Decision-Driven Data Analytics for Well Placement Optimization in Field Development Scenario - Powered by Machine Learning

Peyman Kor

Submitted in accordance with the requirements for the degree of Master of Science in Petroleum Engineering

University of Stavanger
Energy Resources Department

June 2019

Contents

List of Tables

List of Figures

Acknowledgment

I would like to express my gratitude to my supervisor, Professor Reidar B. Bratvold for his guidance and support throughout the tenure of this work. Reidar is an inexhaustible source of critical thinking and a great mentor and I appreciate him giving me the learning opportunity at his research group.

I am also grateful for my co-supervisor, Dr. Aojie Hong who enthusiastically supported and encouraged me in the area of Machine Learning. Aojie helped me with insightful discussions and comments and was essential to fulfill this thesis.

I wish to appreciate my friends at our research group at UiS for providing a friendly environment and helpful discussions about different issues during these months. Special thanks go to Amine, Xin, Camillo for giving invaluable comments on my work. I also wish to thank all my friends and especially for my two awesome friends, Jan and Juan for being with me during the master program.

Last but not least, my master program could not have been completed without unconditional love and support of my family. No words can begin to justify my appreciation for their support and love to me.

Thesis Objectives

Application of Data Analytics and Machine Learning (ML) in petroleum reservoir management have received a lot of attention in recent years, mainly due to the availability of sheer computational resources and recorded big data set. Taking advantage of ML in subsurface modeling for efficient and computationally inexpensive forecast and as well as incorporating ML in the context of decision analysis, this thesis aims to cover the following objectives:

Objective 1: Building a Proxy Model

The thesis aims to utilize the ML models and past data to build the proxy model (less rich model compared to complex, full-physics based model) in order to make decision in a timely manner during the decision making in field development process (decision nodes, in this case, are coordinates of injection wells and injection rate control parameters). The five-spot pattern consisting of 4 injection wells and one producer were considered as the development scenario in this work. The concept of Fast Marching Method (FMM) was utilized to determine the connectivity of any pair point within the heterogeneous reservoir. Further, these connectivities (as well as pore volume of flight) were used as the features of ML model to predict the desired output (Recovery Factor (RF), Net Present Value (NPV)) through training "simulated" data set. Different ML models were tested, and the best one is selected based on predictive capabilities. To generate the output of the field performance, a numerical reservoir simulator was used with some geological realizations in order to capture the uncertainty in the petrophysical parameters of the field.

Objective 2: Quantification of the Value of Data Analytics in Field Development Project

The study as well as cover the concept of Value of Information (VOI) in order to distinguish constructive information gathering than destructive ones. Here, the developed ML model was served as a information in the context of decision making and the aim is to quantify the value of this information. (Note: in this work we consider the geological, petrophysical, production and injection data as a DATA; ML model finds the pattern between these data and desired response (NPV); so that ML model forecast is a INFORMATION in the decision making context). This quantification will be helpful for oil and gas companies to prioritize their investment in data analytics projects based on the potential value that the ML model could add to the decision context.

Objective 3: Robust Field Development Optimization Using the Proxy Model

Then, the fast and computationally inexpensive proxy (developed in Chap.2) was used to optimize the well placement and as well as injection rates in the aforementioned field development scenario. Genetic Algorithm (GA) type of Evolutionary Algorithm was used as the algorithm of the optimization while the proxy model provides the response of the field out (NPV) at each optimization iteration.

Objective 4: Pitfall and Advantages of Data Analytics in Subsurface Modeling

Finally, the obstacles and opportunities of applying the data analytics and ML in subsurface modeling were discussed. First, the challenges in implementing off-the-shelf solutions (models and framework) from other technical domains (mainly from computer science field) for reservoir production and management were discussed. Then, on the upside, the research maps out the areas and activities that data analytics could significantly help decision maker in subsurface modeling to make a better decision with utilizing the insight from the data.

Chapter 1

Introduction

1.1 Background and Motivation

Application of big data analytics and machine learning in subsurface modeling is taking a lot of attention in recent years. More and more oil and gas corporations are considering to increase their investment in their in-house data to get insight into making a better decision. These practices obviously could bring the speed, less computational resources, more informed decisions and potentially opening up the new alternatives for decisions on the hand. However, the author believes couples of aspect of the implementation of the data analytics and machine learning in subsurface modeling need more careful evaluation. Firstly, although it is possible that the ML model could be a less complex model, the more important question is how much accuracy is going to be sacrificed in order to gain fast model. Secondly, what is the value of building machine learning model (which constitutes data gathering, pattern detection, pattern exploration and exploitation-especially in O&G industry all of thses steps are costly ones) in the particular decision context. Here, the motivation is to chain the concept of VOI (Value of Information) and machine learning. Finally, the current trend in utilizing the data is focused on more " Data-Driven" approach where the data is in the center of decision making; however this study is intended to have a look on " Decision-Driven" approach where the "Decision" in the hand specify which data must be gathered and how much the analyzing the data does worth for that particulate decision.

1.2 Novelty of the Work

The specific novelties introduced in this work to the literature of ML application in subsurface modeling can be summarized as below:

- Five-spot pattern was considered in this work while including 55 injection scenarios in 4 injection wells.
- Concept of Value of Information (VOI) was considered in the particular decision context of the development project in order to analyze the value the data analytics project could add to the decision.
- Genetic Algorithm type of optimization of both well location and injection rates (Joint Optimization) was used in the robust manner while the ML model was served as a fitness function during the evolution of solutions.

1.3 Outline of the Thesis

Chapter 2 of the work is devoted to providing a workflow to develop the machine learning model as a physics reduced model in comparison to rich, fully physics-based numerical reservoir simulator. To develop this ML

model, the Fast Marching Method (FMM) is introduced as the features of the ML model. The chapter concludes by testing the predictive capability of the developed model.

Chapter 3 will discuss VOI analysis and sensitivity analysis using an HRPT (High- Resolution Probability Tree Method). The chapter will provide a workflow to determine the Value of ML model in particular decision context.

Chapter 4 is devoted to utilizing the fast ML model as a response function for optimization. The Genetic Algorithm (GA) will be elaborated and comparison will be made about the performance of the ML model and numerical simulator in the joint optimization.

The final chapter will discuss some takeaways from the application of the ML in subsurface modeling and provide an overview of the advantages and downsides of ML in predictive modeling. In this chapter, some recommendation is made about the types of problems that could be benefited most from ML solutions.

Chapter 2

Development of Proxy Model (Less Rich Model) Using a Machine Learning

2.1 Introduction

Proxy reservoir models [ˆ1] traditionally have been used as a computationally efficient method to simulate reservoir and well responses in subsurface modeling. The advantage of this area of application is that parts of the governing physics can be modified to a simpler mathematical equations, or through the use of data-driven methods, thereby sacrificing some accuracy for a significant decrease in the computational time and resource needed. This fast, computationally inexpensive model is especially very useful during the optimization process where iteration in the order of several thousand is required. Herein, we classify the proxy models in the two main categories, physics-based and data-driven approaches.

2.1.1 Physics-Based Proxies

Physics-based proxies incorporate the mathematics of fluid flow in a simpler framework using assumptions that may be deemed appropriate for the situation. Examples of this approach include capacitance-resistance (CRM) modeling which is based on material balance and derived from total fluid connectivity. (Sayarpour et al., 2009), the Fast Marching Method (FMM)(Sethian, 1996, Sharifi et al. (2014)) and random walker particle tracking (RWPT)(Stalgorova et al., 2012). (Sayarpour et al., 2009) used CRM to characterize the resevoir response based on inter-well connectivity. These connectivities, as will be discussed in the rest of this chapter, present a an efficient way to densely include the reservoir geology without explicitly invoking the petrophysical properties of the field. However, as opposed to static connectivities that are used in this work (In CRM model, history matching is performed to calculate connectivities; while in FMM connectivities are found using before production.). In CRM model connectivity are dependent not only on the static reservoir geology, but on the dynamic injection and production rates.

2.1.1.1 Data-Driven Based Proxies

Over the last decades, data-driven based proxies have increased dramatically in popularity, owing to recent advances in big data and a broad wave of emerging ‘Machine Learning’ applications in research and industry. This class of proxies have a entirely data-driven approach, in which a sets of of data observations are trained for forecasting of reservoir out without relying on any specific physical equatipons. The training data set

used for training the model could be found from field measurements, or synthesized using a commercial reservoir simulator. Most studies in this area have used artificial neural networks (ANN) (Ahmadi et al. (2013)),(Yu et al., 2008), as the learning algorithm, although implementation of tree-based methods such as random forests and gradient boosting is common. (Castelletti et al., 2010)

2.1.2 Data-Driven or Physical Based-Proxy?

Whether using the physics-based proxy or data-driven proxy, one should keep in mind the statement of the George Box(Box, 1979), "All models are wrong, some are useful." Therefore, the main purpose of the models are to support the decision maker with providing the insight. On the other hand, (Bratvold et al., 2009),noted that "O& G companies tend to build too much detail in their decision-making models from the beginning". Here, considering the voluminous codes and models in the commercial simulators, the work tends to build proxy model (less rich model) to help decision makers make a decision in a 'timely' manner. Then, choosing between "Data-Driven" or "Physic Based" proxy, herein the research take fully advantages of the both methods in the new method named "Hybrid Proxy". In this work the proxy is neither fully data-driven nor physical based, but combination of the both. In fact, in the hybrid approach, we calculate the 'Features' of the ML algorithm using the physical-based approach while then we take advantage of the recent advance in pattern recognition techniques (Machine Learning), to find the pattern between those 'Features' and response of the reservoir. The Fast Marching Method (FMM) introduced by (Sethian, 1996) and has been used to compute travel times of the pressure front from a source/sink (Sharifi et al., 2014) often called 'diffusive times of flight' are the features used in this work driven from physic-based analytical method.(Nwachukwu et al., 2018a)

2.2 Workflow

In this chapter the method used to build the proxy model will be explained. The methodology to build this proxy model can be explained in the six steps. First, we provide the brief summary of this methodology but in the next pages more complete description of the each step will be further explained. The flow diagram of the development process has been depicted in the Figure 2.1. The development of this proxy consist of the following steps:

1. Geological model and the type of field development scenario (five spot pattern or any injection scenario) should be specified.

2. After specifying the model, the Sequential Gaussian Simulation (SGS) is performed to build the different realizations the petrophysical parameter. (In this work, the permeability was defined as the source of uncertainty and different realizations are representative of the uncertainty in the permeability data)

3. In each realization, the Fast Marching Method (FMM) is performed to find the connectivities between injectors and producers, between injectors and pore volume of the flight.

4. The realizations in step 2 with it's injection scenario are fed to the numerical simulator to produce the monthly Oil production, water injection and water production as the result forward fully physic-based model.

5. In this step, the output of numerical simulator is post processed to extract the monthly production of water and oil.

6. At last step, we calculate the Net Present Value (NPV) of the development scenario to be used as the output of the ML algorithm.

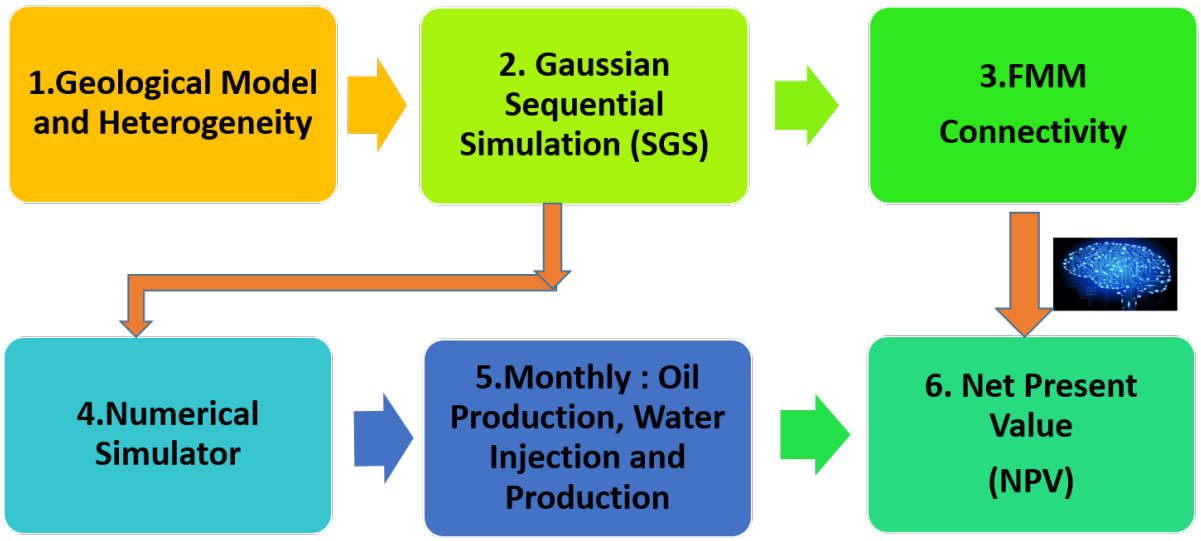

Figure 2.1: Flow Diagram of Development of the Proxy Model

Table 2.1: Rock and Fluid Characteristics of the Geological Model

Parameter	Value
Grid Dimension	45 by 45
Grid Size	10 by 10 by 10 m
Porosity	20%
Compressibility	10^-5 1/psi
Initial pressure	234 bar
Initial Saturation	0.3

2.2.1 Geological Model and Heterogeneity

In this work, two-dimensional reservoir geometry with two phases water/oil system was considered. The grid based geological model with size of the $45 * 45$ was used in this work. The parameters of the model and rock and water properties have shown in the Table 2.1.

The relative permeability curves of the reservoir model in this work as well have been shown in the Figure 2.2. The relative permeability curves cross each other on Sw>0.6, indicating more 'water-wet' system of the fluid and rock of the model.

2.2.2 Generating Geological Realization using Gaussian Sequential simulation (SGS)

In this work, Sequential Gaussian Simulation (SGS) was used to generate realizations of the each permeability. For example, in the 5-spot pattern, the permeability of the 5 wells (4 injectors and one producer) are driven randomly from the distribution of the permeability of the field which has the log-normal distribution.(Pyrcz and Deutsch, 2014)

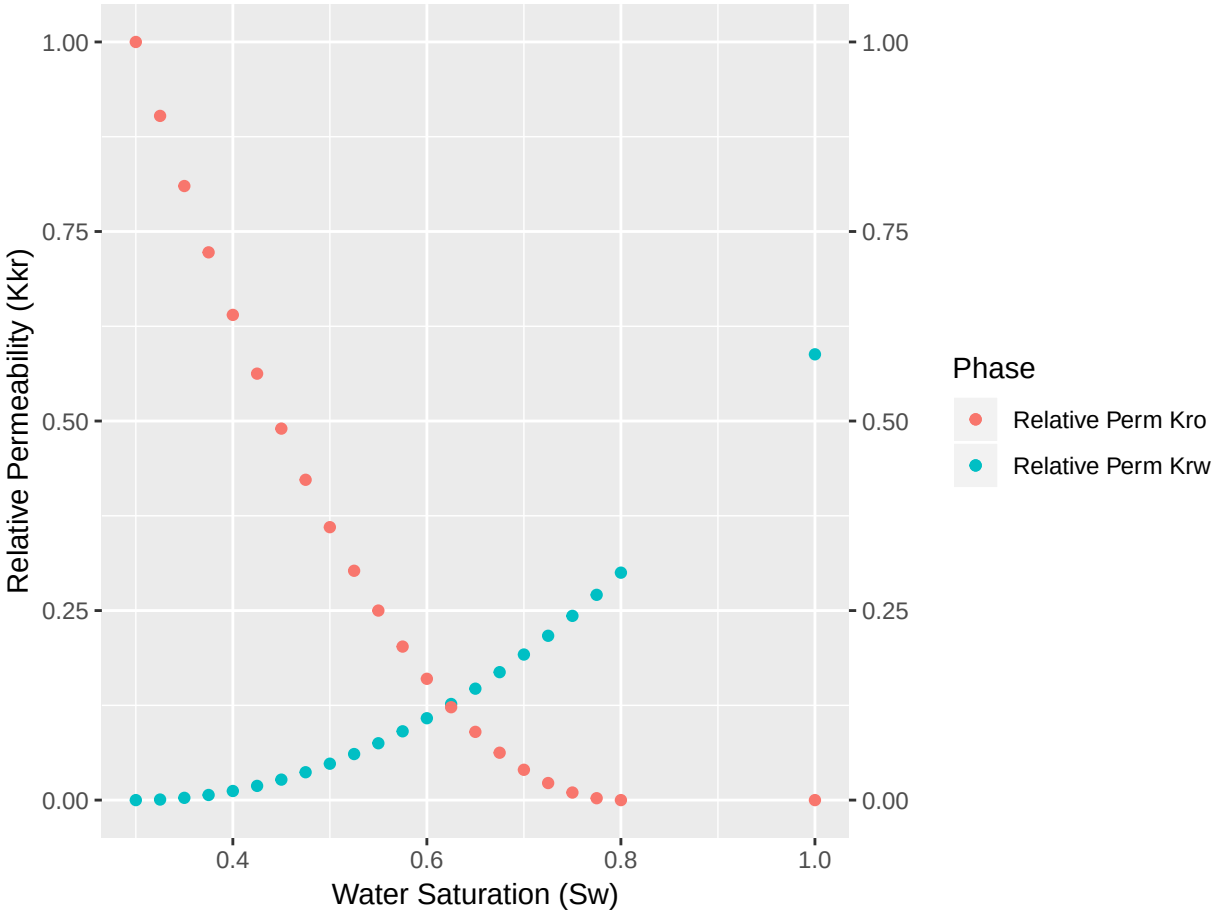

Figure 2.2: Relative Permeability Curves, water/oil System

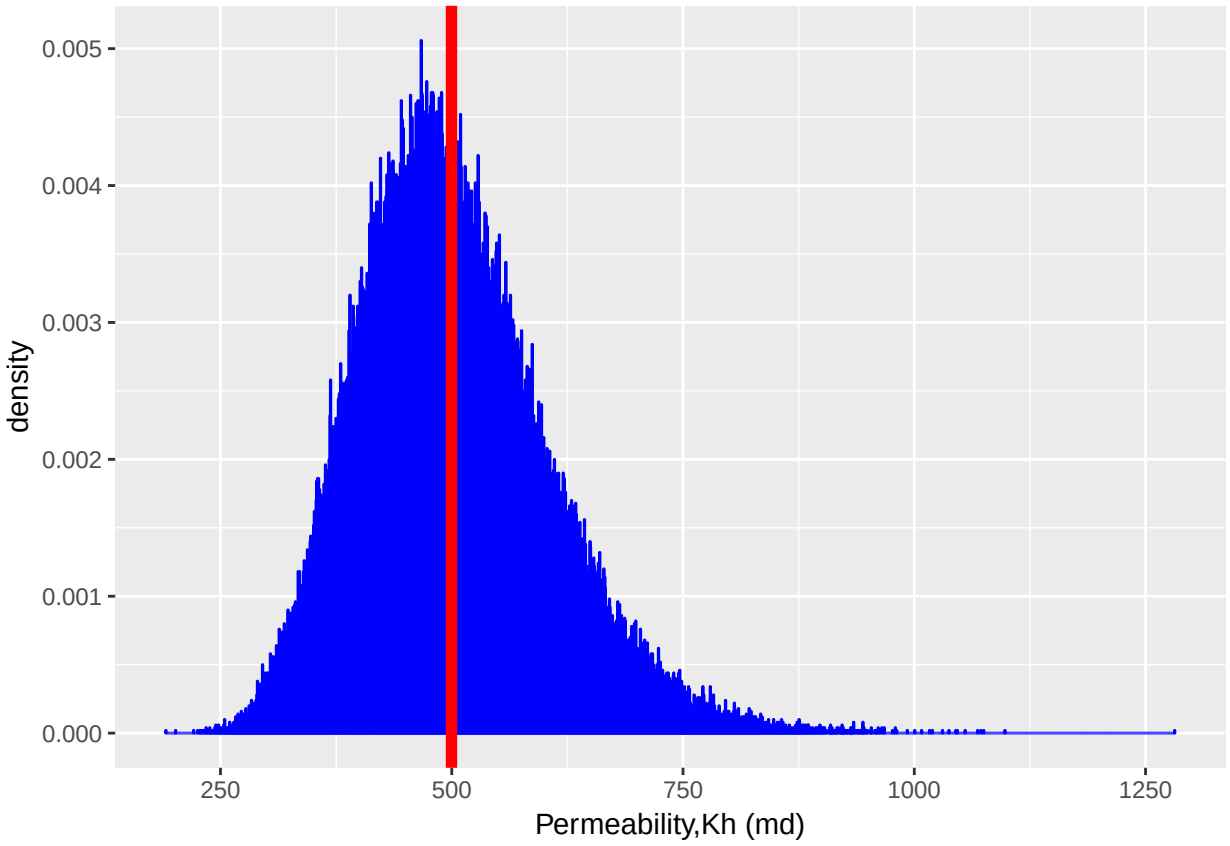

Figure 2.3: Log-Normal Distribution of the Field Permeability

Table 2.2: Assigned Permeability Values for 5 Points of 5-Spot Pattern

X	Y	Perm	logperm
225	225	614.6815	2.788650
45	435	462.9789	2.665561
375	345	558.9657	2.747385
125	15	603.3033	2.780536
325	25	515.1680	2.711949

For the sake of this work, we considered the following distribution in the Figure 2.3 as the distribution of the permeability in the field under study (It is considered $K_h = K_v$)

Now, for every training observation [^2], we build 10 realizations [^3] of the permeability. SGS starts by creating a grid of randomly assigned values drawn from a standard normal distribution (mean = 0 and SD = 1). The co-variance model (from the semivariogram defined in the Simple Kriging layer, which is required as input for generating realizations) is then applied. This ensures that raster values conform to the spatial coordinates found in the input data-set. The resulting raster constitutes one unconditional realization, and many more can be produced using a different raster of random values from gaussian Distribution at each time.

The steps involved in SGS can be summarized as the follows:

Step 1: In this case, the log 10 of permeability data are transformed into Gaussian values using the Q-Q plot.

Step 2: The distance matrix is calculated, that is the distance between the data and the unknown location. Here the unknown locations are the random path.
Step 3: Input the model of spatial continuity in the form of an isotropic spherical variogram and nugget effect (contributions should sum to one).
Step 4: Variogram matrix is calculated by applying the distance matrix to the isotropic variograms model.

Step 5: Covaraince matrix is calculated by subtracting the variogram from the variance (1 for standard normal distribution).

Step 6: The left hand side of the co-variance matrix is inverted.
Step 7: The inverted left hand-side matrix is multiplied by the right hand side matrix to calculate the simple kriging weights.

Step 8: The kriging estimate and kriging variance are calculated with the weights and co-variances.

Step 9: With the Gaussian assumption for the local CDF, the Monte Carlo simulation is applied to draw simulated realizations at the random path.

