AMERICAN UNIVERSITY OF BEIRUT

OIL AND GAS PRODUCTION SYSTEM OPTIMIZATION USING PARTICLE SWARM OPTIMIZATION

by
HUSSEIN SLEIMAN KASSEM

A thesis
Submitted in partial fulfillment of the requirements
for the degree of Master of Engineering
to Baha and Walid Bassatne Department of Chemical Engineering and Advanced Energy
of the Maroun Semaan Faculty of Engineering and Architecture
at the American University of Beirut

Beirut, Lebanon
August 2019

AMERICAN UNIVERSITY OF BEIRUT

Oil and Gas Production System Optimization Using Particle Swam Optimization

by
HUSSEIN SLEIMAN KASSEM

Approved by:

Dr. Kassem Ghorayeb, Assistant Professor — Advisor
Department of Chemical and Petroleum Engineering

Dr. Mohammad N. Ahmad, Professor, Chairman — Member of Committee
Department of Chemical and Petroleum Engineering

Dr. Nesreen Ghaddar, Professor — Member of Committee
Department of Mechanical Engineering

Dr. Elsa Maalouf, Assistant Professor — Member of Committee
Department of Chemical and Petroleum Engineering

Date of thesis defense: August 29, 2019

AMERICAN UNIVERSITY OF BEIRUT

THESIS, DISSERTATION, PROJECT RELEASE FORM

Student Name:_____ Kassem ___________Hussein _________ Sleiman _________
Last First Middle

● Master's Thesis ○ Master's Project ○ Doctoral Dissertation

☐ I authorize the American University of Beirut to: (a) reproduce hard or electronic copies of my thesis, dissertation, or project; (b) include such copies in the archives and digital repositories of the University; and (c) make freely available such copies to third parties for research or educational purposes.

■ I authorize the American University of Beirut, to: (a) reproduce hard or electronic copies of it; (b) include such copies in the archives and digital repositories of the University; and (c) make freely available such copies to third parties for research or educational purposes
after:

One ---- year from the date of submission of my thesis.
Two --X-- years from the date of submission of my thesis.
Three ---- years from the date of submission of my thesis.

Hussein Kassem September 5, 2019

Signature Date

ACKNOWLEDGEMENTS

I would like to express my deepest gratitude for Dr. Kassem Ghorayeb for his support, guidance, and patience through my graduate study. It was a great privilege and honor to work under his guidance,

I would also like to thank The Oil and Gas Research Group at AUB, led by Dr. Kassem Ghorayeb, especially the PhD Candidate Ahmad Harb, for their recommendations, innovative ideas and generous help in completing this work.

I would also like to thank Prof. Mohammad Ahmad, Prof. Nesreen Ghaddar, and Dr. Elsa Maalouf for their valuable comments as members of my thesis committee.

Finally, I owe so much to my family and friends for their support, and unwavering belief.

AN ABSTRACT OF THE THESIS OF

Hussein Sleiman Kassem for Master of Engineering
Major: Chemical Engineering

Title: Oil and Gas Production System Optimization Using Particle Swarm Optimization

Field development planning and optimization of production system are main factors while developing oil and gas fields. The main aim of any oil and gas development is to maximize the economic value of projects under study. Reservoir engineers focus on optimizing wells placement, trajectory, and type. Facility and production engineers have as a role to set and optimize production system elements in terms of placement, sizing and their interconnection. Literature review showed an important research work in the field of production system optimization. This latter is typically classified as tree-like, nodes and segments, multilayers problem, yet recent research work only dealt with two layers optimization problems. In addition, recent literature presented the use of genetic algorithm (GA), which is a population-based method, as an efficient optimizer. Another optimization technique, particle swarm optimization (PSO), is introduced in this type of problems. Both methods' efficiency was compared and PSO outperformed GA in terms of convergence time and value. Literature showed that many improvements were applied to the standard algorithm. Adaptive particle swarm optimization is applied in parallel with two newly introduced improvements in this work: Multiple runs initialization, and Restart improvement. The novel improved method showed better results than the standard PSO as the complexity of the problem is increased starting from two-layers up to four-layers high complexity. This introduced method showed robustness and high efficiency in handling multiple layers problems.

Keywords: Field Development Planning, Production system, Optimization, Evolutionary algorithms, PSO, GA.

