



University of  
Stavanger

**Faculty of Science and Technology**

## **MASTER'S THESIS**

Study Program/Specialization:

Petroleum Engineering/Reservoir Engineering

Spring Semester, 2019

Open/~~Restricted~~ access

Writer:

Mohammad Nezhadali

.....  
(Writer's signature)

Faculty supervisor(s):

Remus Gabriel Hanea

Thesis title:

Hybridization of gradient-based and gradient-free optimization techniques for simultaneous optimization of number of wells, their location and drilling time in a 2-dimensional reservoir

Credits (ECTS): 30

Key words:

Hybrid Optimization  
Well Placement  
Reservoir Optimization  
Robust Optimization  
Ensemble-based Optimization

Pages: 72

+Enclosure: 5

Stavanger, 15.06.2019

## **Abstract**

Optimization has always been a challenging scope in any field of science, and reservoir management is no exception. By the advent of computational power, the use of mathematical algorithms in conjunction with reservoir simulators has prospered drastically. These mathematical optimization algorithms are divided into two main categories, namely gradient-based algorithms and gradient-free algorithms.

The optimization in petroleum industry is mostly focused on maximization of the Net Present Value (NPV) of the project. Utilization of gradient-based optimization techniques for this aim, even though being popular, has some limitations including risk of convergence to local optima and difficulties in optimizing discrete and categorical variables such as well path and well locations. On the other hand, gradient-free optimization techniques, which mostly rely on probabilistic and stochastic principles, have their own limitations.

This study shows that hybridization of the gradient-free and gradient-based optimization techniques has a good potential to result in a robust reservoir optimization scheme which outperforms both gradient-free and gradient-based optimizers separately. In doing so, a serial optimization algorithm is devised by hybridization of genetic algorithm, simulated annealing, and stochastic gradient descent method. Employing this algorithm in several optimization problems in a two-dimensional oil field, has proved that this algorithm outperforms its previous forerunners in optimization of the number of wells in the field, their location, and drilling schedule.

Findings of this study can be used to raise a foundation for generation of new optimization techniques which can boost the standards in both the complexity of problems solved in petroleum industry and their computational efficiency.

## **Acknowledgement**

I would like to express my deepest appreciation to my supervisor, Professor Remus Gabriel Hanea, who has the attitude and the substance of a genius: he revitalized my passion towards mathematics and mathematical problems in petroleum engineering once again both as a teacher and supervisor, and continually and convincingly helped me throughout this last one year of my studies. Without him completion of this thesis would have not been possible.

In addition, I would like thank PhD students, Camilo Malagon Nieto and Brage Strand Kristoffersen, as well as Doctor Alireza Zare and Doctor Mathias Bellout for their help and support who were available for help whenever I faced problems.

I would also show my gratitude to the University of Stavanger for providing good learning environment, and the infrastructure required for the progress of this project.

Finally, I would like to express my deepest gratitude to my family for their continuous and heartfelt support and patience throughout years of my life, and in particular through these two years of my studies.

Mohammad Nezhadali  
Stavanger, Norway  
June 2019

# Table of Content

Abstract .....	ii
Acknowledgement .....	iii
Table of Content .....	iv
List of Figures .....	vi
List of Tables .....	viii
Abbreviations .....	ix
1 Introduction.....	1
2 Theory .....	4
2.1 Gradient-free optimization techniques:.....	4
2.1.1 Nelder-Mead Simplex method: .....	5
2.1.2 Genetic Algorithm.....	7
2.1.3 Particle Swarm Optimization: .....	11
2.1.4 Simulated Annealing.....	12
2.2 Gradient-based optimization techniques: .....	13
2.3 Hybridization of algorithms .....	14
3 Methodology: .....	16
3.1 Ensemble methods and Robust Optimization .....	17
3.2 Optimization program structure: .....	20
3.2.1 Genetic Algorithm: .....	21
3.2.2 Live Genetic Algorithm (LGA) .....	27
3.2.3 Simulated Annealing:.....	29
3.2.4 Gradient-based optimizer:.....	31
3.2.5 Hybridizing the Algorithms: .....	35
4 Results and discussion: .....	37
4.1 Sensitivity analysis: .....	37
4.1.1 Sensitivity analysis for the genetic algorithm .....	37
4.1.2 Simulated Annealing Sensitivity Analysis.....	47
4.2 Solved Problems .....	49
4.2.1 Primary problem .....	49
4.2.2 Main Problem: .....	52
4.2.3 Problem of moderate drilling cost.....	56
4.2.4 Problem of costly drilling .....	59
4.2.5 Problem of different initial well locations .....	62

4.2.6 Problem of single pre-drilled well .....	64
5 Conclusion and future work.....	68
6 References.....	70
Appendix A—The Eclipse DATA file.....	73

## List of Figures

Figure 2. 1 Evolution of Nelder-Mead method in a sample problem .....	5
Figure 2. 2 Probability distribution function example for the children in GA .....	9
Figure 2. 3 Schematic of compromise between particle velocity, its trust in itself and its trust in swarm .....	12
Figure 2. 4 An example of SA advancement .....	13
Figure 2. 5 comparison of Gradient Descent and Stochastic Gradient Descent methods.....	14
Figure 3. 1 3D representation of the field structure .....	17
Figure 3. 2 Permeability fields of a sample ensemble of realizations .....	19
Figure 3. 3 Basic structure of the optimization algorithm .....	20
Figure 3. 4 Different formats of GA fitness function .....	24
Figure 3. 5 Flow chart of the GA used in this study .....	26
Figure 3. 6 The population list in LGA.....	29
Figure 3. 7 A comparison of GD and SGD.....	31
Figure 3. 8 SGD progress example .....	33
Figure 3. 9 A simplistic graph on metaheuristic neighborhood.....	34
Figure 3. 10 Hybrid algorithm flow chart.....	36
Figure 4. 1 Mean of NPV achieved by different inflation rates.....	38
Figure 4. 2 Maximum NPV achieved by different inflation rates .....	38
Figure 4. 3 mean of the NPV achieved by different inflation rates .....	39
Figure 4. 4 Maximum NPV achieved by different inflation rates .....	40
Figure 4. 5 Mean of the NPV of population for different crossover ratios.....	41
Figure 4. 6 Maximum NPV achieved by different crossover ratios .....	41
Figure 4. 7 Mean NPV achieved for different elitism schemes .....	42
Figure 4. 8 Standard deviation of the maximum NPV achieved in several runs of the algorithm for different elitism schemes.....	43
Figure 4. 9 Mean of the NPV of the population in different mutation ratios .....	44
Figure 4. 10 maximum NPV achieved in different mutation rates .....	44
Figure 4. 11 Mean of the NPV of the population for different population sizes .....	45
Figure 4. 12 Maximum NPV achieved for different population sizes.....	46
Figure 4. 13 Maximum NPV achieved for different acceptance chance ratio.....	47
Figure 4. 14 Maximum NPV achieved for different acceptance probability half-lives.....	48
Figure 4. 15 Maximum NPV in different optimization algorithms for primary problem.....	50
Figure 4. 16 Simulations ran in different optimization algorithms for primary problem .....	50
Figure 4. 17 Maximum NPV in different optimization algorithms for the main problem .....	53
Figure 4. 18 Simulations ran in different optimization algorithms for the main problem.....	53
Figure 4. 19 Final water saturation map of the ensemble in the main problem.....	55
Figure 4. 20 NPV evolution of the optimized solution and its ensemble and its comparison to a sub-optimal solution and its ensemble.....	56

Figure 4. 21 Maximum NPV achieved by different optimization algorithms for the problem of moderate drilling cost .....	57
Figure 4. 22 Number of simulations ran in different optimization algorithms for the problem of moderate drilling cost .....	58
Figure 4. 23 Maximum NPV achieved by different optimization algorithms for the problem of costly drilling .....	60
Figure 4. 24 Number of simulations ran in different optimization algorithms for the problem of costly drilling .....	60
Figure 4. 25 Maximum NPV achieved by different optimization algorithms for the problem of different initial well locations .....	63
Figure 4. 26 Number of simulations ran in different optimization algorithms for the problem of different initial well locations .....	63
Figure 4. 27 Maximum NPV achieved by different optimization algorithms for the problem of single pre-drilled well .....	65
Figure 4. 28 Number of simulations ran in different optimization algorithms for the problem of single pre-drilled well .....	66

## List of Tables

Table 3. 1 The structural properties of the simulation case .....	16
Table 4. 1 Primary problem definition.....	49
Table 4. 2 Solution to primary problem.....	51
Table 4. 3 Definition of main problem .....	52
Table 4. 4 Solution to the main problem.....	54
Table 4. 5 Definition of the problem of moderate drilling cost .....	57
Table 4. 6 Solution to the problem of moderate drilling cost .....	58
Table 4. 7 Definition of the problem of costly drilling .....	59
Table 4. 8 Solution to the problem of costly drilling .....	61
Table 4. 9 Definition of the problem of different well initial well locations.....	62
Table 4. 10 Solution to the problem of different initial well locations.....	64
Table 4. 11 Definition of the problem of single pre-drilled well.....	65
Table 4. 12 Solution to the problem of single pre-drilled well.....	67

## Abbreviations

2-D	-	two-dimensional
3-D	-	Three-dimensional
ANN	-	Artificial Neural Networks
AI	-	Artificial Intelligence
BHP	-	Bottom Hole Pressure
GA	-	Genetic Algorithm
GD	-	Gradient Descent
ICV	-	Intake Control Valve
LGA	-	Live Genetic Algorithm
n-D	-	n-Dimensional
NPV	-	Net Present Value
PSO	-	Particle Swarm Optimization
RO	-	Robust Optimization
SA	-	Simulated Annealing
SGD	-	Stochastic Gradient Descent

# 1 Introduction

Defining it in a general way, optimization is the process of finding a minimum or maximum of an objective function in a search space while satisfying a set of constraints. In petroleum reservoir management, normally the objective function is the Net Present Value of the field over a certain period of time, which is a measure of how satisfactory the project is, in economic standards. The search space is made of the parameters forming the field development plan and production schedule including the number of wells, ICVs, timing of drilling, order of wells to be drilled, injection or production switches as well as several IOR methods and many other parameters which form a hyper-complex search space. The constraints such as the limitations on production, environmental issues, well head and piping facilities as well as reservoir restrictions, etc. normally confine the search space and consequently complexify the problem to a higher degree. The problem that is finally in front, is almost impossible to solve. Nonetheless, in order to be able to solve the problem, some simplifications on the search space are required.

In the first step, according to the conventions of industry, some of the parameters are set to be fixed or limited. Setting the well path to be fixed or deciding on the IOR method to be planned prior to the optimization are examples of this simplification. These simplifications should also happen over the variables that are changing with time and need to be optimized. ICV openings, for instance, can change at any point in time; however, since by doing so optimization of the problem becomes impossible, the ICV openings are set to be constant at certain periods of time and their modification is limited to discrete time intervals, every 3 months for example. These are all added to the initial simplification of geological reality to an ensemble of geological models. The hypothesis that the reality and its uncertainty can be modelled with an ensemble of static models and a dynamic model, whose accuracy in prediction of reservoir behavior is still unknown, are other influential simplifications. After all these steps, an optimization problem is formed which is, finally, computationally feasible to solve.

Regardless of the traditional methods of optimization and the novel machine learning techniques, the process of finding the optimized solution in the simplified search space is classically performed using some mathematical optimization methods which are generally divided into two categories: gradient-based optimization techniques and gradient-free optimization techniques [1]. As it can be

inferred from their name, their difference comes from relying of one category of techniques on calculation of the gradient of the objective function over the search space. Despite being very successful and widely used in industry, gradient-based optimization techniques have problems in handling discrete or categorical parameters. In petroleum engineering, the number of wells to be drilled and their location are examples of these parameters. On the other hand, gradient-free optimization techniques are designed to overcome this kind of problems, yet they also have their own limitations.

The main objective of this study is to set up an initiative on hybridization of gradient-based and gradient-free optimization techniques to come up with a more robust optimization scheme which can optimize the number of wells and their location and time of drilling based on the preferred objective function. In doing so, the performance of several gradient-based and gradient-free algorithms on optimization of reservoir and similar problems were studied, and three optimization algorithms were finalized for hybridization. These three algorithms are Genetic Algorithm (GA), Simulated Annealing (SA), and Stochastic Gradient Descent (SGD). In order to make a discrete form of the problem, which is closer to the reality, it was assumed that the well locations are in the center of the grid cells and their drilling time is in a certain year of the project, which reduces the problem into a discrete problem of finding the proper number of wells and the grid in which they are drilled and their year of drilling. Assuming the wells to be controlled by the bottom hole pressure (BHP) for the whole production time and the wells to be vertical, which is the basic form of any optimization problem, and the field to be consisted of 800 (40 by 20) grid cells, initially a sensitivity analysis was performed on the parameters of the above-mentioned algorithms to tune them for the case study, and following that, all the algorithms as well as their hybridization in several ways were tried on the field and the quality of their optimization was assessed quantitatively.

As the previous studies suggest, one of the best hybrid optimization schemes that has proved to work well in similar problems, is the serial use of optimization algorithms that can result in a better optimization scheme, when comparing to the original algorithms [1-5]. Accordingly, the above-mentioned algorithms were firstly, adjusted for optimization of some similar problems on the proposed petroleum field, and then merged serially with each other. Consequently, several case

studies were performed using the algorithms. And six of these problems are presented in this thesis in order to compare the quality of optimization algorithms.

Finally, a brief elaboration of potential extensions of this work and the application its ideas in industry were addressed.

## **2 Theory**

By the developments in technology the computational power has increased drastically. As optimization entails several simulations as part of it, this development in technology has helped a lot in advancements in optimization realm. As early methods were only capable of solving linear programming problems and heuristics, the more novel methods can solve much more complex problems with nonlinearity involved. The use and improvement of many new techniques such as discrete and continuous optimization, stochastic programming, metaheuristics, and genetic algorithms have made us capable of uncertainty handling and solution of much more complex problems [1].

The approach towards optimization in petroleum industry and literature can be categorized into three mainstreams including, sensitivity analysis and use of simulation tools, learning and machine learning based on the intuition and data form the past, and the generic term used in mathematics for optimization which entails mathematical and programming algorithms for optimization [6, 7].

Even though the first two categories of optimization were assumed to be outdated. In recent years thanks to upcoming of big data analysis and machine learning techniques as clustering, regression methods and neural networks, the second category, which relies on data-driven decisions, is emerging again. This emergence has proven to be successful both in optimization and substitution of simulators and models with proxies [8-12].

This study mainly focuses on the last category of the optimization, in which several optimization schemes are investigated, and some new hybridized methods are generated to attack the problems in petroleum industry. The optimization schemes in mathematics are divided into two main categories, gradient free and gradient-based Optimizations. Each of the categories have been widely used in the petroleum industry. A brief representation of the main subcategories and their employment in the petroleum industry will be covered in this chapter.

### **2.1 Gradient-free optimization techniques:**

There are several gradient-free optimization techniques which have been used in petroleum reservoir management. We will cover them in brief here.

### 2.1.1 Nelder-Mead Simplex method:

This method uses a heuristic approach to search for the optimum solution. It can also be referred as nonlinear simplex in the literature.

A simplex in an  $n$ -dimensional space is formed using  $n+1$  points whose any 4 points are not in a same plane. The  $n$ -dimensional space itself is formed using our control variables which are meant to be optimized. The  $n+1$  points forming the simplex are a set of potential solutions that are used as initial guess and are optimized in every step so as to finally converge in our optimum solution.

At any step of the optimization process either the simplex shrinks into a smaller size or it is reflected, expanded or contracted so that the mean of the simplex vertices decreases. This means that in any step we are moving forward towards the optimized solution; assuming that there are no very local good solutions to our problem which is a logical assumption. In petroleum reservoir management the models that we run for optimization are more of smooth functions with only some exceptions in highly heterogeneous fields or fractured reservoirs. Additionally, very local good solutions normally have a higher risk of failure which are not of our interest.

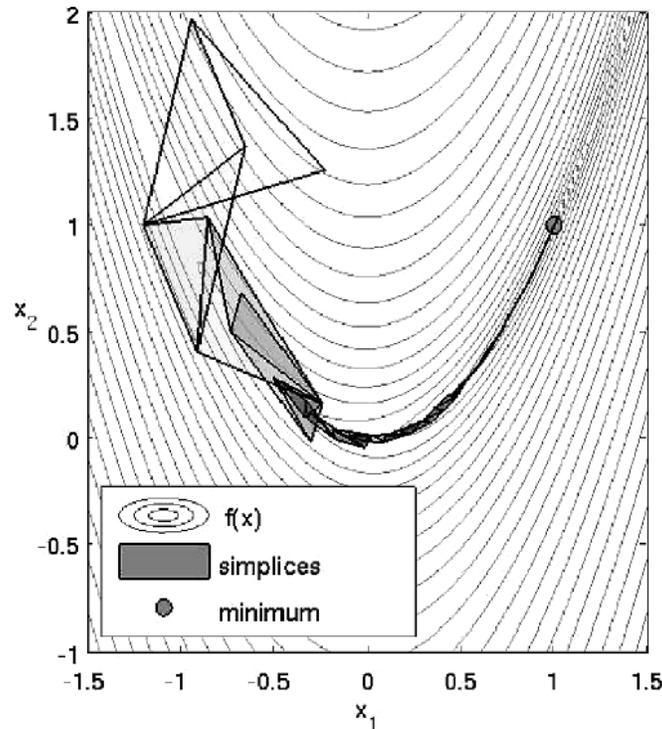


Figure 2. 1 Evolution of Nelder-Mead method in a sample problem

In general even though this method does not require calculation of derivative which benefits us in computational cost, it does not show good convergence speed in high number of dimensions, especially in problems with more than 10 design variables. This method has been used in reservoir production optimization in the literature. In a study by Rahmawati et al. two scenarios were considered for optimization of the NPV in a WAG project[13]. In the first scenario the optimization variables were listed as sales gas fraction, DPC temperature, gas-condensate reinjection fraction, and lean gas-condensate re-injection fraction. In the second scenario, the decision variables were comprised of sales gas fraction, DPC temperature, gas injection rate and water injection rate for WAG scenario and WAG period. This research showed that using nonlinear simplex method for optimizing the NPV could result in 12 to 25 percent of growth in the NPV compared to initial engineering guess.

In another study for a case study of optimization of a reservoir in Brugge the Simplex reflection Nelder-mead method of optimization was compared with four other optimization schemes including Pattern Search Hook-Jeeves method, Generalized Pattern Search method, Line search derivative free method, and sequential quadratic programming method[14]. The control variables were the water cuts of the production wells under which the wells were shut. According to the study, the line-search derivative-free method was the most efficient among them and the Nelder-mead was the slowest in convergence. The reason behind this can be the big number of control variables (54 wells) which has resulted in the slowness of Nelder-Mead method.

The same group in another research conducted on optimization of reservoir NPV using NPV as its objective function, proved that taking a set of water cuts as their optimization variables, a good initial guess can be made to start gradient-based optimization[15]. This study also proved that using a big number of search variables, will result in slow convergence of Nelder-Mead method among other methods for optimization.

In some older studies the simplex method is used to optimize daily oil rate without the use of simulators and by linearizing the optimization problem. These have resulted in sub-optimal solutions [16, 17].

### 2.1.2 Genetic Algorithm

A widely recognized method of optimization which is essentially inspired by the process of natural selection is the genetic algorithm. This method was initially devised and used by John Holland to understand the evolution of life by computational simulation[18]. This method is based on three main principles.

- Survival of the fittest species
- Reproduction of the species and transferring their traits to the next generation
- Mutation and randomized modification of the species characteristics

A short introduction of this useful algorithm in petroleum engineering will be provided in this section.

The algorithm starts with generating a population of possible solutions to the problem. In any optimization technique we are aiming at minimizing an objective function considering some constraints to the problem. Accordingly, based on the objective function and some other side-functions a term called “fitness” is defined. Any of the population members has its own value of objective function, and consequently fitness. This fitness is the essence of GA inspired by the natural selection. A member with a higher fitness is a better fit to the environment, hence has a higher chance of reproduction which means it is more probable that we have that member or some of its traits in the next generation. The good traits or the good members are transferred from one generation to the next and hereby it is assumed that by convergence of the objective function and fitness to its highest value in this probabilistic approach the optimum solution to the problem is found.

The process of reproduction and other concepts as mutation and generation which are adopted from natural phenomena, necessitate us to define their essence as well—the genome.

Any member of the population is a set of control variables which can lead us to an objective function; these control variables are called phenotype, since they are defined in the real environment of optimization. In order to make possibility for definition of the aforementioned concepts of optimization as reproduction, we have to define and encode the characteristics of any solution into a genome, which is called genotype. Consider this example:

Plan A, has drilling well  $W_{1A}$  in location  $L_{1A}$  and time  $T_{1A}$  after that drilling well  $W_{2A}$  in location  $L_{2A}$  and time  $T_{2A}$ . Plan B has drilling well  $W_{1B}$  in location  $L_{1B}$  and time  $T_{1B}$  followed with drilling well  $W_{2B}$  in location  $L_{2B}$  and time  $T_{2B}$ . There are several methods of encoding including binary encryption, decimal encryption, table encryption etc. These two plans can be encoded simply using decimal encryption in the following way: a string of 16 digits with 1<sup>st</sup> four digits showing the location of the 1<sup>st</sup> well the 2<sup>nd</sup> four digits the timing of the 1<sup>st</sup> well, the 3<sup>rd</sup> four digits the location of the 2<sup>nd</sup> well and the 4<sup>th</sup> four digits time of time of the drilling of the 2<sup>nd</sup> well.

Plan A:

$L_{1A}$	$T_{1A}$	$L_{2A}$	$T_{2A}$
----------	----------	----------	----------

Plan B:

$L_{1B}$	$T_{1B}$	$L_{2B}$	$T_{2B}$
----------	----------	----------	----------

If these two plans go into a reproduction process, using crossover we can come up with a pair of children for them. The crossover is switching between the parents at certain locations. For example, if the crossover is to happen after the location of the first well the children will be:

Plan C<sub>1</sub>:

$L_{1A}$	$T_{1B}$	$L_{2B}$	$T_{2B}$
----------	----------	----------	----------

Plan C<sub>2</sub>:

$L_{1B}$	$T_{1A}$	$L_{2A}$	$T_{2A}$
----------	----------	----------	----------

The crossover is normally performed in a probabilistic way and its rate is adjusted in such a way that the average number of crossovers in a pairing would be about 1 which is mimicked from the natural phenomena [19]. Also, the crossovers which break the structure and make the plan meaningless are also avoided, that is why normally the crossover is performed at the end of each gene section, but this is not a rule and in binary encryption it is not enforced. There are also some continuous schemes for crossover. Consider a crossover between well locations for example. In the scheme that was introduced previously, the child either had the traits from one parent or the traits from the other, therefore the trait that is inherited is only close to one of the parents. However, if the variable search space is a continuous one, the location of the well can be anything between the location of its parents or even anywhere in the search space. The probability density function of the location of the wells in one dimensional space with peaks in the location of the wells of the parents in figure 2.2 is an example of continuous crossover [19].

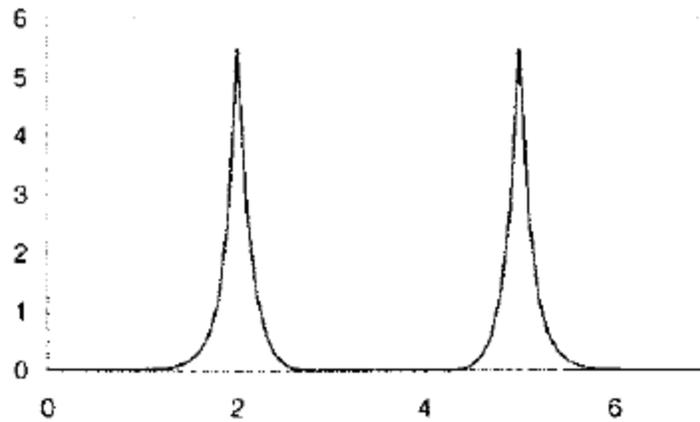


Figure 2. 2 Probability distribution function example for the children in GA

Another technique which is used to mimic natural phenomena in this method of optimization is mutation. As in natural processes this mutation can by chance act positive or negative for our objective function. As for the previous example, the mutation can be introduced in the following way:

Plan:	$L_1$	$T_1$	$L_2$	$T_2$
Mutated:	$L_1$	$T^*_1$	$L_2$	$T_2$

The drilling time of the first well is randomly changed to another time in the search space. The fitness of the mutated plan will decide probabilistically that it will go through reproduction again or not.