For example, for one of the training data set in five-spot pattern, the following permeability (in md) were drawn from the field permeability distribution (Figure 2.2). The X and Y shows spatial coordinates of the production well and injection wells, while the columns three and four provide the permeability value of those points.

To visualize the data shown in the Table 2.2, the Figure 2.4 shows the spatial locations of the 5 points for calculating SGS. As was mentioned, the X and Y coordinates both has 45 grids with the size of the 10 m, therefore the coordination of the plot in Figure 2.4 will be from 0 to 450 m in both X and Y direction. It must be mentioned that in the all training data-set, we considered the production well is located in the center of geological model, in other word in grid (23,23).

In this work the parameters of the semivariogram to generate different realization out of SGS method has been depicted in the Table 2.3. Note to mention that the sill considered for nugget effect is the the variance of the 5 permeability points in the Table 2.2.

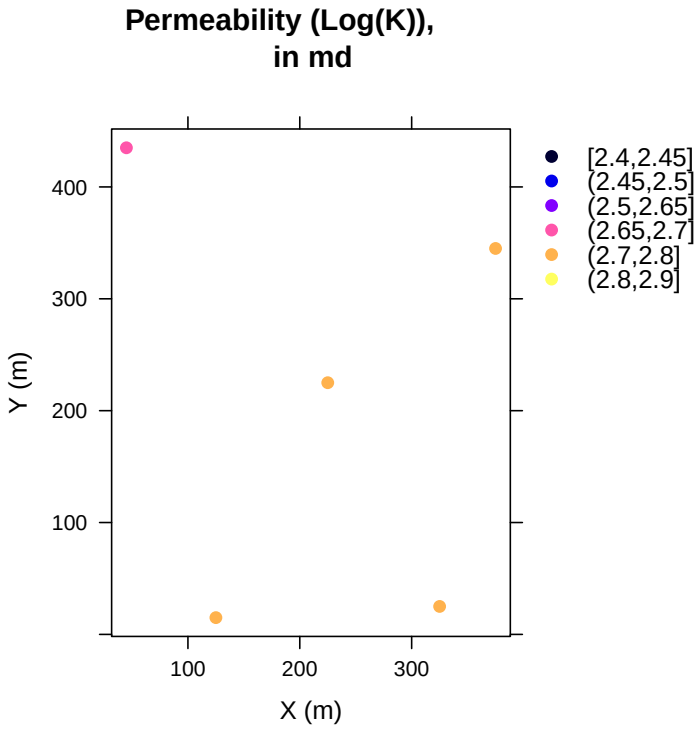

Figure 2.4: Graphical Representation of Five Permeability Points, Shown in the X, Y Coordinates

Table 2.3: Parameters of Semi variogram to Perform SGS

Parameter	Value
Nugget Effect	Sill/2, md^2
Type	Spherical
Range	20 (grid cell)
Anistropy ratio	1
Azimuth	0-degree (North)

Having specified the 5 permeability points and semivariogram parameters the SGS method could be performed. The Figure 2.5 shows the four realizations. In this work, every training observation has 10 permeability realizations.(Note, the range of color-bar at each realization in the Figure 2.5 starts from the minimum of the permeability until maximum of the permeability at THAT realization.)

```
## drawing 10 GLS realisations of beta...
## [using conditional Gaussian simulation]
```

2.2.3 Measure of Connectives

The FMM is a numerical method for the tracking the monotonically advancing surface on a grid based structure. One of the prorogation problem is where the prorogation waves moves in one direction and strictly expands. This boundary values equation is known as Eikonal equation and can be written as:

$$F(x)|\bigtriangledown \tau(x)| = 1 \tag{2.1}$$

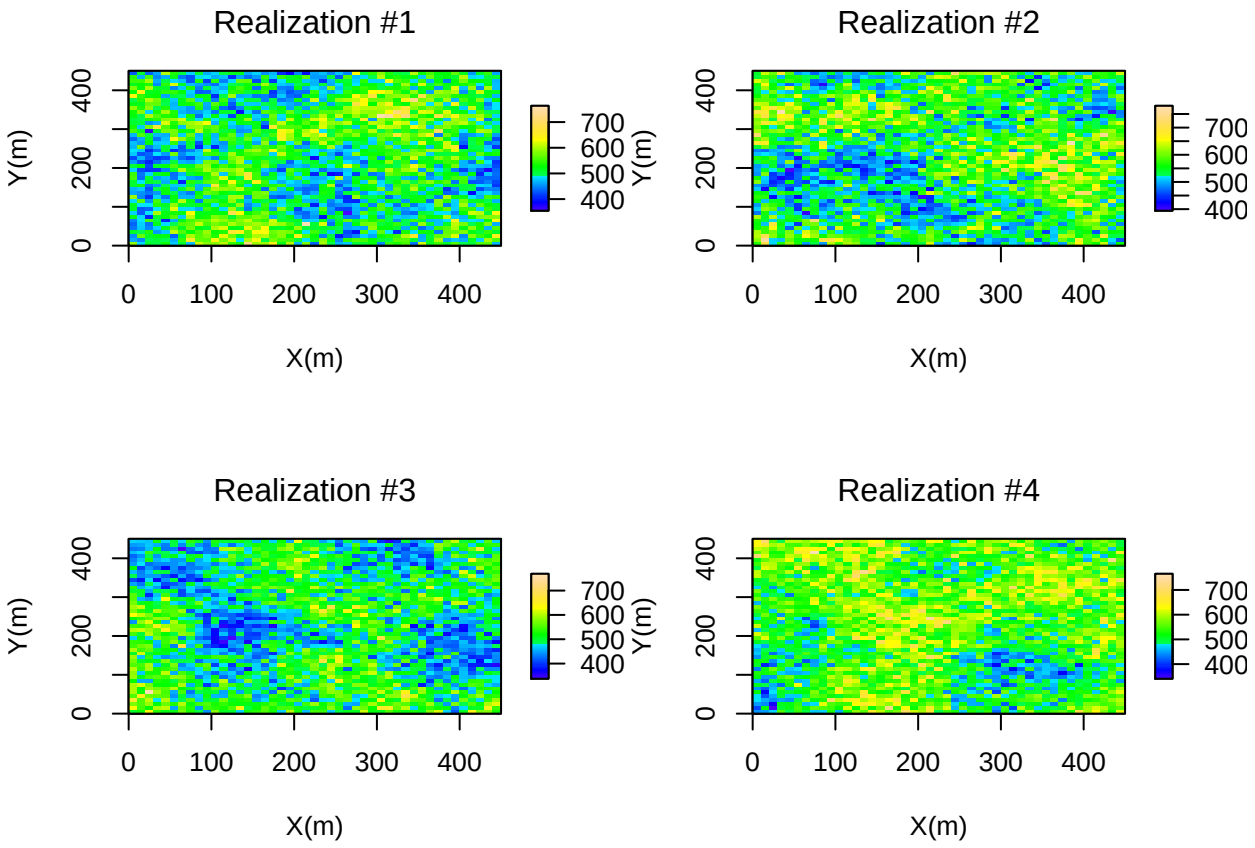

Figure 2.5: Four Realizations of the Permeability Ditribution Found Using SGS Method

where the τ is the arrival time , F is speed function. The FMM is introduced by (Sethian, 1996) as numerical method for solving Equinal form of equation. The first-order fast-marching formulation to calculate the propagation time in each grid ($\tau_{i,j,k}$) can be summarized as:

$$\mu c_t(x)\phi(x)\frac{\partial p(x,t)}{\partial t} - k(x)[\bigtriangledown^2 p(x,t)] - \bigtriangledown k(x).\bigtriangledown p(x,t) = 0 \tag{2.2}$$

where $p(x)$ is the pressure, $\phi(x)$ is the porosity, $k(x)$ is the permeability, μ is the viscosity , and $c_t(x)$ is the total compressible (summation of rock and fluid compressibility). Using the Fourier transform , the Equation could be written as:

$$\frac{\mu c_t \phi(x)}{k(x)} iw\hat{p}(x,w) - \frac{\bigtriangledown k(x)}{k(x)}.\bigtriangledown \hat{p}(x,w) - \bigtriangledown^2 \hat{p}(x,w) = 0 \tag{2.3}$$

Assuming the heterogeneous media the $\frac{\bigtriangledown k(x)}{k(x)}$ is negligible we can write:

$$\bigtriangledown^2 \hat{p}(x,w) - \frac{1}{\eta(x)} iw\hat{p}(x,w) = 0 \tag{2.4}$$

Where the term $\eta(x) = \frac{k(x)}{\mu c_t \phi(x)}$ is called diffusivity.

An asympotic solution gives:

$$\hat{p}(x,w) = e^{[-\sqrt{-iw}\tau(x)]} \sum_{n=0}^{\infty} \frac{A_n(x)}{(\sqrt{-iw})^n} \tag{2.5}$$

where the w is the frequency, $\tau(x)$ is phase propagation, and $A_n(x)$ is the coefficient of in the expansion. The previous equation can be used to calculate the operators:

$$\bigtriangledown \hat{p}(x,w) = e^{[-\sqrt{-iw}\tau(x)]} \sum_{n=0}^{\infty} \frac{1}{(\sqrt{-iw})^n} \times [-\sqrt{-iw}A_n(x)\bigtriangledown\tau(x) + \bigtriangledown A_n(x)] \tag{2.6}$$

$$\bigtriangledown^2 \hat{p}(x,w) = e^{[-\sqrt{-iw}\tau(x)]} \sum_{n=0}^{\infty} \frac{1}{(\sqrt{-iw})^n} \times [-(\sqrt{-iw})^2 A_n(x)\bigtriangledown\tau(x)\bigtriangledown\tau(x) - \sqrt{-iw}\bigtriangledown\tau(x)\bigtriangledown A_n(x) - \sqrt{-iw}A_n(x)\bigtriangledown^2\tau(x) - \sqrt{-iw}\tau \tag{2.7}$$

$$\bigtriangledown^2 \hat{p}(x,w) = e^{[-\sqrt{-iw}\tau(x)]} \sum_{n=0}^{\infty} \frac{1}{(\sqrt{-iw})^n} \times [-(\sqrt{-iw})^2 A_n(x)\bigtriangledown\tau(x) \tag{2.8}$$

Now this equation can be written as:

$$e^{[-\sqrt{-iw}\tau(x)]} \sum_{n=0}^{\infty} \frac{1}{(\sqrt{-iw})^{n-2}} \times [A_n(x)\bigtriangledown\tau(x)\bigtriangledown\tau(x) - \frac{1}{\eta(x)} - 2\bigtriangledown\tau(x)\bigtriangledown A_{n-1}(x) \tag{2.9}$$

one can set zero to the coefficient of the the individual powers of $\sqrt{-iw}$ starting with the highest power $(\sqrt{-iw})^2$, now we can write:

$$\bigtriangledown\tau(x)\bigtriangledown\tau(x) - \frac{1}{\eta(x)} = 0 \tag{2.10}$$

Table 2.4: Model Parameters for Radius of Investigation Calculation

Parameter	value
Grid-block Size	20 by 20 by 20
Porosity	10%
Permeability	1 md
Compressibility	10^-5
Viscosity	1 cp
Initial pressure	4000 psi

The above equation has the Eikonal form and relates the $\tau(x)$ we call it diffusive propagation time to $\eta(x)$ which is the diffusivity coefficient.

$$F(x) = \sqrt{\frac{k(x)}{\mu c_t(x)\phi(x)}} max(\frac{\tau - \tau_1}{\triangle x/F_I}, 0)^2 + max(\frac{\tau - \tau_2}{\triangle/F_J}, 0)^2 + max(\frac{\tau - \tau_3,}{\triangle z/F_K}, 0)^2 = 1 \tag{2.11}$$

2.2.4 Analytical Method

Now, in this section the goal is to compare results of the FMM method in tracking the pressure waves with the well known analytical well testing method. Considering the homogeneous reservoirs model with the characteristics depicted in the Table 2.4, the radius of investigation(Here, the radius of investigation is defined as the the radius, the pressure waves reaches after time (t)).

$$r = \sqrt{\frac{kt}{948\mu c_t \phi}} \tag{2.12}$$

Where t is the time (hours), k is the permeability (md), μ is the viscosity (cp), c_t is the total compressibility (1/psi), and ϕ is the porosity.

The figure plotted in Figure 2.6 shows the time pressure arrives at different radius (as contour plot). As could be seen, the FMM methods performs well in the capturing the pressure propagation compared to the exact analytical solution found from concept of the radius of investigation. However, the analytical method developed was valid in the case of the homogeneous reservoir model, whereas the geological model considered in this study is fully heterogeneous, therefore in he rest of this work, the FMM method will be used to calculate the connectivities between each pair point.

2.2.5 Features and Response

To utilize the Machine Learning method, we need to define the features of the model and as well response we are looking for. In this work the features of models are:

- Features
  - Connectivity
  - Pore Volume Flight
- Response
  - NPV The connectivity is defined by the value of 'diffusive time of flight' and the PV is the defined as the sum of all grid affected by pressure disturbance when the pressure propagation reach the injection well.

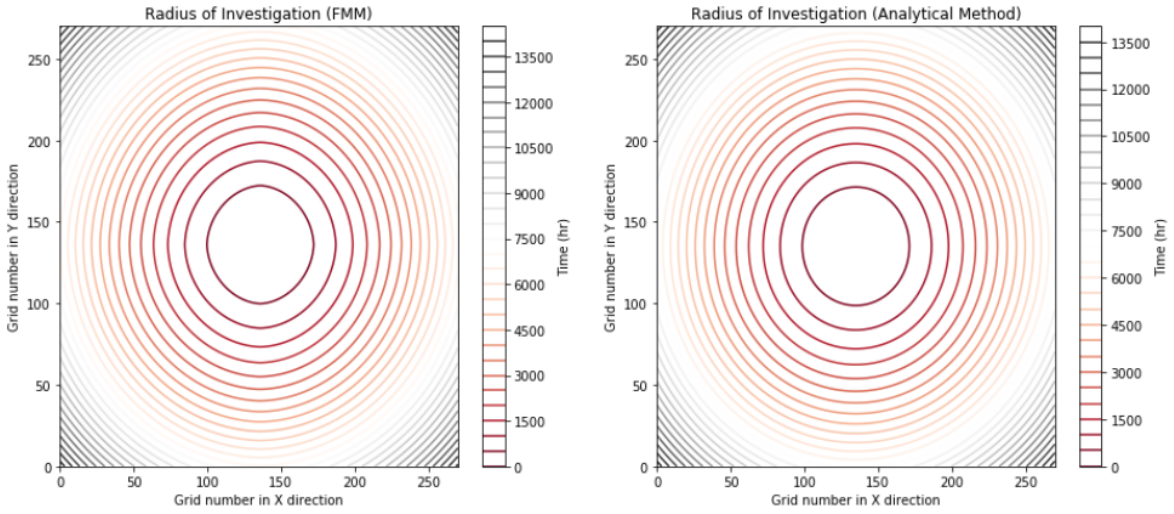

Figure 2.6: Comparison of Pressure Wave Propogation Profiles, FMM Method vs Analytical Solution

Table 2.5: Economic Parameters of the NPV Calculation

Parameter	value
Oil Price, per barrel	70 $
Water Production Cost, per barrel	15$
Water Injection Cost, per barrel	5$
Discount Factor	8%

The NPV is defined as the below:

$$NPV = \sum_{k=1}^{n_T} \frac{[q_o^k P_o - q_w^k P_w - I^k P_{wi}] \triangle t_k}{(1+b)^{t_k/D}} \tag{2.13}$$

Where the parameter is defined is below: q_o^k: is the field oil production rate at time k q_w^k: is the field water production rate at time k I^k: is the field water injection rate at time k P_o: is the oil price $P_w p$: is the water production cost $P_w i$: is the water injection cost b: is the discount factor $t-k$: is the cumulative time for discounting D: is the reference time for discounting (D=365 days if b is expressed as fraction per year and the cash flow is discounted daily) q_o^k,q_w^k and I^k are forecasted by given production model.

In this work, the following economical parameters were considered to calculate the Net Present Value after recessing the Numerical Reservoir Simulator.

2.2.6 Machine Learning Algorithm

Big data vary in These call for different approaches. Wide data consists of :

Thousands/Millions of variables Hundreds of samples

Tall data has a dimension in the following range:

Tens/Hundreds of Variables Thousands/ Millions of Samples

In briefly, the most well-known ML algorithm can be summarized as the follows:

- Linear/Logistics Regression

- k-Nearest Neighbours
- Support Vector Machine
- Tree-based Model *Decision Tree* Random Forest *Gradient Boosting Machine* Neural Networks

In this work, We use the XGBoost (short for extreme Gradient Boosting) ML model since it has showed several successful application in oil and gas industry and as well is the winning model for several Kaggle competitions (Nwachukwu et al., 2018a, Nwachukwu (2019)).

2.2.7 eXtreme Gradient Boosting Model

suppose we have K trees, the model is defined as:

$$\sum_{k=1}^{K} f_k \tag{2.14}$$

Where each f_k is the prediction from a decision tree . The model is a collection of decision trees. Having all the decision trees, we make prediction by:

$$\hat{y}_i = \sum_{k=1}^{K} f_k(x_i) \tag{2.15}$$

where x_i is the feature vector for the $i-th$ data point. Similarly , the prediction at the$t-th$ step can be defined as:

$$\hat{y}_i^{(t)} = \sum_{k=1}^{t} f_k(x_i) \tag{2.16}$$

To train the model, the loss function need to be optimized:

- Rooted Mean Square Error for regression
- $$L = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2$$
- Log- Likelihood for multi-classification
- $$L = -\frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{M} y_{i,j} log(p_{i,j})$$

Regularization is another important part of the model. A good regularization term controls the complexity of the model which prevents over- fitting.

$$\Omega = \gamma T + \frac{1}{2}\gamma \sum_{j=1} T w_j^2 \tag{2.17}$$

Where T is the number of the leaves, and w_j^2 is the score on the j-th leaf. Putting loss function and regularization together, we have the objective of the model:

$$Obj = L + \Omega$$

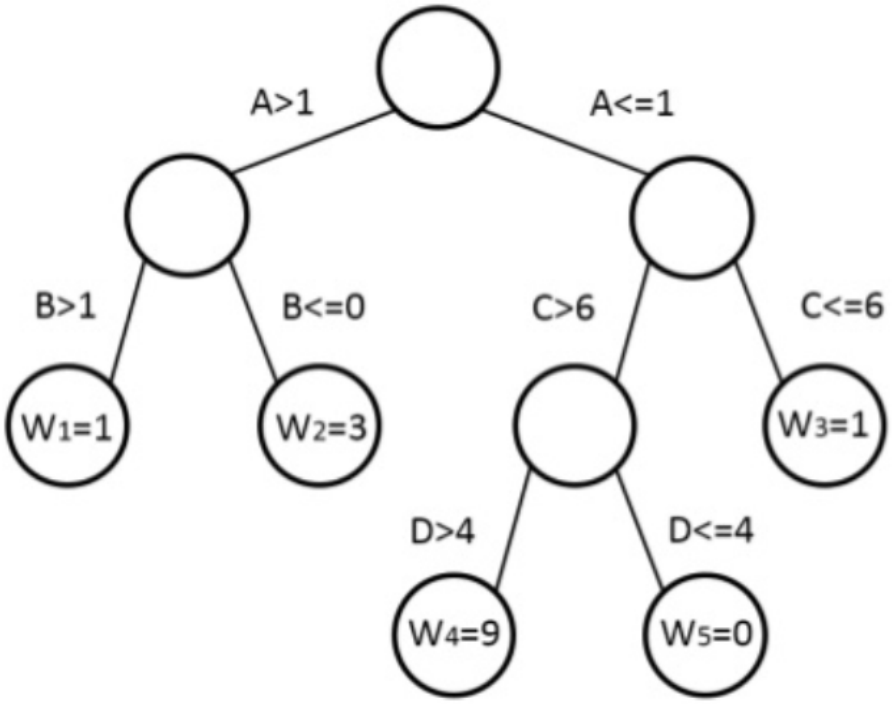

Figure 2.7: Representation of a Decision Tree in XGBOOST

In the XGBoost method, the gradient descent is used to optimize the objective. Given an objective $Obj(y, \hat{y})$ to optimize, gradient descent is an iterative technique which calculate:

$$\partial_{\hat{y}} Obj(y, \hat{y}) \tag{2.18}$$

at each iteration.Then we improve $\hat{y}$ along the direction of the gradient to minimize the objective. In the case of iteration, the Objective function $Obj = L + \Omega$ can be defined as :

$$Obj^{(t)} = \sum_{i=1}^{N} L(y_i, \hat{y}_i^{(t)}) + \sum_{i=1}^{t} \Omega(f_i) = \sum_{i=1}^{N} L(y_i, \hat{y}^{(t-1)} + f_t(x-i)) + \sum_{i=1}^{t} \Omega(f_i) \tag{2.19}$$

The tree structure in XGBoost leads to the core problem: How we can find a tree that improves the prediction along the gradient? The idea of the gradient descent is used to solve this problem.

2.2.8 Model Building and Validation

In this, 500 training observation were used to build the ML model. Each training observation then has 10 realizations that make total training data set equal to 5000. The features of this model has been shown in the table.

Then, the Table 2.6 provides the features that carries the information about the geology and pressure propagation in the reservoir model, not the injection rate parameters. To include the injection scenarios, the exponential form of injection scenario was considered, as the follow:

$$A * exp(-\gamma * t) \tag{2.20}$$

Table 2.6: Connectivity Network as the Feature of the ML

Parameter
Connectivity Pair (Inj1-Inj2)
Connectivity Pair (Inj3-Inj1)
Connectivity Pair (Inj3-Inj2)
Connectivity Pair (Inj4-Inj1)
Connectivity Pair (Inj4-Inj2)
Connectivity Pair (Inj4-Inj3)
Connectivity Pair (Pro1-Inj1)
Connectivity Pair (Pro1-Inj2)
Connectivity Pair (Pro1-Inj3)
Connectivity Pair (Pro1-Inj4)
PV of Flight

Here, we consider the 5 different A values as the starting injection rates and 11 γ. Therefore, the total injection scenario included in this work is $(5 * 11 = 55)$. In the Figure 2.8 we can see the 55 injection scenarios having 11 values for γ (left side of the Figure) and Showing all scenarios (Right side of the figure).

Now, specifying each injection scenarios based on it's A and γ values, we can theses features to our initial features, to complete the features which captures both the physic of the flow and the control parameters.

Now, the full sets of the features shown in the Table 2.7 and the output of the ML (NPV) is ready. The number of training data set is (N=5000) and the algorithm of ML is XGBOOST. The ML algorithm here has six parameter that needed to be tuned. Therefore, hyper parameter optimization process was doe considering 288 combination of the parameters to fund the tuned parameters of the ML. It must be mentioned that, in this work 5-fold Cross- validation was used to avoid any potential over-fitting problem in the development of the ML. The final parameters of the ML in this work can be shown in the Figure 2.8

[ˆ1] Here, we call proxy reservoir model because it is less rich in complexity and in addition is not fully based on solving governing physical equations, rather model reduced form of the physical models. [ˆ2]: In this work training observation means the same geological model which has the different well locations and injection scenarios. [ˆ3]: The realization in this work means the same well placement and and injection rates but different permeability distribution to capture the uncertainty of this petrophysical parameter.