CONTENTS

FIGURES

TABLES

CHAPTER I

INTRODUCTION

With the increasing demand on oil and gas around the world associated with the scarcity of the oil and gas reserves, oil companies are moving towards ultra-deep areas to unlock unexplored hydrocarbon resources. The development of fields in such areas is a very complex task as it involves sophisticated design procedure at huge expenses. The main challenge lies in reducing the capital and operating expenditures of the offshore production system while maintaining a prudent and optimal development performance.

An offshore field development plan mainly consists of three main components: Reservoir, Wells and Subsea production network. At the reservoir level, reservoir engineers conduct several subsurface studies to select the most optimal depletion plan. Once the depletion plan (well number, type, location, rates and pressure) is set, production engineers design the production tubing in a way to further optimize production from well bottom-hole till the wellhead. After establishing the production characteristics at wellheads, facility engineers design the subsea production network to bring production from wells to the facilities for processing and export. Subsea production networks can range in complexity from a single well with a flow linked to a host facility, to several wells linked via a manifold producing and transferring product via subsea processing facilities or directly to an onshore installation. Figure 1 shows a typical subsea production network. The main components of a subsea production systems are:

- Susbea Xtrees/Wellheads incorporating flow and pressure control valves
- Manifolds for controlled gathering of various fluids streams

- Platforms for combining produced hydrocarbon
- Processing unit are processing produced hydrocarbon
- Subsea pipelines/flowlines to convey produced/injected fluids between wellheads, manifolds, platforms and processing units

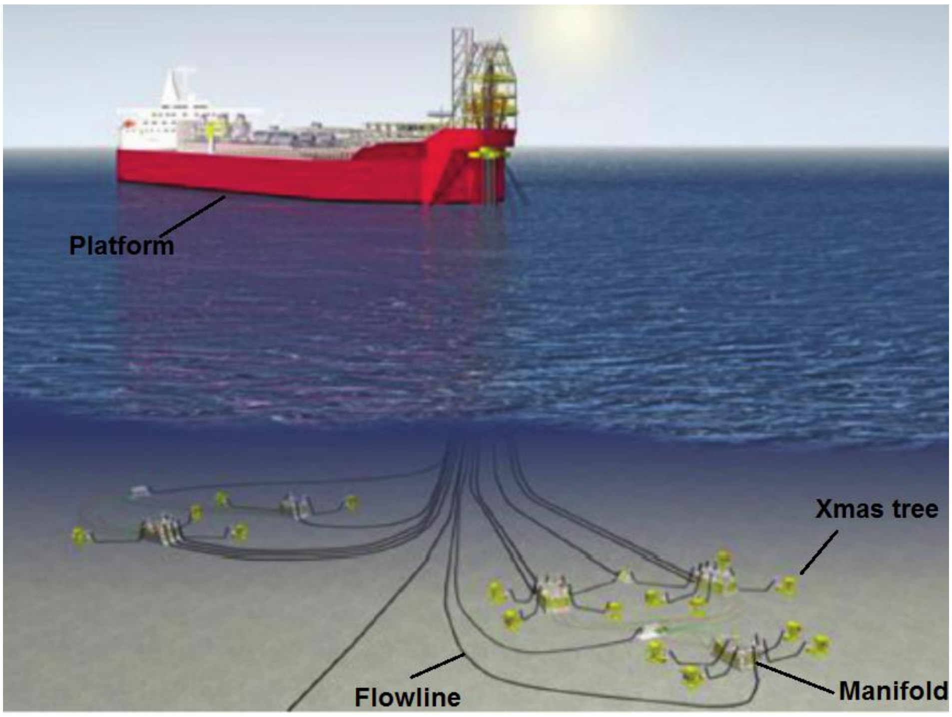


Figure 1- Offshore production system

The components of the subsea production system are interconnected. Manifolds organize wells into clusters. These valves are the hub of oil or gas production from several wells, which is then transported to a platform by a single pipe, the “riser”. Manifolds reduce the number of risers connecting wells to the platform, which in turn reduces the total pipeline length used hence allowing for more flexibility in the field operations. The location of the manifolds, in turn, affect the length of pipelines and the capacity effect the riser diameter etc. The number, location and capacity of each component in a subsea production design must be specified in way to optimize production while minimizing the total cost of the overall design.