There are some other techniques that can help the optimization to progress better. Elitism, for instance, makes it possible that some of the best results of the generation survive to the next with a probability of unity; hence our best results so far will be saved. Niching divides the population into some subpopulations so that each of the subpopulations will occupy a certain proportion of fitness landscape. Diversifying is another technique in genetic algorithms which targets exploration of the search space. It introduces another term in the objective function, distance to other members of the population for example by growth in which the exploration of the search space will improve, and consequently the results of optimization will become more reliable[19].

One of the difficulties when using genetic algorithm is handling nonlinear constraints. When there is a discontinuity in some area of the search space, the children of any reproduction are not necessarily valid scenarios for the optimization process. Suppose one of the constraints tells us that the distance between any two wells has to be at least a certain amount. In the initial population satisfying the criterion is not difficult since human can be involved, but from that point on human intrusion in every step will be both time-consuming and bothersome; hence it has to be resolved in another way. In a study conducted by Emerick et al this problem was resolved using an algorithm called Genocop III[20]. This technique updates any invalid population member several times with the existing valid members so that it finally becomes a feasible production scenario. An article using this technique targeted simultaneous optimization of the number of wells as well as their location and trajectory [20]. The results showed that even though the totally random selection of the initial population had a better result than those of base case engineering guess, they were inferior to those conducted by an initial engineering guess.

An application of the genetic algorithms in reservoir optimization is planning and optimization of the unconventional wells. Defining and optimizing such kind of well is arduous in other methods of optimization since these wells are completely dependent on geological characteristics of the area at which the well is to be drilled. Using the GA though, a fixed chromosome of well characteristics can be defined and by change in the drilling environment the genome will evolve to comply with the geological traits of the reservoir. A study by Yeten et al used this method in conjunction with artificial neural network as proxy for reservoir simulation and a final state hill climbing optimizer to optimize the choice of well type and well plan of unconventional wells. This method was more successful than randomly chosen wells[21].

In another research on well placement for water flooding in the gulf of Mexico, polytope algorithm, artificial neural network, and kriging were introduced to GA algorithm to add some hill-climbing theme to its stochastic approach of optimization. This method showed to be computationally efficient by reducing the computational cost and also avoiding the local minima[22]. This technique has also been used to optimize the pipeline network in use, separator pressure, gas injection rate, well connection systems [23-25].

As of particular interest, the optimization problem of well-placement under uncertainty in addition to involving risk taking capability into objective function is also investigated in some studies [22,

26]. This has been done by designing an objective function which is weighted average based on the compromise in maximization of the objective function and minimization of its variance. However, in another study, a comparison between GA and PSO has been performed and the superiority of PSO over GA in the optimization of well types including vertical, deviated, dual lateral, etc. over single and multiple reservoir realizations has been shown [27].

### 2.1.3 Particle Swarm Optimization:

Another gradient-free optimization technique that is inspired from nature is the particle swarm optimization. This algorithm essentially mimics flocking of the birds. It starts from a random colony of possible answers to the problem and using the fitness function and previous responses to the problem it tries to improve it. The basic idea behind the PSO is as follows.

Initially a population of possible answers are generated. Each of the members is assumed to be a particle moving in the search space, which is assumed to be a n-dimensional space formed by the decision variables with some constraints. Any of these particles at each time-step has its own position, velocity, and direction of movement. Two terms are to be defined here; the best global position, and the best personal position of each particle. The position of the particle in the next time-step is a compromise between the current position, the velocity and direction of movement, the location of the best personal record, and the location of the best global record. The formulation on compromise is shown in equations 2.1 and 2.2.

$$v_{i,j}^{k+1} = v_{i,j}^k + c_1 r_1 (x_{best_{i,j}}^k - x_{i,j}^k) + c_2 r_2 (x_{gbest_j}^k - x_{i,j}^k) \quad (2.1)$$

$$x_{i,j}^{k+1} = x_{i,j}^k + v_{i,j}^{k+1} \quad (2.2)$$

In the above-mentioned formula,  $v_{i,j}^k$  is the  $j$ th component of velocity of  $i$ th particle in  $k$ th time-step.  $x_i^k$  is the position of  $i$ th particle in  $k$ th time-step.  $r_1$  and  $r_2$  are two random numbers with uniform probability in (0,1).  $x_{best_i}$  and  $x_{gbest}$  are the best position of  $i$ th component and the best position of the whole group so far.  $c_1$  and  $c_2$  are factors representing the particle's confidence in itself and swarm respectively. A graphical representation of the update is shown in figure 2.3 [28].

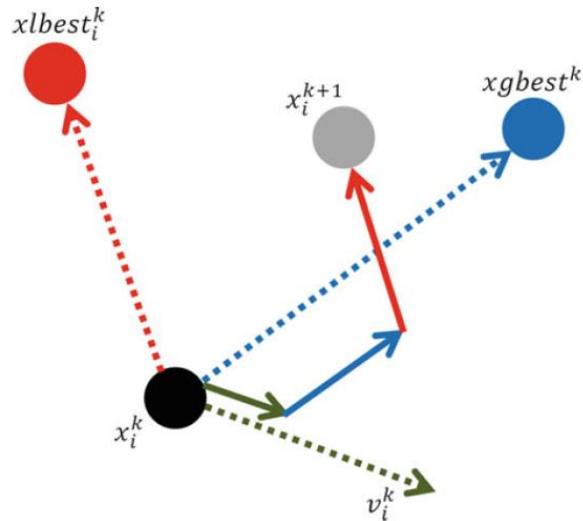


Figure 2. 3 Schematic of compromise between particle velocity, its trust in itself and its trust in swarm

The velocities of particles adaptively slow down as they converge to the solutions, and the solutions are obtained [29]. This method has recently become very popular in optimization of the problems faced in petroleum industry.

In a study by Shirang and Durlofsky, the development plan for number, type, location, and controls of the new wells is optimized using robust optimization. The uncertainty is taken into account using multiple realizations and the objective function has been the net present value of the entire project [30]. In some other studies the categorical variables such as well locations, are defined as integer-valued parameters and are optimized simultaneously with continuous variables such as valve openings using the PSO [31-33]. The objective function varies from the NPV, to total oil production of the field, and some other multi-objective functions that were defined previously.

### 2.1.4 Simulated Annealing

Simulated annealing is a probabilistic simulation method inspired by the annealing process in metallurgy. In metallurgy a piece of metal is tempered and cooled down several times consecutively so as to finally have the metal frozen in its least energy configuration. This method is mimicking the mentioned physical phenomenon to compute the minimum of the objective function after several stochastic moves which get more and more limited by the progress of the algorithm. As the metal reaches to its minimum state of energy, the simulated annealing algorithm

is aimed to conduct the control variable to its minimum state of energy or in other words optimized solution of the problem [34].

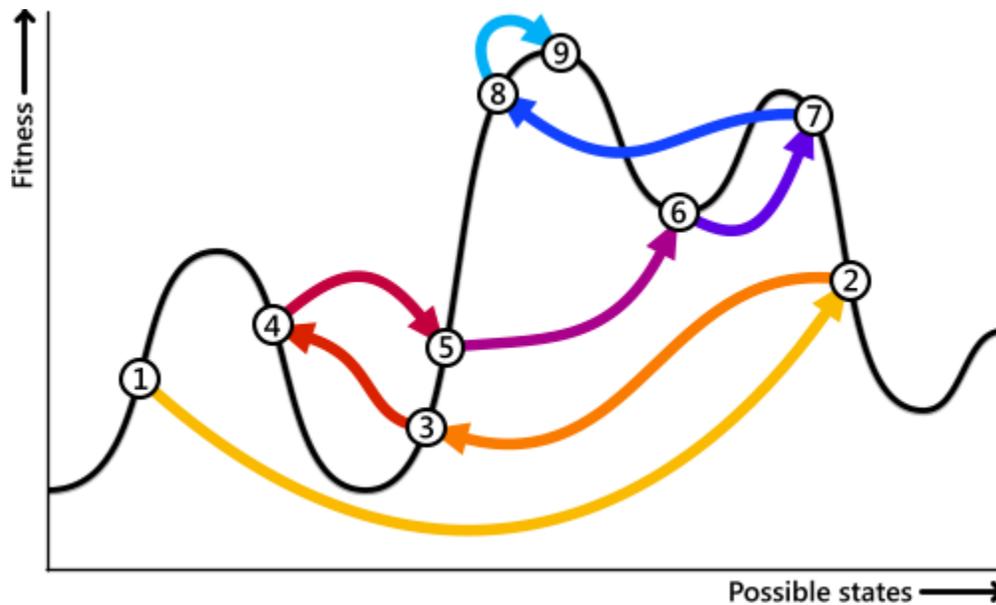


Figure 2. 4 An example of SA advancement

This method has been used in optimization of a wide range of problems in petroleum industry from minimum miscibility pressure prediction to fracture network modeling [5, 35].

## 2.2 Gradient-based optimization techniques:

Gradient-based techniques of optimization are both diverse and widespread in the academic literature and industry. All of these methods entail 4 main steps [36];

1. Convergence criteria check
2. Search direction computation
3. Step length calculation
4. Design variables update

Most of the gradient-based techniques including steepest descent method, conjugate gradient method, nonlinear conjugate gradient method, Newton's method, modified and quasi-Newton method, Davidon-Fletcher-Powell (DFP) Method, trust region method etc. entail computation of the derivative of the objective function, e.g. gradient, Hessian matrices. As we use commercial simulators in the petroleum industry for calculation of the objective function, calculation of these

matrices is either very costly or even impossible. Accordingly, instead of using gradient-based techniques for optimization of the reservoir we mostly use stochastic gradient descent techniques which are numerical estimations of the gradient based techniques, which are not very good approximates of steepest descent method; however, they prove to be practically very successful.

In this method a stochastic direction and a step length in the search space is chosen. The algorithm then updates to the next search location based on these two parameters and the current location, satisfying the condition that the objective function is to be improved. Figure 2.5 depicts a comparison between gradient descent method and stochastic gradient descent method in a two-dimensional search space [37].

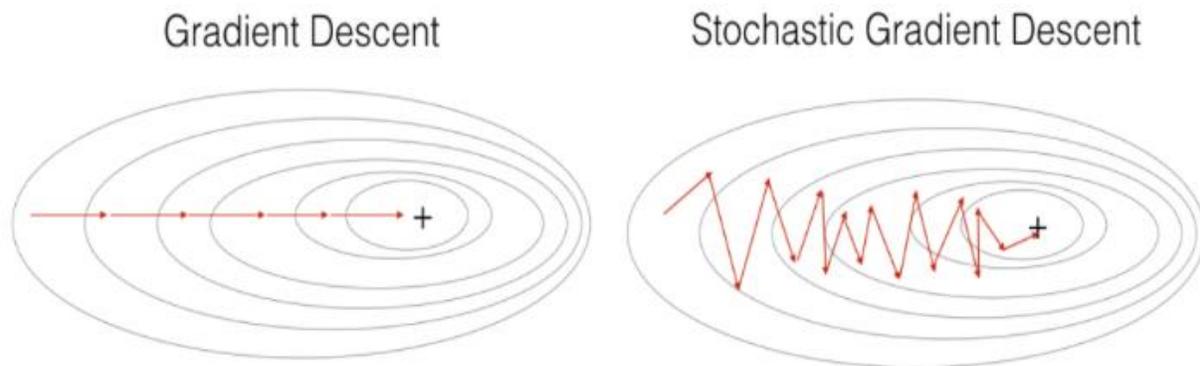


Figure 2. 5 comparison of Gradient Descent and Stochastic Gradient Descent methods

These techniques are widely used in the optimization section of the closed loop management for several purposes basically focusing in optimization of the continuous variables of the problem [38-42]. This technique has also been used in well-placement optimization by taking the hypothesis that wells are continuous variables in a constrained search space [43].

### 2.3 Hybridization of algorithms

The attraction of simultaneous benefit from pros of both categories of optimization methods in addition to minimizing their weakness in practice has encouraged many scientists to devise some hybridized techniques by which the problems are handled more efficiently. The joint use of

gravitational search and Nelder-Mead algorithm has been used to improve thin-wall structures in auto-motive industry [44]. Hybridization of joint genetic algorithm and particle swarm optimization has been used for training neural network-based artificial intelligence (AI) in oceanography and rainfall forecast [45]. Particle swarm optimization has been used in conjunction with simulated annealing to improve the prediction of minimum miscibility prediction of petroleum fluids and a combination of genetic algorithm and simulated annealing is used stochastic reservoir modeling improvement [5, 46]. Also, hybrid algorithms made from combination of genetic algorithm and simulated annealing are used for optimization of multiple water reservoirs in the field of hydrology [47, 48]. The probabilistic genetic algorithm has also been used in a hybridized format with the deterministic gradient-based approach to optimize the total number and placement of wind turbines in a wind farm which is a close problem to the problem of our interest [4].

Siavashi et al., in their study, have compared utilization of genetic algorithm and particle swarm optimization and a hybrid of the two on water flooding optimization in which the hybrid optimization has showed excellence over the pure GA and pure PSO [2]. The genetic algorithm and finite difference gradient methods have also been used in conjunction with artificial neural networks and kriging to optimize the number, location and type of the wells in a petroleum reservoir [3, 49]. The genetic algorithm has also been hybridized with polytope search in order to optimize the injection strategy in a reservoir so as to optimize the oil production [50].

### 3 Methodology:

In order to implement the optimization process in test cases, we need both the optimization algorithms compiled in a computer platform and the case designed for this purpose. As for this to happen, we should start the test with simple cases. As usual for the initiation of any research on optimization, our basic simple case is a 2-dimensional reservoir. The structural view and a sample of permeability ensemble of this reservoir is represented in figure 3.1. The problem that is going to be solved is summarized in table 3.1, and a more comprehensive elaboration on the reservoir model characteristics can be found in appendix A.

#### Property

Reservoir Type	2-phase, black oil
Dimensions:	20 x 40 x 1
Grid Size:	100m x 100m x 20m
Top Depth:	2700 m
Initial Pressure:	2000 psi
Porosity:	0.2
Initial water saturation	0.25
Fault 1:	Between rows 13 and 14, Transmissibility: 0.2
Fault 2:	Between rows 27 and 28, Transmissibility: 0.05
Fetkovich Aquifer properties:	PI:5, Volume: 1.0E9, Eastern side of the field
Wells:	BHP controlled; set be 500 psi

Table 3. 1 The structural properties of the simulation case

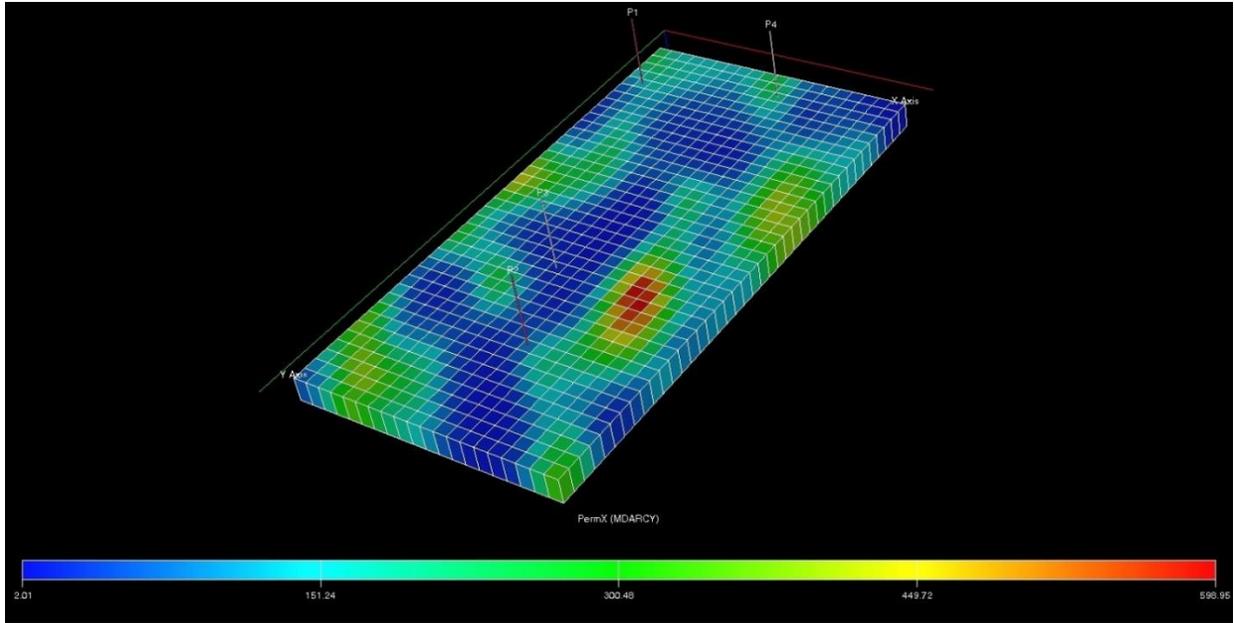


Figure 3. 1 3D representation of the field structure

### 3.1 Ensemble methods and Robust Optimization

One of the most difficult tasks in petroleum engineering is uncertainty quantification. One of the most popular methods to tackle this problem is using ensembles of models instead of a single deterministic model. In this research, as usual in the literature and industry, we put our uncertainty on the static models; the dynamic model which is our simulator is supposed to work accurately for now. The uncertain parameter in this case is assumed to be the permeability field; accordingly, an ensemble of realizations is acquired to account for this uncertainty. A sample of these permeability fields can be found in figure 3.2. All of these permeability fields are assumed equiprobable. This entails that when calculating the objective function of the optimization process the arithmetic mean of the NPVs of the ensemble is calculated and reported as the final result for the objective function of each case. This will be referred to as Robust Optimization (RO).

One of the properties of the objective function formed out of an ensemble of realizations is that it is smoother than any of them. This is a direct consequence of normalization. Any of the NPV functions of each member of the ensemble forms a potential surface in the multi-dimensional search space of the optimization problem. Adding up and normalization of two or several of these potential surfaces will result in alleviation of local exclusive features and more emphasis on the

common features of all ensemble members. This results in a smoother potential surface of the objective function. Accordingly, finding the optimized solution in an ensemble-based method becomes easier relative to single reality method, simply because the chance of getting stuck in the local minima decreases.

In order to prove the quality of our algorithm we try optimization of both single reality and ensemble-based reality models and provide the results.

The field that we are planning to optimize, has two pre-drilled wells in the first year of field development plan. There is also a constraint in the number of wells that can be drilled in each year. In this case, it is supposed that we can drill only one well per year. There exists an aquifer in the eastern side of the field so that it supports the pressure drop of field due to production. Problem is defined by the number of the wells to be drilled in this field, their location, as well as the time of drilling the wells limited by the afore-mentioned constraint. All of the wells are producing at constant pressure. Since the time they are drilled, they start to produce at constant pressure. We are aiming at maximizing the NPV function for the whole project. Drilling any well has its own constant cost which is initially set to be 5 million dollars for drilling and completion of the well. The objective function for this problem is the Net Present Value of the project. It is calculated based on the formula 3.1 [51]:

$$NPV(i, N) = \sum_{t=0}^N \frac{R_t}{(1+i)^t} \quad (3.1)$$

In this formula,  $t$  denotes time of the cash flow,  $i$  is the discount rate, and  $R_t$  is the cash flow rate.

As for this problem, the parameters for the cash flow rate are oil price per barrel, water treatment cost per barrel, water injection cost per barrel, and well drilling cost per well.

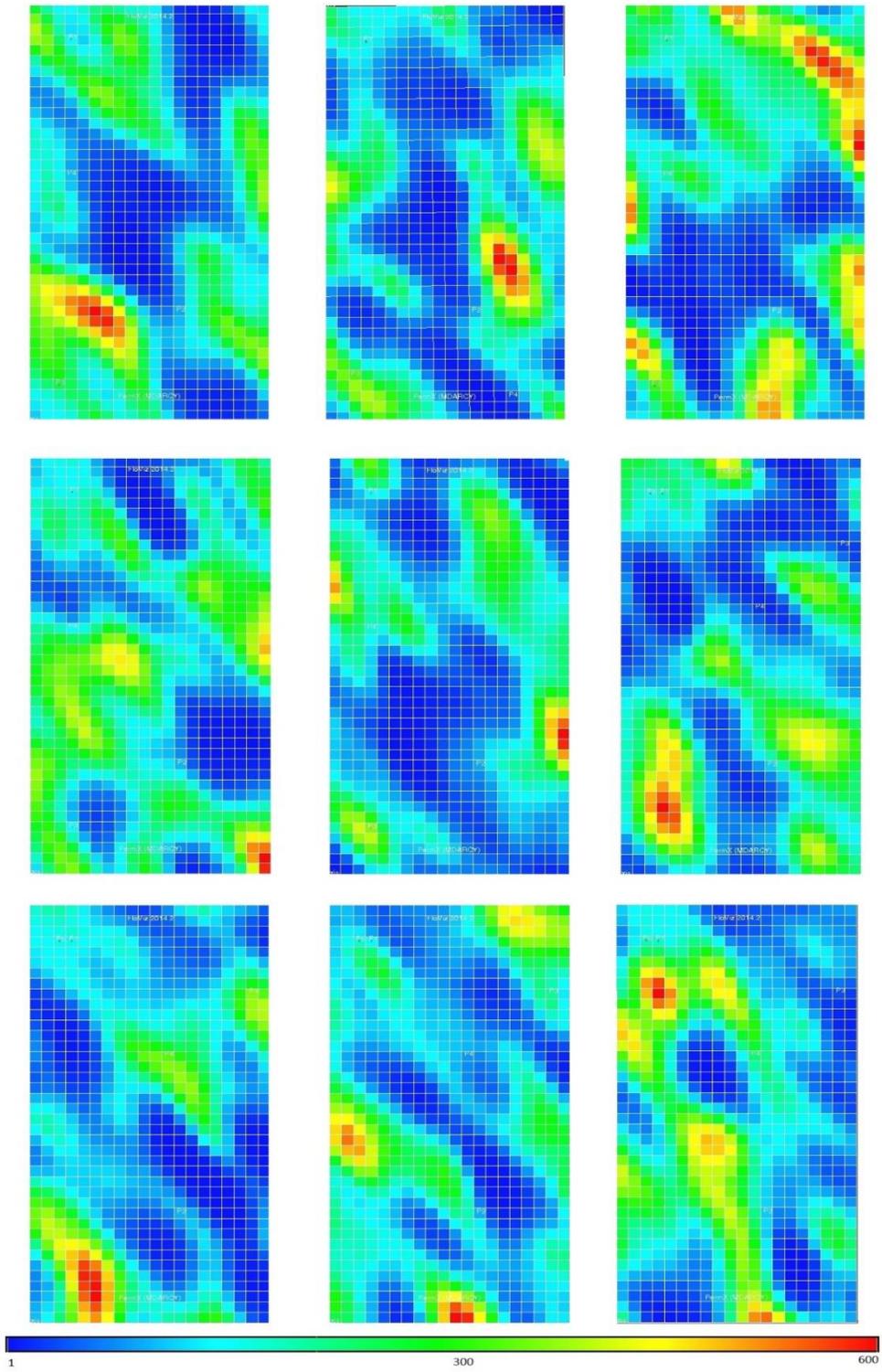


Figure 3. 2 Permeability fields of a sample ensemble of realizations

### 3.2 Optimization program structure:

The optimization program consists of two basic parts designing a test case and running the simulation on that. The simulations are run using commercial Schlumberger Eclipse E100. Designing the test cases on the other hand is done using a MATLAB program which is written to construct test cases based on the employed optimization technique. These optimization techniques are including genetic algorithm with several schemes, local search and metaheuristics, simulated annealing, and stochastic gradient descent.

The structure of the algorithm design code is depicted in figure 3.4. More comprehensive depictions of particular algorithms will follow in next sub-chapters.