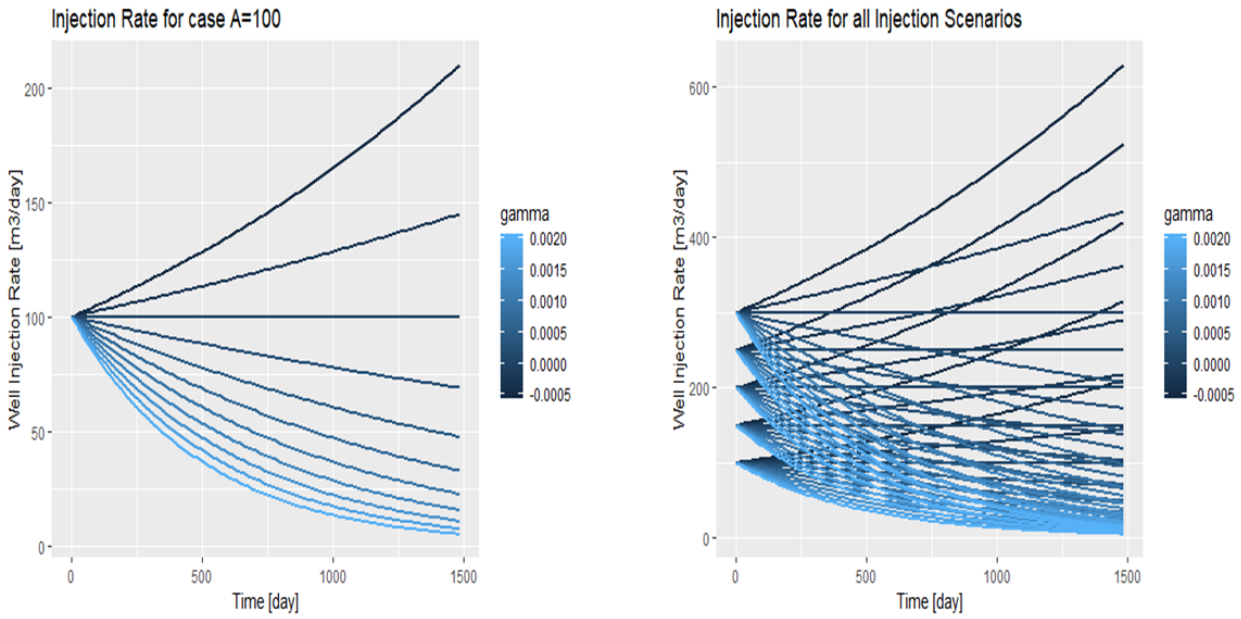

Figure 2.8: Left: 11 Injection Scenarios at A=100, Right: All Injection Schemes

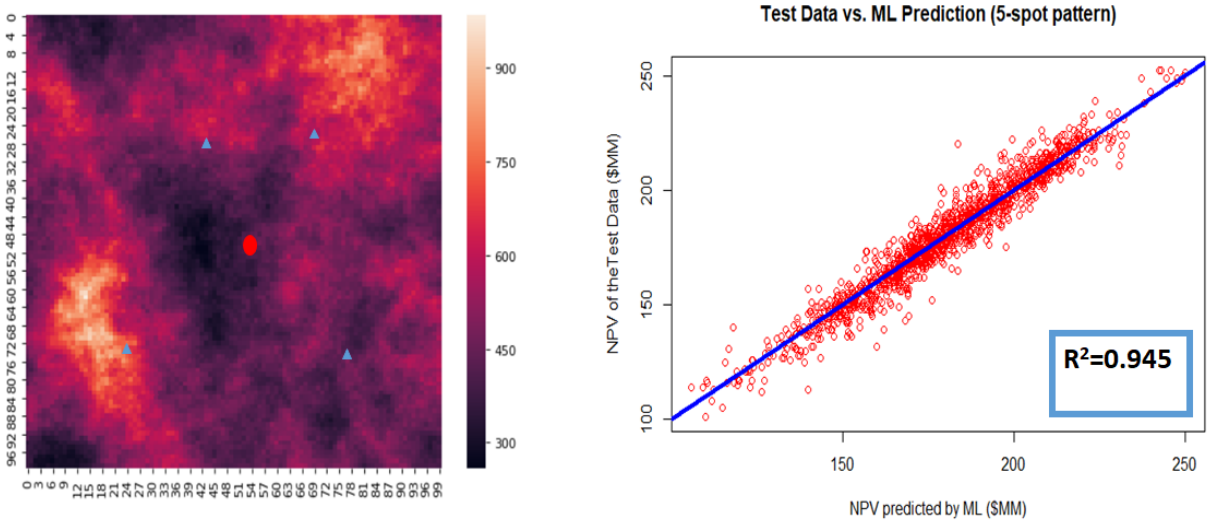

Figure 2.9: An Example of a Training Data Set and Testing Model Capability

Table 2.7: Connectivity Network as the Feature of the ML + Control Parameters for Injection Rates

Features
Connectivity Pair (Inj1-Inj2)
Connectivity Pair (Inj3-Inj1)
Connectivity Pair (Inj3-Inj2)
Connectivity Pair (Inj4-Inj1)
Connectivity Pair (Inj4-Inj2)
Connectivity Pair (Inj4-Inj3)
Connectivity Pair (Pro1-Inj1)
Connectivity Pair (Pro1-Inj2)
Connectivity Pair (Pro1-Inj3)
Connectivity Pair (Pro1-Inj4)
PV of Flight
A value for Inje1
A value for Inj2
A value for Inj3
A value for Inj4
Gamma value for Inj1
Gamma value for Inj2
Gamma value for Inj3
Gamma value for Inj4

Table 2.8: Tuned Parameters of the XGBOOT Model

XGBOOST Parameter	Value
nrounds	4000
Max_depth	6
Etta	0.001
Gamma	0
Min_Child_Weight	2.25
Sub-sample	1

Chapter 3

Value of Data Analytics in Field Development Project (VOI Analysis)

3.1 Background

The concept of Value of Information (VOI) first was used in Oil&Gas industry in making decision related for drilling the reservoir(Grayson, 1960). This concept originally comes from Decision Analysis (DA) community . (Schlaifer, 1959) first defined the concept of Value of Information in the context of business decisions. More recently, (Bratvold et al., 2009) provided the overview the application and future recommendation of applying VOI analysis in the oil and gas industry. (Eidsvik et al., 2015) has provided the overview of the use cases of the VOI in the earth science applications. According to the (Bratvold et al., 2009) any information gathering is concerned with two fundamental uncertainties,

1. The uncertainties we hope to learn about it,but can not directly observe it
2. The test result, which we refer to as the observable event

Genteelly, in this discussion (as well in the included codes), we denote the x to the distinction of the interest and y denote the observable distinctions.

The VOI is the defined as the "maximum value the decision maker should pay for additional information gathering regarding to the distinction of interest (x), and is defined as the below,(considering the risk neutral case):

VOI = EV with additional Information - EV without additional

Here, the EV means the Expected Value of the Decision. For example, for this particular context of decision making (Figure 3.1), assuming the three outcomes, the Value of Information can be shown mathematically as the Equation 6.2:

$$VOI = \sum_{j=1}^{N} P(z_j).max(\sum_{i=1}^{N} P(x_i|z_j)x_i, v) - max(\sum_{i=1}^{N} P(x_i)x_i, v) \tag{3.1}$$

Where,

- D_k: denotes alternative k for a decision,
- x_i denotes realization i of payoff,
- z_j denotes realization j of information signal (a test result),

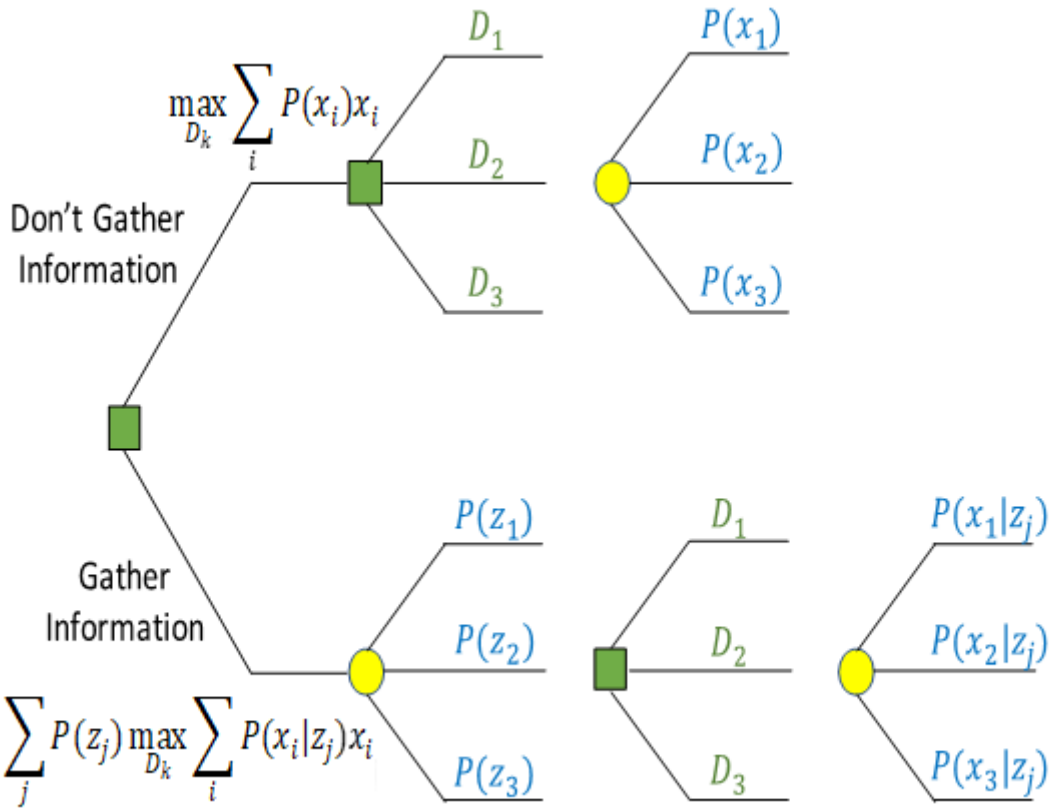

Figure 3.1: Decision Context for the Oil Company (without and with information)

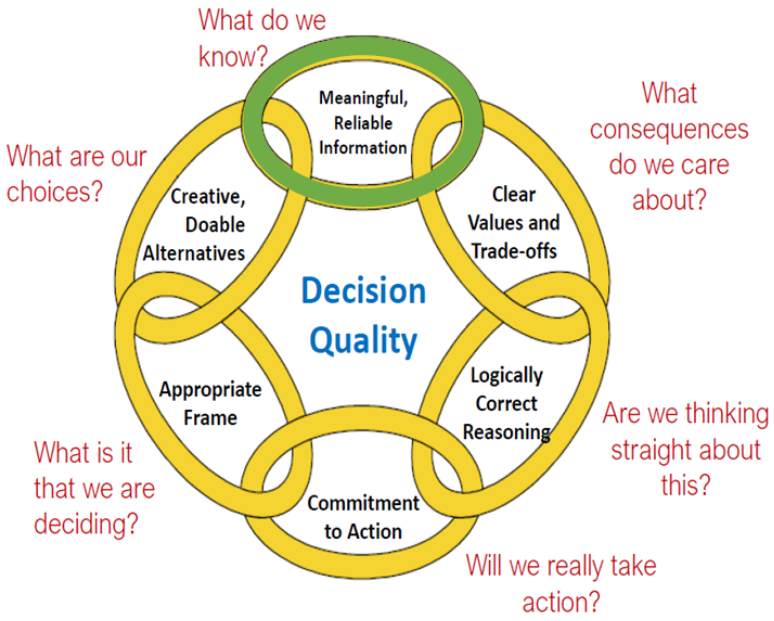

Figure 3.2: Six Dimesnions of High-Quality Decision Making

- P(x_i) is the probability of payoff x_i before gathering information z_j (i.e., P(x_i) denotes the prior),
- P(x_i|z_j) is the conditional probability of payoff x_i given information signal z_j (i.e., P(x_i|z_j) denotes the posterior),
- and P(z_j) is the marginal probability of information signal z_j (i.e., P(z_i) denotes the preposterior).

VOI analysis, helps us in providing the framework to distinguish the constructive than wasteful information gathering process. Considering six dimension of the high-quality decision making (Figure 3.2), it could be considered that the ML developed in the previous chapter, work as information with updating our prior belief regarding decision to develop (start drilling) the field or walk away from the development.

This decision considering the three possible outcomes Low, Medium, High (In the term of the it's NPV value that again depends on well locations and injection scenarios) can be depicted in the Figure 3.3 . In this case the decision maker has two decision to make, whether to drill or walk away(It was considered the walking away the decisions no cost to the decision maker in this case.)

3.2 High-Resolution Probability Tree Method (HRPT) for VOI Analysis

In this, we will use the High-Resolution Probability Method (HRPT) developed by (Bratvold et al., 2014). First, to demonstrate this techniques, we use the HRPT method for one simple decision context with the known analytical solution and then we apply this method in the decision context described in the Figure 3.4. In this example, we suppose that a risk-neutral Oil company is considering drilling a well in an undeveloped area where the outcome (Net Present value) in this case is normally distributed with the mean of the $\mu = 10MM$ and $SD = 20MM$, in Dollar.

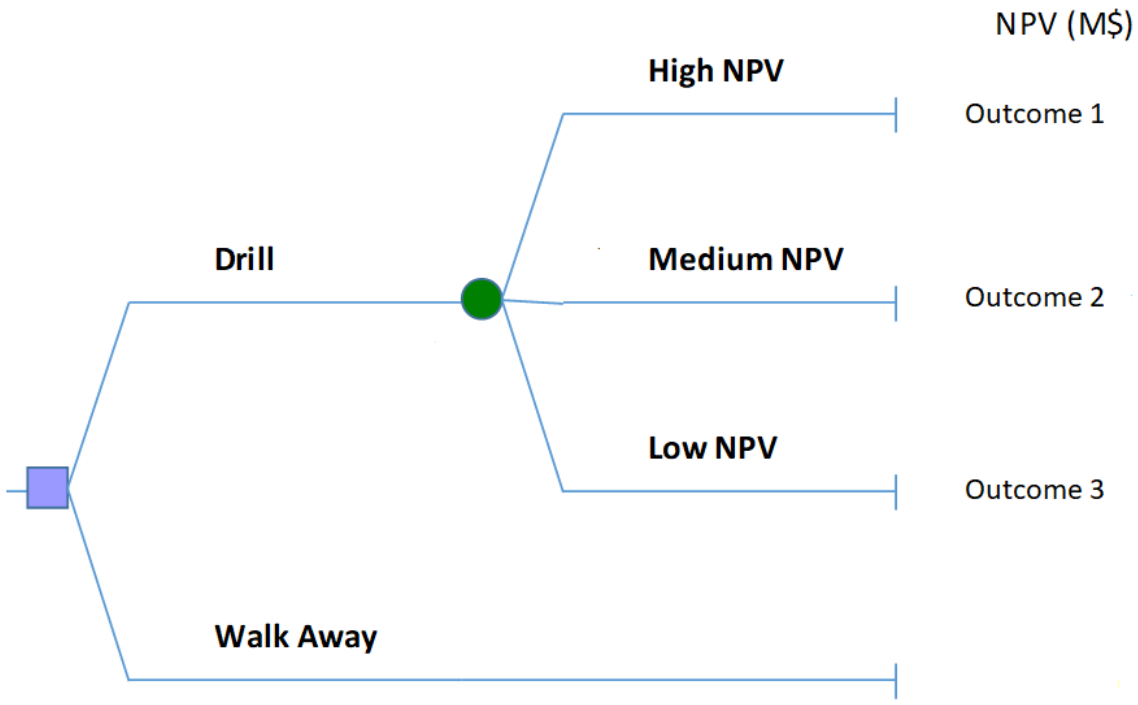

Figure 3.3: Simple Decision Tree for the Case of the Making Decision for Drilling 5-Spot Pattern (discussed in chap2)

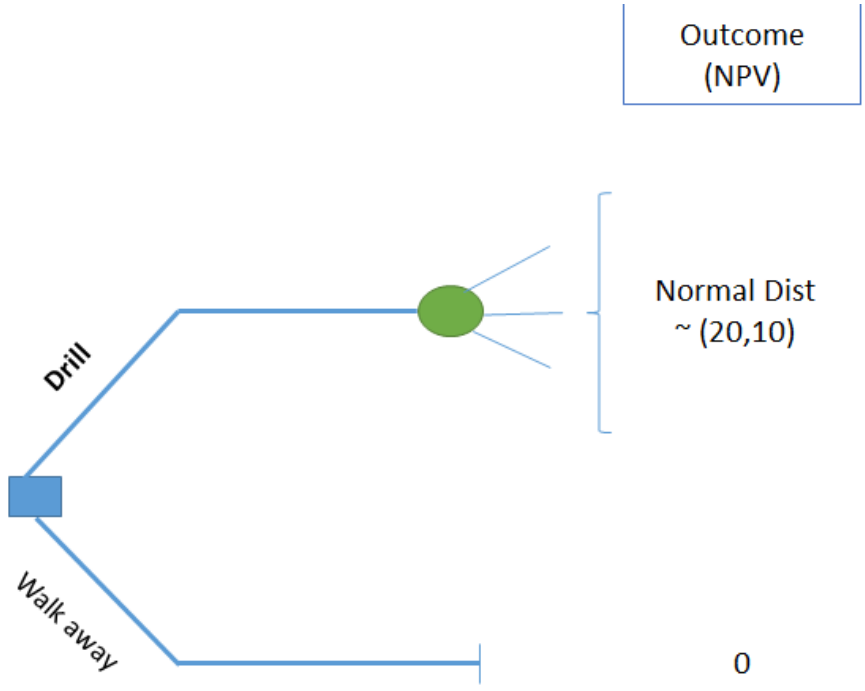

Figure 3.4: Decision Tree for TALL-N Problem (without information)

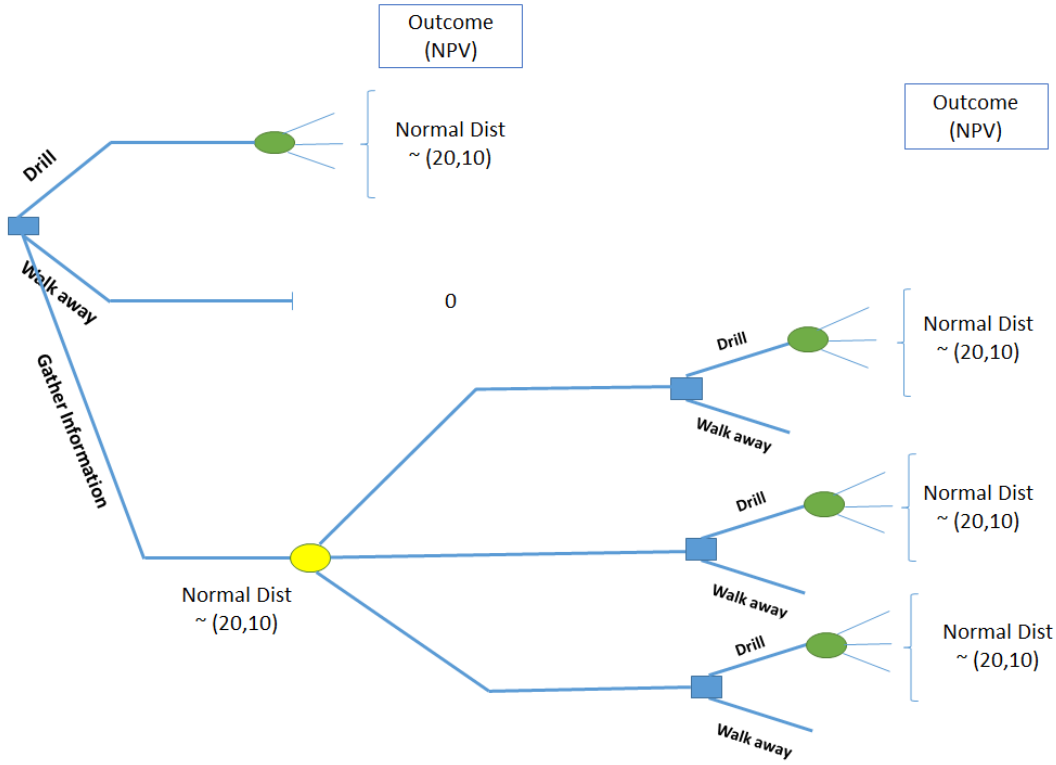

Figure 3.5: Decision Tree for TALLN Problem (with information)

In the effort to make a better decision, the company considering the acquisition of seismic survey. The expert in the company believes that the seismic results are correlated with true value of the well with the correlation coefficient of $\rho = 0.6$. It is assumed that in this case, the signal (seismic survey) will have a normal distribution with the same mean and standard deviation ,Figure (3.5. Here the decision makers face three decisions to make,1) Start the Drilling 2) Walk away from the Drilling, 3) Gather the information about the uncertainty of the out put. However, the information gathering has a specific value to could add to the decision, (VOI) that need be analyzed before "Gathering the Information."

Now, considering the prior distribution and signal distribution of this problem (is named TALL-N), Figure (3.6) shows both the prior distribution as well signal.

Considering this information, Now we can use the (HRPT) method to calculate the Value of information (VOI) in this case. It is worth to mention that in this case, we used the the racket and mean method for discritizing the prior as well the signal distribution. In addition, the number of nodes (Nodes) in this case was 1000. Therefore, the the conditional probability will have the dimension of the 1000 in 10000.

Full other related data regarding this VOI case has been depicted in the Table 3.1.