A. Objective

This work presents an efficient optimization method applying for both onshore and offshore fields. In onshore fields, the problem will tackle the planning of the number and placement of drill centres as well as gathering centres. However, in offshore fields the proposed method applies for setting the subsea production network design, mainly the number, capacity and location of manifolds and platforms to be installed, in addition to well and manifold designation to manifold and platform respectively. The proposed model considers the production layout as a multi-layered system connecting nodes and segments, where nodes represent wellheads, manifolds, platforms and segments represent the flowlines connecting different components. This model is optimized using hybrid evolutionary algorithm: improved particle swarm optimization (PSO). The total cost of the subsea production system is used as the objective function of the proposed algorithm.

B. Thesis Outline

This thesis is divided into five chapters. This chapter presented the scope of our work, motivation and its objective

In Chapter 2, an extensive literature review was conducted on evolutionary optimization methods mainly, particle swarm optimization (PSO) and genetic algorithms (GA) that are applied for optimization problems in general. The workflow along with the technical improvements of these methods are described.

In Chapter 3, the optimization problem along with the input parameters, optimization variables, and objective function are discussed.

In Chapter 4, a hybrid PSO algorithm is introduced for multi-layer production system optimization problem. Improvements were introduced to the standard PSO by manipulating the algorithmic parameters and applying a smart initialization which helped in speeding up convergence and reaching a consistent result.

In Chapter 5, results of the proposed method in comparison to the GA and the standard PSO are presented. We first compare the performance GA and the Standard PSO on two layered system (wellheads and manifolds). We then compare the performance of our proposed method to that of the standard PSO on four different optimization problems with different level of complexity. Results show the superiority of the standard PSO over GA and the newly proposed algorithm over the standard PSO.

CHAPTER II

LITERATURE REVIEW

Field development is a key activity in an oil and gas project as it comes at a huge expense especially in offshore development where investments may reach billions of dollars. Consequently, a group consisting of geoscientists, reservoir engineers, production engineers and facility engineers team up to come out with the most optimal field development plan. Field development planning can be characterized by two optimization problems; Well placement and production layout design. In well placement, the number, trajectory, location, type and control are altered to maximize one or more driving values; e.g. net present value, plateau length or recovery factor. Upon optimizing well placement, another important factor arises, that is the design of the production system. The production system layout design involves defining the optimal number of production system elements (i.e. manifolds, PLEMs, Platforms, and facility), location, sizing, and allocation of flowline between different elements [1].

Many optimization methods were used for well placement optimization. These methods can be classified into gradient based and gradient free methods. Gradient-based methods (e.g. Conjugate gradient, Newton's, and steepest descent methods) require computation of the gradient of the objective function. These methods are not widely used in well placement optimization problems because of their need to smooth objective functions which is not the case in both well placement and platform placement problems [2-6].Another family of optimization's methods used in field development planning is the stochastic gradient-free methods such as particle swarm optimization (PSO) and genetic algorithm (GA). Stochastic algorithms acquire their robustness of overcoming premature converging (local optima) from their inherently randomness. Another feature of these methods is their capability to address a wide range of optimization problems irrespective of their complexity. Stochastic optimization methods can also be simply modified, tuned, and assisted by other optimizers to enhance their performance, thus work in a hybrid manner[6]. Though, no deterministic mathematical convergence can be demonstrated for these methods; the convergence can be assessed by a maximum number of iterations, prescribed optimal goal value [2], or an "external convergence criterion" as is the case in the of production system optimization problem.

Another important aspect is the production system optimization. Production system layout comprises decision about the number, positions, and capacity of assets i.e. manifolds, PLEMs, platform, and facility. Optimizing production units' locations and the way these units are connected is an important decision variable in field development planning in terms of both drilling cost and enhanced hydrocarbon recovery. This optimization problem has as objective function the total production system cost (i.e. cost of units in use and cost of connections between the units such as pipelines, flowlines, and risers costs). The problem is optimized regarding two main aspects the number of units that should be used, and the total length of the connections. Thus, an equilibrium is needed between these two factors so that the problem is optimized. Regarding the high cost of the units (manifolds, PLEMs, etc..) relatively to the connections' costs, minimizing the number of units will be a critical factor in the optimization process. However, if the small number of units will lead to very high connections' distance this will affect the optimal result. In addition, the importance of minimizing the total length of the connections arises from two main factors: not only as aforementioned by minimizing the drilling cost and investment related to the distances, but, more importantly, from enhancing the productivity of the reservoir. The productivity of wells and hence of the reservoir is affected by the well tubing, risers and pipelines length through the associated hydrostatic pressure drop in the production system due to fluid lifting. Consequently, the shorter the distances, the lower the pressure drop and hence the higher the well productivity. [7, 8]