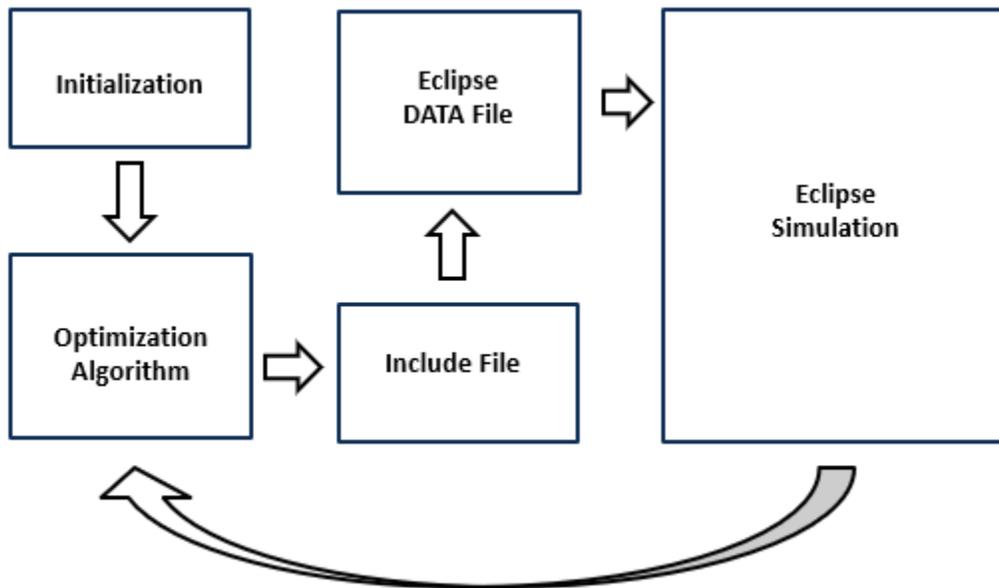


Figure 3. 3 Basic structure of the optimization algorithm

In the initialization process, eclipse files, proper population of initial engineering guess for beginning of the optimization process, optimization parameters setup, and several other parameters are decided and set.

Then the output of this section is fed to the optimization algorithm. There the algorithm modifies the include eclipse file which is the well plan of the field and using that the Eclipse DATA file is ready to be run in the Eclipse simulator.

The DATA file is finally run in Eclipse simulator and the results of the run are fed back to the optimizer so as to calculate the objective function and decide on the next run of the algorithm or convergence criteria.

### 3.2.1 Genetic Algorithm:

In this algorithm initially a population of initial guess which can be both random and meaningful is generated. The structure of the genome is designed in a completely discrete manner. For any of the possible scenarios we would either drill a well or not, the location of the well can be a completely discrete encoded number which denotes the number of the grid cell at whose center the well is drilled. In this case we have used a column following row order for the grid numbering. The drilling schedule will simply be the order of the wells in the genome. Consider the following structure for example:

389	1	47	1	221	0	180	1	98	0
-----	---	----	---	-----	---	-----	---	----	---

In the above-mentioned 5-year drilling schedule, any of the integer codes in odd cells denotes location of the grid cell at which the well is drilled and the binary codes at the even cells shows whether that well is drilled or not. The size of the genome simply represents the possible drilling period at the end of which the Net Present Value will be calculated.

After designing the genome, we should also design a proper evolutionary algorithm for the designed genome. In practice a genetic algorithm consists of two main parts, both of which are inspired by the nature: crossover and mutation.

Crossover, as mentioned in the previous chapter, consists of choosing two members of the population as parents and getting their genomes for the breeding process. In this process a location in the genome is chosen and the first part of the genomes are retained while the second part is exchanged. As for this research, the location of the genome cut is chosen after the binary code, as other than that it would be meaningless; however, it may add to the exploration capability of the code. The rate of crossover which means what is the probability of crossover happening while the genome is traversed is set to be proportional to inverse of genome size which is a rule of thumb and sensitivity analysis would be carried out on it.

The mutation on the other hand is responsible for adding to the exploration capability of the code by a completely random change in the genome the rate at which this genome change happens also is a factor to be investigated.

Elitism is another technique in GA which is also considered in this research. Based on elitism some members of the code with the highest fitness or objective function always survive to the next generation regardless of the probability function and probabilistic approach of the algorithm. The ratio of the Elites that are chosen from one generation to the next is also a number to be changed and decided. With the increase in this ratio the exploitation capacity of the code will be boosted, but there will be a chance that the algorithm gets stuck in local optima. The optimum ratio for that should be decided based on the problem characteristics and via sensitivity analysis.

The fitness function and its calculations are among the core segments of the genetic algorithm. Initially the objective function itself was assumed to be the fitness function which literally means that the chance of any species to survive is proportional to its objective function. However, there is a problem with this approach. If the objective function potential hyper-surface is flat, the fitness function for both the worst and best solutions of the problem will be the same. This implies that in any upcoming generation of schemes the probability of survival of the best solution and the worst solution will be very close to each other. This will result in very low convergence speed. In an attempt to attack this issue, we modified the fitness function to be not the objective function itself but an increasing function of it. Two different functions were proposed in this study and the sensitivity analysis of them are also carried out.

The first function that is introduced keeps the linearity of the fitness function. In fact, it is a linear transform of this function to a space where the chance of the best solution to survive is K times the chance of the worst solution. Being an increasing function the order of the solutions is also maintained, which means that if scheme A has a higher objective function than scheme B, it will have a higher chance of survival in the new fitness function probability space as well. The formulation of this function is presented as equation 3.2.

$$f_n = f_o - \frac{K * Min(f_o) - Max(f_o)}{K - 1} \quad (3.2)$$

$f_n = new\ fitness\ function$

$f_o$  = old fitness function

$K$  = the ratio of the chance of the best solution to that of the worst

$Min$  = minimum of the fitness function among the most recent population

$Max$  = maximum of the fitness function among the most recent population

A final normalization of the function is required for the sum of probabilities to be equal to one. A representation of the comparative probability functions is presented in figure 3.4.

Another scheme that is also examined in this study uses exponential transform function to build the new fitness function. In this scheme the transform function gives a respectively higher chance to the solutions with higher objective function than those with lower. In other words, the probability density function is denser around the best solutions. This will result in more exploitation in the algorithm which has both its own positive and negative points which has to be balanced. A mathematical formulation of the function is represented in equation 3.3.

$$f_n = \exp\left(\frac{\ln(K) * (f_o - Min(f_o))}{Max(f_o) - Min(f_o)}\right) \quad (3.3)$$

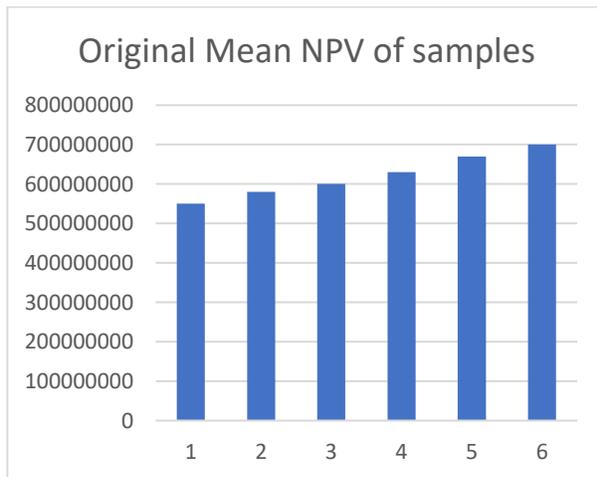
$f_n$  = new fitness function

$f_o$  = old fitness function

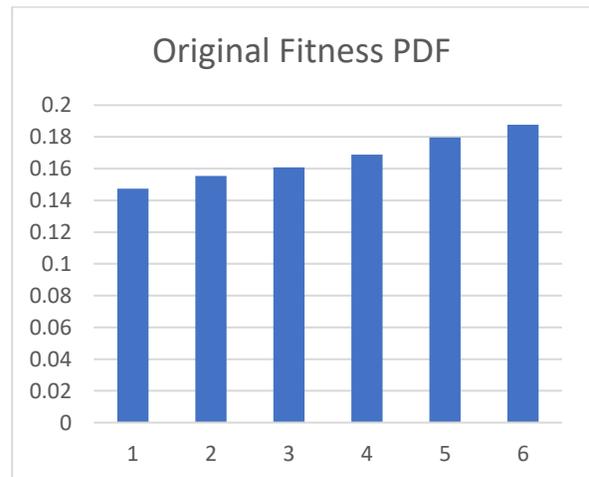
$K$  = the ratio of the chance of the best solution to that of the worst

$Min$  = minimum of the fitness function among the most recent population

$Max$  = maximum of the fitness function among the most recent population



(a)



(b)

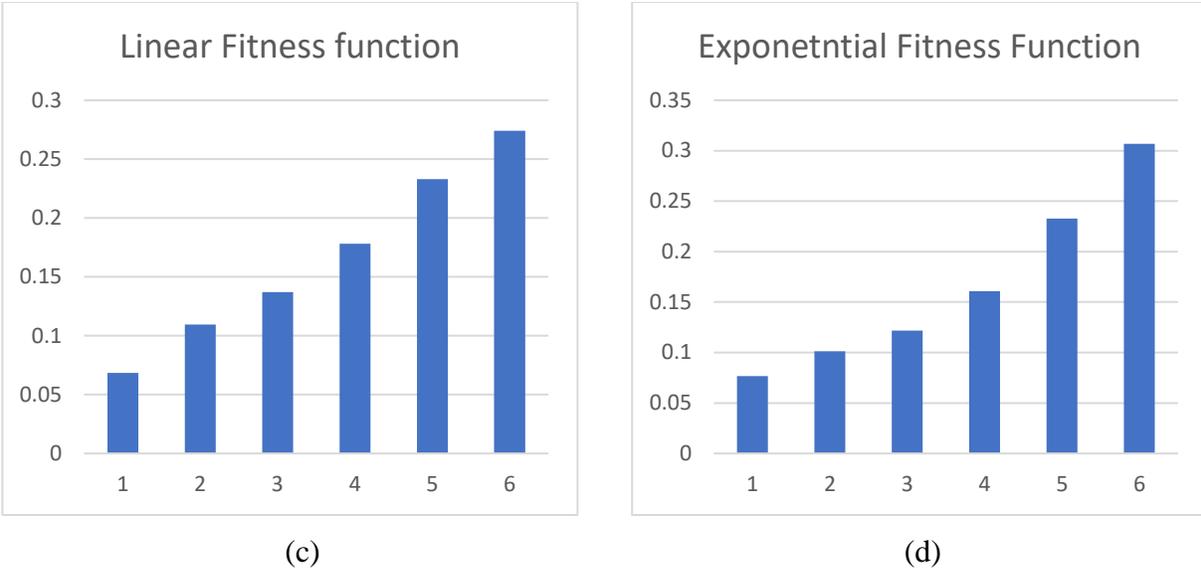


Figure 3. 4 Different formats of GA fitness function

As it can be seen in figure 3.4, the fitness function initially was proportional to the mean NPV of the ensemble for any of the drilling schedules (b). This will imply that in the example above (a), when we have cases with completely random location of the wells which generate NPV values of about 550 million dollars due to good quality of the reservoir, and some much better optimized schemes that produce about 700 million dollars of NPV, the probability of survival of the former would be 0.15 and that of the latter would be 0.18, which does not give a much higher privilege to the optimized solutions. Using this fitness function without any modification will result in very slow convergence speed.

On the other hand, we have proposed two different schemes, one linear (c) and one exponential (d), that will set the exploitation/exploration ratio to our preferred adaptive rate.

Another constraint that we have forced in the problem is the distance between the production wells. As for the production to be closer to optimal point, the distance of the wells is adjusted so that the drainage area of two production wells does not collide. Even though it is not a mathematically provable idea to keep the wells apart; however, this idea is implemented widely in industry, so we add this constraint to our list of constraints. Since we are in 2-dimensional discrete space in this problem we will define the distance of two wells with Manhattan distance, which can be defined as number of cells a rook in chess moves to reach its destinations times the dimension of each cell movement. The constraint on the spacing between the wells is the Manhattan distance between

any two wells has to be more than or equal to a certain value  $D$ , which is set to be a positive constraint so as to help us avoid any unnecessary calculations, in this study we avoided neighboring cells to have wells simultaneously by implementation of this method i.e.  $D=2$ .

The Genetic Algorithm used in this study can be defined as follows:

1. Generate a of meaningful ensemble of realizations
2. Generate an ensemble of initial guess or initial population of drilling scheme genotypes--  
This can be both done randomly be machine or continuously by engineer
3. Transform the genotypes to phenotypes and simulate for all the drilling schemes and their realizations using Schlumberger Eclipse E100
4. Calculate the mean NPV of all the realizations for any of the drilling schemes based on the Eclipse files
5. Calculate the fitness function for any of the phenotypes and make a lottery pot using this population and the elites list
6. Choose  $N$  pair of parents by lottery and intersect them to come up with a population of  $N$  children
7. Check if the children are fit to the self-appointed constraint otherwise substitute the unfit children using the lottery pot and new parents
8. Transform the children genotypes to phenotypes and simulate for all the drilling schemes and their realizations using Schlumberger Eclipse E100
9. Calculate the mean NPV of all the realizations for any of the drilling schemes based on the Eclipse files
10. Compare the new generation with the previous generations and if the convergence criteria are not satisfied go to 6
11. Report scheme with the highest mean NPV as the solution to the problem

A flow chart of this algorithm can be found in figure 3.5.

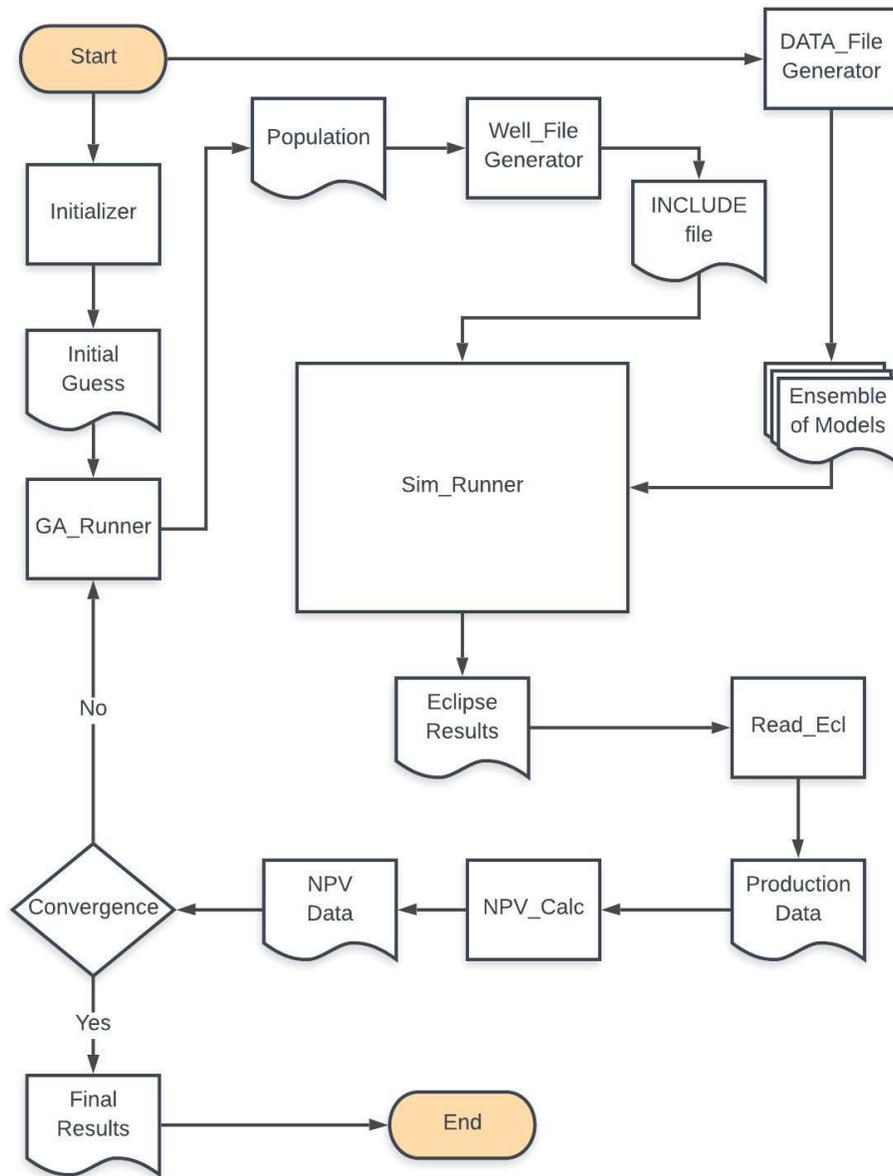


Figure 3. 5 Flow chart of the GA used in this study

The population size which is also a very critical parameter in the runtime of the code is to be decided. A prevailing number for that in the literature is the genome size [19]. As the genome size increases, with any unit of this increase, another dimension is added to the search space which makes the problem incomparably bigger but with adding one member to the population the computational cost will only increase linearly.

Another problem that is faced in robust optimization is that we should generate several realizations and simulate all of them. The objective function for the optimization algorithm will finally be the mean of the NPV or any other fitness function of the ensemble. This would imply that with increase in the size of the ensemble the computational cost will increase linearly. And due to the limited capacity of the accessed clusters this would mean that a normal robust optimization with an ensemble of size 10 will take 10 times more computation runtime than normal single-realization optimization.

### **3.2.2 Live Genetic Algorithm (LGA)**

As it can be simply recognized in the problem and its size, the use of clusters for this problem seems inevitable. One of the problems with the traditional sequential genetic algorithms is that the results of any full cycle of breeding, simulation, and objective function calculation should be determined at the end of any cycle for the next generation to be formed. This implies that if we have a cluster of 100 and our population size is 60 for example, in this case 60 of the cores are assigned to the computational costs pertaining to each of the population members and the other 40 will be out of use or a more complex algorithm should be implemented to use the whole 100 cores for different realizations in case of robust optimization. But even in this case, using traditional genetic algorithm, reaching 100 percent of computational power usage is impossible. In order to attack this problem, we design a new scheme in genetic algorithms called “Live Genetic Algorithm” (LGA) using which we can attain the 100 percent usage of computational cluster capacity.

The LGA as the GA itself is inspired by the nature. As we do not have any discrete form of generations in the nature, which means that second generation of some species will not stay for the first generation to end their lifetime so that they can start their lifespan. Different generations and genome lines coexist together but have a certain life expectancy. This is exactly what we have mimicked to design LGA. Starting with a population of genomes, we specify a lifetime “L” for each of the population members. This means that any of the population members can take part in L lotteries to be one of the parents. After pairing if the child was fit to the environment, which is the set of constraints, the child goes for simulation in all of its realizations and the necessary data about each of them in addition to its own genome is stored in a live list. So each of the cores can parallelly come to this list, choose a pair of parents from the last L members of the list, generate a

fit child genome, run the simulations of all of the realizations based on the phenotype built on the child's genome, and store the chief data at the end of the live list and update it. The algorithm and a schematic of the live GA can be seen below.

The Live Genetic Algorithm outline:

1. Generate an initial live list of drilling schemes of size L
2. Simulate the realizations of all the members in the initial population (live list)
3. Calculate the mean NPV for any of the population members based on the simulations
4. Calculate the fitness function for any of the members of the list
5. Add the last L members of the list to the lottery pot together with the elites list
6. If any core is free choose two parents from the lottery pot based on the probabilistic approach generate a child member
7. Simulate all the realizations based on the child scheme
8. Calculate the NPV of all the realizations of the child member
9. Calculate the mean NPV and the fitness function accordingly
10. Add the child member and the chief data of all of its realizations to the end of the list
11. Update the elites list
12. If the convergence criteria are not met go to 5
13. Report the top of elites as the optimized scheme of this algorithm

As it can be seen in figure 3.6 in any step the child is generated by selection of parents in lottery among the last L members of the list and an elite list, and after simulations and calculation of its mean NPV it is added to the end of the list.

First Member	Data for the first realization
	...
	Data for the $N_{th}$ realization
· · ·	
Last Member of Initial Generation	Data for the first realization
	...
	Data for the $N_{th}$ realization
· · ·	
The Youngest Member So Far	Data for the first realization
	...
	Data for the $N_{th}$ realization
· · ·	

Figure 3. 6 The population list in LGA

### 3.2.3 Simulated Annealing:

This method is widely used specially in the problems with complex search spaces. This is due to its probabilistic approach in direction of the movement. Thanks to this probabilistic approach, the algorithm becomes capable of dealing with the problem of local optima.

As for the problem of our interest, the technique should be modified for the discrete search space [52]. For this purpose, the same graph that will be introduced in the gradient-based search is constructed. The two algorithms differ in the point that in stochastic gradient descent method the new neighboring scheme that we try at any step is only accepted if it improves the objective function, which increases the risk of getting stuck in the local optima, while in simulated annealing the worse solution is also accepted with some probability. This probability decreases with time so that finally the method will be very close to stochastic gradient descent method.

In simulated annealing method, the probability of accepting the decrease in objective function has to be a function of the step at which the algorithm is. The functionality in equation 3.4 is proposed in this research for that purpose.

$$P(k) = P_0 \exp(-\alpha k) \quad (3.4)$$

*k = the step (an integer increasing one unit by every full set of simulations)*

*P(k) = probability of accepting a decrease in NPV in  $k_{th}$  step*

*P<sub>0</sub> = probability of accepting a decrease in NPV initially*

*α = the decay rate of probability*

The decay rate is to be adjusted based on how fast we want to avoid the moves towards the decreasing objective function. In order to quantify it we use the half-life terminology. If we want the probability of accepting the reduction in objective function to be halved every  $K_{th}$  step the formula 3.5 can be used to determine  $\alpha$ .

$$\alpha = \frac{\ln(2)}{K} \quad (3.5)$$

The  $\alpha$  in equation 3.5 should be determined in a way that the exploitation to exploration ratio be adjusted well for the problem.

The following is an algorithm proposed in this research for the simulated annealing.

1. Take one of the vertices as the initial guess
2. Update the probability of accepting reduced objective function— $P(k)$
3. Randomly choose either time or space as for search in this step (not equal chances)
4. Take one of the wells as for the trial in this step
5. Randomly choose one of the directions to which the cell can move (2 directions in time domain and 4 directions in space domain)
6. Simulate the neighboring scheme that is formed
7. Calculate the objective function and compare it with the objective function in the previous cell
8. Generate a random number R between 0 and 1 with uniform probability.

9. If the objective function has increased or  $R$  is smaller than  $P(k)$  move to the new cell
10. If the convergence criteria are not satisfied move to step 2
11. Report the final scheme and its NPV as the solution to the problem

### 3.2.4 Gradient-based optimizer:

As mentioned in the previous chapter, the optimizers of this category are trying reach the optimum solution by moving along the direction with the steepest dip. Even though there are many successful gradient-based techniques in the literature, most of them are not applicable in petroleum industry for a couple of major reasons. Firstly, the calculation of derivatives is not technically possible as we normally use commercial software for simulation of our study cases. Secondly, if want to calculate the derivative matrices in a numerical way, it will take a gigantic computational cost, as most of the time that optimization takes is due to simulation of the cases. Because of these two main reasons and some other minor ones, the industry tries to cope with the problem through use of stochastic gradient methods. In this subcategory, the searching move will not be in the direction with the steepest slope. Instead of that, the move will be in a random direction with our preferred sign of slope. This would be a very bad approximate of the best move direction, but in practice it has shown to be the most successful among the gradient-based techniques. A two-dimensional symbolic representation of this technique is shown in the figure3.7 [37].

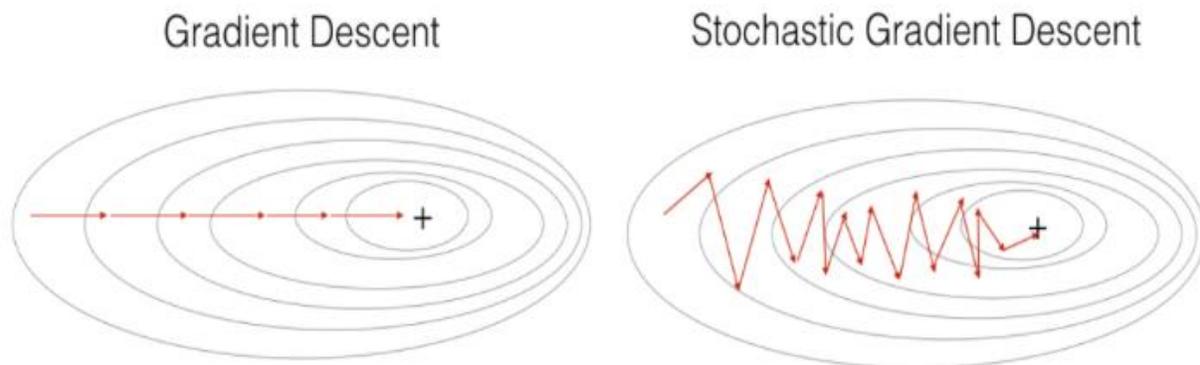


Figure 3. 7 A comparison of GD and SGD

When we are in the discrete search space, such as the discrete  $(2+1)*n$  dimensional search space that we have in this problem, where 2 stands for two spatial dimensions, 1 for time dimension, and  $n$  for the number of wells, we can modify the continuous search into a metaheuristic search. By

discrete, we mean that the well can be drilled only in the center of a grid, not at any point of interest which is also prevailing in industry, where the location of the wells are mainly discrete. The same goes for the timing of drilling.