On the other hand we know that this TALL_N problem has a analytical solution as the follow:(Sethian, 1996)

$$EVII = \begin{cases} \rho\sigma[\phi(\rho^{-1})c - \rho^{-1}c\psi(-\rho^{-1}c)] & \text{if }, \mu > v \\ \rho\sigma[\phi(\rho^{-1})c + \rho^{-1}c\psi(\rho^{-1}c)] & \text{if }, \mu < v \end{cases} \tag{3.2}$$

Where the, μ is the mean of the prior; σ is the standard deviation of the prior; v is the value of alternative decision-the best decision without gathering more information); $c = (\mu - v)/\sigma$ is known as the "coefficient of the divergence"; σ is the standard normal probability density function ; ψ is the standard normal cumulative

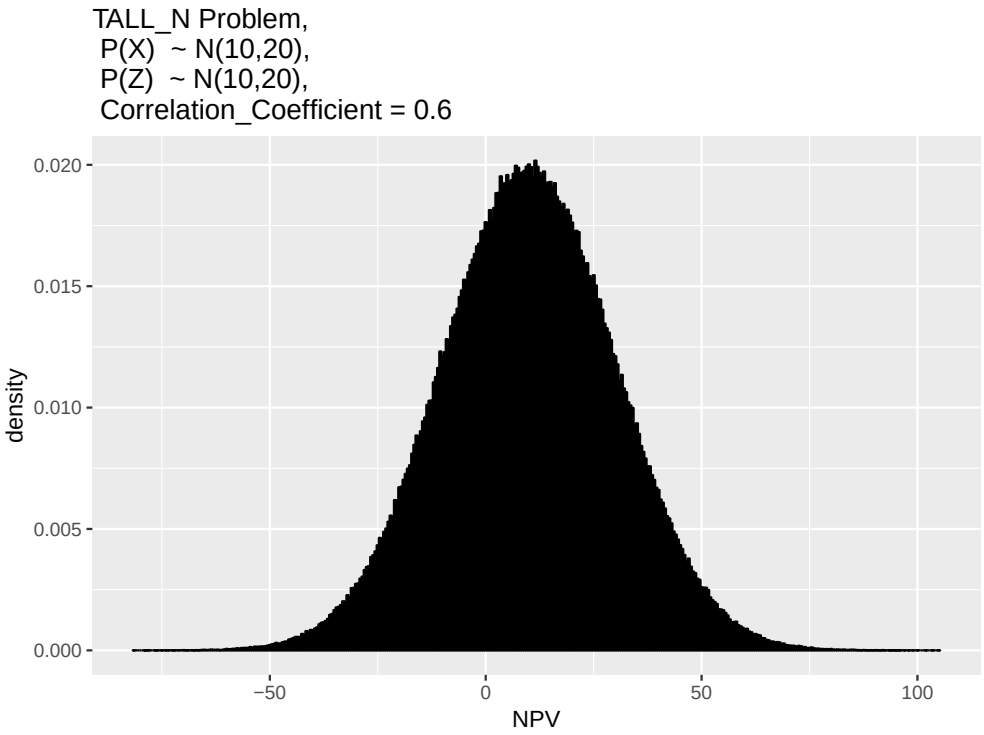

Figure 3.6: Permeability Distribution of the Field

Table 3.1: Prior and Information System Characteristics in the TALL-N Problem

Parameter	Value
Mean Prior(x_mean)	10.0
Standard Deviation (Prior)	20.0
Mean Signal (z_mean)	10.0
Standard Deviation Signal (z_sd)	20.0
Correlation Coefficient (rho)	0.6
Cost	0.0

distribution function, ρ is the positive correlation coefficient between the prior and the observed signal obtained through information system.

Here, the result of HRPT method and as well exact solution were found and the comparison of these two results has been made in the Figure @ref(fig:compa. The comparison was made to capture also range of the correlation coefficient and gain the number of nodes in discritization method in this case was (nodes=1000). In context of the TAll_N problem, the Table provide the value of the VOI found from the HRDT method, Exact solution and the (@ Bratvold et al., 2014) paper.

3.3 Sensitivity Analysis of VOI to Prior, Likelihood and CAPEX

In VOI discussion, the value of information must be assesd before gathering the information. This provides insight to the decision maker about the maximum value that worth to pay for gathering the data. Here, in the context of the Machine Learning the cost of Information includes the following:

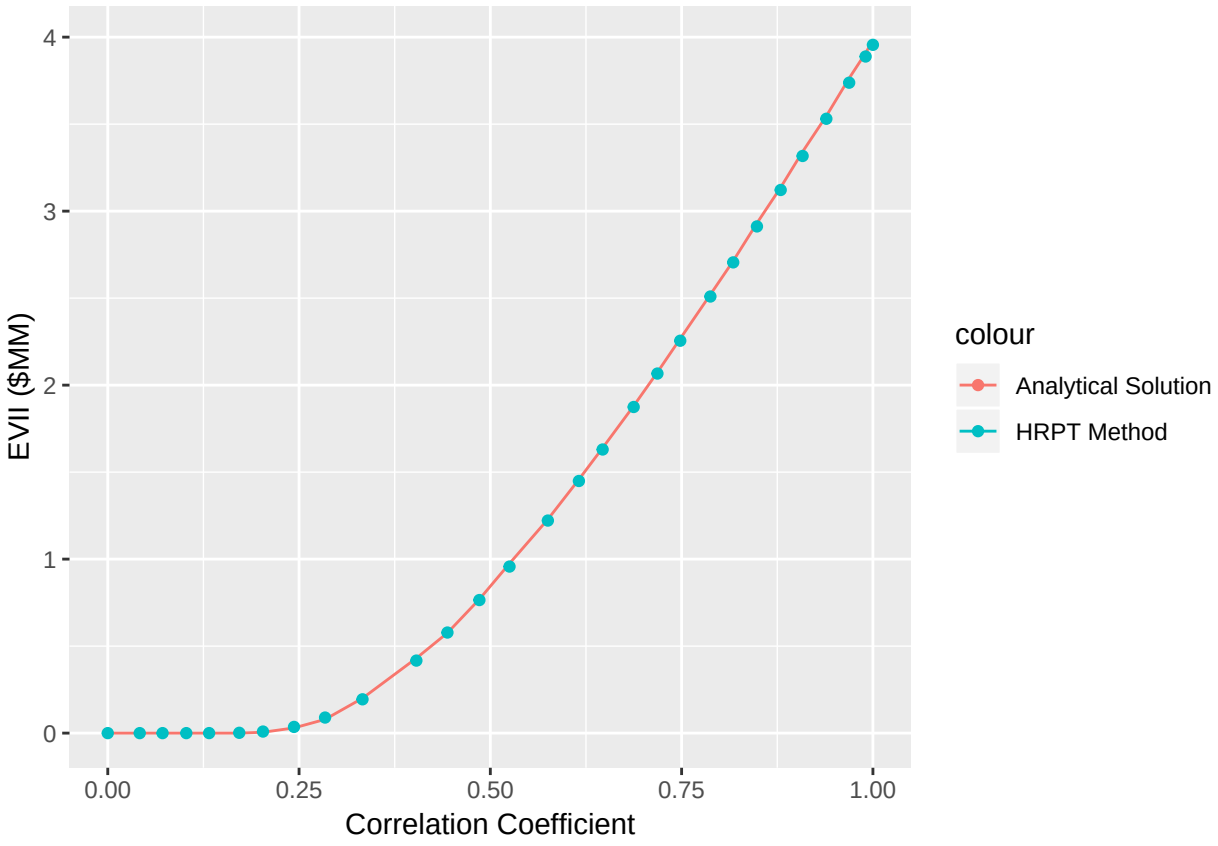

Figure 3.7: Comparison of the VOI found in Analytical Solution Vs. HRPT Method

Table 3.2: The Result of VOI from HRPT Method, N(nodes)=1000

VOI	
Value of Information (VOI) is:	1.35953174492042

- Data Acquisition
- Pattern Detection
- Pattern Recognition
- Pattern Exploration
- Pattern Exploitation

On the other hand, the reliability of the ML model can be evaluated only if after acquisition of data and building the ML model. However, in the concept of VOI, the value of information must be asses **before** gathering the data. Therefore, we make a sensitivity analysis not only on the reliability of the result of data analytics (ML model) but also to the prior and as well CAPEX of the project. It must be mention that in the VOI analysis, the outcome of the event NPV is defined as the follows that includes the CAPEX cost (All the spending cost of the development project before production of the field):

$$NPV = \sum_{k=1}^{n_T} \frac{[q_o^k P_o - q_w^k P_w - I^k P_{wi}] \triangle t_k}{(1+b)^{t_k/D}} - CAPEX \tag{3.3}$$

Now, the discussion above aims to provide the framework to find the VOI in the cases of different scenarios for the following parameters:

- Mean of Prior Distribution
- Standard Deviation of Prior Distribution
- CAPEX (Capital Cost)
- Reliability of the Information

Mean of Prior Distribution:

In the work developed in the Chapter 2, we had 5000 total training set. These data-set represents the historical data set for this specific '5-Spot Pattern'. Well, in fact the prior is defined as the *historic data' and 'Expert Knowledge', in this work while we consider the 'Historic data' as the prior, we will have different cases to include the several scenarios of the Prior distribution. To get the hint regarding which range of the prior distribution to be considered in sensitivity analysis, first we have look on the historical data-set of the 5-spot patter. The Figure 3.8 shows the histogram of the NPV of 'Historical' data-set. The box plot of this distribution was plotted in Figure 3.9. we can see that the historical data has the shape of the Normal distribution with the mean of 54 MM$ and standard Deviation of the 5.6 MM.

Now this gives us the idea about the different scenario of the prior we could have in the sensitivity analysis. In this work, we considered 5 different possible cases for prior distribution. As shown in the Figure 3.10, these five distribution were considered as the possible scenarios of the prior (all in $MM):

1. Normal Distribution, $\mu = 30$
2. Normal Distribution, $\mu = 40$
3. Normal Distribution, $\mu = 50$
4. Normal Distribution, $\mu = 60$
5. Normal Distribution, $\mu = 70$

Standard deviation of Prior Distribution:

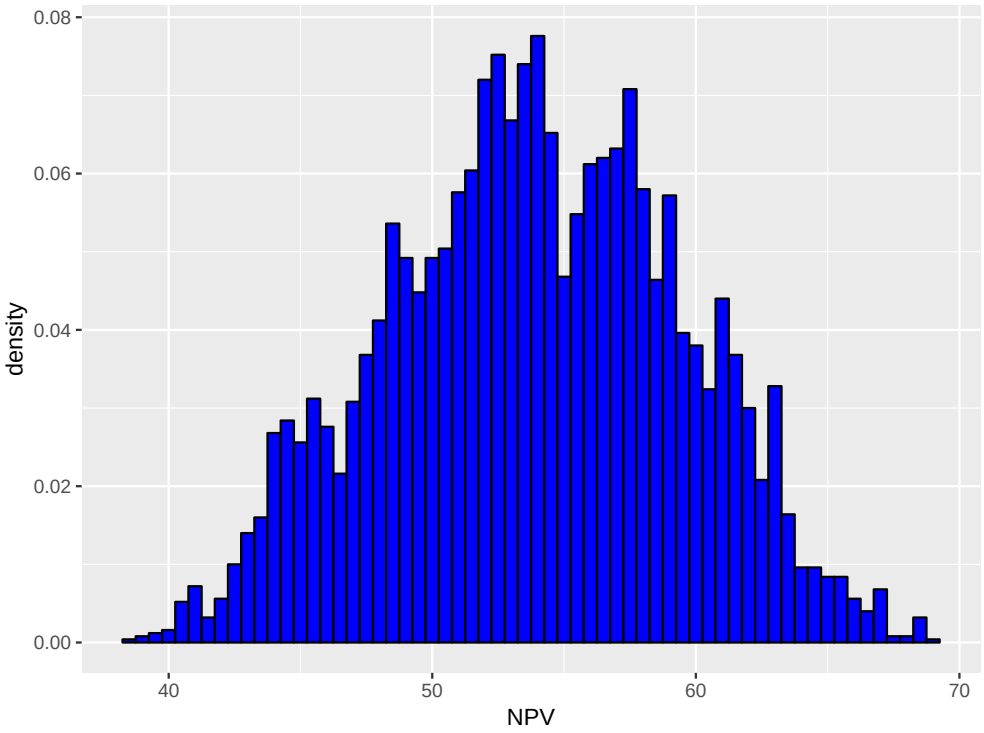

Figure 3.8: Distribution of the Historical NPV

However, considering the only different mean values of the prior distribution is not sufficient. Therefore, in this study we considered the both the change in the mean and as well spread of the data from the mean with considering the 4 different possible standard deviation. Again, as it was discussed in the for the mean analysis, we had the look on the standard deviation of the "Historical data" to get the insight about the range of the standard deviations to be included in the sensitivity analysis. The Standard Deviation of the the past data-set was, $SD = 5.6$, then the following were considered for the range of SD values. (Note: Each mean prior, will have 4 different SD values in the analysis, therefore, the total number of the prior distribution are $5 * 4 = 20$).

1. Normal Distribution, $sd = 2.5$
2. Normal Distribution, $sd = 5$
3. Normal Distribution, $sd = 10$
4. Normal Distribution, $sd = 15$

The Figure 3.11 shows the 4 different assigned standard deviation at the mean prior distribution of the $\mu = 50$.

CAPEX:

The evaluation of the CAPEX (Capital Cost before the oil production) must be done through economic expert of the the decisions. Since the NPV defined in this case is all consider the money discount factor, the CAPEX values in this study as well must be expresses in the Net Present Values, that is why we call the CAPEX in the figures as PV_CAPEX Here, We considered the four possible range for the CAPEX in order to capture the range of CAPEX values, these are as follows (in MM$):

1. CAPEX $PV(Capex) = 30$
2. CAPEX, $PV(Capex) = 40$

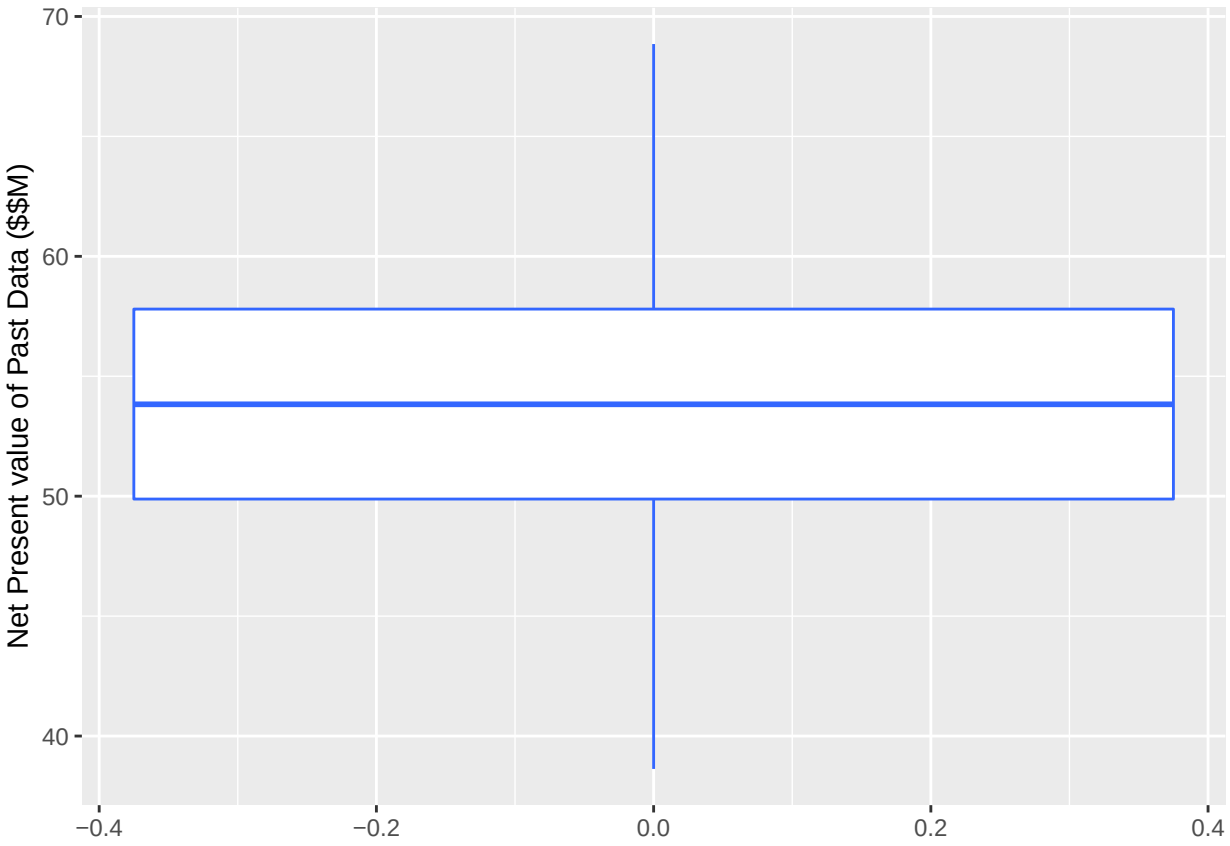

Figure 3.9: Box Plot of Permeability Distribution

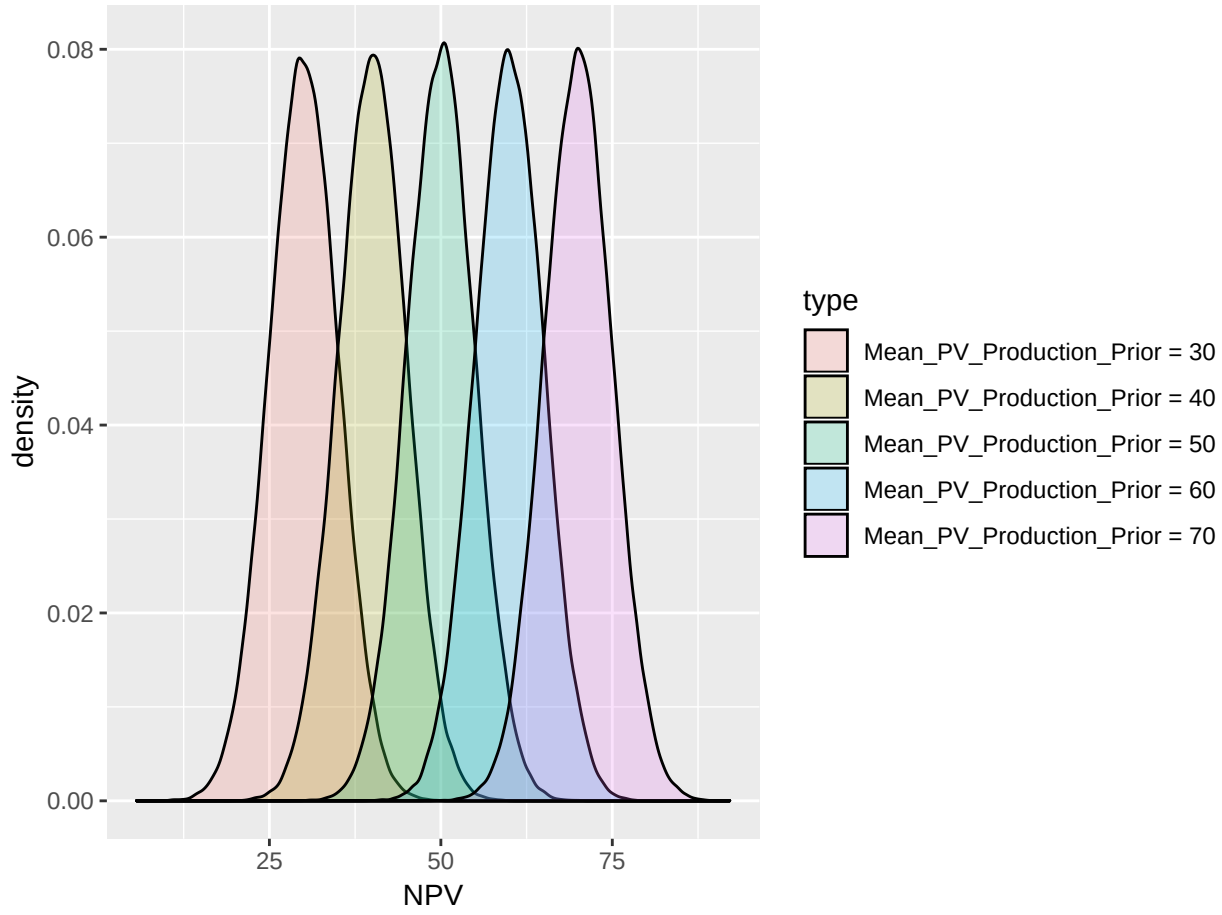

Figure 3.10: Range of Prior Distribution in VOI Analysis-Normal Distribution

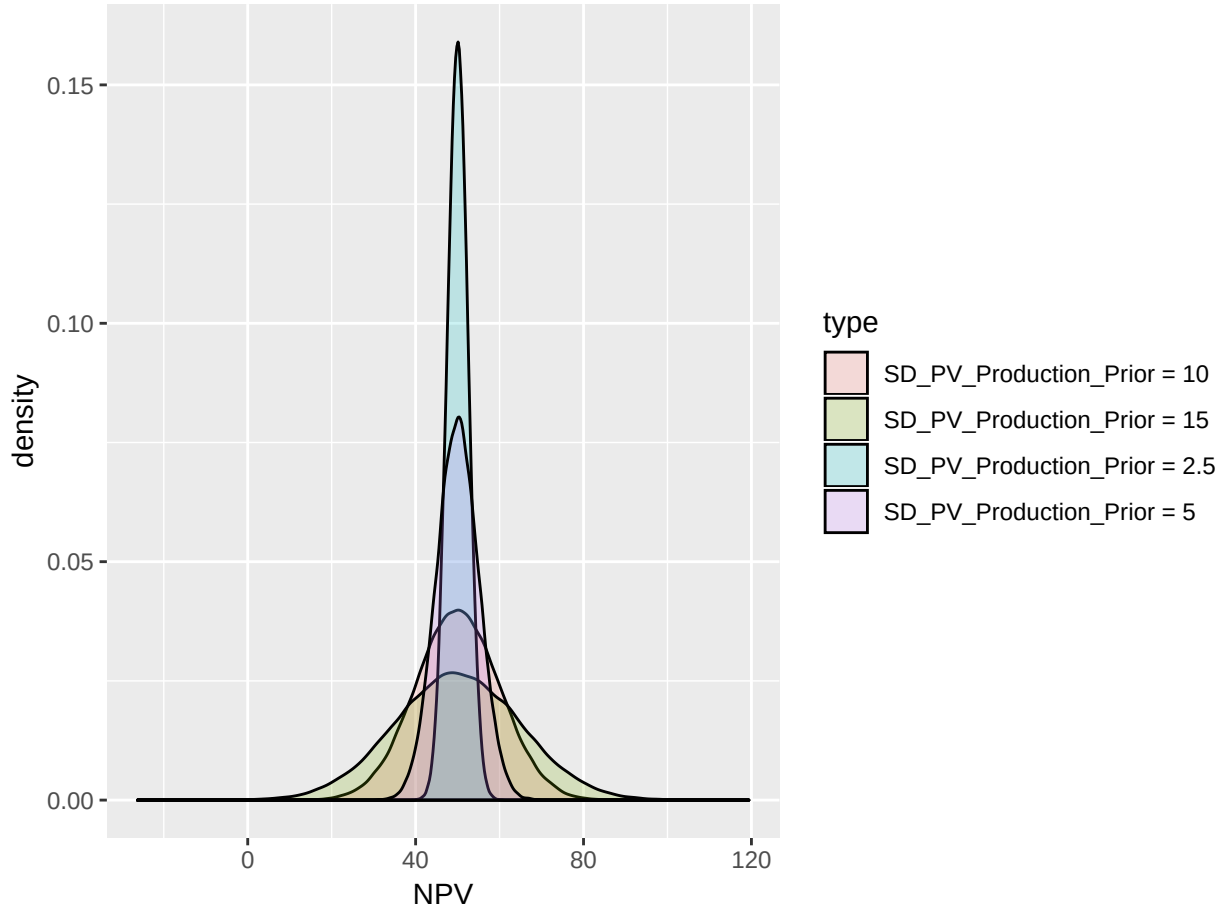

Figure 3.11: Prior Distribution for Four SD Values at Mean = 50 MM(Dollar)

3. CAPEX, $PV(Capex) = 50$
4. CAPEX, $PV(Capex) = 60$

The Figures (3.12, 3.13,3.14,3.15) show the results of sensitivity analysis of the VOI at different standard deviation of the prior, SD=2.5,5,10,15. For example when we have a normal distribution withe the SD = 2.5, the decsion context with the following characteristics,:

- Mean of prior distribution: 50 MM (Dollar)
- Correlation Coefficient: 0.8
- CAPEX: 50 MM (Dollar);

The VOI for this case is equal to 0.8 MM (Dollar). In addition we can see two different trends:

- The VOI increases with increasing the reliability of the information (Correlation Coefficient)
- The VOI has the highest value when the CAPEX is equal to the mean of the prior distribution, showing that the VOI is more valuable when it has stronger capability to change our decisions.
- With increasing standard deviation values, the VOI get more valuable. The main reason that could be attributed is that in the case of high standard deviation, there is more down side in distribution of the prior, so that the reliable information could be more valuable in order to avoid that downside.

