{\mu}_p \end{bmatrix}, \begin{bmatrix} \mathbf{K} & \mathbf{K}_{o,p} \\ \mathbf{K}_{p,o} & \mathbf{K}_p \end{bmatrix}\right), \tag{C.25}$$

and

$$p(\mathbf{j}_o|\mathbf{j}) = \mathcal{N}(\mathbf{j}, \sigma_n^2\mathbf{I}), \tag{C.26}$$

where $\mathbf{K}$ is $N_{tr} \times N_{tr}$ covariance matrix of the function space at observed points, and $\mathbf{I}$ is $N_{tr} \times N_{tr}$ dimensional unit matrix. Note that observation data ($\mathbf{j}_o$) are different than the function evaluated at observed points ($\mathbf{j}$). Actually, the integration with respect to $\mathbf{j}$ is because we want to get rid of $\mathbf{j}$ since we do not know what is the function space evaluated at the observed points. However, we know observation data, $\mathbf{j}_o$. The term σ_n^2 in Eq. C.26 is variance of the noise in the data.

The integration term at the right-hand side of the Eq. C.24 gives us joint PDF of prediction points and observation points, $p(\mathbf{j}^*, \mathbf{j}_o)$. Since, both factors in that integral ($p(\mathbf{j}_o|\mathbf{j})$ and $p(\mathbf{j}, \mathbf{j}^*)$) are Gaussian, the integral can be evaluated in closed form to give

$$p(\mathbf{j}_o, \mathbf{j}^*) = \mathcal{N}\left(\begin{bmatrix} \boldsymbol{\mu} \\ \boldsymbol{\mu}_p \end{bmatrix}, \begin{bmatrix} \mathbf{K} + \sigma_n^2\mathbf{I} & \mathbf{K}_{o,p} \\ \mathbf{K}_{p,o} & \mathbf{K}_p \end{bmatrix}\right). \tag{C.27}$$

If we define $\mathbf{K} + \sigma_n^2\mathbf{I} = \mathbf{K}_o$, we see that Eq. C.27 is exactly the same as Eq. C.11.

Normalizing constant in Eq. C.24, $p(\mathbf{j}_o)$, can be found similarly

$$p(\mathbf{j}_o) = \int p(\mathbf{j}_o|\mathbf{j})p(\mathbf{j})d\mathbf{j}. \tag{C.28}$$

Again, since both factors in the integral in Eq. C.28 are Gaussian, the integral can be evaluated in closed form to get

$$p(\mathbf{j}_o) = \mathcal{N}(\boldsymbol{\mu}, \mathbf{K} + \sigma_n^2 \mathbf{I}), \tag{C.29}$$

which is exactly the same Eq. C.12, defining $\mathbf{K} + \sigma_n^2 \mathbf{I} = \mathbf{K}_o$. The rest of the derivation follows from Eq. C.13 to Eq. C.22.

C.2 Bayesian Model Selection in GPR

Here we discuss Bayesian model selection in general and its application to GPR. Any model can be specified hierarchically. At the first level are the parameters of the model, which is denoted as $\mathbf{w}$ also known as weights. Since the Gaussian process is a non-parametric model, we do not have weights; instead, we have function space directly. As we know the uncertainty on the weights induces uncertainty to the function space. All the derivations of the GPR, its training procedure shown above, can also be derived using the weight space view. For the derivations using the weight space view of GPR, please check Section 2.1 of the book Williams and Rasmussen (2006). At the second level, we have hyperparameters, which affect the distribution of the weights at the first level. And at the third level, model structures stay. It is a discrete set of models $\mathcal{H}_i$. Therefore, the probability assigned them is to discrete probability.