In order to modify the problem into a metaheuristic search, we consider any of the schemes as node in a graph. Two nodes in this graph are connected to each other, if one can be switched to the other by a “minor” change. The word minor here refers to either changing the location of a well to one of its neighboring cells or changing the drilling time of a well one year forth or back. Depending on the number of wells that are tried to be optimized, a node in this graph is connected to between 3 to 36 other nodes. As in stochastic gradient descent method, which tries different directions around a point, in this metaheuristic search, the algorithm tries to calculate the mean NPV for a random neighboring node and if its objective function was greater than the previous cell, the algorithm leads itself to the new node, if not it will stay in the same node. If a node has an objective function higher than all of its neighbors, it can be assumed at least as a local maximum. In order to prove that this search will help us to obtain our optimum solution, we should prove that the graph that we have built for our problem is a connected graph, which means that there is a path between any pair of vertices in this graph. This assumption seems axiomatic, since if we want to transform the first node into the second one, we can simply move the first well in the former node cell by cell to its location in the latter. The same can be done with the second well and the rest of the wells one by one. In the next step we will move the wells in the time domain and the first node is converted to the second node through a path which passes through some other valid nodes. This simply proves our basic assumption. A hidden presumption that we took was that the number of the wells have to be the same in two schemes which has to be taken into account. A depiction of the two sample neighboring plans is shown in figure 3.8 (a) and 3.8 (d).

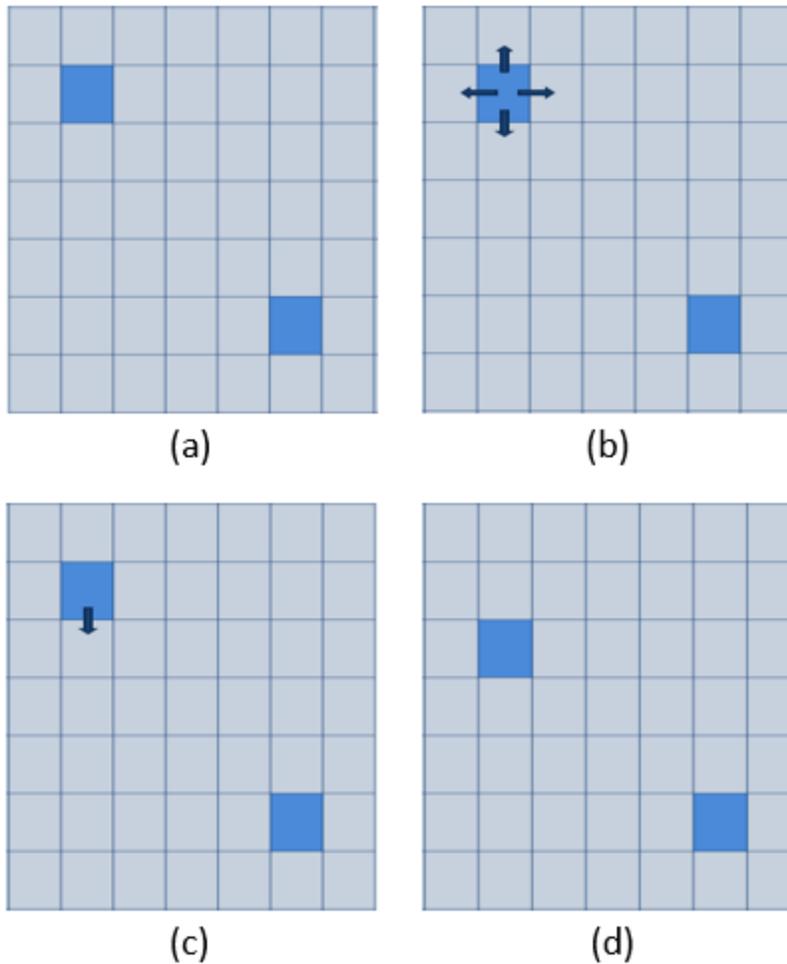


Figure 3. 8 SGD progress example

As it can be seen in figure 3.8, a well is to be drilled in the top left blue cell and another one in the bottom right blue one as shown in sub-figure (a), which together form the spatial location of the wells in drilling plan, a group of neighbors of this drilling plan are formed by moving the top left blue cell to one of its neighboring cells as shown in sub-figure (b), for example its neighbor to the bottom side, which finally forms a new drilling schedule (d) which is connected to the previous drilling schedule in the search graph.

When looking at the problem in large scale all of the aforementioned vertices in the problem form a gigantic graph. The algorithm does not traverse all the vertices in this graph though. Many of the are excluded from the problem search space by imposing the self-constraint boundaries. The algorithm only traverses a tiny proportion of the vertices in the graph but if the search space is not

very complex it without several local optima it can be simply proved that the algorithm is capable of finding the solution for us. Figure 3.9 is a two-dimensional depiction of how a graph would look like. The size of a graph is defined by the number of vertices in it. The graph in figure 3.8 is of size 22. The gigantic graph in our study is of an incomparable size.

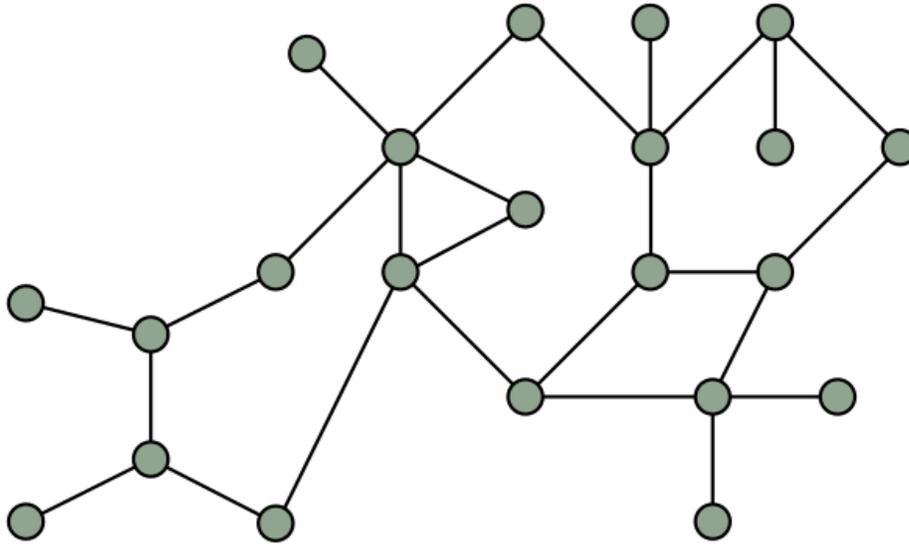


Figure 3. 9 A simplistic graph on metaheuristic neighborhood

The following is a representation of the metaheuristic search proposed in this research:

1. Take one of the vertices as the initial guess
2. Randomly choose either time or space as for search in this step (not equal chances)
3. Take one of the wells as for the trial in this step
4. Randomly choose one of the directions to which the cell can move (2 directions in time domain and 4 directions in space domain)
5. Simulate the neighboring scheme that is formed
6. Calculate the objective function and compare it with the objective function in the previous cell
7. If the objective function has increased move to the new cell
8. If the convergence criteria are not satisfied move to step 2
9. Report the final scheme and its NPV as the solution to the problem

### 3.2.5 Hybridizing the Algorithms:

As suggested by the academic literature and some tested results, genetic algorithm shows to be very successful for finding the locality of the optimized solution but becomes very slow in convergence. On the other hand, gradient descent and simulated annealing methods are very robust in finding the local optima. Accordingly, the sequential usage of these algorithms seems natural. In doing so, based on the afore-mentioned algorithms in this chapter, a hybridized algorithm is devised in which, initially, an initial guess is found using the Genetic Algorithm, and after that, it is optimized using the Simulated Annealing and Stochastic Gradient Descent Algorithms respectively. Or in other words, the algorithms work sequentially to solve the optimization problem. The optimized solution from the genetic algorithm is fed to simulated annealing to be optimized, and consequently the solution of this algorithm is the initial guess for the stochastic gradient descent method to be optimized. The final solution would then be the optimized solution which is generated after all the 3 steps (GA-SA-SGD). Other schemes were also tried which are including the sequential implementation of genetic algorithm and simulated annealing (GA-SA), and sequential implementation of genetic algorithm and stochastic gradient descent method (GA-SGD). The algorithms that are finally used for optimization are including pure GA, pure SA, pure SGD, hybrid GA-SA, hybrid GA-SGD, and fully Hybrid GA-SA-SGD. In all the mentioned algorithms a sequential implementation of the algorithms from left to right is performed. A Flow chart of the fully Hybridized GA-SA-SGD Algorithm which is the most complex among all is as depicted in figure 3.10. Since transformation of simulated annealing to stochastic gradient descent method entails only the change of probability of acceptance of decrease in the objective function to zero, they are shown in a same block in the flow chart.

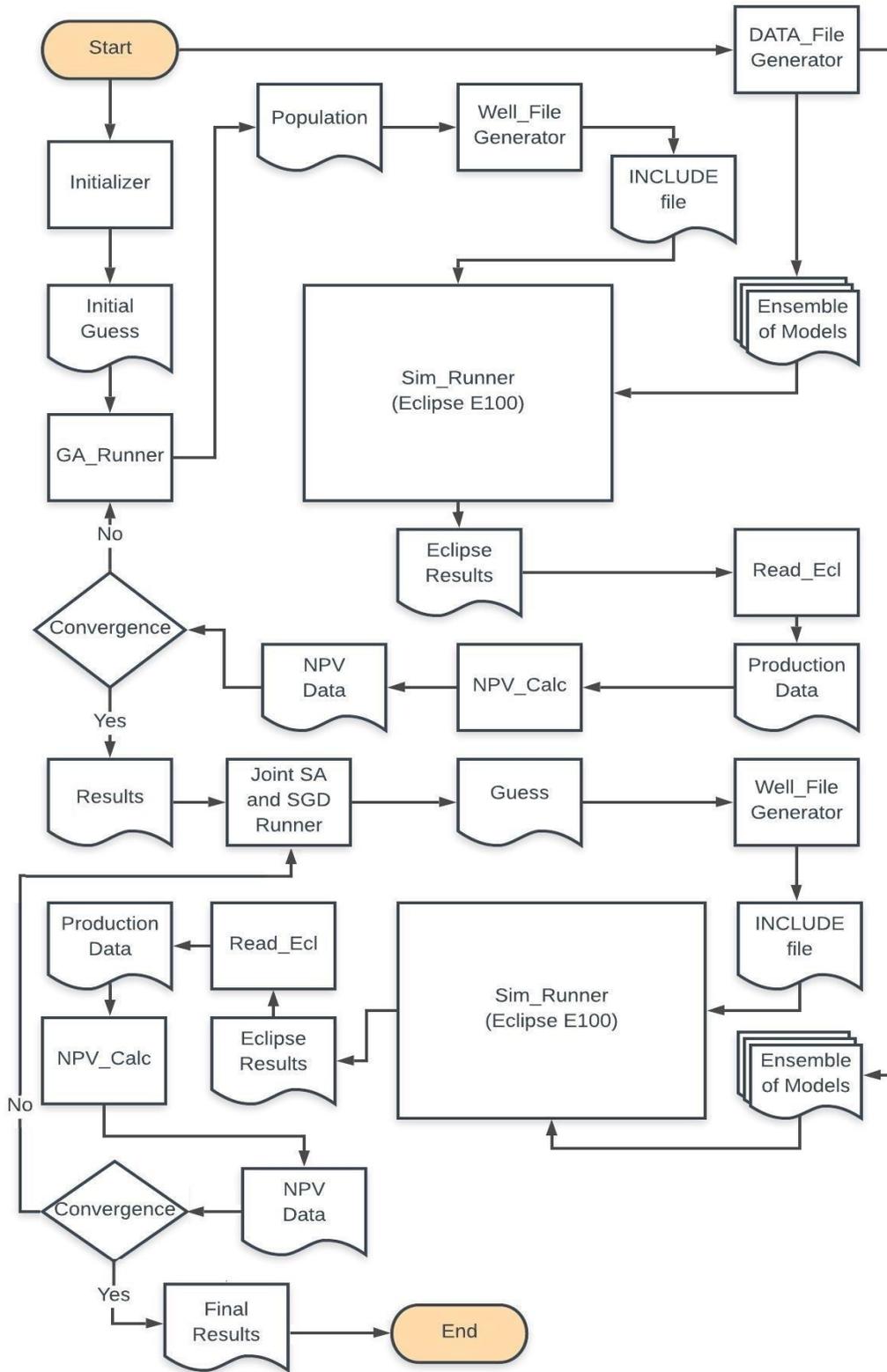


Figure 3. 10 Hybrid algorithm flow chart

## **4 Results and discussion:**

In this chapter the results and achievements obtained in this study will be presented. For any of the algorithms that we have used in this study, there is an optimized version which is suitable for solving the problem described in the previous chapter. Any of the parameters involved in these algorithms are optimized using sensitivity analysis, meaning that the best performance obtained through running the algorithm using several values for the parameter is chosen and utilized in the final hybridized algorithm.

### **4.1 Sensitivity analysis:**

#### **4.1.1 Sensitivity analysis for the genetic algorithm**

There are several parameters in the genetic algorithm that can be tuned for the algorithm to work better, among which fitness function design parameters, crossover probability, mutation ratio, elitism number, and population size are of more importance. We will go through them one by one.

##### **4.1.1.1 Linear fitness function format**

As elaborated in the previous chapter, two formats of manipulation of the fitness function are presented in this research. And for each of them the maximum to minimum chance ratio is a crucial parameter in the algorithm. We would call this parameter “the inflation rate”. For the linear scheme, various values for the inflation rate are tried on a test problem. Based on that, the mean of NPV of population at any generation, maximum value of the NPV achieved at any generation, and the convergence speed of the algorithm are recorded. Figures 4.1 and 4.2 present the mean and maximum of NPV achieved at any generation by different values of inflation rate.

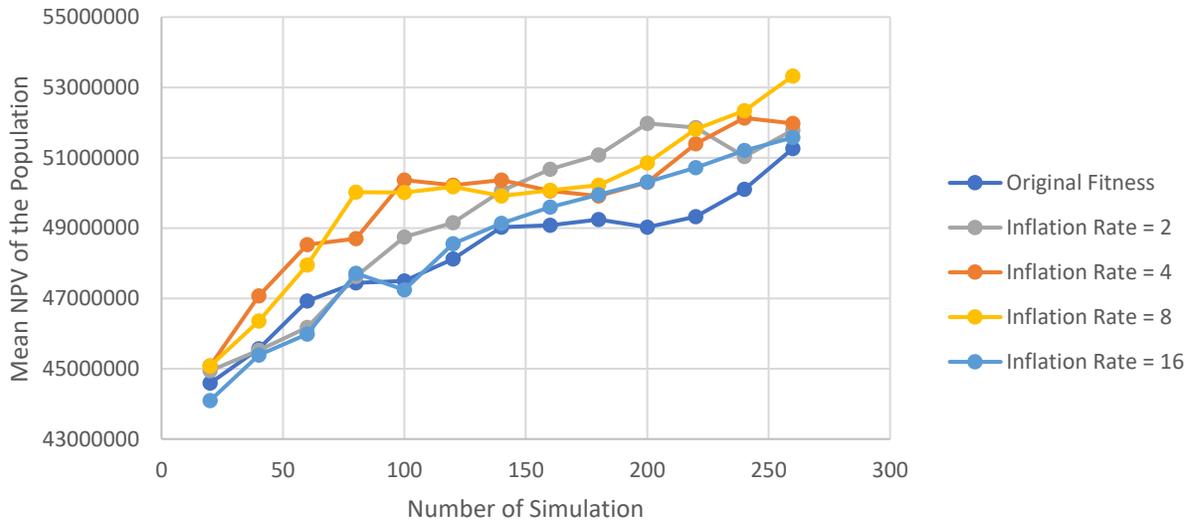


Figure 4. 1 Mean of NPV achieved by different inflation rates

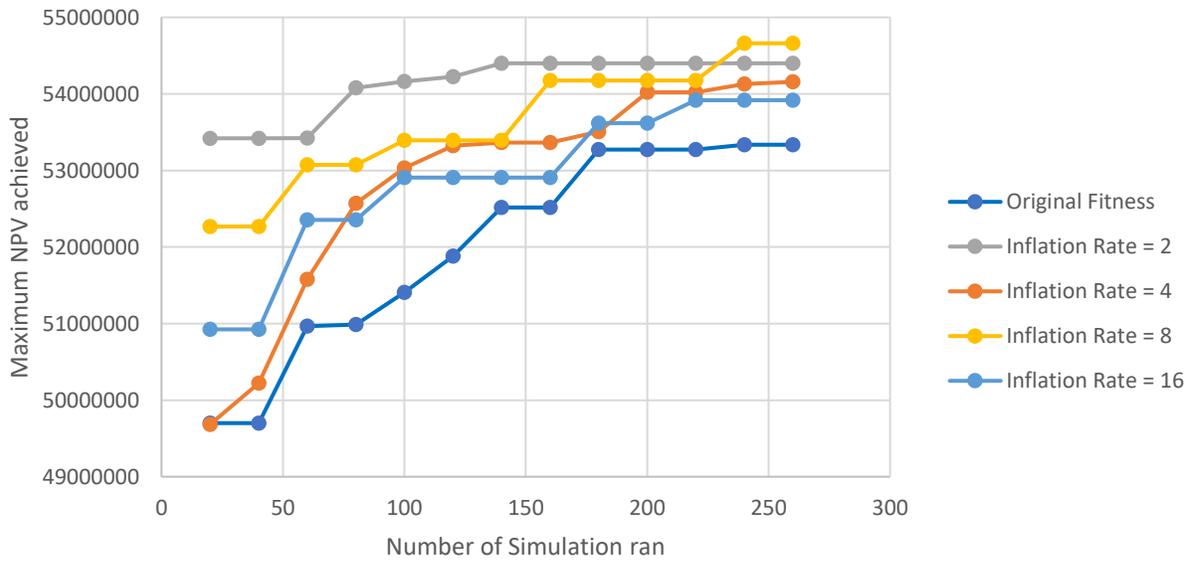


Figure 4. 2 Maximum NPV achieved by different inflation rates

As it can be seen in the figures 4.1 and 4.2, the best results are obtained by the inflation rate between 2 and 8. The higher values for inflation rate would result in hyper-emphasis on the best local results which have been achieved in that certain generation which decreases the exploration

capability of the algorithm. As it can be seen in this figure 4.2, after 8 generations, or 160 simulations, the improvements on the objective function slows down drastically and the mean of the NPV of the generations are moving towards the maximum NPV achieved which means that variance of the population decreases over time, and this is a sign of convergence of the genetic algorithm to its fittest members.

#### 4.1.1.2 Exponential fitness function format

Another fitness function designed in this study is the exponential scheme of the fitness function transform. The chance of the fittest or best solution divided by the chance of the worst solution in any generation of the genetic algorithm is, similarly, called the inflation rate. As it can be seen in the figures 4.3 and 4.4, all the inflation rates that are examined improve the optimization convergence speed and show better results than the original fitness function. This can be due to high objective function values for all of the solutions which decreases the comparative chance of the best solutions relative to solutions with completely random location and drilling schedule of the wells.

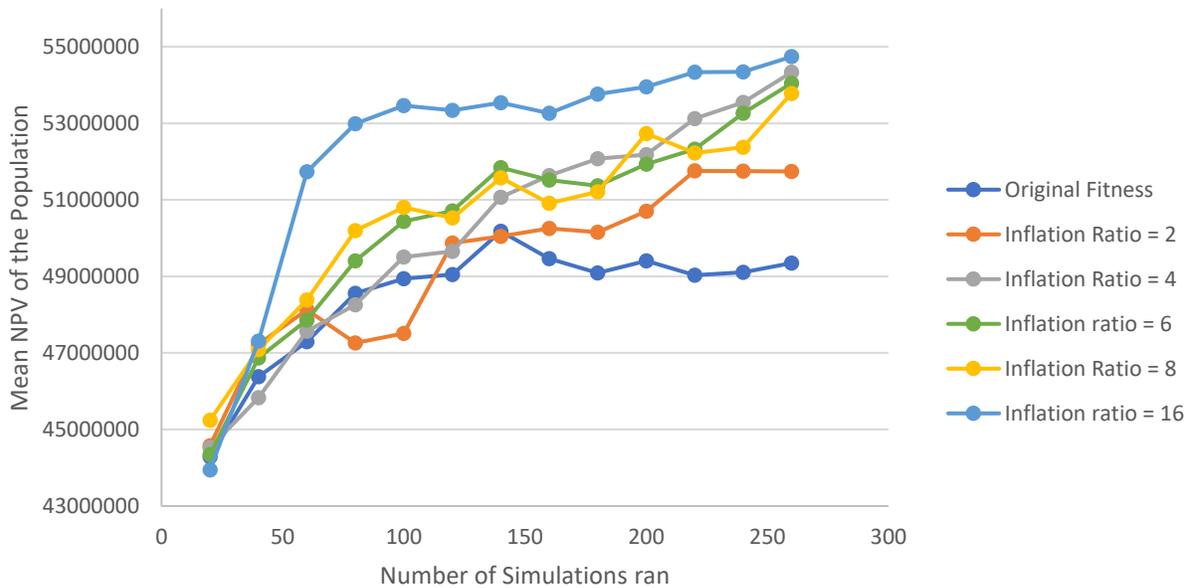


Figure 4. 3 mean of the NPV achieved by different inflation rates

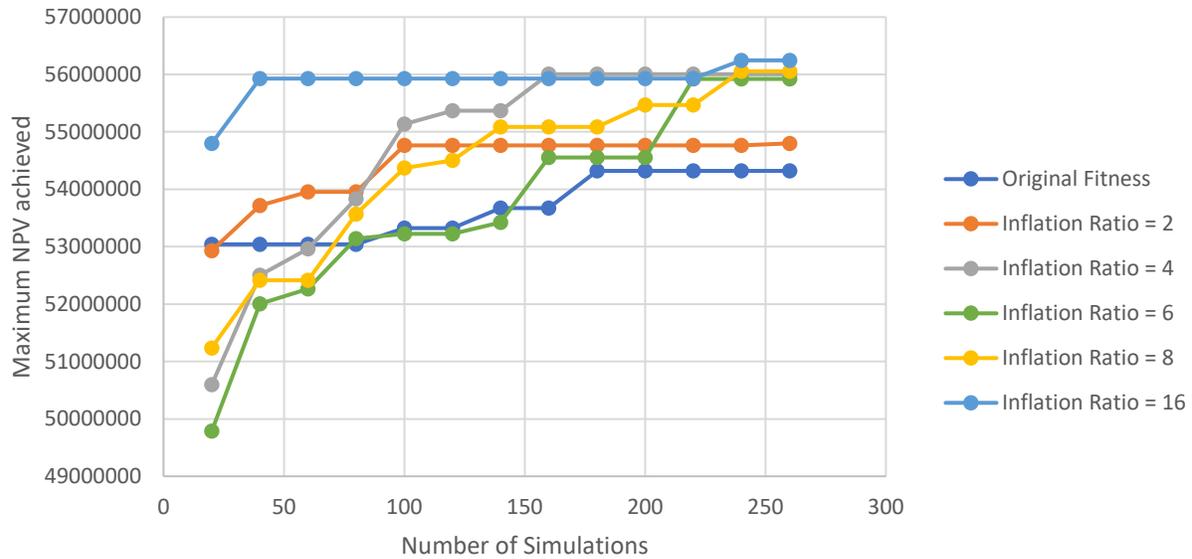


Figure 4. 4 Maximum NPV achieved by different inflation rates

The high values as 16 on the other hand may have higher achievements in the objective function but no improvements are seen in the objective function maximum for a long while and the good performance of this scheme may be due to good initial guess for this scheme to start up with. Inflation rates between 4 and 8 seem to show reasonable results in optimization.