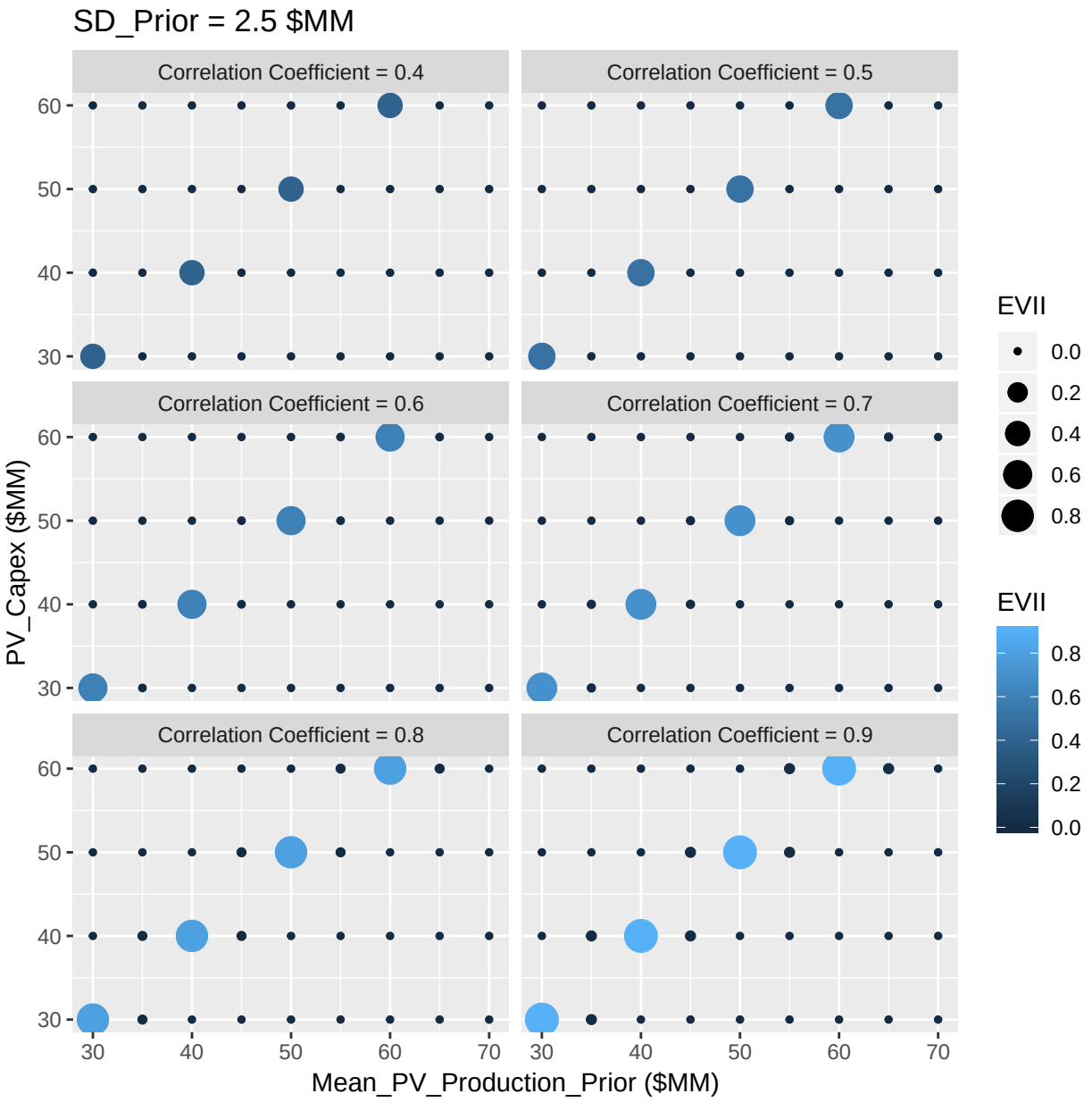

Figure 3.12: Sensitivity Analysis of VOI at SD=2.5

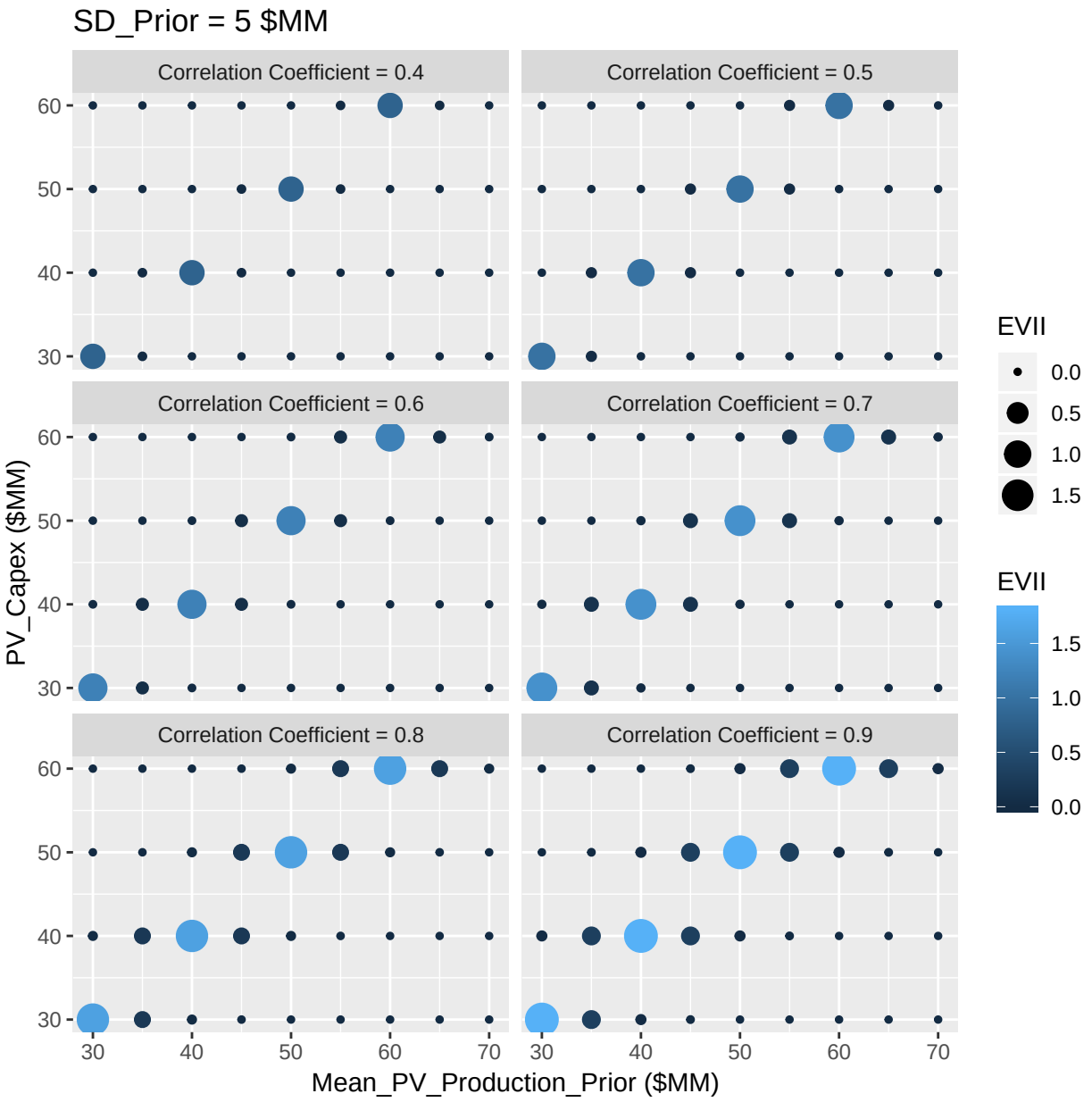

Figure 3.13: Sensitivity Analysis of VOI at SD=5

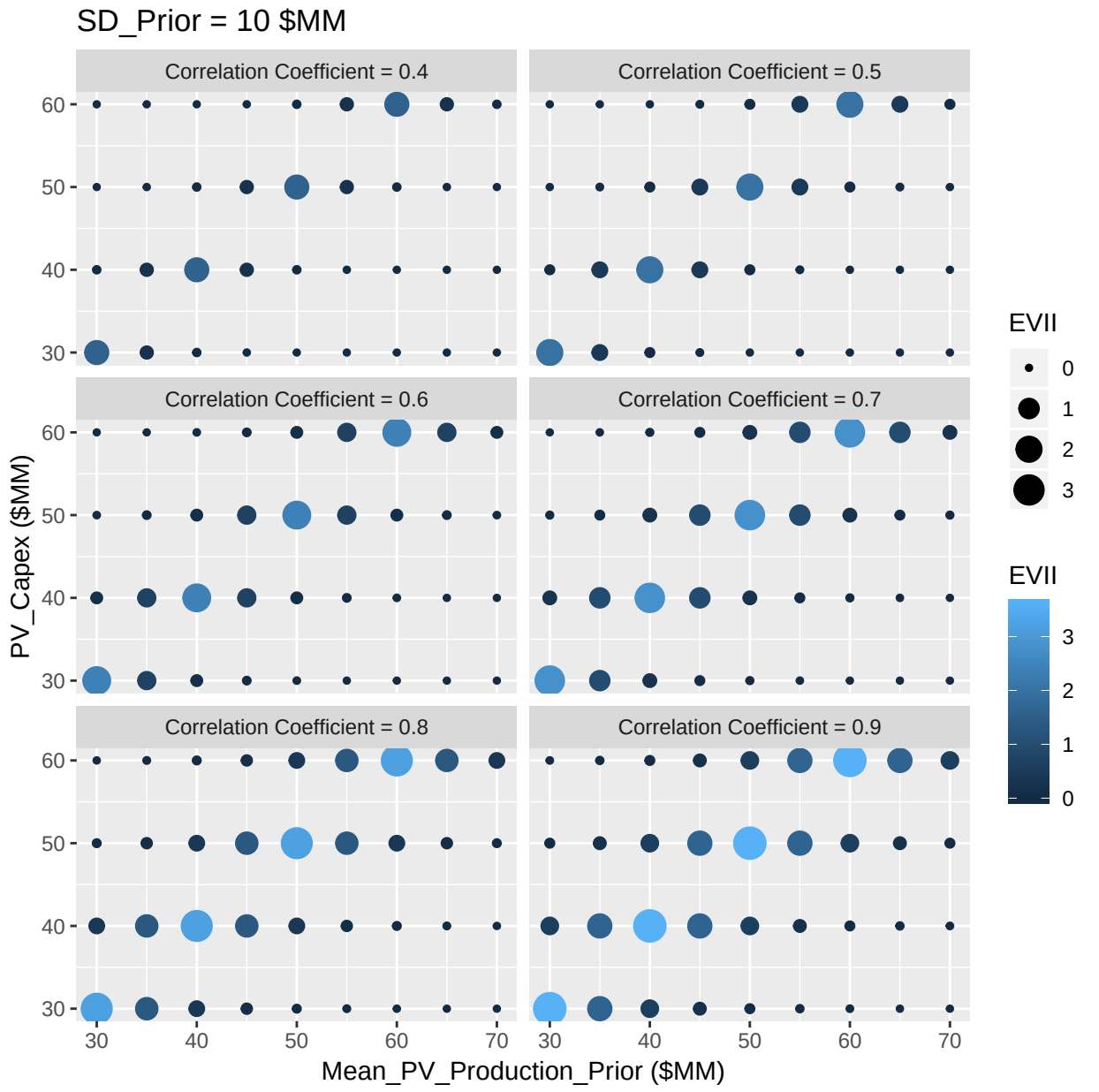

Figure 3.14: Sensitivity Analysis of VOI at SD=10

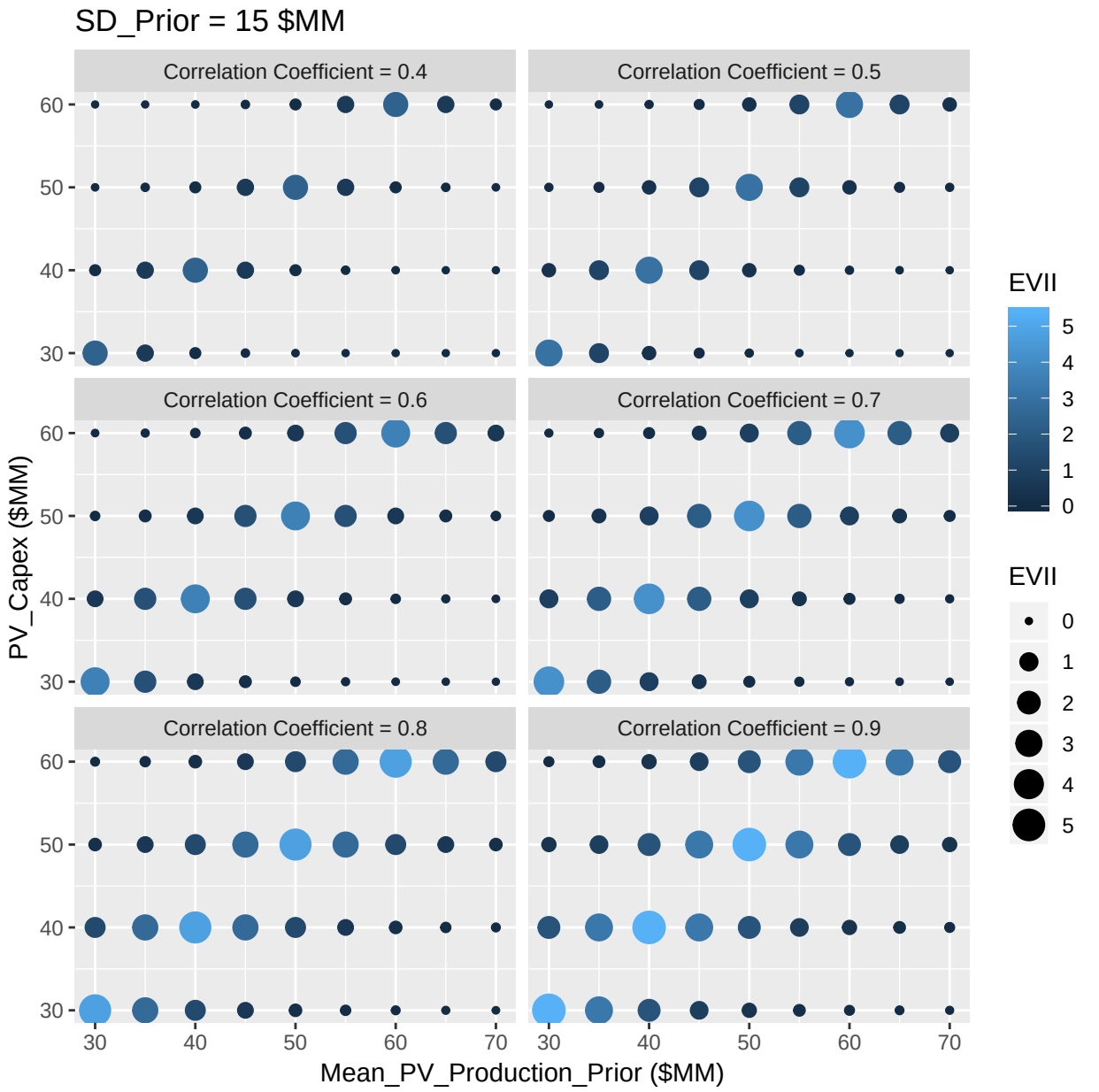

Figure 3.15: Sensitivity Analysis of VOI at SD=15

Chapter 4

Robust Field Development Optimization Using the Proxy Model

4.1 Introduction

So far, in the chapter 2 we developed a proxy model utilizing machine learning algorithm. In the chapter 3, we discussed the Value of Information in decision making for field development project. In this chapter the optimization algorithm will be utilized to optimize the field development scenario. This chapter presents a robust ,multidimensional, simultaneous optimization of well locations and controls (injection rate) using a Genetic Algorithm (GA) type of Evolutionary Algorithm (EA). Here, we demonstrate the robust optimization (RO) to the 5-spot pattern. This optimization algorithm is developed to fully take advantage of the computational efficiency and speed of the proxy model. Within a Robust Optimization procedure, the geological uncertainties are included by set of realizations generated using the geostatistical methods. Then, the objective of RO is to find a control vector (here both injection wells coordinates and water injection scheme) that optimize the expected value (EV) of the objective function (NPV) over the all geological realizations. This workflow of is named RO since optimum control vector is robust to geological uncertainty. (Hong et al., 2017, van Essen et al. (2009))

4.2 Brief Summary of the Proxy-Model

We begin with a summary of the approach presented in Chapter 3. As with other machine learning algorithm, the procedure starts with the collection and manipulation of feature and response data. This is achieved by running reservoir simulations using randomly sampled well locations within the domain of interest. It must be noted that special care has been taken to not to avoid inefficient well placement of injectors leading to weak sweep efficiency. The injectors are not only have distance from the producers as well as are in distant place from each other.

For each training observation, well-to-well connectivities and Pore Volume (PV) of flight are computed between every pair of wells in the reservoir using FMM. Together, these parameters make up the predictor variables that constitute the feature labels used for training. Each each training, characterized by a unique well configuration and combination of predictor parameters, is evaluated in a reservoir simulator to obtain a true output. A well known reservoir-wide objective function, net present value (NPV), can then be computed from the simulator output, to comprise the corresponding response observation. Then, the feature and response data are fed into the XGBoost algorithm to create a prediction model. Tuning parameters for the XGBoost algorithm (number of trees, shrinkage factor, bag fraction, and tree depth) are optimized by cross-validation to minimize the mean square error and avoid over-fitting.

4.3 Robust Optimization of Well Placement and Water Injection Scheme

The manner in which the joint problem has been handled in recent studies has varied. (van Essen et al., 2009) considered the geological uncertainty in optimization water-flooding using a gradient-based optimization which the gradients are obtained with an adjoint formulation.(Chen et al., 2009) proposed a ensemble-based optimization method(EnOpt) which is adjoint-free and gradient is approximated by the co-variance between the control variables and objective function. (Forouzanfar and Reynolds, 2014) conducted a joint optimization in which both the location and injection schemes are handled using gradient projection. (Bellout et al., 2012) proposed a hybrid approach where well placement was solved using derivative-free methods based on pattern search while injection rate is solved by adjoint-based method. (Hong et al., 2017) proposed a Ensemble-based optimization (EnOpt) where the capacitance-resistance model (CRM) was used as a proxy for optimization of injection scheme. (Nwachukwu et al., 2018b) proposed a scheme similar Mesh Adaptive Direct Search (MADS)for joint optimization where Extreme Gradient Boosting method (XGBoost) was build for making forecast for given any set of observations.

In this chapter, we propose a Genetic Algorithm (GA) (Holland, 1975, Goldberg (1989)) as a stochastic search algorithm which are are able to solve optimization of problems of the and development of an optimization strategy that efficiently combines the proxy and a reservoir simulator to solve the joint well locations and controls problem in a multidimensional manner.following type:

$$\Theta^* \equiv argmax f(\theta) = \theta^* \in \Theta : f(\theta^*) > f(\theta), \vee \theta \in \Theta \tag{4.1}$$

Where $\Theta \subseteq R$ defines the search spaces, $\theta = (\theta 1, \theta 2, .., \theta p)$ is domain of parameters where each θi varies between lower and upper bound. The joint Optimization in this work can be classified as the type of equation above where the S is the search space for (NPV) while the domain of parameters of the problem are defined as well locations and water injection scheme.(Scrucca et al., 2013)

GA utilize evolutionary strategies inspired by the basic principles of biological adaption. At each stage of the evolution, a population is composed of a number of individuals, also called chromosomes or strings . each chromosomes is made of units (genes, features, characters) which control the inheritance of one or several characters. Genes of specific characters are placed along the chromosome, and the corresponding string positions are called loci. Each genotype would represent a potential solution to a problem. The decision variables, or phenotypes, in a GA are obtained by applying some mapping from the chromosome representation into the decision variable space, which represent potential solutions to an optimization problem. A suitable decoding function may be required for mapping chromosomes onto phenotypes.

The genetic algorithm process could be summarized as follow:

1. type of variables, fitness function, GA parameters and convergence criteria must be defined to initiate the GA process,
2. Initial random population of size n is generated, so for step k = 0 we may write $\theta_1^0, \theta_2^0, ..., \theta_n^0$.
3. convergence criteria are checked and if it is meet, GA stops.
4. The fitness of each member of the population at any step k, $f(\theta_i^k)$ is computed and probabilities p_i^k are assigned to each individual in the population, proportional to their fitness.
5. The reproducing population is formed (selection) by drawing with replacement a sample where each individual has probability of surviving equal to p_i^k.
6. A new population$\theta_1^{k+1}, \theta_2^{k+2}, ..., \theta_n^{k+1}$ is formed from the reproducing population using crossover and mutation operators.
7. Then, set $k = k+1$ and the algorithm returns to the fitness evaluation step. When convergence criteria are met the evolution stops, and the algorithm deliver $\theta^* \equiv argmax f(\theta_i^k)$ arg max as the optimum. The flow chart of this algorithm is shown in the Figure 4.1.