As can be seen from the name of the section this model selection uses the Bayes rule to go from the first level to the last level of the hierarchy. For the purpose of not confusing the reader, we will use the function space view since all our derivations stand for this view. The first level posterior over the function space is given as

$$p(\mathbf{j}|\mathbf{j}_o, \mathbf{U}, \theta, \mathcal{H}_i) = \frac{p(\mathbf{j}_o|\mathbf{U}, \mathbf{j}, \mathcal{H}_i) p(\mathbf{j}|\theta, \mathcal{H}_i)}{p(\mathbf{j}_o|\mathbf{U}, \theta, \mathcal{H}_i)}, \tag{C.30}$$

where the first two probability distributions at the numerator is called *likelihood* distribution of data given function space, and *prior* distribution of the function space before seeing the

data, respectively. The term at the denominator is independent of the function space and called the *marginal likelihood* or *evidence*

$$p(\mathbf{j}_o|\mathbf{U},\theta,\mathcal{H}_i) = \int p(\mathbf{j}_o|\mathbf{U},\mathbf{j},\mathcal{H}_i)p(\mathbf{j}|\theta,\mathcal{H}_i)d\mathbf{j} \tag{C.31}$$

For the second level, posterior is given for hyperparameters, which controls probability distribution over function space, $(\mathbf{j}|\theta,\mathcal{H}_i)$, and given as

$$p(\theta|\mathbf{j}_o,\mathbf{U},\mathcal{H}_i) = \frac{p(\mathbf{j}_o|\mathbf{U},\theta,\mathcal{H}_i)p(\theta|\mathcal{H}_i)}{p(\mathbf{j}_o|\mathbf{U},\mathcal{H}_i)}, \tag{C.32}$$

where the second term at the nominator is a prior distribution over hyperparameters. The term at the denominator is normalizing constant and given by

$$p(\mathbf{j}_o|\mathbf{U},\mathcal{H}_i) = \int p(\mathbf{j}_o|\mathbf{U},\theta,\mathcal{H}_i)p(\theta|\mathcal{H}_i)d\theta. \tag{C.33}$$

At the first level, which is the top level of the hierarchy, we have posterior over the model (type of kernel function in our case), which controls probability distribution over hyperparameters, $p(\theta|\mathcal{H}_i)$, and given by

$$p(\mathcal{H}_i|\mathbf{j}_o,\mathbf{U}) = \frac{p(\mathbf{j}_o|\mathbf{U},\mathcal{H}_i)p(\mathcal{H}_i)}{p(\mathbf{j}_o|\mathbf{U})}, \tag{C.34}$$

where the normalizing constant is the marginal distribution of data, which is discrete

$$p(\mathbf{j}_o|\mathbf{U} = \sum_{i}^{N_{model}} p(\mathbf{j}_o|\mathbf{U},\mathcal{H}_i)p(\mathcal{H}_i), \tag{C.35}$$

where N_{model} is the number of models (number of possible kernel functions in our case). Since it is hard to analytically track these integrals, one may have to use integral approximation techniques such as the Markov chain Monte Carlo (MCMC) method. Instead of evaluating the integral in Eq. C.33, it is preferable to maximize marginal likelihood (Eq. C.31) with

respect to θ. For the prior distribution over model is chosen to be uniform distribution, which means we do not prefer any model to others.

Marginal likelihood (Eq. C.31) has the specific property that shows the trade-off between model fit and model complexity automatically. Therefore, we mainly focus on Eq. C.31 for model selection. To apply model selection process to GPR process we, therefore, apply Eq. C.30 and Eq. C.31 for the first level inference. The posterior distribution given by C.30 is predictive distribution, and we derived it in Appendix C.1. Its mean and covariance are given by Eq. C.22 and Eq. C.20, respectively. Marginal likelihood for GPR is given in Eq. C.12. However, we will rewrite it in the logarithmic form, assuming zero mean for the convenience of the derivations, as

$$\log p(\mathbf{j}_o|\mathbf{U},\boldsymbol{\theta}) = -\frac{1}{2}\mathbf{j}_o^T\mathbf{K}_o^{-1}\mathbf{j}_o - \frac{1}{2}\log|\mathbf{K}_o| - \frac{N_{tr}}{2}\log 2\pi. \tag{C.36}$$

Taking the derivation of Eq. C.36 with respect to hyperparameters (θ), we maximize the log marginal likelihood.

In the case we have multiple data sets, we simply optimize summation of log marginal likelihoods with respect to hyperparameters, which is known as *multi-task learning* (Caruana, 1997).

"Luck is where opportunity meets preparation."
–Seneca