#### 4.1.1.3 Crossover probability

Another parameter that needs to be adjusted is the crossover probability. By decrease in this parameter the rest of population members are generated using random mutation and elitism. As it can be seen in figures 4.5 and 4.6 except for the crossover probabilities as small as 70 percent which reduce the rest of population to only a random search in the search space or copying the previous good solutions, the rest of crossover probabilities show the same quality in optimization. Hence, we can conclude that if the crossover probability is big enough, more improvements do not affect the optimization quality. Accordingly, we choose crossover probability of 85 percent for the final algorithm as it has a slight superiority over other examined options.

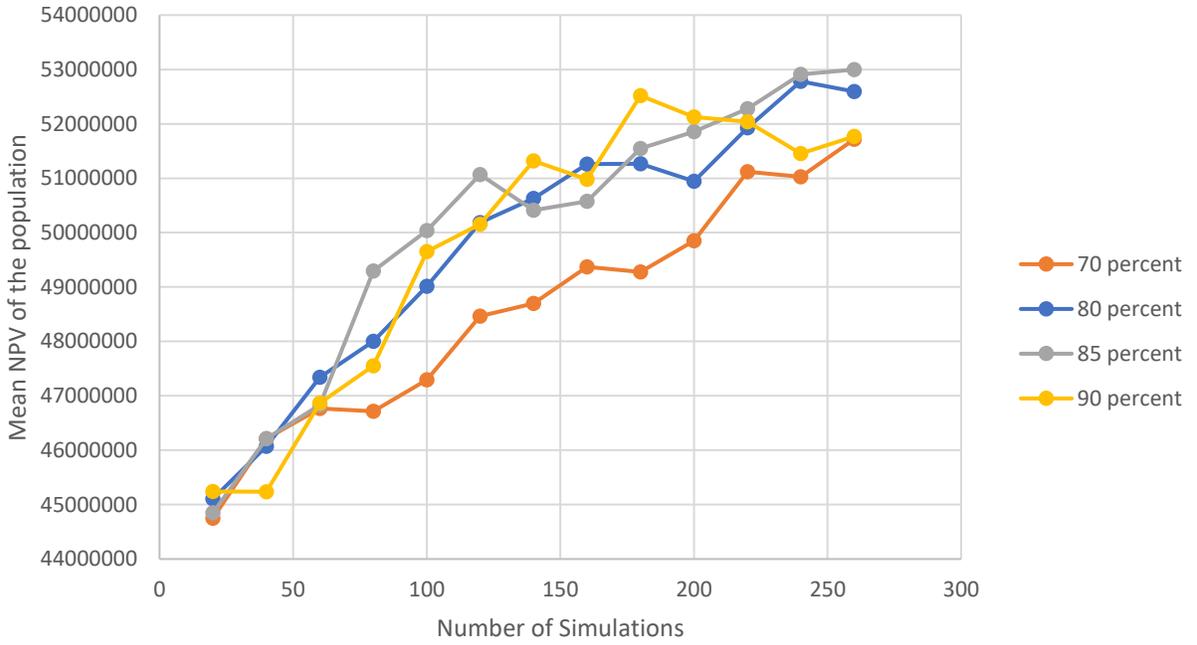


Figure 4. 5 Mean of the NPV of population for different crossover ratios

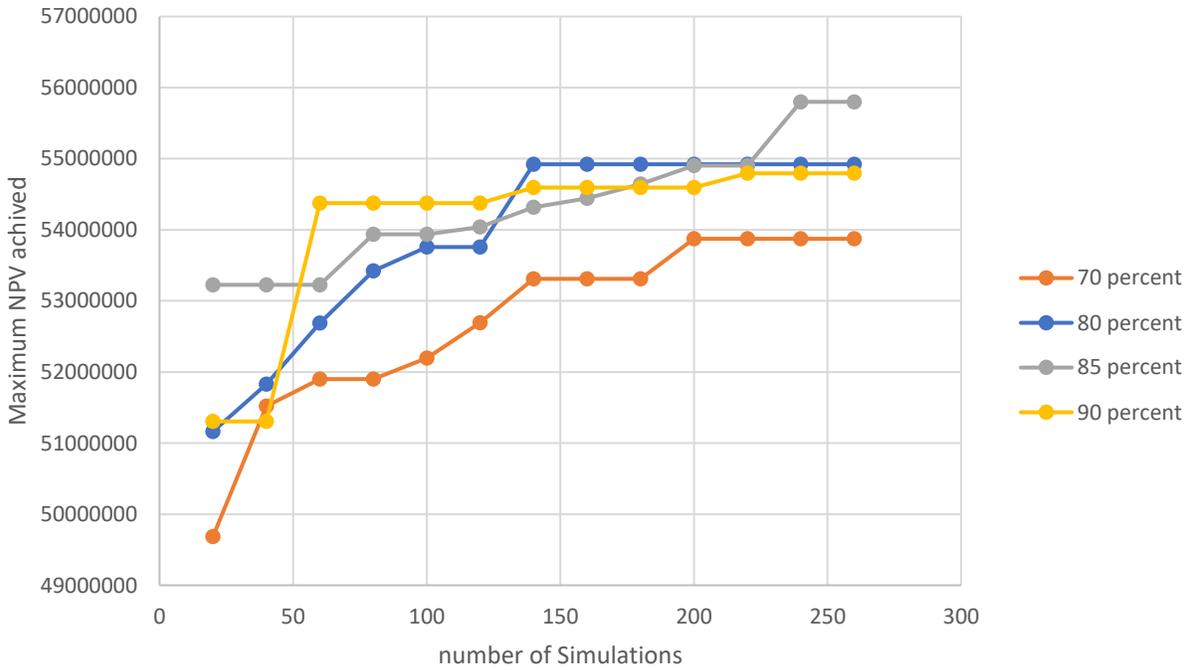


Figure 4. 6 Maximum NPV achieved by different crossover ratios

#### 4.1.1.4 Elitism rate

Elitism is used in order to keep the best solutions that are achieved under more investigation. Some of the best results are directly copied to the next generation without going through any parenting process. In order to assess the effect of this technique in our algorithm, 4 different schemes are investigated. Between 0 and 4 members of any generation are chosen to be elites. As our population size is 20 in this case, the numbers will refer to 0 to 20 percent elitism respectively.

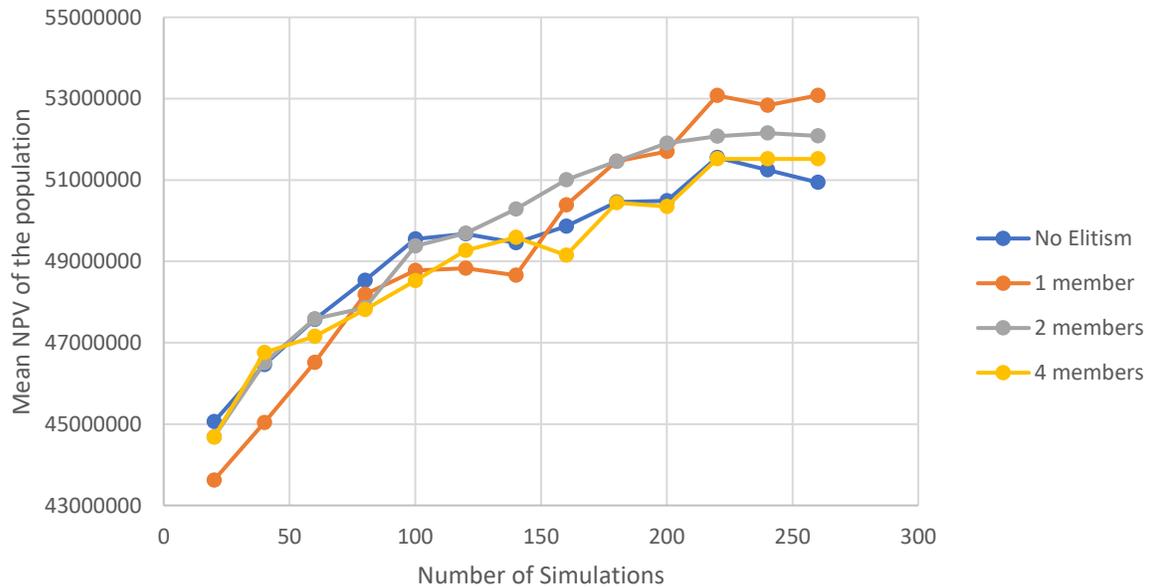


Figure 4. 7 Mean NPV achieved for different elitism schemes

Different schemes do not show drastically different results as it is presented in figure 4.7; however, as it is clearly seen in figure 4.8, the elitism rate of 20 percent which refers to the scheme with 4 elite members, has a very high standard deviation in different runs. This shows that this scheme lacks stability; converging sometimes to good and some other times to bad solutions based on the initial guess and random path that it goes during the optimization. If good results are achieved the scheme keeps the results in the population, on the other hand if bad results are obtained in previous steps the scheme keeps them in the process. Hence, high values for the elitism rate should be avoided. We choose this rate to be less than 10 percent.

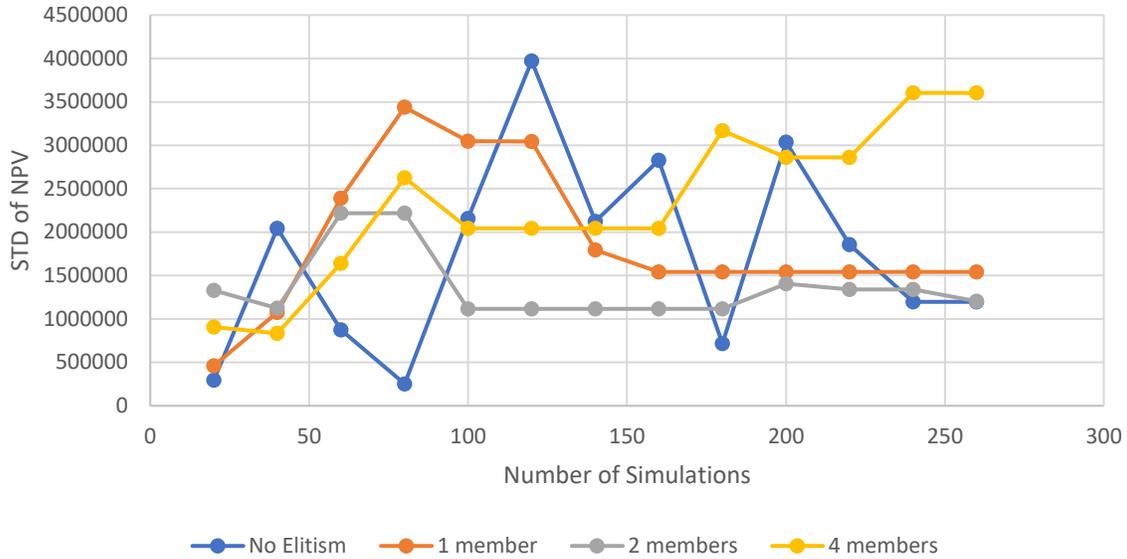


Figure 4. 8 Standard deviation of the maximum NPV achieved in several runs of the algorithm for different elitism schemes

#### 4.1.1.5 Mutation rate:

Another key parameter in the genetic algorithm is the mutation rate which expands our exploration area to a higher degree than the crossover technique does. This parameter should be tuned in a way that it does not convert the algorithm to a mere random search.

The scheme with no mutation, will be equal to only using crossover for the search. This would simply result in finding the optimized solution in a sub-space of our big search space and accordingly, low objective functions, this can clearly be seen in the figure 4.10. On the other hand, mutation rates as high as 20 percent reduce the algorithm capability by making it closer to a mere random search in the search space. Low objective function values are good evidence of this occurrence. Mutation rates of about 10 percent, which on average result in between one and two changes in each genome shows the best results which is in close to the literature norms.

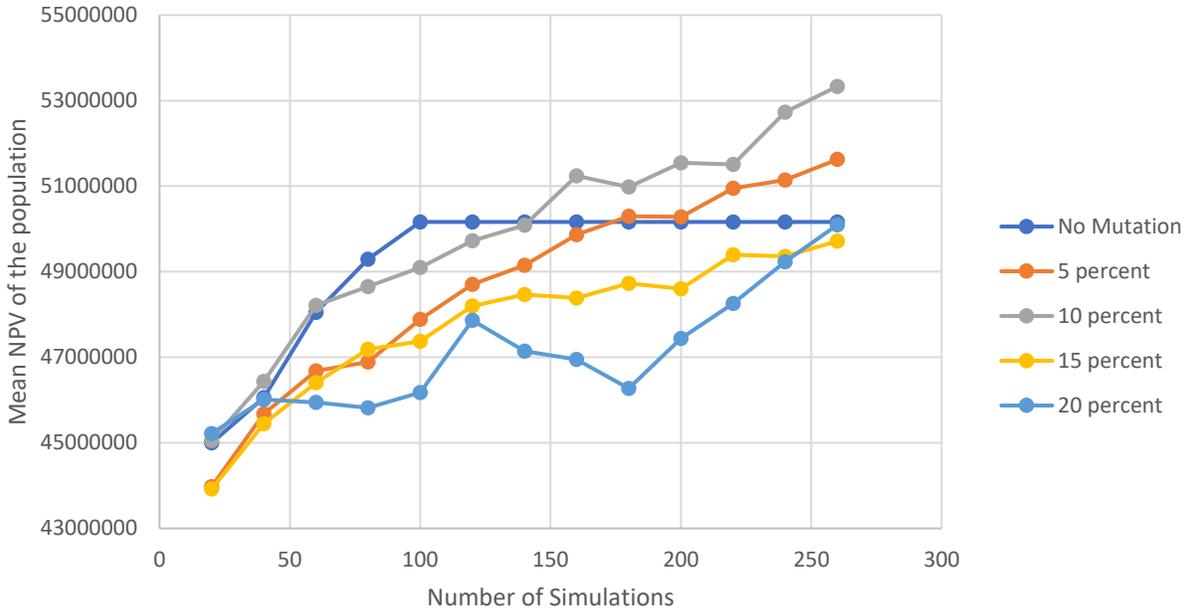


Figure 4. 9 Mean of the NPV of the population in different mutation ratios

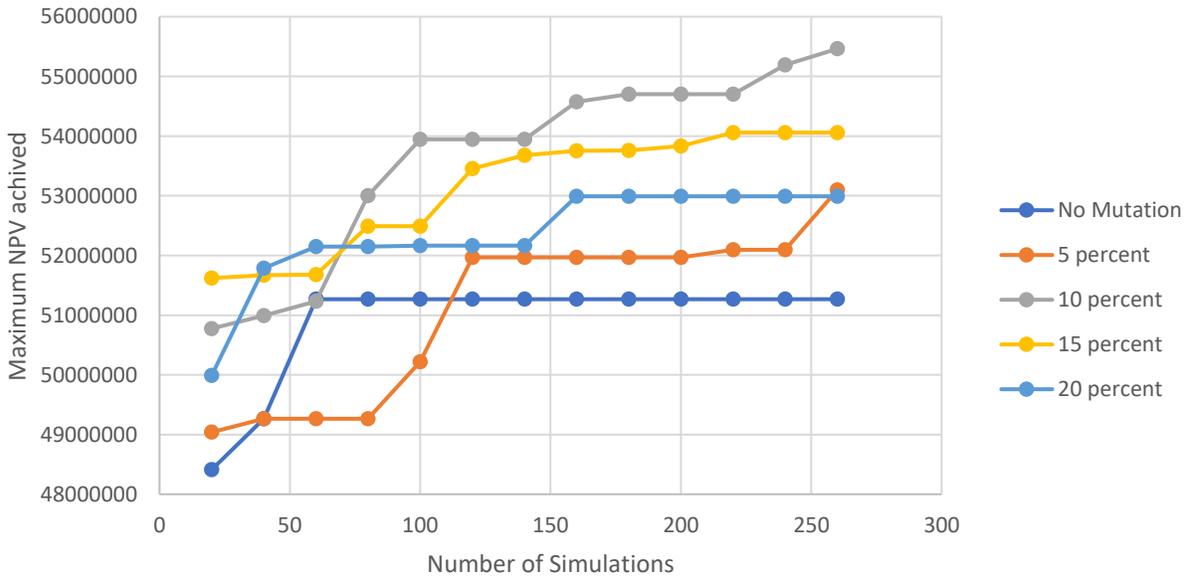


Figure 4. 10 maximum NPV achieved in different mutation rates

#### 4.1.1.6 Population size

Population size is one of the most important parameters to be tuned in the genetic algorithm. As it is clear, the bigger the population size, the better coverage of the search space is achieved, which results in better optimization scheme and higher probability of converging to the global maxima. However, this implies more computational cost. In this trade off we should decide the population size which does not converge to sub-optimal solutions and does not dictate a big computational burden simultaneously.

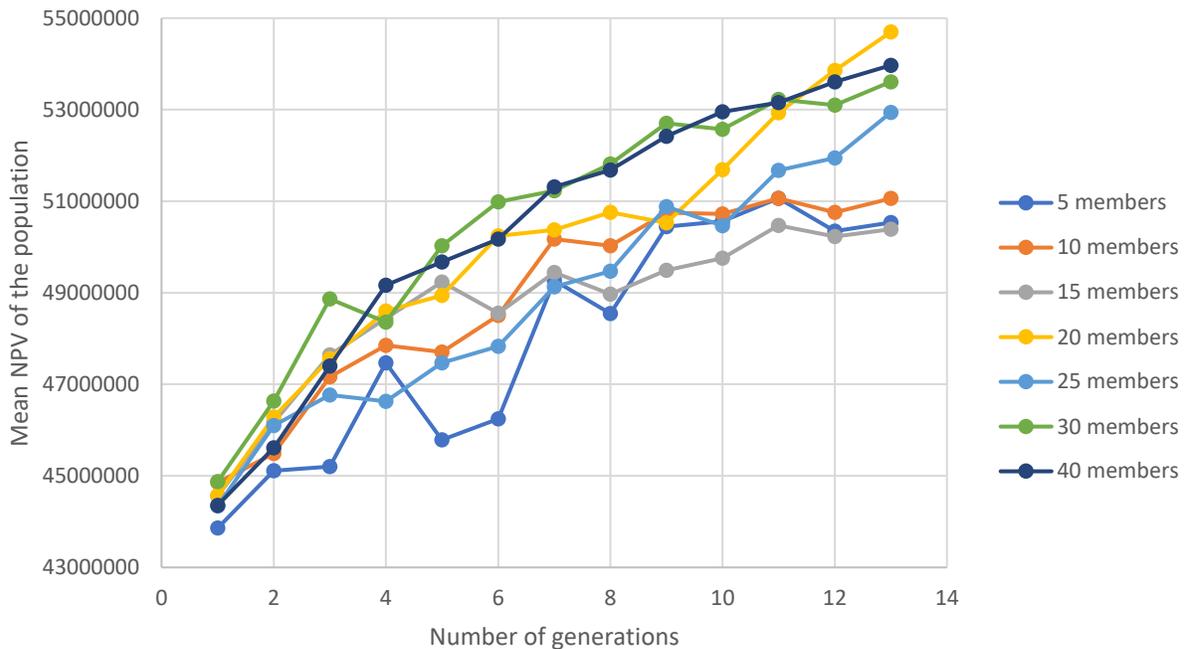


Figure 4. 11 Mean of the NPV of the population for different population sizes

As it can be seen in the figures 4.11 and 4.12 the population sizes of 5 to 10 are resulting in much lower values comparing to the rest. Population size of 25 shows a lower final maximum NPV than the rest of schemes with large population size. This can be a result of lower objective function values of the initial population that has been given as input to this algorithm. On the other hand, population size of 15 fails to converge to results as high as those of population size of 25 even though it has had an initial population with higher average NPV. For the population size of 20 to 40, the bigger the size, the better the results, but this improvement is very marginal. Accordingly, we choose 20 members for the finalized algorithm. As the genome size in this problem is 18, the

rule of thumb for population size in the literature also leads us to a population size the same as the genome size, where 20 is a good choice. A counter-intuitive point that can be seen in figure 4.11 is that the average NPV of the population in population size of 20 is higher than that of population size of 40. This may be due to 2 main reasons, firstly it may be due to convergence of all the population towards the best member of population, as it can be seen in figure 4.12 the best result obtained by the GA has not changed from 7<sup>th</sup> to 14<sup>th</sup> generation in population size of 20, this means that due to existence of elitism in the algorithm, the best solution is dragging other population members towards itself. This can also be seen in steep linear growth of its mean NPV in figure 4.11. Another reason maybe the stochastic nature of the genetic algorithm. As both the initial population and the process that the two population sizes have gone through differ, sometimes the genetic population size of 20 may have better results than population size of 40. However, the outcome of the genetic algorithm is not the mean of the NPV of the population, it is the maximum NPV of the population. This maximum NPV represents the mean of the NPV of the ensemble of realizations in the best scheme of the population. As it can be seen in figure 4.12, the maximum NPV achieved by population size of 40 is greater than the maximum NPV achieved by the population size of 20, which is how it should normally be.

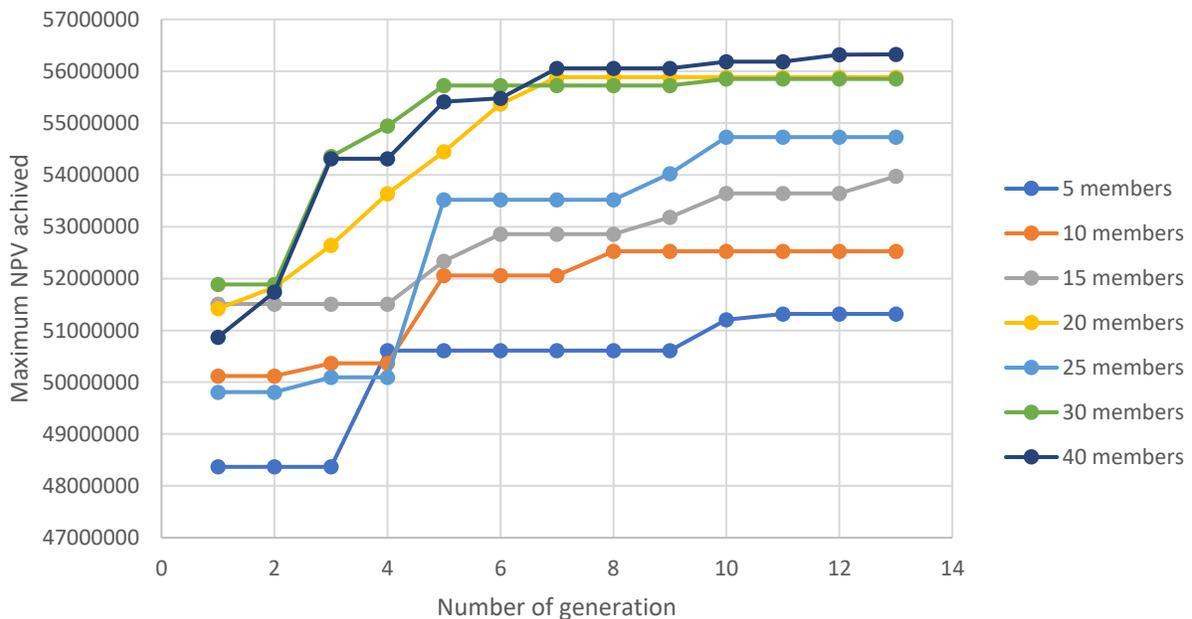


Figure 4. 12 Maximum NPV achieved for different population sizes

## 4.1.2 Simulated Annealing Sensitivity Analysis

In order to use this algorithm both individually and hybridized with other algorithms, it should be tuned for solving this problem. The quality of implementation of this algorithm highly relies on two parameters; namely, initial chance of accepting reduction in objective function and its decay rate. Two experiments have been conducted to see the effect of these parameters on the quality of the algorithm. The value of objective function obtained at the end of these algorithms are aimed to be maximized.

### 4.1.2.1 Initial acceptance chance

As in classical simulated annealing, the moves towards increase in the objective function are always accepted by the algorithm. Additionally, in order to add to the exploration capability of the algorithm, the reductions in the objective function are also accepted by a probability which decreases exponentially. The sensitivity of the final NPV to the initial chance in the first move is investigated and the results are provided in figure 4.13.