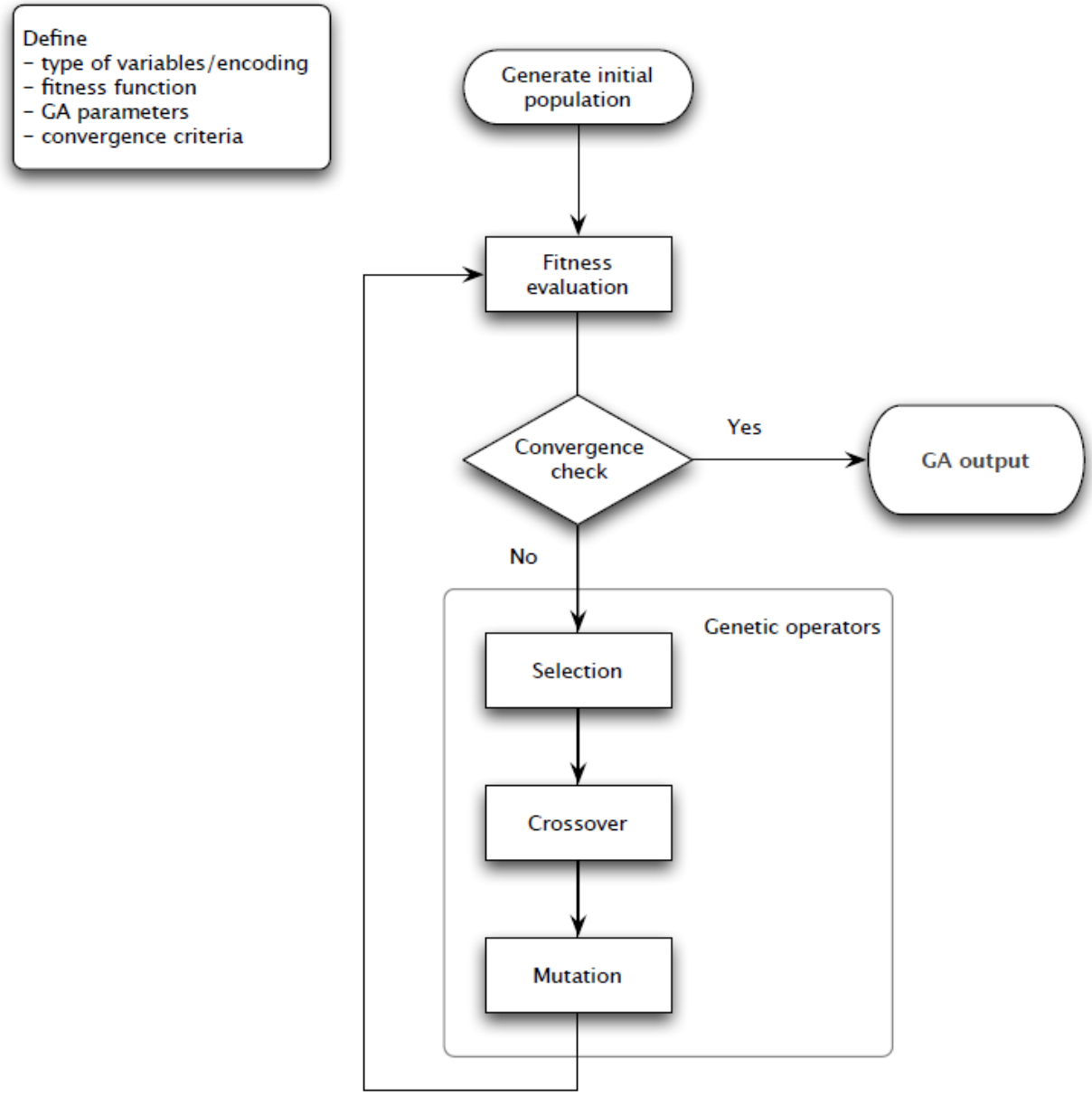

Figure 4.1: Flow Chart for Genetic Algorithem Process

Table 4.1: Parameters of Semi variogram to Perform SGS

Parameter	Value
Nugget Effect	Sill/2, md^2
Type	Spherical
Range	20 (grid cell)
Anistropy ratio	1
Azimuth	0-degree (North)

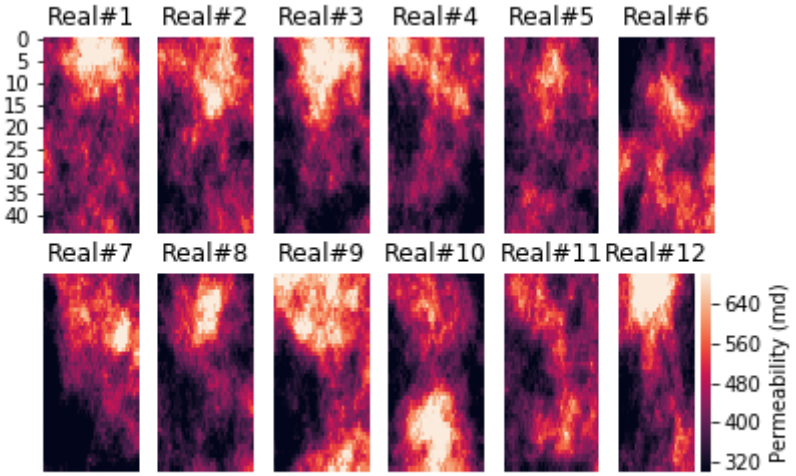

Figure 4.2: Twelve Realizations (out of 50) of Permeability in the Geological Model

4.4 Optimization Process

The proposed methods presented in the preceding sections were applied to a synthetic field under five spot pattern with 4 production wells and 1 injection well. Geological model were built on a 45 × 45 two-dimensional grid measuring 450 m on each side and 10 m in thickness. In order to take the geologic uncertainty into consideration, geological realizations were generated using Sequential Gaussian Simulation (SGS) (Pyrcz and Deutsch, 2014) with the semivariogram parameters given in Table 4.1.

The geological model has the petrophysical uncertainty (in this case permeability) and 50 realizations were considered to capture the uncertainty. 10 realization (out of 50) of permeability of the reservoir has been shown in the Figure 4.2.

Considering the Robust Optimization defined in the previous sections, the problem can be stated as the follows: *The Optimization algorithem search solution(s) for injection well coordinates and water injection scheme to maximize EV(NPV) ove all realizations.*

The GA optimization was conducted using the parallel Computing in R programming language with GA package. (Scrucca et al., 2013) The initial setup of the algorithm (Optimization parameters) has been shown in the Table 4.2.

As reminder, the NPV value is calculated as the following:

$$NPV = \sum_{k=1}^{n_T} \frac{[q_o^k P_o - q_w^k P_w - I^k P_{wi}] \triangle t_k}{(1+b)^{t_k/D}} \tag{4.2}$$

Table 4.2: Initial Setup of GA for Optimization

GA Parameters	Value
Population Size	25.0
robability of crossover between pairs of chromosomes	0.8
Probability of mutation in a parent chromosome	0.1
Maximum Iteration	20.0
Number of consecutive generations without any improvement	5.0

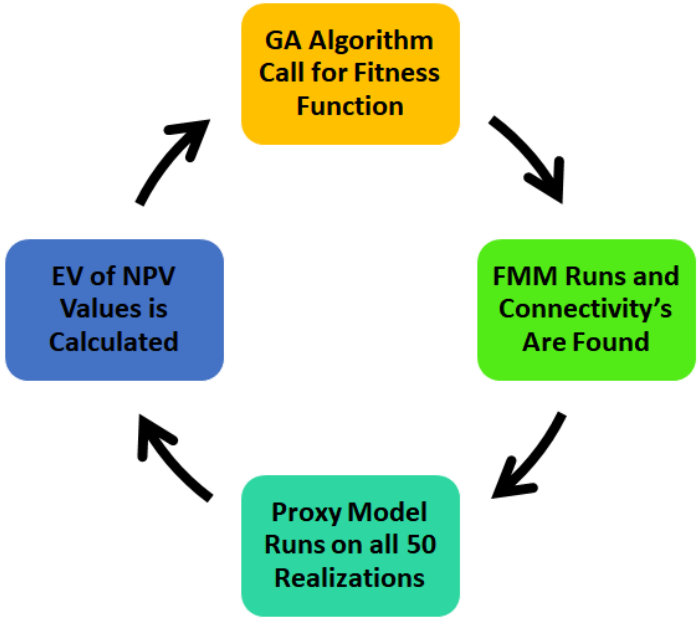

Figure 4.3: Flow Chart of the Optimization Alghorithem

In each iteration of GA algorithm, the fitness function is the mean NPV of all 50 realizations, defined as the below:

$$EV(NPV(u)) = \sum_{i=1}^{n_r=50} \frac{NPV_i(u)}{N} \tag{4.3}$$

Where u in this case is the injection well coordinates and injection rate scenarios(defined in Chapter 3).

The EV(NPV(u)) is found using ML proxy developed in the Chapter2, in each iteration. The flow chart of the optimization process in this chapter can be depicted as the Figure 4.3.

4.5 Results of Optimization

After running the 500 (25*20) iterations (which it means 25,000 proxy model iteration because of the 50 realizations), the result of the improvement in the best solution over the 20 generations could be found in the figure (4.4)

The best solution after 20 Iteration was found 56.7 $MM. This value is the mean NPV of the all realization at the U(Control vector) with the following parameters of the optimization in the Table 4.3.(Scrucca et al., 2013)

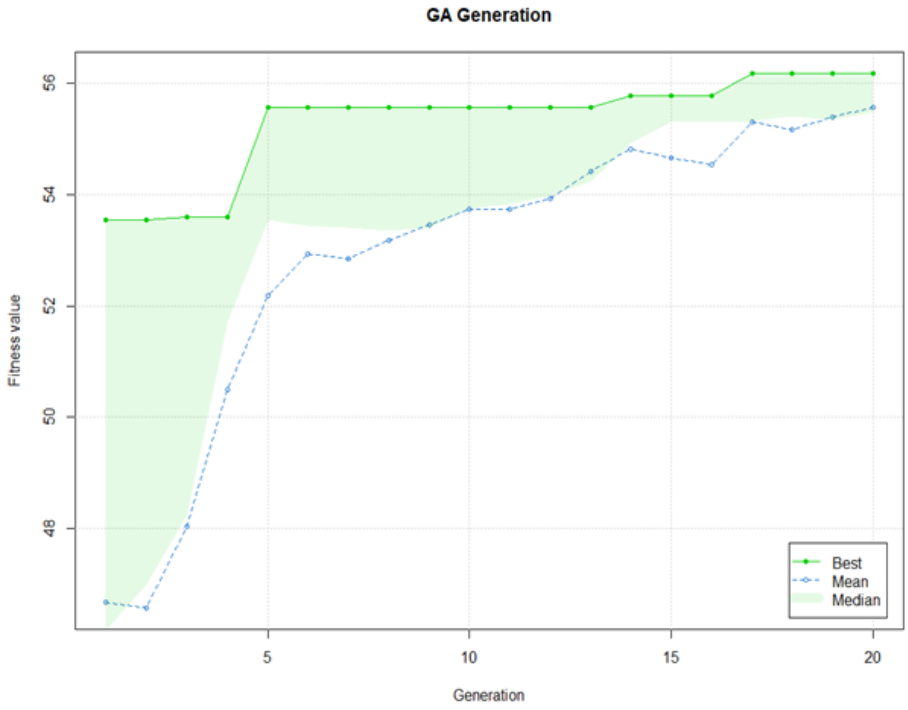

Figure 4.4: Evolution of GA Algorithem Solution over Generations

Now, the other area that was studied in this work is at the optimum control vector, how much the the distribution of NPV found from the Proxy Model differs from the the results of the Numerical Reservoir Simulator at that optimum location. This result will provide some insight about what is the deviation of the result of optimum location in the proxy model compared to complex, physics based model. The Figure 4.5 shows the distribution of the NPV over 50 realizations. The blue and red vertical lines are representative of the mean values.

Finally, we analyze the Value of Complexity in this case. The Value of Complexity is defined as:

- Value Of Complexity = Value Created from Rich Model - Value Created from Physics-Reduced Model*

This Difference, that in this study is shown with the difference in the NPV values of the Machine Learning Model and Eclipse reservoir Simulator has shown in the Figure 4.6

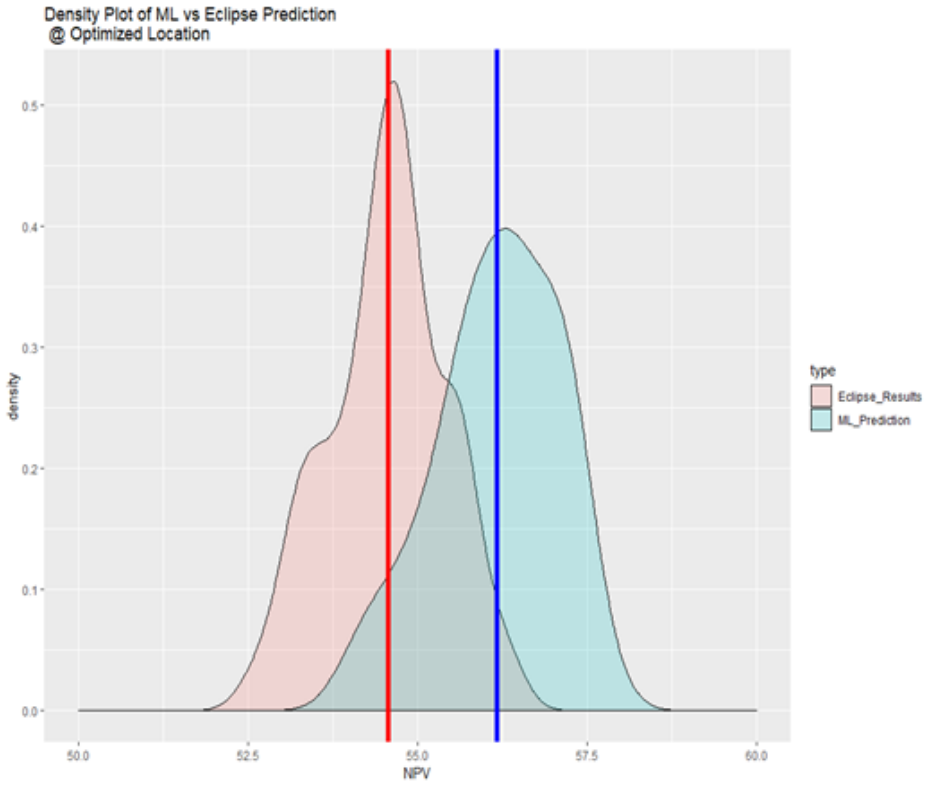

Figure 4.5: Comparison of the NPV Found (at Optimum Locations) in Machine Learning Vs. Numerical Simulator

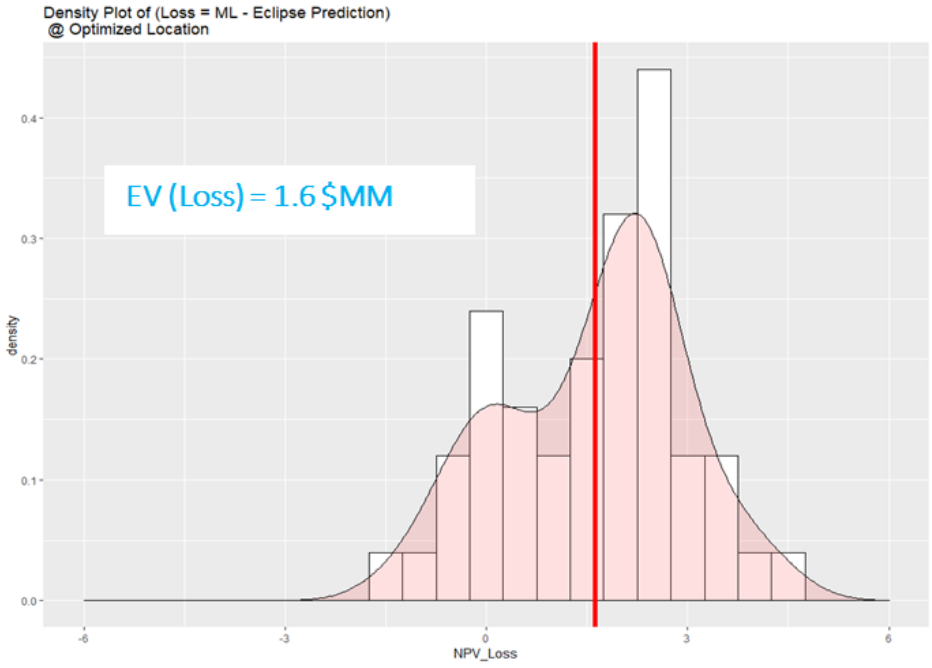

Figure 4.6: Value of Complexity (Deviation of Proxy Model from Complex One)

Table 4.3: Solution of Optimization Algoritem

Componenet of u Vector	Value
X and Y Coordinate of Injector Number 1	3,19
X and Y Coordinate of Injector Number 2	40,9
X and Y Coordinate of Injector Number 3	9,41
X and Y Coordinate of Injector Number 4	34,41
Starting Rate of Injector Number 1	250
Starting Rate of Injector Number 2	250
Starting Rate of Injector Number 3	100
Starting Rate of Injector Number 4	200
Gamma Rate of Injector Number 1	0.001
Gamma Rate of Injector Number 2	0
Gamma Rate of Injector Number 3	-0.005
Gamma Rate of Injectior Number 4	0.005

Chapter 5

Final Remarks on ML Application and Conclusions

5.1 Final Remarks on ML Application

This work was attempted to bring new insight to the oil and gas companies in the era of big data and data analytics. Here in this work, we call for "Decision-Driven" approach rather than "Data-Driven" approach. In the case f building the machine learning model, the main difference between these two approaches is in the Decision Driven approach, we only collect, analyze, find the pattern in the data that has following four characteristics(Bratvold and Begg, 2010);

- We must be able to view the result of the model (here the model is machine learning algorithm)
- The model found from the data must have the potential to change our prior belief in the decision context
- The model must have the ability to change the decision on the hand
- The value added to the decision through model must exceed its cost

We argue that the Data-Driven approach where the data in it's oneself considered as the value and it is the main focus of the decision may ultimately lead to sub-optimal allocation of time and resource of the corporation.

Then, another area that is needed to be discussed is answering this question:

- Which type of "Decision" has the most qualification to be addressed and answered using a machine learning method?

The author believes that the classification in the Figure 5.1 could be very helpful to answer this question. According to that, for any Decision that has three following characteristics, machine learning could provide the best insight for making a better decision in the 'timely' manner:

- The "Decision" repeats often (it means the same decision is faced repeatably)
- The "Decision" needs the data to be made
- The "Decision" is not consequential (having a very important impact)

The author concludes that while the "Decision" for well placement maybe not the best candidate to be addressed using the machine learning models (Since it does not repeats much and most of the time it is an important decision for corporations); the following areas can be gain the most advantages from ML models (considering the aforementioned three characteristics):

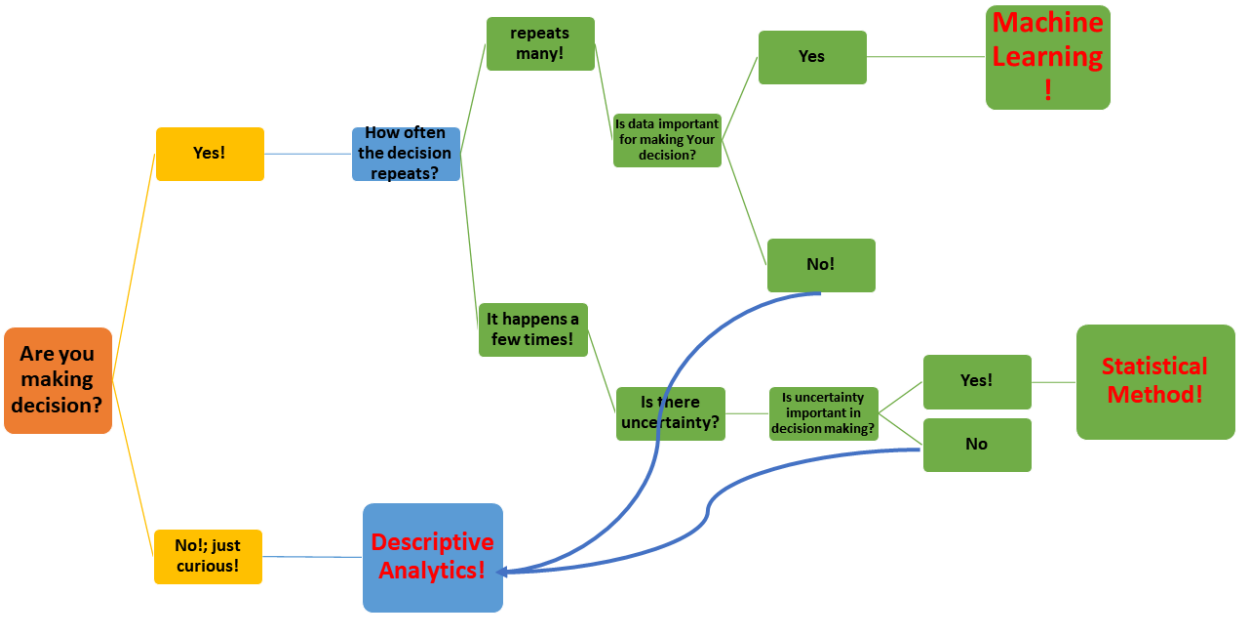

Figure 5.1: From Cassie Kozyrkov (Chief Decision Scientist at Google), Keynote speaker in DAAG 2019 conference

- Geo-steering the well in order to achieve a best possible zone of the reservoir
- Classification and characterization of the petrophysical characteristics of the reservoir rock

These two areas are among the potential application to get benefit from ML models in order to "Automate the Decision Making".

5.2 Conclusions

Considering this work, the key messages and conclusion of this work could be summarized as follows:

1. The subsurface modeling in the Reservoir Engineering deals with sparse data sets with tens of the features. One approach for feature selection is using fully data-driven techniques such as Principal Component Analysis (PCA) or Backward Feature Elimination (BFE). We found here that the Physical - Based features like the Fast Marching Method (FMM) carry important geological information and could be used as an alternative representation of geology while retaining the physical parameters involved in the flow of the oil and gas.

2. The machine learning based model could provide the very efficient and fast proxy for complex and slow full physical based simulators which in application like optimization, could be very helpful. However, two areas need further attention:

- When coming from a complex model to the more simplification of the 'truth model', the part of accuracy will be sacrificed. The research in ML application must have analysis about the trade-off between speed and accuracy.

- In the ML model developed in this work, it was assumed the training data set here is "Historical Data" that was fed to the ML to make a prediction about the future field. In fact, that training data set is the 'sample' while we are considering to make a prediction about the population. Here, the challenging statistical question comes that how much training data set is representative of the population of the data set?

3. In the framework of decision analysis, in this study, the value of machine learning in a particular decision context was analyzed. This analysis (considering the final result of the data analytics as the information for decision) will be helpful for the companies to prioritize the value of future data analytic project based on their expected value. The framework explained in chapter 3 can fully be used for any other data gathering process.

4. There is an essential need in the domain of the Reservoir Engineering to build a fast model for optimization where the current commercial numerical simulators take even days to have several thousand iterations. The Proxy model and the robust optimization (RO) framework proposed in this work has a clear application: thousands of forwarding reservoir evaluations can be realized in 100X faster to help decision maker make a decision in a "timely" manner. This offers a practical new alternative to full reservoir simulations which requires massive computational resources and time to perform the same numbers of iterations.

Chapter 6

Appendix

6.1 Sequential Guassian Simulation (R Code)

```
#install.packages(c('gstat','sp','sp','plyr','fields'))

library(gstat)                                  # geostatistical methods by Edzer Pebesma
library(sp)                                     # spatial points addition to regular
                                                # data frames
library(plyr)                                   # manipulating data by Hadley Wickham
library(fields)                                 # required for the image plots


nx = 45                                        # number of cells in the x direction
ny = 45                                        # number of cells in the y direction
xsize = 10.0                                    # extent of cells in x direction
ysize = 10.0                                    # extent of cells in y direction



xmin = 0
ymin = 0
xmax = xmin + nx * xsize
ymax = ymin + ny * ysize
x<-seq(xmin,xmax,by=xsize)                      # used for axes on image plots
y<-seq(ymin,ymax,by=ysize)


colmap = topo.colors(100)                       # define the color map and
                                                # descritation

nscore <- function(x) {                         # by Ashton Shortridge, 2008
  # Takes a vector of values x and calculates their normal scores. Returns
  # a list with the scores and an ordered table of original values and
  # scores, which is useful as a back-transform table. See backtr().
  nscore <- qqnorm(x, plot.it = FALSE)$x  # normal score
  trn.table <- data.frame(x=sort(x),nscore=sort(nscore))
  return (list(nscore=nscore, trn.table=trn.table))
```

```
}

addcoord <- function(nx,xmin,xsize,ny,ymin,ysize) { # Michael Pyrcz, March, 2018
  # makes a 2D dataframe with coordinates based on GSLIB specification
  coords = matrix(nrow = nx*ny,ncol=2)
  ixy = 1
  for(iy in 1:nx) {
    for(ix in 1:ny) {
      coords[ixy,1] = xmin + (ix-1)*xsize
      coords[ixy,2] = ymin + (iy-1)*ysize
      ixy = ixy + 1
    }
  }
  coords.df = data.frame(coords)
  colnames(coords.df) <- c("X","Y")
  coordinates(coords.df) =~X+Y
  return (coords.df)
}
sim2darray <- function(spdataframe,nx,ny,ireal) { # Michael Pyrcz, March, 2018
  # makes a 2D array from realizations spatial point dataframe
  model = matrix(nrow = nx,ncol = ny)
  ixy = 1
  for(iy in 1:ny) {
    for(ix in 1:nx) {
      model[ix,iy] = spdataframe@data[ixy,ireal]
      ixy = ixy + 1
    }
  }
  return (model)
}


sim2vector <- function(spdataframe,nx,ny,ireal) { # Michael Pyrcz, March, 2018
  # makes a 1D vector from spatial point dataframe
  model = rep(0,nx*ny)
  ixy = 1
  for(iy in 1:ny) {
    for(ix in 1:nx) {
      model[ixy] = spdataframe@data[ixy,ireal]
      ixy = ixy + 1
    }
  }
  return (model)
}


mydata1=read.csv('Welllocation5SGS_inj.csv')


j=1
cumdata <- matrix(0,nrow = 1500,ncol = 4)
cumdata <- as.data.frame(cumdata)
for (i in 1:1){
```

```
  sample <- matrix(data.matrix(mydata1[i,2:9]),nrow = 4,ncol = 2)
  locdata <- data.frame(rbind(matrix(c(23,23),nrow = 1,ncol = 2),sample))
  locgrid <- matrix(0,nrow=5,ncol = 2)
  for (k in 1:5) {
    locgrid[k,1] <- (locdata[k,1]-1)*xsize + 5
    locgrid[k,2] <- (locdata[k,2]-1)*ysize + 5
  }
  m <- 500
  s <- 100
  location <- log(m^2 / sqrt(s^2 + m^2))
  shape <- sqrt(log(1 + (s^2 / m^2)))
  Perm <- matrix(rlnorm(5,location,shape),nrow = 5,ncol = 1)
  logperm <- log10(Perm)
  logperm <- matrix(logperm,nrow = 5,ncol = 1)
  mydata2 <- data.frame(cbind(locgrid,Perm,logperm))
  z <- i*5
  cumdata[j:z,1:4] <- mydata2
  colnames(mydata2) <- c('X','Y','Perm','logperm')
  #head(mydata2)
  j=j+5


coordinates(mydata2) = ~X+Y

npor.trn = nscore(mydata2$logperm)              # normal scores transform
mydata2[["NPermeability"]]<-npor.trn$nscore     # append the normal scores transform

cuts = c(2.4,2.45,2.5,2.65,2.7,2.8,2.9)
cuts.var = c(0.05,.1,.15,.20,.25,.3,.35,.4,.45,.5,.55,.6,.65,.7,.75,.8,.85,.9,.95)

spplot(mydata2, "logperm", do.log = TRUE,      # location map of porosity data
       key.space=list(x=1.05,y=0.97,corner=c(0,1)),cuts = cuts,
       scales=list(draw=T),xlab = "X (m)", ylab = "Y (m)",main ="Permeability (Log(K)),
       in md")

  coords <- addcoord(nx,xmin,xsize,ny,ymin,ysize) # make a dataframe with
                                                  #all the estimation locations
#summary(coords)                                   # check the coordinates

sill = var(mydata2$logperm)                      # calculate the variance of the property
                                                  #of interest as the sill
min = min(mydata2$logperm)                       # calculate the property min and max
                                                  #for plotting
max = max(mydata2$logperm)
zlim = c(min,max)            # define the property min and max in a 2x1 vextor

vm.nug1 <- vgm(psill =0.5*sill, "Sph", 200, anis = c(000, 1.0),nugget=0.5*sill)
condsim.nug1 = krige(logperm~1, mydata2, coords, model = vm.nug1, nmax = 100, nsim = 10)



par(mfrow=c(2,2))
real1 <- sim2darray(condsim.nug1,nx,ny,1)     # extract realization #1 to a
```

```
                                                #  2D array and plot
image.plot(10^real1,x=x,y=y,xlab="X(m)",ylab="Y(m)",
zlim = c(min(10^real1),max(10^real1)),
           col=colmap,legend.shrink = 0.6);
mtext(line=1, side=3, "Realization #1", outer=F);box(which="plot")

real2 <- sim2darray(condsim.nug1,nx,ny,2)     # extract realization #2 to a
                                                #2D array and plot
image.plot(10^real2,x=x,y=y,xlab="X(m)",ylab="Y(m)",
zlim =c(min(10^real2),max(10^real2)) ,
           col=colmap,legend.shrink = 0.6);
mtext(line=1, side=3, "Realization #2", outer=F);box(which="plot")

real3 <- sim2darray(condsim.nug1,nx,ny,3)      # extract realization #3 to a
                                                # 2D array and plot
image.plot(10^real3,x=x,y=y,xlab="X(m)",ylab="Y(m)",zlim = c(min(10^real3),max(10^real3)),
           col=colmap,legend.shrink = 0.6);
mtext(line=1, side=3, "Realization #3", outer=F);box(which="plot")

real4 <- sim2darray(condsim.nug1,nx,ny,4)      # extract realization #4 to a
                                                #2D array and plot
image.plot(10^real4,x=x,y=y,xlab="X(m)",ylab="Y(m)",
zlim = c(min(10^real4),max(10^real4)),
           col=colmap,legend.shrink = 0.6);
mtext(line=1, side=3, "Realization #4", outer=F);box(which="plot")

}
```

6.2 Calculation of Connectivities (FMM) for 5-Spot Pattern (Python Code)

```
!pip install pandas
!pip install numpy
!pip install pandas
!pip install seaborn
!pip install scikit-fmm

import os
import pandas as pd
import numpy as np
import skfmm

loc_well = pd.DataFrame(np.array([[30,80,11,19,18,12,87,88]]),
                  columns=['X1', 'X2', 'X3','X4','Y1','Y2','Y3','Y4'])

input = pd.DataFrame(np.array([(np.arange(1,18))]),
columns=['ConI1I2', 'ConI1I3', 'ConI1I4',
'ConI2I1','ConI2I3','ConI2I4','ConI3I1',
'ConI3I2','ConI3I4''ConI4I1','ConI4I2',
'ConI4I3','ConP1I1','ConP1I2','ConP1I3',
```

```
'ConP1I4','PV_flight'])


import timeit
start = timeit.default_timer()
i=1
j=1
ii=str(i)
jj=str(j)
ss='C:\\Users\\243886\\OneDrive - Universitetet i Stavanger\\ML-5spot-SGS\\R&\\En$'
pathnew=ss.replace("&", ii)
pathnews=pathnew.replace("$", jj)
os.chdir(pathnews)
os.getcwd()
data=pd.read_csv('SPEED.csv')
speed=data
speed = speed.iloc[:,0:100]
phi=np.ones((100,100))
phi[loc_well.loc[i-1,'X1']-1,loc_well.loc[i-1,'Y1']-1]=0
t_va=skfmm.travel_time(phi,speed,dx=10)
TimeFMM_hr=t_va**2/4/3600
input.loc[0,'ConI1I2']=TimeFMM_hr[loc_well.loc[i-1,'X2']-1,loc_well.loc[i-1,'Y2']-1]
input.loc[0,'ConI1I3']=TimeFMM_hr[loc_well.loc[i-1,'X3']-1,loc_well.loc[i-1,'Y3']-1]
input.loc[0,'ConI1I4']=TimeFMM_hr[loc_well.loc[i-1,'X4']-1,loc_well.loc[i-1,'Y4']-1]

###############################################################################################

phi=np.ones((100,100))
phi[loc_well.loc[i-1,'X2']-1,loc_well.loc[i-1,'Y2']-1]=0
t_va=skfmm.travel_time(phi,speed,dx=10)
TimeFMM_hr=t_va**2/4/3600
input.loc[0,'ConI2I1']=TimeFMM_hr[loc_well.loc[i-1,'X1']-1,loc_well.loc[i-1,'Y1']-1]
input.loc[0,'ConI2I3']=TimeFMM_hr[loc_well.loc[i-1,'X3']-1,loc_well.loc[i-1,'Y3']-1]
input.loc[0,'ConI2I4']=TimeFMM_hr[loc_well.loc[i-1,'X4']-1,loc_well.loc[i-1,'Y4']-1]


###############################################################################################


phi=np.ones((100,100))
phi[loc_well.loc[i-1,'X3']-1,loc_well.loc[i-1,'Y3']-1]=0
t_va=skfmm.travel_time(phi,speed,dx=10)
TimeFMM_hr=t_va**2/4/3600
input.loc[0,'ConI3I1']=TimeFMM_hr[loc_well.loc[i-1,'X1']-1,loc_well.loc[i-1,'Y1']-1]
input.loc[0,'ConI3I2']=TimeFMM_hr[loc_well.loc[i-1,'X2']-1,loc_well.loc[i-1,'Y2']-1]
input.loc[0,'ConI3I4']=TimeFMM_hr[loc_well.loc[i-1,'X4']-1,loc_well.loc[i-1,'Y4']-1]


###############################################################################################

phi=np.ones((100,100))
phi[loc_well.loc[i-1,'X4']-1,loc_well.loc[i-1,'Y4']-1]=0
t_va=skfmm.travel_time(phi,speed,dx=10)
```

```
TimeFMM_hr=t_va**2/4/3600
input.loc[0,'ConI4I1']=TimeFMM_hr[loc_well.loc[i-1,'X1']-1,loc_well.loc[i-1,'Y1']-1]
input.loc[0,'ConI4I2']=TimeFMM_hr[loc_well.loc[i-1,'X2']-1,loc_well.loc[i-1,'Y2']-1]
input.loc[0,'ConI4I3']=TimeFMM_hr[loc_well.loc[i-1,'X3']-1,loc_well.loc[i-1,'Y3']-1]


###############################################################################################
phi=np.ones((100,100))
phi[49, 49]=0
t_va=skfmm.travel_time(phi,speed,dx=10)
TimeFMM_hr=t_va**2/4/3600
input.loc[0,'ConP1I1']=TimeFMM_hr[loc_well.loc[i-1,'X1']-1,loc_well.loc[i-1,'Y1']-1]
input.loc[0,'ConP1I2']=TimeFMM_hr[loc_well.loc[i-1,'X2']-1,loc_well.loc[i-1,'Y2']-1]
input.loc[0,'ConP1I3']=TimeFMM_hr[loc_well.loc[i-1,'X3']-1,loc_well.loc[i-1,'Y3']-1]
input.loc[0,'ConP1I4']=TimeFMM_hr[loc_well.loc[i-1,'X4']-1,loc_well.loc[i-1,'Y4']-1]

###############################################################################################


phi=np.ones((100,100))
phi[loc_well.loc[i-1,'X1']-1,loc_well.loc[i-1,'Y1']-1]=0
phi[loc_well.loc[i-1,'X2']-1,loc_well.loc[i-1,'Y2']-1]=0
phi[loc_well.loc[i-1,'X3']-1,loc_well.loc[i-1,'Y3']-1]=0
phi[loc_well.loc[i-1,'X4']-1,loc_well.loc[i-1,'Y4']-1]=0
t_va=skfmm.travel_time(phi,speed,dx=10)
TimeFMM_hr=t_va**2/4/3600
PV=(sum(sum(TimeFMM_hr<TimeFMM_hr[49, 49])))/10000
input.loc[0,'PV_flight']=PV
stop = timeit.default_timer()
print('Time: ', stop - start)

input
```

6.3 XGBOOST Machine Learning Model (R Code)

```
setwd("C:/Users/243886/OneDrive - Universitetet i Stavanger/ML-5spot-SGS/Rcode")
NPVpl <- data.frame(N)
NPVpl <- NPVpl[, c(1)]
input1 <- data.frame(input)
geoin <- read.csv('INPUTPCA489.csv',nrows = F)
Final_Input <- cbind(input1,NPVpl)
install.packages(c("e1071", "caret", "doSNOW", "ipred", "xgboost"))
install.packages(c('lattice','ggplot2'))
library(caret)
library(doSNOW)
INPUTMEAN <- data.frame(INPUTMEAN)
train <- INPUT
#=================================================================
# Data Wrangling
#=================================================================
# Set up factors.
```

```
# Subset data to features we wish to keep/use.

features <- c('MC1','MC2','MC3','MC4','MPV','NPVpl')
c('En1C1','En2C1','En3C1','En4C1','En5C1','En6C1','En7C1','En8C1','En9C1',
              'En10C1','En1C2','En2C2','En3C2','En4C2','En5C2','En6C2','En7C2','En8C2'
              ,'En9C2','En10C2','En1C3','En2C3','En3C3','En4C3','En5C3','En6C3','En7C3',
              'En8C3','En9C3','En10C3','En1C4','En2C4','En3C4','En4C4','En5C4','En6C4',
              'En7C4','En8C4','En9C4','En10C4','En1PV','En2PV','En3PV','En4PV','En5PV',
              'En6PV','En7PV','En8PV','En9PV','En10PV','NPVpl')
train <- train[, features]

#=====================================================================
# Split Data
#=====================================================================

names(train)[26]<-"NPVpl"
# Use caret to create a 70/30% split of the training data,
# keeping the proportions of the Survived class label the
# same across splits.
set.seed(54321)
indexes <- createDataPartition(train$NPVpl,
                               times = 1,
                               p = 0.7,
                               list = FALSE)
profs.train <- train[indexes,]
profs.test <- train[-indexes,]


# Examine the proportions of the Survived class lable across
# the datasets.
prop.table(table(train$NPVpl))
prop.table(table(proonebyone.train$NPVpl))
prop.table(table(proonebyone.test$NPVpl))

#=====================================================================
# Train Model
#=====================================================================
# nrounds max_depth  eta gamma colsample_bytree min_child_weight subsample
#4    4000         6 0.01     0              0.4                2         1
#   nrounds max_depth   eta gamma colsample_bytree min_child_weight subsample
#10    4000         6 0.025     0              0.4             2.25         1
# Set up caret to perform 10-fold cross validation repeated 3
# times and to use a grid search for optimal model hyperparamter
# values.
train.control <- trainControl(method = "repeatedcv",
                              number = 10,
                              repeats = 3,
                              search = "grid")


# Leverage a grid search of hyperparameters for xgboost. See
# the following presentation for more information:
# https://www.slideshare.net/odsc/owen-zhangopen-sourcetoolsanddscompetitions1
```

```
tune.grid <- expand.grid(eta = c(0.0025),
                         nrounds = c(4000),
                         max_depth = 6,
                         min_child_weight = c(2.25),
                         colsample_bytree = c(0.4),
                         gamma = 0,
                         subsample = 1)
View(tune.grid)


# Use the doSNOW package to enable caret to train in parallel.
# While there are many package options in this space, doSNOW
# has the advantage of working on both Windows and Mac OS X.
#
# Create a socket cluster using 10 processes.
#
# NOTE - Tune this number based on the number of cores/threads
# available on your machine!!!
#
cl <- makeCluster(10, type = "SOCK")

# Register cluster so that caret will know to train in parallel.
registerDoSNOW(cl)

library(foreach)
install.packages('doParallel')
library(doParallel)
cl <- makeCluster(30)
registerDoParallel(cl)


# Train the xgboost model using 10-fold CV repeated 3 times
# and a hyperparameter grid search to train the optimal model.
library('xgboost')
caret.cv <- train(NPVpl ~ .,
                  data = profs.train,
                  method = "xgbTree",
                  tuneGrid = tune.grid,
                  trControl = train.control)
stopCluster(cl)
caret.cv$bestTune
# Examine caret's processing results
# Make predictions on the test set using a xgboost model
# trained on all 625 rows of the training set using the
# found optimal hyperparameter values.
preds <- predict(caret.cv, profs.test)
plot(preds,profs.test$NPVpl,col='red',type = 'p',pch=1 ,xlab =
       'NPV predicted by ML ($MM)',ylab = 'NPV of the real Test Data ($MM)',
     main = 'Test Data vs. ML Prediction')
abline(a=0,b=1,col=4,lwd=3)
mylabel = bquote(italic(R)^2 == .(format(r2, digits = 2)))
text(x = 40, y = 15, labels = mylabel)
caret::R2(preds,profs.test$NPVpl)
```

```
# Use caret's confusionMatrix() function to estimate the
# effectiveness of this model on unseen, new data
caret.cv$bestTune
```

6.4 Modyfing the Eclipse Data File + Running the Eclipse Simulator from R Shell (R Code)

```
setwd("C:/Users/243886/OneDrive - Universitetet i Stavanger/ML-5spot-INJ-SGS/Rcode")
library(readr)
X2D_4Inj_1Prod_R1 <- read_delim("2D_4Inj_1Prod_R.DATA",
                                "\t", escape_double = FALSE, na = "null",
                                trim_ws = TRUE)
run <- 'C:/ecl/macros/eclrun.exe eclipse "C:/Users/243886/OneDrive
- Universitetet i Stavanger/ROOPT_Ide/En#/Ensemble.DATA"'

Eclipse <- function(X1,Y1,X2,Y2,X3,Y3,X4,Y4,A1,A2,A3,A4,gam1,gam2,gam3,gam4,En){
  wd <- 'C:/Users/243886/OneDrive - Universitetet i Stavanger/ROOPT_Ide/En#'
  run1 <- gsub('#',En,run)
  newwd <- gsub('#',En,wd)
  setwd(newwd)
  X1 <- round(X1)
  X2 <- round(X2)
  X3 <- round(X3)
  X4 <- round(X4)
  Y1 <- round(Y1)
  Y2 <- round(Y2)
  Y3 <- round(Y3)
  Y4 <- round(Y4)
  slope <- c(-0.0005,-0.00025,0,0.00025,0.00050,0.00075,
             0.00100,0.00125,0.00150,0.00175,0.00200)
  Avalue <- c(100,150,200,250,300)
  A_1 <- Avalue[findInterval(A1,Avalue)]
  A_2 <- Avalue[findInterval(A2,Avalue)]
  A_3 <- Avalue[findInterval(A3,Avalue)]
  A_4 <- Avalue[findInterval(A4,Avalue)]

  Gam1 <- slope[findInterval(gam1,slope)]
  Gam2 <- slope[findInterval(gam2,slope)]
  Gam3 <- slope[findInterval(gam3,slope)]
  Gam4 <- slope[findInterval(gam4,slope)]

  data <- X2D_4Inj_1Prod_R1
  g1 <- data[131,1]
  g2 <- data[132,1]
  g3 <- data[133,1]
  g4 <- data[134,1]
  g5 <- data[138,1]
  g6 <- data[139,1]
  g7 <- data[140,1]
  g8 <- data[141,1]
```

```
  G1 <- gsub('101',X1, g1)
  G2 <- gsub('102',Y1, G1)
  G3 <- gsub('103',X2, g2)
  G4 <- gsub('104',Y2, G3)
  G5 <- gsub('105',X3, g3)
  G6 <- gsub('106',Y3, G5)
  G7 <- gsub('107',X4, g4)
  G8 <- gsub('108',Y4, G7)
  G9 <- gsub('109',X1, g5)
  G10 <- gsub('110',Y1, G9)
  G11 <- gsub('111',X2, g6)
  G12 <- gsub('112',Y2, G11)
  G13 <- gsub('113',X3, g7)
  G14 <- gsub('114',Y3, G13)
  G15 <- gsub('115',X4, g8)
  G16 <- gsub('116',Y4, G15)
  data[131,1] <- G2
  data[132,1] <- G4
  data[133,1] <- G6
  data[134,1] <- G8
  data[138,1] <- G10
  data[139,1] <- G12
  data[140,1] <- G14
  data[141,1] <- G16
  #'C:/Users/243886/OneDrive - Universitetet i Stavanger/ROOPT/Ecl'
  write.table(data, file ="Ensemble.DATA", sep = "\t",quote = F,
              row.names = F,col.names = F)
  int <- seq(from=0,to=1485,by=30)
  InjectionRatesex <- Injec_up1
  InjectionRatesex[seq(from=2,to=400,by=8),5] <- A_1*exp(-Gam1*int)
  InjectionRatesex[seq(from=3,to=400,by=8),5] <- A_2*exp(-Gam2*int)
  InjectionRatesex[seq(from=4,to=400,by=8),5] <- A_3*exp(-Gam3*int)
  InjectionRatesex[seq(from=5,to=400,by=8),5] <- A_4*exp(-Gam4*int)
  write.table(InjectionRatesex, 'InjectionRates.inc',quote = F,
              row.names = F,col.names = F, na = '')
  a1 <- A_1*exp(-Gam1*int)
  a2 <- A_2*exp(-Gam2*int)
  a3 <- A_3*exp(-Gam3*int)
  a4 <- A_4*exp(-Gam4*int)
  AA <- a1+a2+a3+a4
  shell(run1)
  return(aa(AA,newwd))

}
```