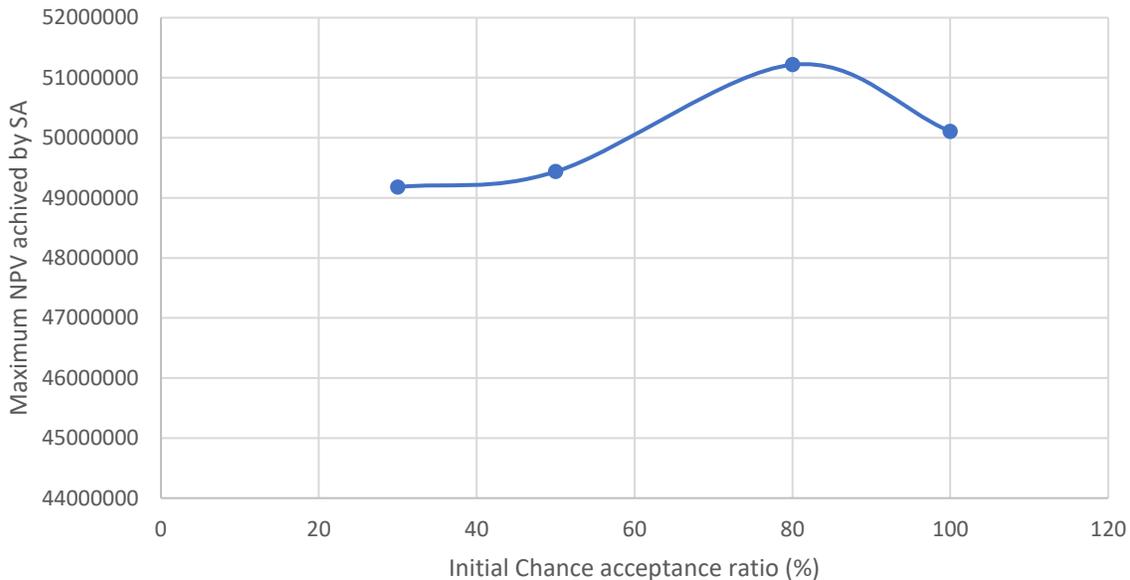


Figure 4. 13 Maximum NPV achieved for different acceptance chance ratio

Low chances as 30 percent after a few steps reduce this algorithm to mere gradient descent method, while taking the same number of tries, starting with probabilities as high as 100 percent will reduce the algorithm to mere random search in the search space. The Initial chance between 70 and 80

percent shows to be the most successful among the initial chances. Accordingly, we choose our optimized initial chance to be 70 percent.

#### 4.1.2.2 Decay rate

The rate of reduction in the acceptance chance of decrease in the objective function is called the decay rate. In order to make it more sensible, we use another term called the half-life. Half-life simply refers to the number of steps at which the chance of acceptance of reduction in objective function reduces to its half. Several values have been tried and the best of them is chosen in the rest of experiments onward.

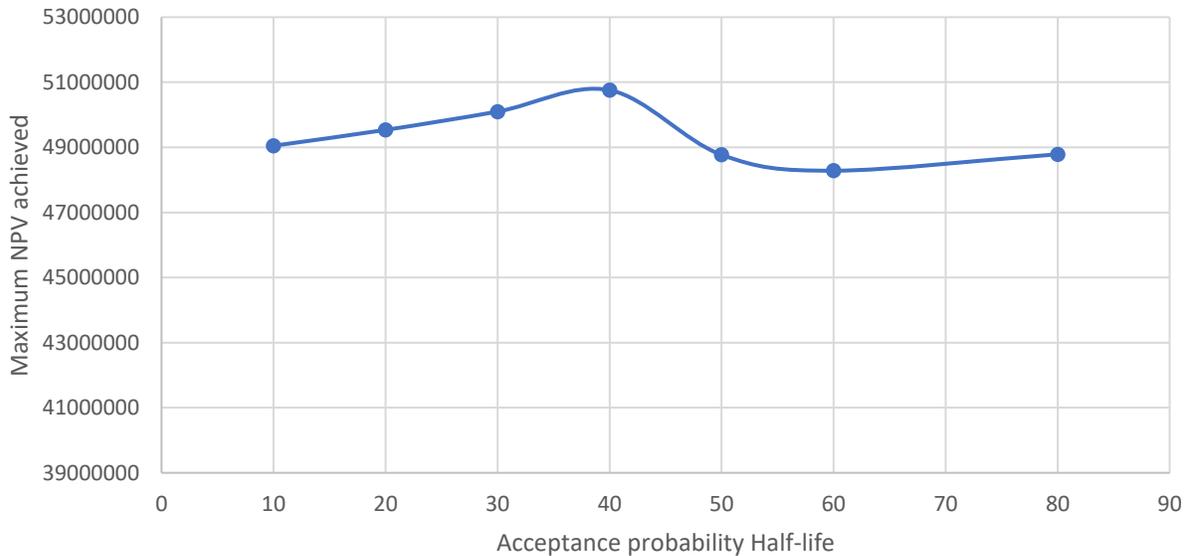


Figure 4. 14 Maximum NPV achieved for different acceptance probability half-lives

Figure 4.14 shows small half-lives as 10 after only 30 steps will result in acceptance chance less than 1/8 of its initial value which simply reduces the algorithm to gradient descent method very fast. On the other hand, half-lives as high as 80 will result in mere random search in the search space for a long while which is a waste of computational power. We choose the half-life to be 40 which shows to be the most successful in the experiment above.

## 4.2 Solved Problems

After implementation, hybridization, and tuning the algorithms using sensitivity analysis, we try to solve some optimization problems on the field using the hybridized algorithms and compare them with pure individual algorithms. As mentioned in the previous chapter, in order to prove the quality of the algorithm we provide example problems both on ensemble and deterministic models, which are more difficult to optimize.

### 4.2.1 Primary problem

In this problem the aim is optimization of number, location, and drilling time of the wells in the single-realization field introduced in the previous chapter, in which two pre-drilled wells are planned. A summary of problem definition can be found in table 4.1.

#### Problem Definition

Single Realization

Objective Function:	NPV
Field Optimization time:	8 years
Well Drilling Cost	3 MM USD per well
Oil Price	50 USD per barrel
Water production cost	6 USD per barrel
Discount Rate	8 percent
Pre-drilled well 1 location:	Cell 79
Pre-drilled well 2 location:	Cell 447
Number wells to be drilled:	?
Location of them	?
Timing of their drilling	?

#### Pre-drilled wells location

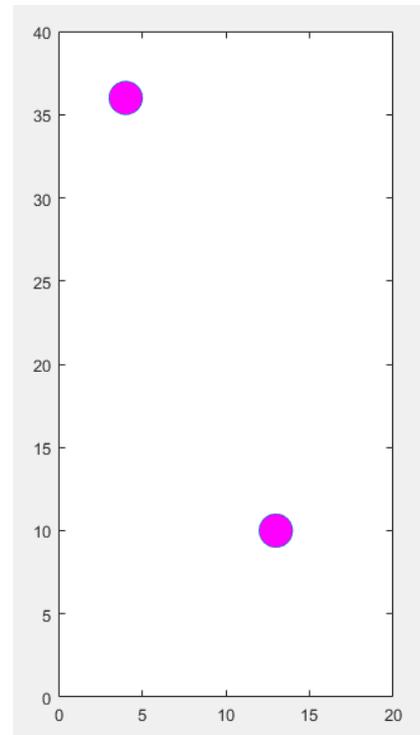


Table 4. 1 Primary problem definition

As it can be seen in figure 4.15, the hybrid algorithm has proved to be the most successful among all the listed techniques. Simulated annealing and stochastic gradient descent method show the

poorest performance by getting stuck in the local minima. Among the rest of algorithms, GA followed by stochastic gradient descent is closest in performance to the hybrid method.

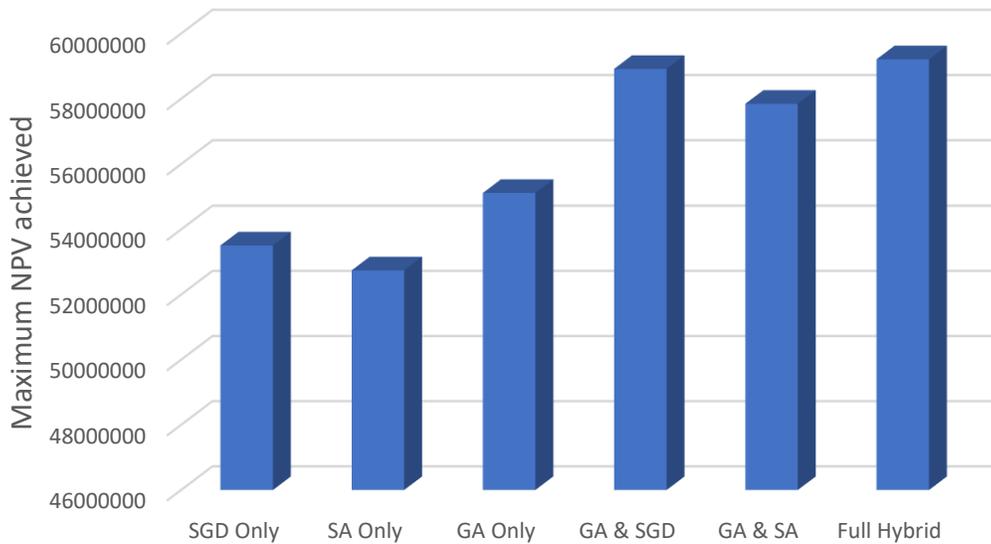


Figure 4. 15 Maximum NPV in different optimization algorithms for primary problem

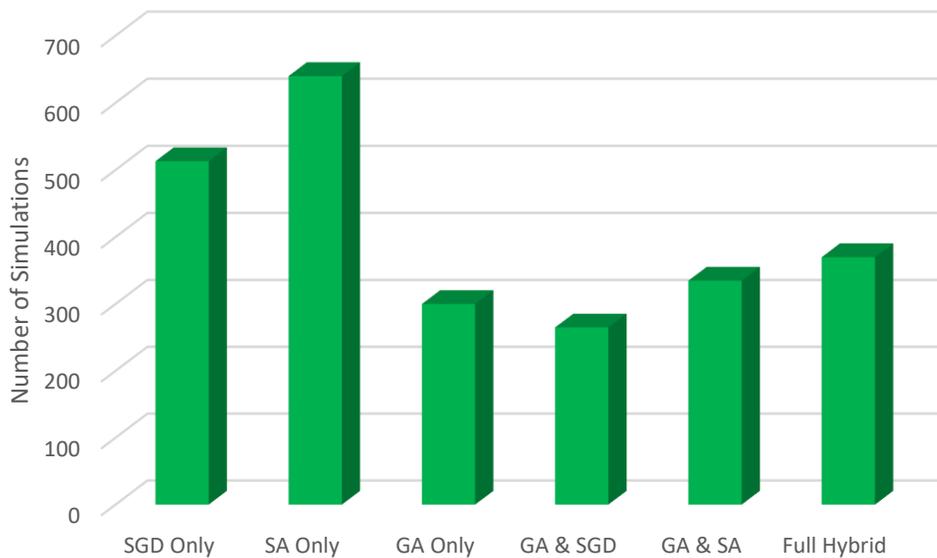


Figure 4. 16 Simulations ran in different optimization algorithms for primary problem

As in can be seen in figure 4.16, the computational cost of the SGD and SA for this problem is the hugest. The reason for this to happen is that due to getting stuck in the local optima, several initial

guesses should be tried for this problem to be tackled; however, these algorithms do not show comparable results to the rest of algorithms. The pure GA is let to explore its search twice the GA which is implemented and fed to the other algorithms, but still it has failed to provide results as good as the hybrid algorithms. The GA followed SGD results in both better solution in terms of the final NPV and lower computational cost compared to the GA. The Fully hybrid method which has the highest computational cost among the GA involved algorithms, also shows the best results in the final NPV; the big computational cost of this algorithm, accordingly, is a sign of its good exploration of the search space.

The solution that the hybrid algorithm gives is summarized in table 4.2. The two wells are proposed to be drilled in the western side of the field, which is in good harmony with intuition since there is an active aquifer in the eastern side of the field. The number of the wells is also tuned well, as one fewer well would have resulted in less production of oil and one more would have imposed additional 3 million dollars.

### Problem Solution

Final NPV:	59.2 MM USD
Number of additional wells to be drilled:	2
Pre-drilled well 1 location:	Cell 79
Pre-drilled well 2 location:	Cell 447
Additional well 1 location:	Cell 18
Additional well 1 drilling time:	Year 2
Additional well 2 location:	Cell 113
Additional well 2 drilling time:	Year 3

Location of the wells

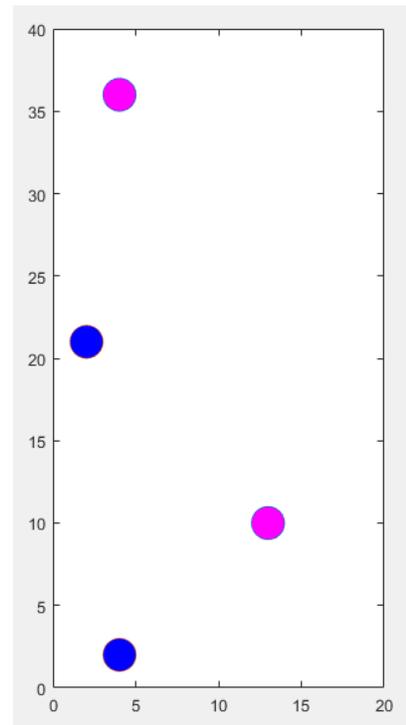


Table 4. 2 Solution to primary problem

### 4.2.2 Main Problem:

The main problem that we aimed to solve was the ensemble format of the previous problem in which the objective function changes to the mean of the NPV of the ensemble in any of the scenarios. A summary of the problem description is provided in table 4.3.

#### Problem Definition

Multiple Realizations	Ensemble size:5
Objective Function:	Mean NPV
Field Optimization time:	8 years
Well Drilling Cost	3 MM USD per well
Oil Price	45 USD per barrel
Water production cost	6 USD per barrel
Discount Rate	8 percent
Pre-drilled well 1 location:	Cell 79
Pre-drilled well 2 location:	Cell 447
Number wells to be drilled:	?
Location of them	?
Timing of their drilling	?

Pre-drilled wells location

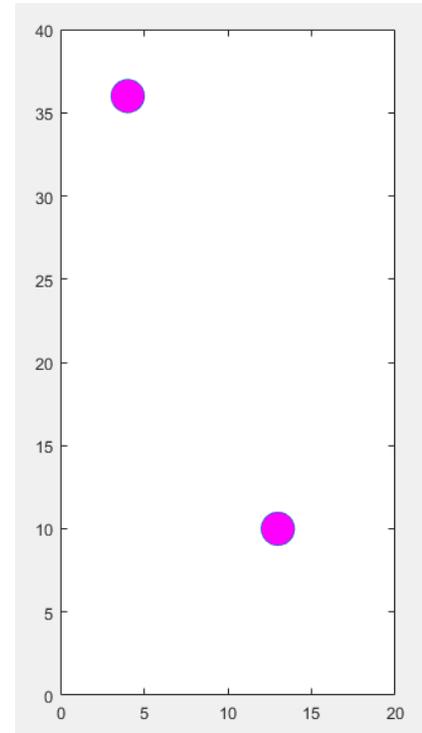


Table 4. 3 Definition of main problem

As it can be seen in 4.17, the two non-population-based optimization schemes; namely stochastic gradient descent and simulated annealing, show unsuccessful results and due to their big computational cost and sub-optimal solutions do not seem to be good methods for optimization of this field unless good engineering guess are available, which we provide them with using genetic algorithms. The rest of algorithms show comparable results and like the previous case the hybrid scheme proves to be the most successful among all.

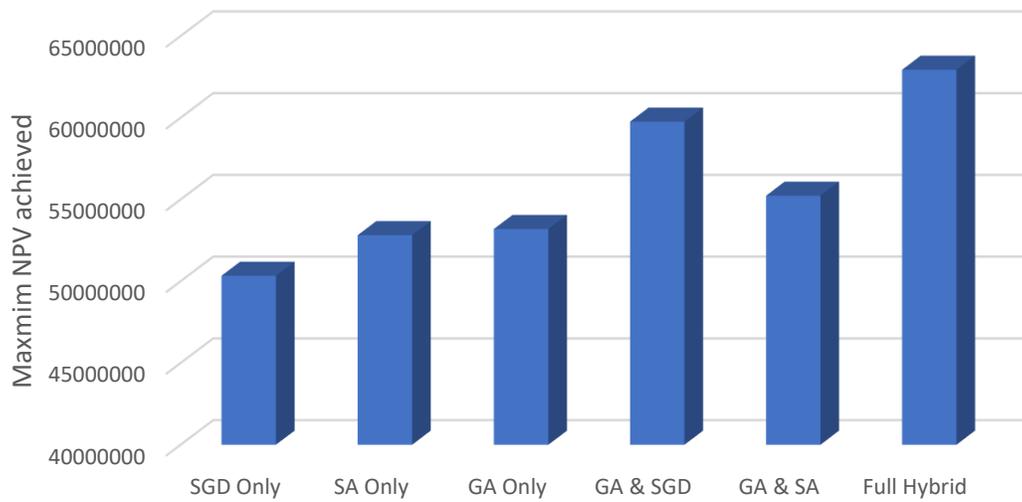


Figure 4. 17 Maximum NPV in different optimization algorithms for the main problem

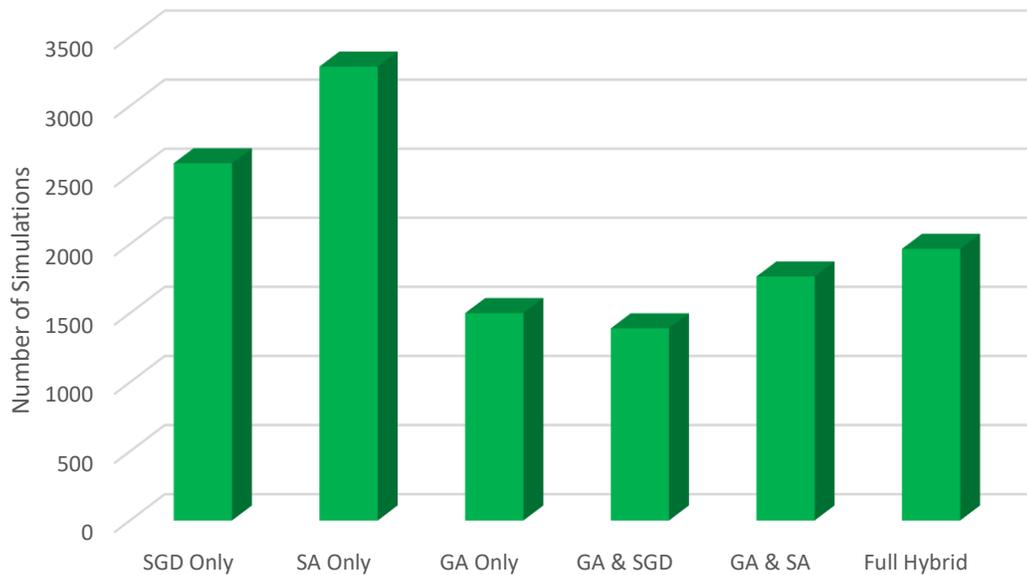


Figure 4. 18 Simulations ran in different optimization algorithms for the main problem

As it is evident in figure 4.18, the computational cost of the algorithms is on average 5 times more than the computational cost in the previous problem, simply because the ensemble size is 5 times the previous ensemble. The GA followed by SGD provides the lowest computational cost among all the algorithms, and Full Hybrid method gives the best results even though being

computationally costly. This bigger computational cost, nonetheless, as mentioned in the previous problem is due to better coverage of the search space since it finally results in higher NPVs.

### Problem Solution

Final NPV:	62.9 MM USD
Number of additional wells to be drilled:	2
Pre-drilled well 1 location:	Cell 79
Pre-drilled well 2 location:	Cell 447
Additional well 1 location:	Cell 18
Additional well 1 drilling time:	Year 2
Additional well 2 location:	Cell 34
Additional well 2 drilling time:	Year 3

Location of the wells

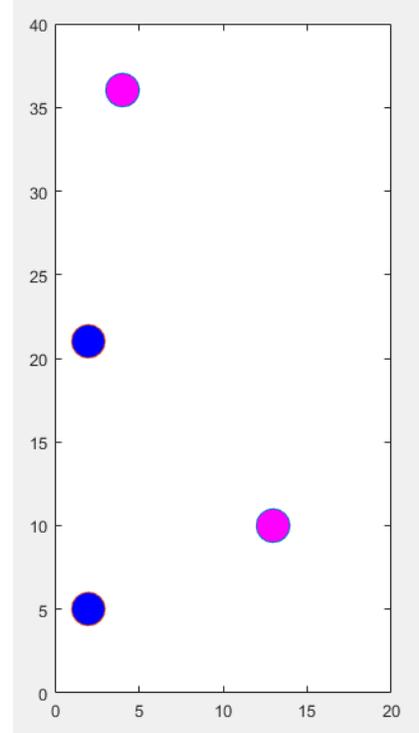


Table 4. 4 Solution to the main problem

Drilling proposal of the 2 additional wells in the western most side of the reservoir, due to existence of an aquifer in the eastern side of the reservoir could have been logically guessed as usual and can be assumed as a sign of good reservoir optimization, knowing that the initial guess has been totally random, and having different number of wells. The final water saturation map of the five ensemble members is given in figure 4.19. The saturation map in the top left corner is the initial oil saturation of uniform 75 percent. As it can be seen almost all of the wells face water breakthrough in all of the realizations, which is a good visual sign of maximum reservoir exploitation. Taking into account that the strategy in this problem has basically been exploitation of the reservoir with the help of aquifer, facing these water breakthrough almost at the end of the simulation time (8 years) is a sign of successful reservoir optimization.

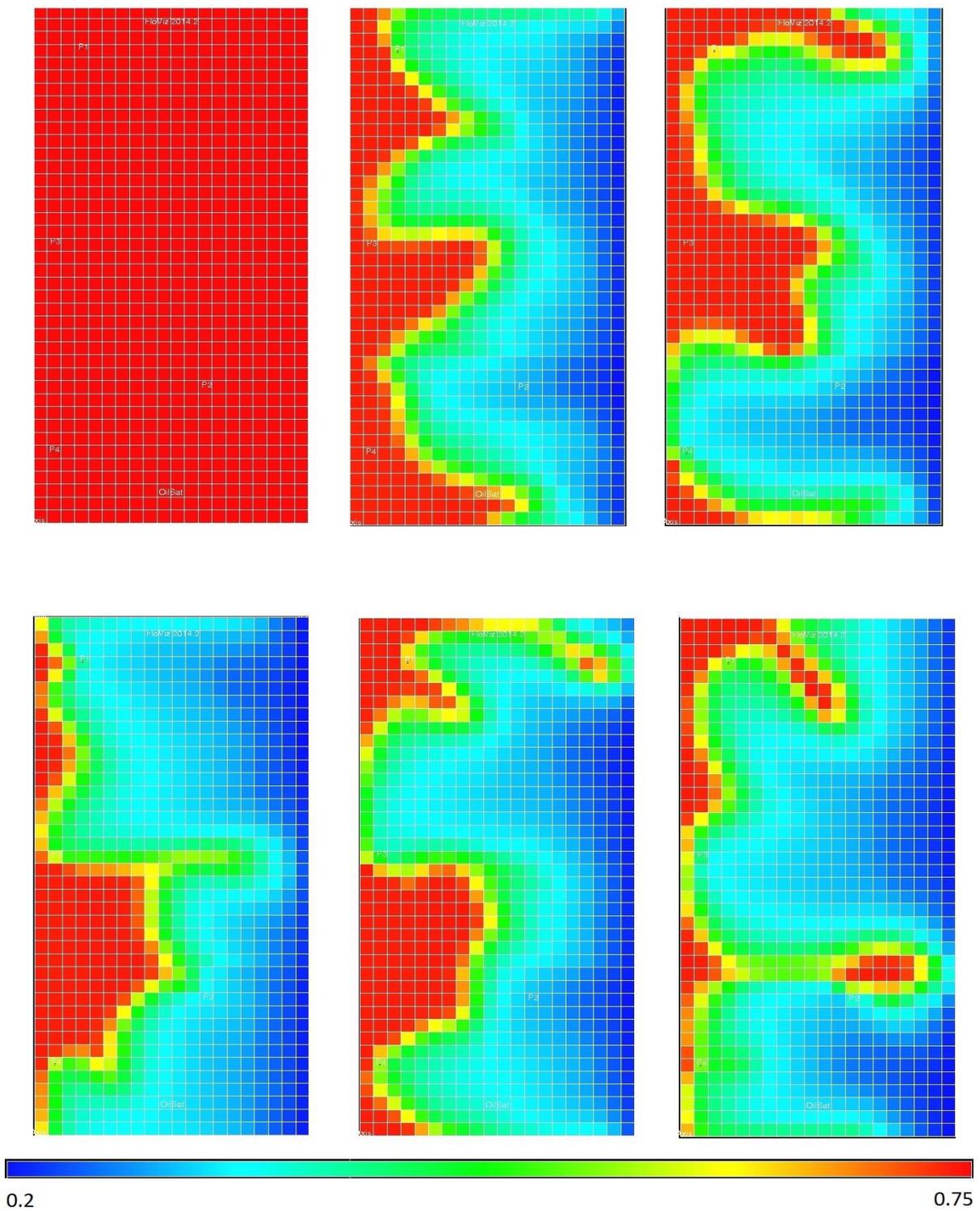


Figure 4. 19 Final water saturation map of the ensemble in the main problem

The NPV growth of our optimized case is calculated and plotted in figure 4.20. It is compared to one of the sub-optimal solutions proposed by the genetic algorithm in the plot.