6.5 Net Present Value Calculation - Processing the Eclipse Output Data (R Code)

```
setwd("C:/Users/243886/OneDrive - Universitetet i Stavanger/ML-5spot-INJ-SGS
      /R1/En1")
```

```
ens <- "ENSEMBLE.RSM"
NPV <- read.delim(ens,  header=FALSE, comment.char="#")
NPVcalc <- NPV[, c(2,5,6)]
colnames(NPVcalc) <- c('Days', 'Oilrate', 'waterrate')
NPVcalc <- na.omit(NPVcalc)
NPVcalc <- NPVcalc[-seq(9,400,by = 2), ]
rownames(NPVcalc) <- 1:nrow(NPVcalc)
days <- as.numeric(levels(NPVcalc$Days))[as.integer(NPVcalc$Days)]
days <- na.omit(days)
l1 <- min(which(days==c(1), arr.ind = TRUE))
l2 <- min(which(days==c(1500), arr.ind = TRUE))
days <- days[c(l1:l2)]
oil <- as.numeric(levels(NPVcalc$Oilrate))[as.integer(NPVcalc$Oilrate)]
oil <- na.omit(oil)
oil <- oil[c(l1:l2)]
water <- as.numeric(levels(NPVcalc$waterrate))[as.integer(NPVcalc$waterrate)]
water <- na.omit(water)
water <- water[c(l1:l2)]
FWCT <- numeric(length = length(water))
FWCT <- water/(water+oil)
t <- seq(0,days[length(which(FWCT<0.85, arr.ind = TRUE))],by = 30)
t[1] <- 1
cashflow <- numeric(length(t)-1)
discashflow <- numeric(length(t)-1)
b <- 0.08
for (j in 1:(length(t)-1)) {
  m1 <- t[j]
  z1 <- which(days==c(m1), arr.ind = TRUE)
  m2 <- t[j+1]
  z2 <- which(days==c(m2), arr.ind = TRUE)
  cashflow[j] <- (mean(oil[z1],oil[z2])*6.29*75-mean
                  (water[z1],water[z2])*6.29*19-A[j]*5*6.29)*30
  discashflow[j] <- cashflow[j]/((1+b)^(t[j+1]/360))
}
sum(discashflow)
```

6.6 Alghorithem of VOI Calculation in HRDP Method (R Code)

```
fn_EVII <- function(x_mean,x_sd,z_mean,z_sd,rho,Cost) {
set.seed(1234)
N <- 1000          # Number of grid nodes
rho <- rho           # Correlation Coefficient
x_mean <- x_mean-Cost     # mean of prior
x_sd <- x_sd       # standard Deviation Prior
z_mean <- z_mean-Cost      # standard Deviation signal
z_sd <- z_sd        # standard Deviation signal
range <- quantile(rnorm(10^6,x_mean,x_sd),c(0.000001,0.99999))
xx <- seq(range[[1]],range[[2]],length.out = N+1)
ss <- seq(range[[1]],range[[2]],length.out = N+1)
X <- c(rep(0,N))
S <- c(rep(0,N))
```

```
for (i in 1:N) {
  X[i] <- (xx[i]+xx[i+1])/2
}

for (i in 1:N) {
  S[i] <- (ss[i]+ss[i+1])/2
}
# Prior Plot
z <- S
x <- X

f_x<- dnorm(x, x_mean,x_sd)
f_x_N <- f_x/sum(f_x)

# Signal Plot
f_z <- dnorm(z,z_mean,z_sd)
f_z_N <- f_z/sum(f_z)

# Liklihood Table
Lik <- matrix(0,nrow = N,ncol = N)          # Liklihood table, x in columns and z in rows

for (j in 1:N) {
  mean <- z_mean + (rho*z_sd*(x[j]-x_mean)/x_sd)
  sd <- ((1-rho^2)*x_sd^2)^0.5
  z_di <- dnorm(z, mean,sd)
  z_di_N <- z_di/sum(z_di)
  Lik[,j] <- z_di_N
}

Prepos <- c(rep(0,N))                       # Preposterior Rows
Pos <- matrix(0,nrow = N,ncol = N)    # Posterior Table
for (j in 1:N) {
  Prepos[j] <- sum(Lik[j,]*f_x_N)
  for (i in 1:N) {
    Pos[i,j] <- Lik[j,i]*f_x_N[i]/Prepos[j]
  }
  Pos[,j] <- Pos[,j]/sum(Pos[,j])
}

sum <- 0
for (j in 1:N) {
  VOI <- Prepos[j]*max(sum(Pos[,j]*x),0)
  sum <- sum + VOI
}
EVII <- sum-max(sum(x*f_x_N),0)
return(EVII)
}
```

6.7 Alghorithem of Sensitivity Analysis of VOI to Mean of Prior (R Code)

```
X <- data.frame(NPV =rnorm(100000,30,5))
Y <- data.frame(NPV = rnorm(100000,40,5))
Z <- data.frame(NPV = rnorm(100000,50,5))
E <- data.frame(NPV = rnorm(100000,60,5))
ee <- data.frame(NPV = rnorm(100000,70,5))

# Now, combine your two dataframes into one.
# First make a new column in each that will be
# a variable to identify where they came from later.
X$type <- 'Mean_PV_Production_Prior = 30'
Y$type <- 'Mean_PV_Production_Prior = 40'
Z$type <- 'Mean_PV_Production_Prior = 50'
E$type <- 'Mean_PV_Production_Prior = 60'
ee$type <- 'Mean_PV_Production_Prior = 70'

# and combine into your new data frame vegLengths
vegLengths <- rbind(X,Y,Z,E,ee)
ggplot(vegLengths, aes(NPV, fill = type)) + geom_density(alpha = 0.2)
```

6.8 Alghorithem of Sensitivity Analysis of VOI to Standard Deviation of Prior (R Code)

```
X <- data.frame(NPV =rnorm(100000,50,2.5))
Y <- data.frame(NPV = rnorm(100000,50,5))
Z <- data.frame(NPV = rnorm(100000,50,10))
E <- data.frame(NPV = rnorm(100000,50,15))

# Now, combine your two dataframes into one.
# First make a new column in each that will be
# a variable to identify where they came from later.
X$type <- 'SD_PV_Production_Prior = 2.5'
Y$type <- 'SD_PV_Production_Prior = 5'
Z$type <- 'SD_PV_Production_Prior = 10'
E$type <- 'SD_PV_Production_Prior = 15'

# and combine into your new data frame vegLengths
vegLengths <- rbind(X,Y,Z,E)
ggplot(vegLengths, aes(NPV, fill = type)) + geom_density(alpha = 0.2)
```

6.9 Optimization Alghorithem, Fittness Function: Machine Learning Model (R Code)

```
modelxgb <- readRDS('NEWMODELUP.rds')
speed_En1_Re <- read.csv('SPEED.csv',header=T)
speed <- speed_En1_Re[,2:46]
install.packages(c('GA','fastmaRching'))
install.packages(c('parallel','doParallel'))
library(doParallel)
library(GA)
library(fastmaRching)

ML_fun_grid <- function(X1,Y1,X2,Y2,X3,Y3,X4,Y4,A1,A2,A3,A4,
                        gam1,gam2,gam3,gam4,speed){
  X1 <- round(X1)
  X2 <- round(X2)
  X3 <- round(X3)
  X4 <- round(X4)
  Y1 <- round(Y1)
  Y2 <- round(Y2)
  Y3 <- round(Y3)
  Y4 <- round(Y4)
  slope <- c(-0.0005,-0.00025,0,0.00025,0.00050,0.00075,0.00100,0.00125,
             0.00150,0.00175,0.00200)
  Avalue <- c(100,150,200,250,300)
  A_1 <- Avalue[findInterval(A1,Avalue)]
  A_2 <- Avalue[findInterval(A2,Avalue)]
  A_3 <- Avalue[findInterval(A3,Avalue)]
  A_4 <- Avalue[findInterval(A4,Avalue)]

  Gam1 <- slope[findInterval(gam1,slope)]
  Gam2 <- slope[findInterval(gam2,slope)]
  Gam3 <- slope[findInterval(gam3,slope)]
  Gam4 <- slope[findInterval(gam4,slope)]

  con_fm <- c(rep(0,11))
  grid <- speed
  seed <- c(X1,Y1,0,1)
  fm <- gridFastMarch(grid, seed, spatial.res=10)
  con_fm[1] <- fm$arrival.time[X2,Y2]^2/4/3600


  seed <- c(X3,Y3,0,1)
  fm <- gridFastMarch(grid, seed, spatial.res=10)
  con_fm[2] <- fm$arrival.time[X1,Y1]^2/4/3600
  con_fm[3] <- fm$arrival.time[X2,Y2]^2/4/3600


  seed <- c(X4,Y4,0,1)
  fm <- gridFastMarch(grid, seed, spatial.res=10)
  con_fm[4] <- fm$arrival.time[X1,Y1]^2/4/3600
  con_fm[5] <- fm$arrival.time[X2,Y2]^2/4/3600
```

```
  con_fm[6] <- fm$arrival.time[X3,Y3]^2/4/3600


  seed <- c(23,23,0,1)
  fm <- gridFastMarch(grid, seed, spatial.res=10)
  con_fm[7] <- fm$arrival.time[X1,Y1]^2/4/3600
  con_fm[8] <- fm$arrival.time[X2,Y2]^2/4/3600
  con_fm[9] <- fm$arrival.time[X3,Y3]^2/4/3600
  con_fm[10] <- fm$arrival.time[X4,Y4]^2/4/3600
  seed <- cbind(c(X1,Y1,0,1),c(X2,Y2,0,1),c(X3,Y3,0,1),c(X4,Y4,0,1))
  fm <- gridFastMarch(grid, seed, spatial.res=10)
  fmm_N <- fm$arrival.time^2/4/3600
  PV=(sum(sum(fmm_N<fmm_N[23, 23])))*1000/20250
  con_fm[11] <- PV

  NPV_Ml <- predict(modelxgb,data.frame(ConI1I2=con_fm[1],ConI3I1=con_fm[2],
            ConI3I2=con_fm[3],ConI4I1=con_fm[4],
            ConI4I2=con_fm[5],
            ConI4I3=con_fm[6],ConPI1=con_fm[7],
            ConPI2=con_fm[8], ConPI3=con_fm[9],ConPI4=con_fm[10],
            PV_flight=con_fm[11],A1=A_1,A2
           =A_2,A3=A_3,A4=A_4,gam1=Gam1,gam2=Gam2,gam3=Gam3,gam4=Gam4))
  return(NPV_Ml)
}


ML_fun_En <- function(X1,Y1,X2,Y2,X3,Y3,X4,Y4,A1,A2,A3,A4,gam1,gam2,gam3,gam4) {
  result <- c(rep(0,10))
  for (i in 1:10) {
    grid <- my.list[[i]]
    result[i] <- ML_fun_grid(X1,Y1,X2,Y2,X3,Y3,X4,Y4,A1,A2,A3,A4,gam1,gam2,gam3
                             ,gam4,speed=grid)
  }
  mean_result <- mean(result)
  return(mean_result)
}

A <- ga(type = "real-valued",
fitness = function(x) + ML_fun_En(x[1],x[2],x[3],
x[4],x[5],x[6],x[7],x[8],x[9],                                                    x[10],x[11],
x[16]),lower = c(rep(1,8),rep(100,4),
rep(-0.0005,4)), upper = c(rep(45,8),rep(300,4),
rep(0.002,4)),popSize = 50, maxiter = 50,
run = 10,parallel = T)
```

6.10 Optimization Alghorithem, Fittness Function: Eclipse Reservoir Simulator (R Code)

```
setwd("C:/Users/243886/OneDrive - Universitetet i Stavanger/ML-5spot-INJ-SGS/Rcode")
library(readr)
```

```
X2D_4Inj_1Prod_R1 <- read_delim("2D_4Inj_1Prod_R.DATA",
                               "\t", escape_double = FALSE, na = "null",
                               trim_ws = TRUE)
run <- 'C:/ecl/macros/eclrun.exe eclipse "C:/Users/243886/OneDrive -
Universitetet i Stavanger/ROOPT_Ide/En#/Ensemble.DATA"'

Eclipse <- function(X1,Y1,X2,Y2,X3,Y3,X4,Y4,A1,A2,A3,A4,gam1,gam2,gam3,gam4,En){
  wd <- 'C:/Users/243886/OneDrive - Universitetet i Stavanger/ROOPT_Ide/En#'
  run1 <- gsub('#',En,run)
  newwd <- gsub('#',En,wd)
  setwd(newwd)
  X1 <- round(X1)
  X2 <- round(X2)
  X3 <- round(X3)
  X4 <- round(X4)
  Y1 <- round(Y1)
  Y2 <- round(Y2)
  Y3 <- round(Y3)
  Y4 <- round(Y4)
  slope <- c(-0.0005,-0.00025,0,0.00025,0.00050,0.00075,0.00100,0.00125,
             0.00150,0.00175,0.00200)
  Avalue <- c(100,150,200,250,300)
  A_1 <- Avalue[findInterval(A1,Avalue)]
  A_2 <- Avalue[findInterval(A2,Avalue)]
  A_3 <- Avalue[findInterval(A3,Avalue)]
  A_4 <- Avalue[findInterval(A4,Avalue)]

  Gam1 <- slope[findInterval(gam1,slope)]
  Gam2 <- slope[findInterval(gam2,slope)]
  Gam3 <- slope[findInterval(gam3,slope)]
  Gam4 <- slope[findInterval(gam4,slope)]

  data <- X2D_4Inj_1Prod_R1
  g1 <- data[131,1]
  g2 <- data[132,1]
  g3 <- data[133,1]
  g4 <- data[134,1]
  g5 <- data[138,1]
  g6 <- data[139,1]
  g7 <- data[140,1]
  g8 <- data[141,1]
  G1 <- gsub('101',X1, g1)
  G2 <- gsub('102',Y1, G1)
  G3 <- gsub('103',X2, g2)
  G4 <- gsub('104',Y2, G3)
  G5 <- gsub('105',X3, g3)
  G6 <- gsub('106',Y3, G5)
  G7 <- gsub('107',X4, g4)
  G8 <- gsub('108',Y4, G7)
  G9 <- gsub('109',X1, g5)
  G10 <- gsub('110',Y1, G9)
  G11 <- gsub('111',X2, g6)
  G12 <- gsub('112',Y2, G11)
```

```
  G13 <- gsub('113',X3, g7)
  G14 <- gsub('114',Y3, G13)
  G15 <- gsub('115',X4, g8)
  G16 <- gsub('116',Y4, G15)
  data[131,1] <- G2
  data[132,1] <- G4
  data[133,1] <- G6
  data[134,1] <- G8
  data[138,1] <- G10
  data[139,1] <- G12
  data[140,1] <- G14
  data[141,1] <- G16
  #'C:/Users/243886/OneDrive - Universitetet i Stavanger/ROOPT/Ecl'
  write.table(data, file ="Ensemble.DATA", sep = "\t",quote = F,
              row.names = F,col.names = F)
  int <- seq(from=0,to=1485,by=30)
  InjectionRatesex <- Injec_up1
  InjectionRatesex[seq(from=2,to=400,by=8),5] <- A_1*exp(-Gam1*int)
  InjectionRatesex[seq(from=3,to=400,by=8),5] <- A_2*exp(-Gam2*int)
  InjectionRatesex[seq(from=4,to=400,by=8),5] <- A_3*exp(-Gam3*int)
  InjectionRatesex[seq(from=5,to=400,by=8),5] <- A_4*exp(-Gam4*int)
  write.table(InjectionRatesex, 'InjectionRates.inc',quote = F,
              row.names = F,col.names = F, na = '')
  a1 <- A_1*exp(-Gam1*int)
  a2 <- A_2*exp(-Gam2*int)
  a3 <- A_3*exp(-Gam3*int)
  a4 <- A_4*exp(-Gam4*int)
  AA <- a1+a2+a3+a4
  shell(run1)
  return(aa(AA,newwd))

}


aa <- function(AA,newwd){
  setwd(newwd)
  ens <- "ENSEMBLE.RSM"
  NPV <- read.delim(ens,  header=FALSE, comment.char="#")
  NPVcalc <- NPV[, c(2,5,6)]
  colnames(NPVcalc) <- c('Days', 'Oilrate', 'waterrate')
  NPVcalc <- na.omit(NPVcalc)
  NPVcalc <- NPVcalc[-seq(9,400,by = 2), ]
  rownames(NPVcalc) <- 1:nrow(NPVcalc)
  days <- as.numeric(levels(NPVcalc$Days))[as.integer(NPVcalc$Days)]
  days <- na.omit(days)
  l1 <- min(which(days==c(1), arr.ind = TRUE))
  l2 <- min(which(days==c(1500), arr.ind = TRUE))
  days <- days[c(l1:l2)]
  oil <- as.numeric(levels(NPVcalc$Oilrate))[as.integer(NPVcalc$Oilrate)]
  oil <- na.omit(oil)
  oil <- oil[c(l1:l2)]
  water <- as.numeric(levels(NPVcalc$waterrate))[as.integer(NPVcalc$waterrate)]
  water <- na.omit(water)
```

```
  water <- water[c(l1:l2)]
  FWCT <- numeric(length = length(water))
  FWCT <- water/(water+oil)
  t <- seq(0,days[length(which(FWCT<0.85, arr.ind = TRUE))],by = 30)
  t[1] <- 1
  cashflow <- numeric(length(t)-1)
  discashflow <- numeric(length(t)-1)
  b <- 0.08
  for (j in 1:(length(t)-1)) {
    m1 <- t[j]
    z1 <- which(days==c(m1), arr.ind = TRUE)
    m2 <- t[j+1]
    z2 <- which(days==c(m2), arr.ind = TRUE)
    cashflow[j] <- (mean(oil[z1],oil[z2])*6.29*75-
                      mean(water[z1],water[z2])*
                      6.29*19-AA[j]*5*6.29)*30
    discashflow[j] <- cashflow[j]/((1+b)^(t[j+1]/360))
  }
  return(sum(discashflow))}

Eclipse1 <- function(X1,Y1,X2,Y2,X3,Y3,X4,Y4,A1,A2,A3,A4,
                     gam1,gam2,gam3,gam4) {
  a <- foreach(En=1:50, .export=c("Eclipse")) %dopar% {
    Eclipse(X1,Y1,X2,Y2,X3,Y3,X4,Y4,A1,A2,A3,A4,gam1,gam2,
            gam3,gam4,En)
  }
  return(mean(unlist(a)))
}

GA_Eclipse <- ga(type = "real-valued",
        fitness = function(x) Eclipse1(x[1],x[2],x[3],
        x[4],x[5],x[6],x[7],x[8],x[9]
       ,x[10],x[11],x[12],x[13],x[14],x[15],x[16])
        ,lower = c(rep(1,8),rep(100,4),rep(-0.0005,4)),
        upper = c(rep(45,8),rep(300,4),rep(0.002,4)),
            popSize = 25, maxiter = 20,run = 10,parallel = T,
            suggestions = suggestedSol)
```

Bibliography

Ahmadi, M. A., Ebadi, M., Shokrollahi, A., and Majidi, S. M. J. (2013). Evolving artificial neural network and imperialist competitive algorithm for prediction oil flow rate of the reservoir. *Applied Soft Computing*, 13(2):1085–1098.

Bellout, M. C., Ciaurri, D. E., Durlofsky, L. J., Foss, B., and Kleppe, J. (2012). Joint optimization of oil well placement and controls. *Computational Geosciences*, 16(4):1061–1079.

Box, G. E. (1979). All models are wrong, but some are useful. *Robustness in Statistics*, 202.

Bratvold, R. and Begg, S. (2010). *Making good decisions*. Society of Petroleum Engineers.

Bratvold, R. B., Bickel, J. E., Lohne, H. P., et al. (2009). Value of information in the oil and gas industry: past, present, and future. *SPE Reservoir Evaluation & Engineering*, 12(04):630–638.

Bratvold, R. B., Thomas, P., et al. (2014). Robust discretization of continuouos probability distributions for value-of-information analysis. In *International Petroleum Technology Conference*. International Petroleum Technology Conference.

Castelletti, A., Galelli, S., Restelli, M., and Soncini-Sessa, R. (2010). Tree-based reinforcement learning for optimal water reservoir operation. *Water Resources Research*, 46(9).

Chen, Y., Oliver, D. S., Zhang, D., et al. (2009). Efficient ensemble-based closed-loop production optimization. *SPE Journal*, 14(04):634–645.

Eidsvik, J., Mukerji, T., and Bhattacharjya, D. (2015). *Value of information in the earth sciences: Integrating spatial modeling and decision analysis*. Cambridge University Press.

Forouzanfar, F. and Reynolds, A. (2014). Joint optimization of number of wells, well locations and controls using a gradient-based algorithm. *Chemical Engineering Research and Design*, 92(7):1315–1328.

Goldberg, D. E. (1989). Genetic algorithms in search, optimization and machine learning. addsion-wesley longman. *Reading*.

Grayson, C. J. (1960). *Decisions under uncertainty: Drilling decisions by oil and gas operators*. Ayer.

Holland, J. H. (1975). Adaptation in natural and artificial systems ann arbor. *The University of Michigan Press*, 1:975.

Hong, A., Bratvold, R., and Nævdal, G. (2017). Robust production optimization with capacitance-resistance model as precursor. *Computational Geosciences*, 21(5-6):1423–1442.

Nwachukwu, A., Jeong, H., Pyrcz, M., and Lake, L. W. (2018a). Fast evaluation of well placements in heterogeneous reservoir models using machine learning. *Journal of Petroleum Science and Engineering*, 163:463–475.

Nwachukwu, A., Jeong, H., Sun, A., Pyrcz, M., Lake, L. W., et al. (2018b). Machine learning-based optimization of well locations and wag parameters under geologic uncertainty. In *SPE Improved Oil Recovery Conference*. Society of Petroleum Engineers.

Nwachukwu, C. (2019). *Machine learning solutions for reservoir characterization, management, and optimization*. PhD thesis.

Pyrcz, M. J. and Deutsch, C. V. (2014). *Geostatistical reservoir modeling*. Oxford university press.

Sayarpour, M., Kabir, C. S., Lake, L. W., et al. (2009). Field applications of capacitance-resistance models in waterfloods. *SPE reservoir evaluation & engineering*, 12(06):853–864.

Schlaifer, R. (1959). Probability and statistics for business decisions.

Scrucca, L. et al. (2013). Ga: a package for genetic algorithms in r. *Journal of Statistical Software*, 53(4):1–37.

Sethian, J. A. (1996). A fast marching level set method for monotonically advancing fronts. *Proceedings of the National Academy of Sciences*, 93(4):1591–1595.

Sharifi, M., Kelkar, M., Bahar, A., Slettebo, T., et al. (2014). Dynamic ranking of multiple realizations by use of the fast-marching method. *SPE Journal*, 19(06):1–069.

Stalgorova, E., Babadagli, T., et al. (2012). Field-scale modeling of tracer injection in naturally fractured reservoirs using the random-walk particle-tracking simulation. *SPE Journal*, 17(02):580–592.

van Essen, G., Zandvliet, M., Van den Hof, P., Bosgra, O., Jansen, J.-D., et al. (2009). Robust waterflooding optimization of multiple geological scenarios. *Spe Journal*, 14(01):202–210.

Yu, S., Zhu, K., and Diao, F. (2008). A dynamic all parameters adaptive bp neural networks model and its application on oil reservoir prediction. *Applied mathematics and computation*, 195(1):66–75.