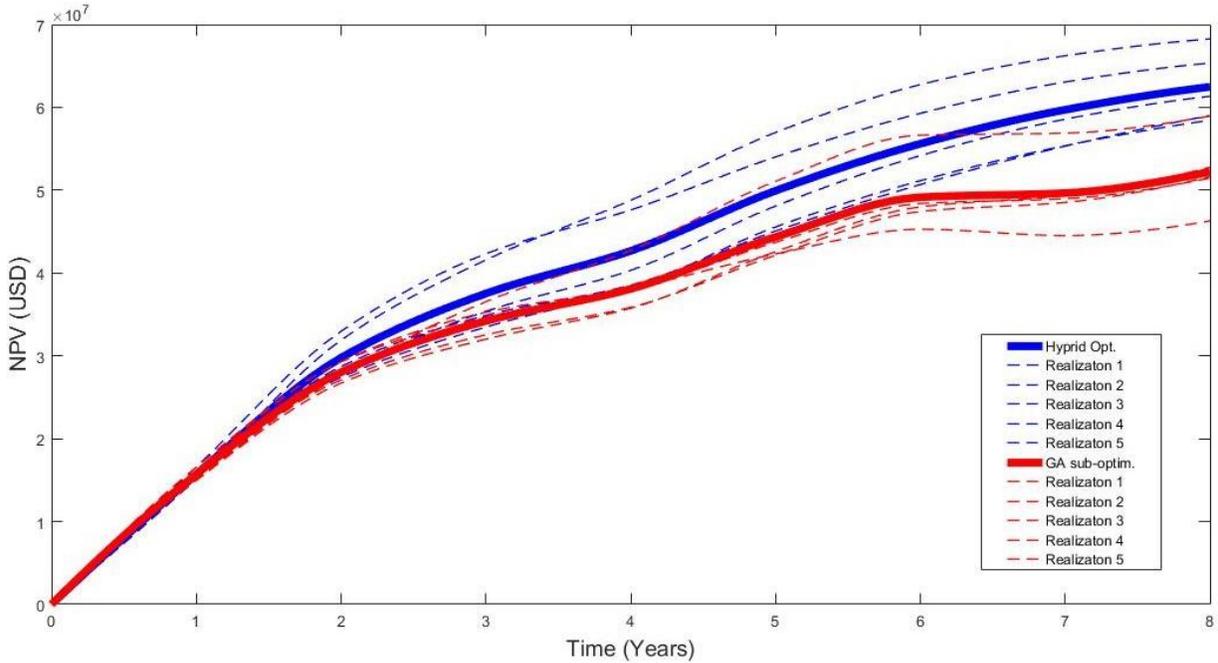


Figure 4. 20 NPV evolution of the optimized solution and its ensemble and its comparison to a sub-optimal solution and its ensemble

Except for the overall negative second derivative of the NPV curve which is due to the reduction in the production of the wells by time, any other downturn or decrease in the second differential before sixth year is due to drilling of additional wells which imposes 3 million dollars per well to us. The decrease in the 2<sup>nd</sup> differential at 6<sup>th</sup> year in GA sub-optimal solution seems to be due to the water breakthrough in one the wells.

### 4.2.3 Problem of moderate drilling cost

In the next two problems we aim at investigating the effect of increase in the drilling cost on the optimization. Accordingly, we try two different well drilling prices of 5 and 10 million dollars per well. A small addition to the water production cost is also exerted. In this part due to big computational cost of sole stochastic gradient descent method and sole simulated annealing as well as their convergence to sub-optimal solutions, these algorithms are only used together with other optimization techniques and sole implementation of them is ignored. A description of the problem of moderate drilling cost is provided in table 4.5.

## Problem Definition

Single Realization

Objective Function:	Mean NPV
Field Optimization time:	8 years
Well Drilling Cost	5 MM USD per well
Oil Price	45 USD per barrel
Water production cost	7 USD per barrel
Discount Rate	8 percent
Pre-drilled well 1 location:	Cell 79
Pre-drilled well 2 location:	Cell 447
Number wells to be drilled:	?
Location of them	?
Timing of their drilling	?

## Pre-drilled wells location

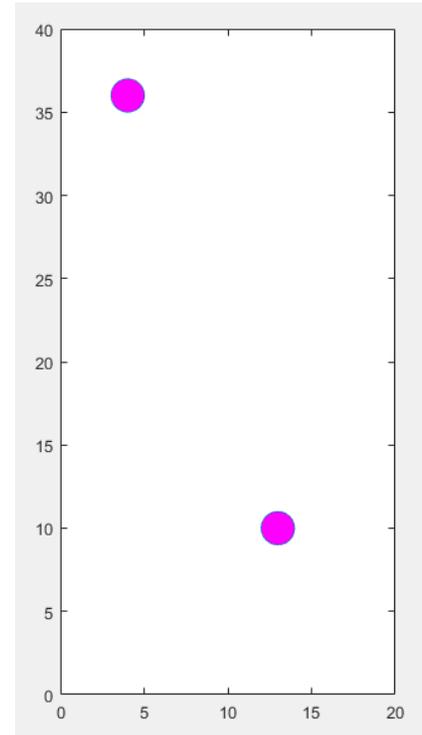


Table 4. 5 Definition of the problem of moderate drilling cost

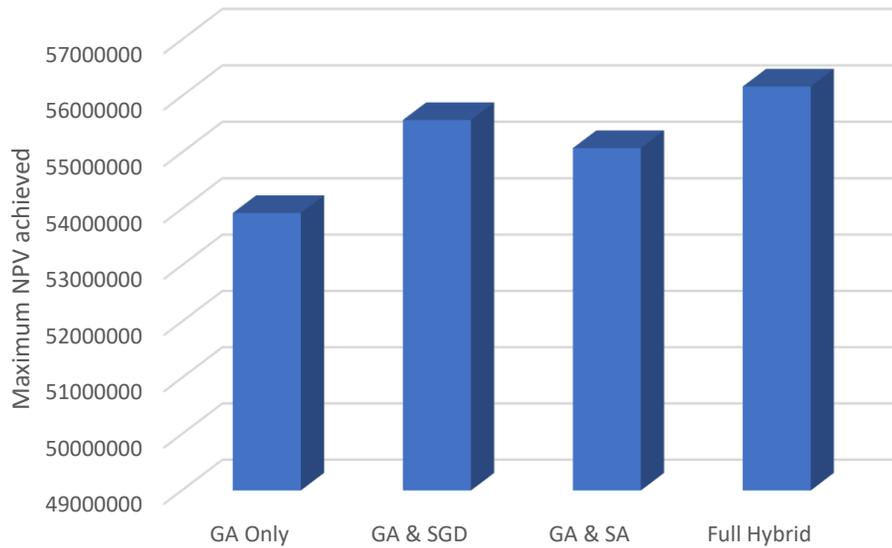


Figure 4. 21 Maximum NPV achieved by different optimization algorithms for the problem of moderate drilling cost

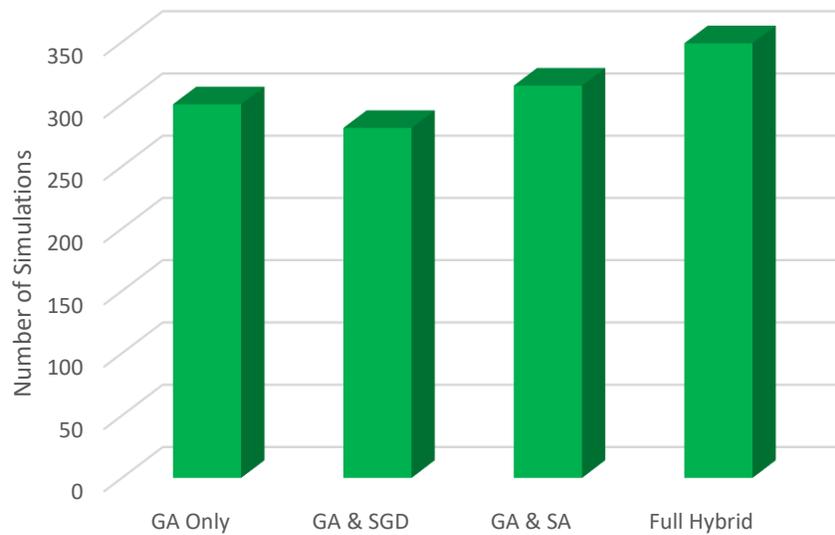


Figure 4. 22 Number of simulations ran in different optimization algorithms for the problem of moderate drilling cost

### Problem Solution

Final NPV: 56.1 MM USD  
 Number of additional wells to be drilled: 2  
 Pre-drilled well 1 location: Cell 79  
 Pre-drilled well 2 location: Cell 447  
 Additional well 1 location: Cell 18  
 Additional well 1 drilling time: Year 2  
 Additional well 2 location: Cell 74  
 Additional well 2 drilling time: Year 3

### Location of the wells

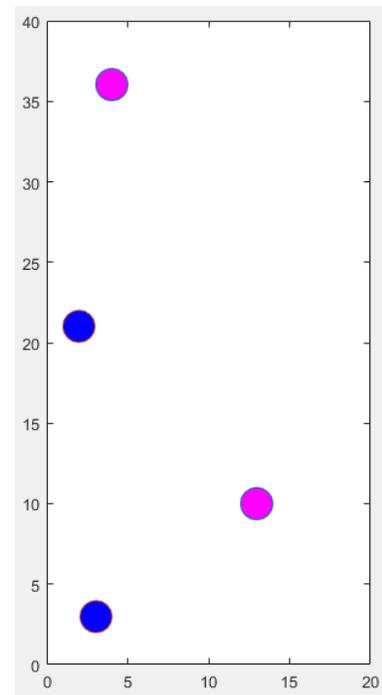


Table 4. 6 Solution to the problem of moderate drilling cost

As we check the final population of solutions in the genetic algorithm, number of the wells vary between 3 and 4 in them. This shows that the well drilling cost of 5 million dollars is in the transition zone of changing of the optimized plan with 4 wells to 3 wells. As it can be seen in figures 4.21 and 4.23, the hybrid algorithm has again converged to a plan which visually seems rational, in which due to the existence of the aquifer the wells are forced towards the western side of the field and also shows the highest objective function obtained among all the tested algorithms. The more expensive wells and addition to the water production cost has reduced the final NPV to 56.1 million dollars. The computational cost of the main four algorithms, as it can be seen in figure 4.22, follows the trend of the previous problems with small changes.

#### 4.2.4 Problem of costly drilling

In this problem the cost of drilling has been increased so as to investigate the sensitivity of the optimization algorithm to the parameters of the objective function. A summary of the problem definition can be found in table 4.7.

#### Problem Definition

Single Realization

Objective Function:	Mean NPV
Field Optimization time:	8 years
Well Drilling Cost	10 MM USD per well
Oil Price	45 USD per barrel
Water production cost	6 USD per barrel
Discount Rate	8 percent
Pre-drilled well 1 location:	Cell 79
Pre-drilled well 2 location:	Cell 447
Number wells to be drilled:	?
Location of them	?
Timing of their drilling	?

#### Pre-drilled wells location

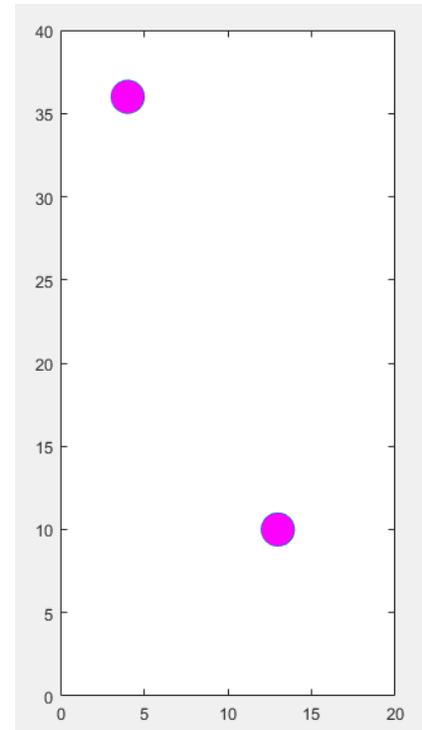


Table 4. 7 Definition of the problem of costly drilling

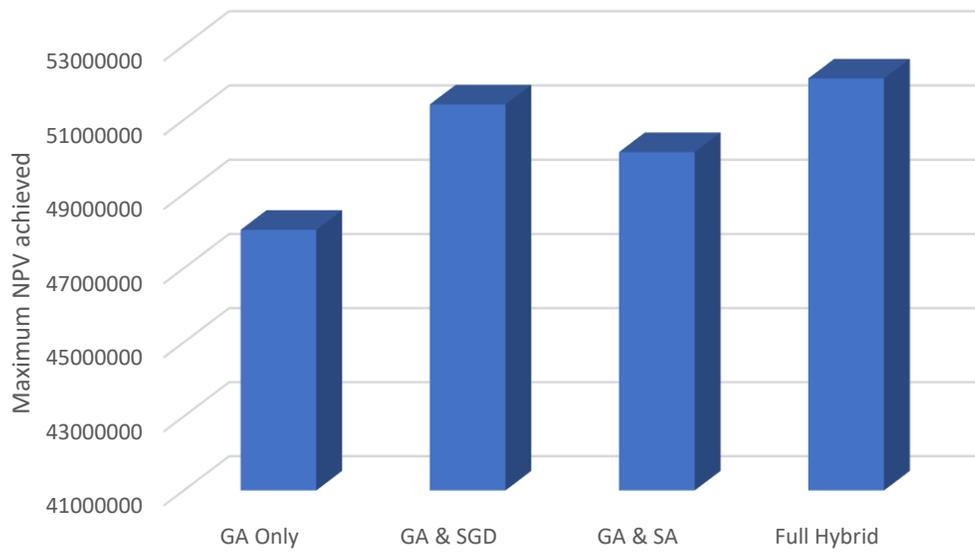


Figure 4. 23 Maximum NPV achieved by different optimization algorithms for the problem of costly drilling



Figure 4. 24 Number of simulations ran in different optimization algorithms for the problem of costly drilling

## Problem Solution

Final NPV:	50.8 MM USD
Number of additional wells to be drilled:	1
Pre-drilled well 1 location:	Cell 79
Pre-drilled well 2 location:	Cell 447
Additional well 1 location:	Cell 20
Additional well 1 drilling time:	Year 3

Location of the wells

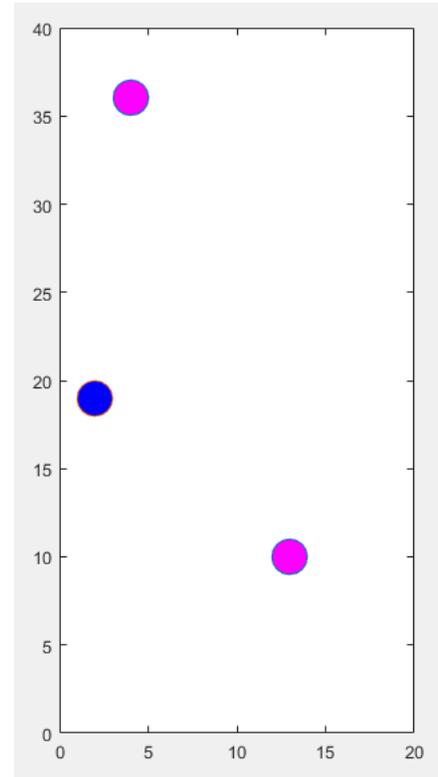


Table 4. 8 Solution to the problem of costly drilling

As it can be seen in figures 4.23 and 4.24, the ranking of quality of optimization remains the same. Full Hybrid optimization gives the best solutions even though being computationally costlier than other methods, and GA followed by SGD yields to acceptable results with the least computational cost.

The increase in the drilling cost of the wells has resulted in removal of one well by the algorithm compared to the previous problem and reducing the number of planned wells to only one. Also drilling of this one well is forced to the 3<sup>rd</sup> year to benefit from 8 percent discount rate; one year of delay in drilling is recommended by the algorithm. These two have only decreased the objective function for about 5 million USD decrease in NPV. This plan is absolutely better than continuing with the previous plan since drilling 1 more well would have costed 10 million dollars which even if drilled in the 3<sup>rd</sup> year would have resulted in more than 7 million dollars of present cost for us. This shows that the algorithm can adapt itself with costlier drilling expenses and suggest suitable well numbers to different problems.

### 4.2.5 Problem of different initial well locations

The location of the pre-drilled wells is a very important parameter in the form of objective function potential surface. In this problem, it is assumed that the location of the initial wells which are drilled in the exploration phase are totally different from those of the first four problems. A visual representation of the location of the new pre-drilled wells and the description of new problem is provided in table 4.9.

#### Problem Definition

Single Realization

Objective Function:	NPV
Field Optimization Time:	8 years
Well Drilling Cost	5 MM USD per well
Oil Price	45 USD per barrel
Water production cost	6 USD per barrel
Discount Rate	8 percent
Pre-drilled well 1 location:	Cell 400
Pre-drilled well 2 location:	Cell 600
Number wells to be drilled:	?
Location of them	?
Timing of their drilling	?

Pre-drilled wells location

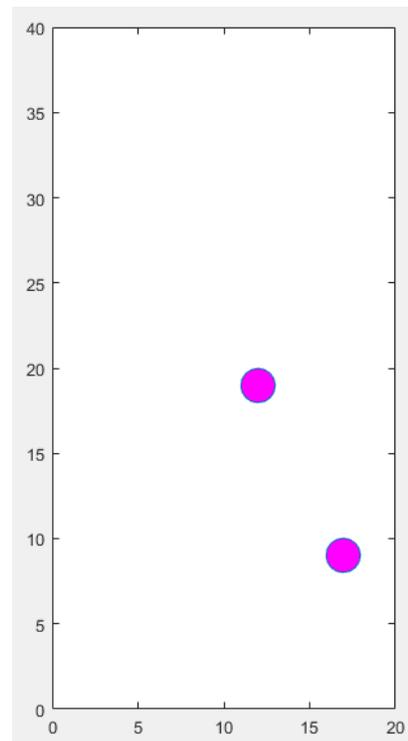


Table 4. 9 Definition of the problem of different well initial well locations

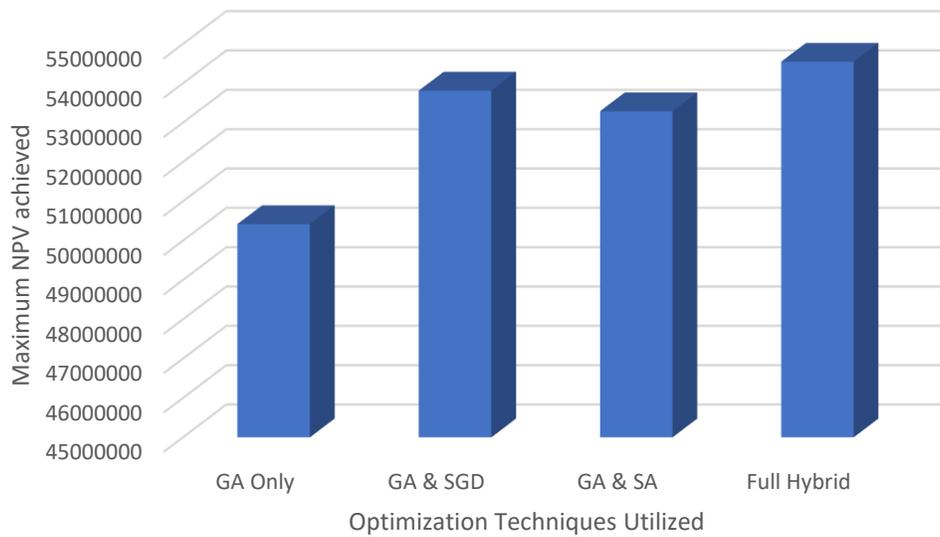


Figure 4. 25 Maximum NPV achieved by different optimization algorithms for the problem of different initial well locations

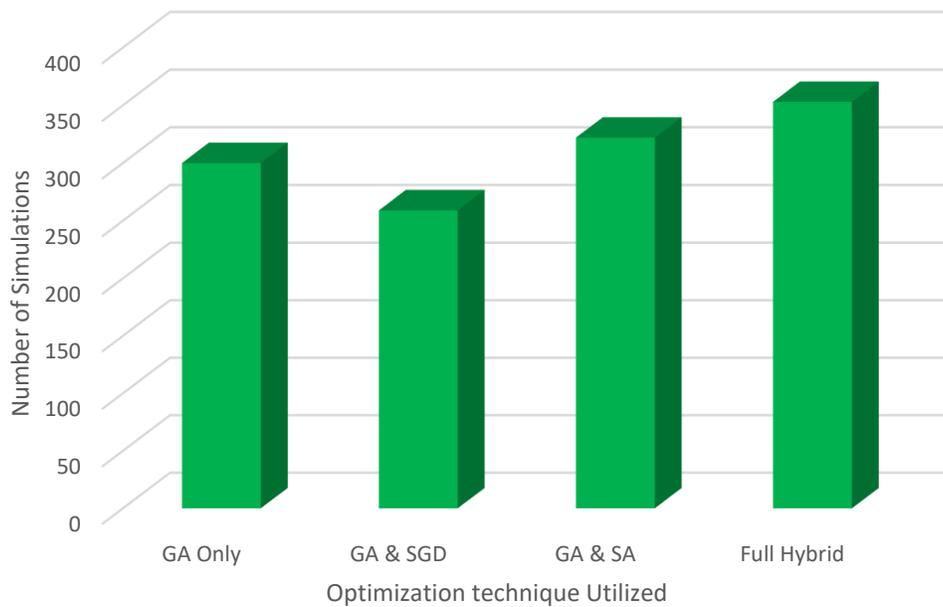


Figure 4. 26 Number of simulations ran in different optimization algorithms for the problem of different initial well locations

Bad location of one of the fixed wells compared to previous problem, has decreased the production ability of four wells. The hybrid algorithm, accordingly, suggests drilling of one additional well to

add up their number to 5 wells in total. The final NPV is accordingly 54.5 in total which is about 2 million dollars less than the plan with more appropriate fixed well locations. Both the final NPV of the algorithms and their computational cost follows the same trends from the previous problems. The location of the suggested wells by the hybrid algorithm is in the zones with highest permeability which is a good visual quality check on the algorithm. The distance between the proposed wells is also close to its maximum value and again they are forced to the side of the reservoir by the active aquifer. All these geological and intuitive quality controls prove the quality of our optimization.

### Problem Solution

Final NPV:	54.5 MM USD
Number of additional wells to be drilled:	3
Additional well 1 location	Cell 17
Additional well 1 drilling time:	Year 2
Additional well 2 location:	Cell 113
Additional well 2 drilling time:	Year 3
Additional well 3 location:	Cell 2
Additional well 3 drilling time:	Year 4

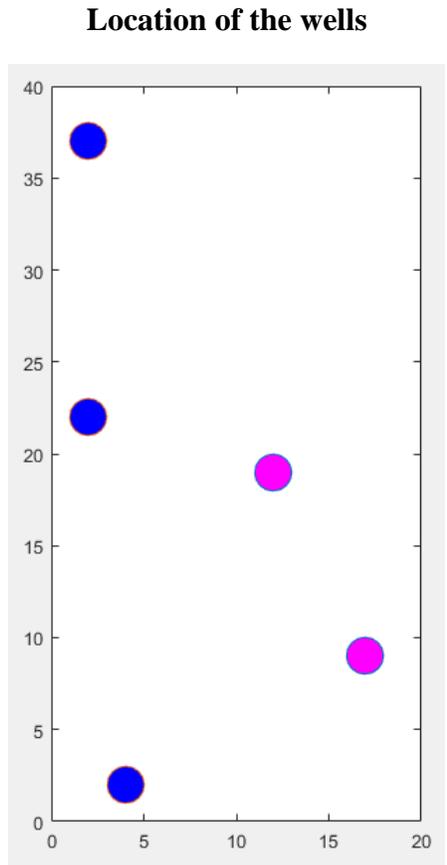


Table 4. 10 Solution to the problem of different initial well locations

### 4.2.6 Problem of single pre-drilled well

In this section, the previous problem is modified by reduction of one of the fixed wells. The description of this problem can be found in table 4.11.

## Problem Definition

Single Realization

Objective Function:	NPV
Field Optimization time:	8 years
Well Drilling Cost	5 MM USD per well
Oil Price	45 USD per barrel
Water production cost	6 USD per barrel
Discount Rate	8 percent
Pre-drilled well 1 location:	Cell 400
Number wells to be drilled:	?
Location of them	?
Timing of their drilling	?

## Pre-drilled wells location

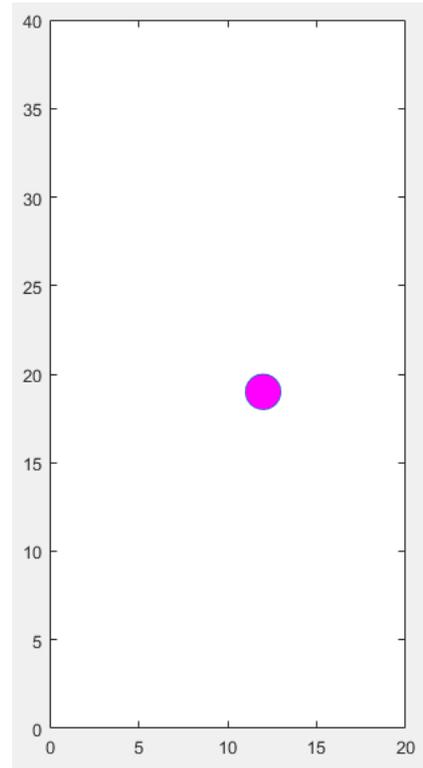


Table 4. 11 Definition of the problem of single pre-drilled well

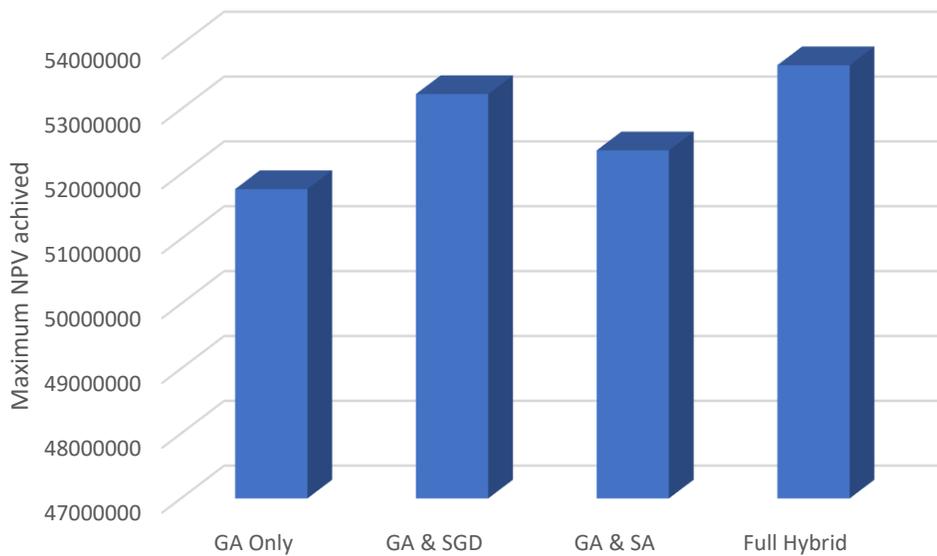


Figure 4. 27 Maximum NPV achieved by different optimization algorithms for the problem of single pre-drilled well

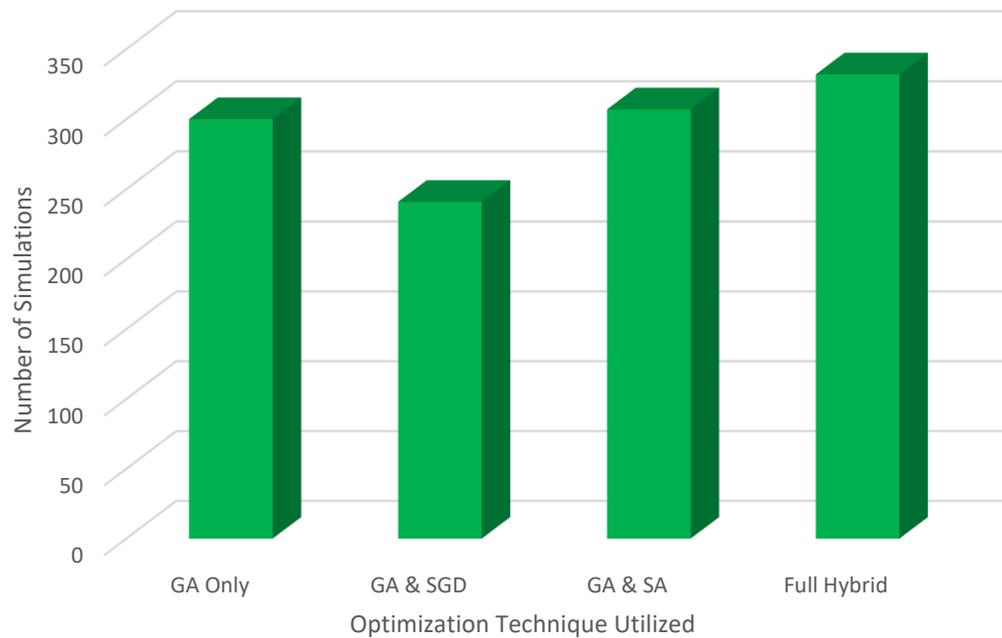


Figure 4. 28 Number of simulations ran in different optimization algorithms for the problem of single pre-drilled well

As it is seen in table 4.12, this reduction in the number of fixed wells has not resulted in drastic change in the objective function value. This is because of bad location of the removed fixed well which was very close to the aquifer and is not a big help for reservoir exploitation, due to early water breakthrough. The final NPV is 53.6 million dollars with 4 active wells in 8 years of production. The distance between the wells is again close to its maximum and the wells are proposed to be drilled in high permeability zones. Comparative quality of the algorithms and their computational cost, as it can be found in figures 4.27 and 4.28 follow the pattern from the previous problems; the full hybrid algorithm being the most successful in optimization of the problem, and the hybrid GA-SGD having the least computational cost.

**Problem Solution**

Final NPV: 53.6 MM USD  
 Number of additional wells to be drilled: 3  
 Additional well 1 location: Cell 18  
 Additional well 1 drilling time: Year 2  
 Additional well 2 location: Cell 151  
 Additional well 2 drilling time: Year 3  
 Additional well 3 location: Cell 77  
 Additional well 3 drilling time: Year 4

**Location of the wells**

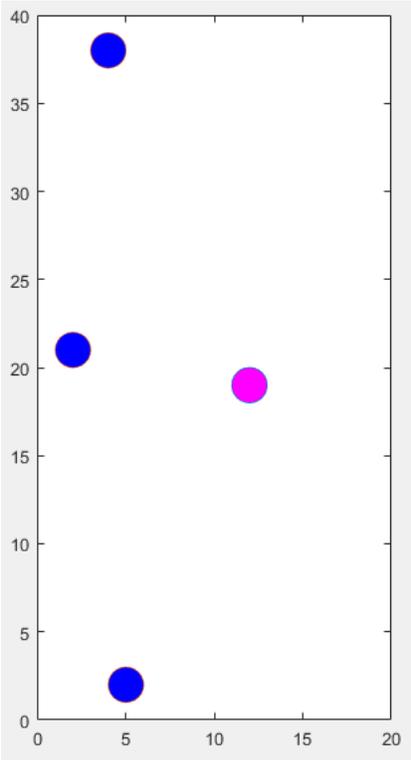


Table 4. 12 Solution to the problem of single pre-drilled well

## 5 Conclusion and future work

In this study the performance of several gradient-free and gradient-based optimization techniques as well as the hybridization of these schemes in solving petroleum industry problems were investigated. A set of optimization problems on the number of wells, well placement and their time of drilling on a 2-dimensional reservoir were defined and several optimization techniques were tried on them. The results show that the sequential hybridization of genetic algorithm, simulated annealing, and stochastic gradient descent method (GA-SA-SGD) results in the highest NPV values after optimization among all the examined methods. The other methods investigated in this study are including pure genetic algorithm (GA), pure simulated annealing (SA), pure stochastic descent method (SGD), hybrid genetic algorithm and simulated annealing (GA-SA), as well as hybrid genetic algorithm and stochastic gradient descent method (GA-SGD). Both pure simulated annealing and pure stochastic gradient descent methods were prone to convergence to local optima and their convergence to comparable results with other algorithms were highly dependent on the initial guess. This helped the hybrid algorithm to perform very well when it is designed by hybridization of genetic algorithm, which yields into a good initial guess, and local optimizers, which can convert the good initial guess into an optimal solution.

The computational cost of the hybrid GA-SGD method is the lowest on average among all the studied optimization schemes. This algorithm is both more successful than hybrid GA-SA and more computationally efficient. All the hybrid algorithms and the pure GA are successful in finding the locality of the optimized planned wells.

The full hybrid algorithm (GA-SA-SGD) proved to be robust by providing good reactions to change in the drilling cost and removal of suitably positioned pre-drilled wells in the problem. It decreased the number of wells and delayed their drilling time so as to benefit from the discount rate in the NPV function when the drilling cost increased. Also, in the case of removal of suitably positioned pre-drilled wells, the algorithm increased the number of wells for compensation. The algorithm also recommends the wells to be drilled in high permeability zones and as distant as possible from the aquifer, both of which intuitively prove the quality of optimization. These were inferred by analysis of the results of six different optimization problems in this study.

A study that can be conducted as continuation of this work is incorporation of real geological models, which can be followed by simultaneous optimization of continuous and categorical variables. This can be done by introduction of real field data into the models and slight changes in the algorithms. Assessment of hybridization of other algorithms which have good potential in exploration of the search space such as PSO with the algorithms introduced in this study would also be a good extension of this research.

Another future work which seems helpful for the petroleum industry is inspired by the computational cost of the simulation process. As most of the computational cost is dictated by simulation phase, introduction of multi-level optimization into this work would be a reasonable way to tackle this problem. This can be done by using proxies or simpler simulators in simulation cases where the objective function accuracy is of less importance, i.e. initial steps of GA, and using more accurate simulators when getting close to the optimal solutions.

## 6 References

1. Khor, C.S., A. Elkamel, and N. Shah, *Optimization methods for petroleum fields development and production systems: a review*. Optimization and Engineering, 2017. **18**(4): p. 907-941.
2. Siavashi, M. and M. Yazdani, *A Comparative Study of Genetic and Particle Swarm Optimization Algorithms and Their Hybrid Method in Water Flooding Optimization*. Journal of Energy Resources Technology, 2018. **140**(10): p. 102903.
3. Khademi, G. and P. Karimaghaee, *Hybrid FDG optimization method and kriging interpolator to optimize well locations*. Journal of Petroleum Exploration and Production Technology, 2016. **6**(2): p. 191-200.
4. Mittal, P., K. Kulkarni, and K. Mitra, *A novel hybrid optimization methodology to optimize the total number and placement of wind turbines*. Renewable energy, 2016. **86**: p. 133-147.
5. Norouzi, S., M. Nazari, and M.V. Farahani. *A Novel Hybrid Particle Swarm Optimization-Simulated Annealing Approach for CO<sub>2</sub>-Oil Minimum Miscibility Pressure (MMP) Prediction*. in *81st EAGE Conference and Exhibition 2019*. 2019.
6. Vasantharajan, S., R. Al-Hussainy, and R. Heinemann, *Applying optimization technology in reservoir management*. Journal of Petroleum Technology, 2006. **58**(05): p. 82-88.
7. Kosmidis, V.D., J.D. Perkins, and E.N. Pistikopoulos, *Optimization of well oil rate allocations in petroleum fields*. Industrial & Engineering Chemistry Research, 2004. **43**(14): p. 3513-3527.
8. Cullick, S., et al., *Optimizing multiple-field scheduling and production strategy with reduced risk*. Journal of petroleum technology, 2004. **56**(11): p. 77-83.
9. Stoitsits, R., et al. *Application of nonlinear adaptive modeling for rigorous representation of production facilities in reservoir simulation*. in *SPE Annual Technical Conference and Exhibition*. 1992. Society of Petroleum Engineers.
10. Stoitsits, R., et al. *Production optimization at the Kuparuk river field utilizing neural networks and genetic algorithms*. in *SPE mid-continent operations symposium*. 1999. Society of Petroleum Engineers.
11. Horesh, L., et al., *Reduced Space Dynamics-Based Geo-Statistical Prior Sampling for Uncertainty Quantification of End Goal Decisions*, in *Numerical Analysis and Optimization*. 2015, Springer. p. 191-221.
12. Wang, H., et al., *Optimal well placement under uncertainty using a retrospective optimization framework*. Spe Journal, 2012. **17**(01): p. 112-121.
13. Rahmawati, S.D., et al., *Integrated field operation and optimization*. Journal of Petroleum Science and Engineering, 2012. **81**: p. 161-170.
14. Asadollahi, M., et al., *Production optimization using derivative free methods applied to Brugge field case*. Journal of Petroleum Science and Engineering, 2014. **114**: p. 22-37.
15. Asadollahi, M., G. Nævdal, and A. Shafieirad, *Efficient workflow for optimizing well controls*. Journal of Petroleum Science and Engineering, 2012. **82**: p. 66-74.
16. Lo, K. and C. Holden. *Use of well management schemes for rate forecasts*. in *SPE western regional meeting*. 1992. Society of Petroleum Engineers.
17. Fang, W. and K. Lo, *A generalized well management scheme for reservoir simulation*. SPE Reservoir Engineering, 1996. **11**(02): p. 116-120.
18. Holland, J.H., *Genetic algorithms*. Scientific american, 1992. **267**(1): p. 66-73.
19. Carter, J.N., *Chapter 3 Introduction to using genetic algorithms*. 2003. p. 51-76.
20. Emerick, A.A., et al. *Well placement optimization using a genetic algorithm with nonlinear constraints*. in *SPE reservoir simulation symposium*. 2009. Society of Petroleum Engineers.
21. Yeten, B., L.J. Durlofsky, and K. Aziz, *Optimization of nonconventional well type, location, and trajectory*. SPE Journal, 2003. **8**(03): p. 200-210.
22. Güyagüler, B., et al., *Optimization of well placement in a Gulf of Mexico waterflooding project*. SPE Reservoir Evaluation & Engineering, 2002. **5**(03): p. 229-236.

23. Fujii, H. and R. Horne, *Multivariate optimization of networked production systems*. SPE Production & Facilities, 1995. **10**(03): p. 165-171.
24. Palke, M.R. and R.N. Horne. *Nonlinear optimization of well production considering gas lift and phase behavior*. in *SPE Production Operations Symposium*. 1997. Society of Petroleum Engineers.
25. Wang, P., M. Litvak, and K. Aziz. *Optimization of production from mature fields. Paper No. 32152*. in *17th World Petroleum Congress, Rio de Janeiro, Brazil, September*. 2002.
26. Ozdogan, U. and R.N. Horne, *Optimization of well placement under time-dependent uncertainty*. SPE Reservoir Evaluation & Engineering, 2006. **9**(02): p. 135-145.
27. Onwunalu, J.E. and L.J. Durlofsky, *Application of a particle swarm optimization algorithm for determining optimum well location and type*. Computational Geosciences, 2010. **14**(1): p. 183-198.
28. Kaveh, A., *Advancers in Metaheuristic Algorithms for Optimal Design of Structures*. Springer, 2017: p. 12-40.
29. Olsson, A.E., *Particle swarm optimization: theory, techniques and applications*. 2010: Nova Science Publishers, Inc.
30. Shirangi, M.G. and L.J. Durlofsky. *Closed-loop field development optimization under uncertainty*. in *SPE Reservoir Simulation Symposium*. 2015. Society of Petroleum Engineers.
31. Isebor, O.J., D. Echeverría Ciaurri, and L.J. Durlofsky, *Generalized field-development optimization with derivative-free procedures*. SPE Journal, 2014. **19**(05): p. 891-908.
32. Aliyev, E. and L.J. Durlofsky. *Multilevel field-development optimization using a sequence of upscaled models*. in *SPE Reservoir Simulation Symposium*. 2015. Society of Petroleum Engineers.
33. Humphries, T.D. and R.D. Haynes, *Joint optimization of well placement and control for nonconventional well types*. Journal of Petroleum Science and Engineering, 2015. **126**: p. 242-253.
34. Chibante, R., *Simulated Annealing, Theory with Applications*. 2010.
35. Mahmoodpour, S. and M. Masihi, *An improved simulated annealing algorithm in fracture network modeling*. Journal of Natural Gas Science and Engineering, 2016. **33**: p. 538-550.
36. [http://adl.stanford.edu/aa222/Lecture\\_Notes\\_files/chapter3\\_gradient.pdf](http://adl.stanford.edu/aa222/Lecture_Notes_files/chapter3_gradient.pdf).
37. <https://engmrk.com/mini-batch-gd/>.
38. Sarma, P., L.J. Durlofsky, and K. Aziz. *Efficient closed-loop production optimization under uncertainty (spe94241)*. in *67th EAGE Conference & Exhibition*. 2005.
39. Sarma, P., et al. *Production optimization with adjoint models under nonlinear control-state path inequality constraints*. in *Intelligent Energy Conference and Exhibition*. 2006. Society of Petroleum Engineers.
40. Sarma, P., et al., *Efficient real-time reservoir management using adjoint-based optimal control and model updating*. Computational Geosciences, 2006. **10**(1): p. 3-36.
41. Sarma, P., L. Durlofsky, and K. Aziz, *Kernel Principal Component Analysis for an Efficient, Differentiable Parameterization of Multi-point Geostatistics, submitted to*. Mathematical Geology, 2006.
42. van den Heever, S.A., et al., *A Lagrangean decomposition heuristic for the design and planning of offshore hydrocarbon field infrastructures with complex economic objectives*. Industrial & engineering chemistry research, 2001. **40**(13): p. 2857-2875.
43. Volkov, O. and M. Bellout, *Gradient-based constrained well placement optimization*. Journal of Petroleum Science and Engineering, 2018. **171**: p. 1052-1066.
44. Yıldız, A.R., et al., *Optimization of thin-wall structures using hybrid gravitational search and Nelder-Mead algorithm*. Materials Testing, 2016. **58**(1): p. 75-78.
45. Wu, J., J. Long, and M. Liu, *Evolving RBF neural networks for rainfall prediction using hybrid particle swarm optimization and genetic algorithm*. Neurocomputing, 2015. **148**: p. 136-142.

46. Sen, M.K., et al., *Stochastic reservoir modeling using simulated annealing and genetic algorithm*. SPE Formation Evaluation, 1995. **10**(01): p. 49-56.
47. Li, X.-G. and X. Wei, *An improved genetic algorithm-simulated annealing hybrid algorithm for the optimization of multiple reservoirs*. Water Resources Management, 2008. **22**(8): p. 1031-1049.
48. Chiu, Y.C., L.C. Chang, and F.J. Chang, *Using a hybrid genetic algorithm-simulated annealing algorithm for fuzzy programming of reservoir operation*. Hydrological Processes: An International Journal, 2007. **21**(23): p. 3162-3172.
49. Salam, D.D., I. Gunardi, and A. Yasutra. *Production optimization strategy using hybrid genetic algorithm*. in *Abu Dhabi International Petroleum Exhibition and Conference*. 2015. Society of Petroleum Engineers.
50. Azamipour, V., M. Assareh, and G.M. Mittermeir, *An improved optimization procedure for production and injection scheduling using a hybrid genetic algorithm*. Chemical Engineering Research and Design, 2018. **131**: p. 557-570.
51. HAJDASIŃSKI, M.M., *NPV-Compatibility, project ranking, and related issues*. The Engineering Economist, 1997. **42**(4): p. 325-339.
52. Hwang, C.-R., *Simulated annealing: theory and applications*. Acta Applicandae Mathematicae, 1988. **12**(1): p. 108-111.

## Appendix A—The Eclipse DATA file

RUNSPEC

TITLE

REMA

DIMENS

20 40 1 /

AQUDIMS

4\* 2 40/

FIELD

OIL

WATER

START

01 JAN 2008 /

WELLDIMS

-- Item 1: maximum number of wells in the model

-- - there are two wells in the problem; injector and producer

-- Item 2: maximum number of grid blocks connected to any one well

-- - must be one as the wells are located at specific grid blocks

-- Item 3: maximum number of groups in the model

-- - we are dealing with only one 'group'

-- Item 4: maximum number of wells in any one group

-- - there must be two wells in a group as there are two wells in total

15 1 2 15 /

UNIFOUT

UNIFIN

NSTACK --Linear solver stack size

10/

FAULTDIM

-- FAULTDIM

-- This keyword specifies the maximum number of segments of

-- fault data entered with the FAULTS keyword in the GRID

-- section.

2 /

-----

GRID

GRIDFILE --Control output of the grid geometry file. 2 produce EGRID file

2 /

INIT

TOPS

800\*2700.0 /

DX

800\*100.0 /

DY

800\*100.0 /

DZ

800\*20.0 /

PORO

800\*0.2 /

INCLUDE

'PERMXTrue1.GRDECL' /

--PERMX

-- 800\*200 /

COPY

'PERMX' 'PERMY' /

/

PERMZ

800\*0.001 /

FAULTS

-- IX1 IX2 IY1 IY2 IZ1 IZ2 FACE

'South' 1 20 13 13 1 1 Y /

'North' 1 20 27 27 1 1 Y /

/

MULTFLT

-- Fault Multiplier

South 0.2 /

North 0.05 /

/

EDIT

PROPS

PVDO

0.0 1.01202 3.0

1000.0 1.01101 3.0

2000.0 1.01 3.0 /

PVTW

2000.0 1.00 5.00E-07 1.0 /

ROCK

2000.0 3.00E-05 /

DENSITY

48.623232 62.312542 0.06054 /

SWOF

0.2 0 0.81 0

0.25 0 0.81 0

0.27 0.002 0.7512 0

0.3 0.005 0.663 0

0.33 0.0092 0.5622 0

0.35 0.012 0.495 0

0.4 0.021 0.362 0

0.45 0.032 0.261 0

0.5 0.048 0.185 0

0.55 0.068 0.128 0

0.6 0.089 0.081 0

0.65 0.119 0.049 0

0.67 0.137 0.035 0

```
0.7    0.158  0.022  0
0.73   0.181  0.016  0
0.75   0.211  0.009  0
0.8    0.301  0      0
```

```
/
```

```
-----  
REGIONS
```

```
SOLUTION
```

```
PRESSURE
```

```
800*2000.0 /
```

```
SWAT
```

```
800*0.25 /
```

```
RPTSOL
```

```
-- Controls on output from SOLUTION section
```

```
-- RESTART 2: the inter block flows of each phase are written to the Restart files.
```

```
'RESTART=2' /
```

```
AQUFETP
```

```
-- Defines Fetkovich Aquifers properties
```

```
-- Aq# D P@D Vol Comp PI
```

```
1 800 1* 1.0E9 1E-5 5 /
```

```
-- 1 861 1* 1.0E7 1E-5 5 / -- Original
```

```
/
```

```
AQUANCON
```

```
-- connects analytic aquifers
```

```
-- Face: of the cube where the aq acts
```

```
-- AQ# i1 in j1 jn k1 kn Face
```

```
1 20 20 1 40 1 1 I+ /
```

```
/
```

```
-----  
SUMMARY
```

```
RUNSUM
```

SEPARATE

RPTONLY

WOPR

'P1' 'P2' 'P3' 'P4' 'P5' 'P6' 'P7' 'P8' 'P9' /

WWCT

'P1' 'P2' 'P3' 'P4' 'P5' 'P6' 'P7' 'P8' 'P9' /

FOPT

FWPT

FWIT

-----

SCHEDULE

RPTSCHED -- Controls on output from SCHEDULE section

RESTART=2 /

RPTRST

BASIC=3 4\* 1 21\* 999999 /

INCLUDE

'WELSPECS\_COMPDAT.INC' /

end