The optimization of the production units can be performed post-well location optimization problem or can be treated in a joint manner (simultaneously). Dogru considered nonlinear mixed integer programming (MINLP) in his work on optimization of platform and wells locations. Their algorithm limits are set to be 5 platforms and 1000 points as a search space. In contrast, our algorithm is not limited by the number of platforms or the search space; the latter is open with only lower and higher limits [9]. Although, Dogru's method is smooth and fast for a low number of wells, time constraints and physical limitations were observed for a well count exceeding 30. Hansen et al. tackled the problem as multi-capacitated plant problem (MCPLP) where they considered the location and capacity of the platforms as optimization parameters in order to minimize the investment costs. They used mixed integer programming (MIP) with Tabu search heuristic. Tabu search heuristic is a gradient-based optimization tool that searches in the neighbourhood of existing solutions for local optimal solutions. The proposed models showed computational problems for more than 30 possible locations and 100 wells [1]. Rosa and Ferreira studied the problem of platform and manifolds location. They related the problem to the pressure drop taking place in pipelines and used MIP to maximize an objective function; an NPV equation related to the placement of platforms and manifolds. Their method is exhaustive in computational load as the number of optimization parameters increases [7, 10]. Campozona et al. used an optimization algorithm combining meta-heuristic methods such as Tabu search with integer programming (IP). This optimizer is coupled with a reservoir simulator that receives the platforms locations from the optimizer, calculates the associated NPV and send it back to the optimizer in a loop manner until a convergence criterion is met. This method is applied on a simple case of 12 wells and 1 platform and showed high computational load so that only 200 iterations were performed [11]. Similarly, Rodrigues et al. used a multi-capacitated Platforms and Wells Location Problem (MPWLP) algorithm that employs linear programming and aims at minimizing costs by defining the number, location, and capacities of offshore platforms. This method only considered one scenario with 1 platform case and, as stated by the authors, it is not tested with various levels of complexity [12]. Recently, Sales et al. used GA integrated with Monte Carlo simulation to handle the platform location optimization problem. Their method showed better results with less computational load compared to the deterministic methods (e.g. IP, MPWLP, MINLP and Tabu search). They considered uncertainties associated to the production profile which is, in turn, related to the platform positioning. Multiple scenarios are simulated with quick and optimal solutions for each case [10].

In contrast to all aforementioned methods, the used method in this work is dealing with multi-layer problem i.e. it is not restricted to wells-Platform positioning, but it can handle problems that are of higher complexity such as "wells-manifolds-PLEMs-Platform-Facility" problem.

A. Overview of Evolutionary algorithms

1. Genetic Algorithm

Genetic algorithm (GA) is an evolutionary algorithm that mimics the natural laws of genetics such as crossover, mutation and other genetic operators of chromosomes coming from two parents. Natural selection proposed by Darwin consists that all these genetic operators applied in the propagation of the populations are focusing on making the survival chance of fittest individuals favored.

GA is a stochastic algorithm; it also deals with a population of solutions that can be recombined to get more fit solutions. It is a robust method as it can deal with multidimensional problems. It is a search method that deals with large population of solutions. Solutions are represented as chromosomes and this is the first step in GA method to encode solutions into chromosomes. The GA consists of an iterative process in a way that the population evolves toward the needed criteria through several steps as follows:

a. Initialization Phase

Population of solutions with a set of properties is initialized, each solution is represented by a chromosome that is composed of many subparts known as genes which are the properties to be optimized.

b. Encoding

These solutions can be expressed or encoded into a binary form or can be expressed as a real value expression. In case of binary representation, the parameters to be optimized are represented in a vector of binary digits. Each parameter is consisting of a constant length sub-vector in the chromosome. The length of the parameter representation vector is dependent on the lower and upper limits of this factor take as example if the boundaries of an optimization parameter are [0 50] so the maximum number of binary representations is that corresponding of the conversion of 50 (=110010), so a vector of length 6 is preserved for this parameter. The same is applied for other parameters constituting the chromosome and are set in a conventional predefined order.

c. Decoding

Each set of properties forming a solution presented in a chromosome which is a subpart of a group of solutions (Population). As discussed in the encoding part, know the inverse is done by translating the binary representation of each optimization parameter and this value is used in setting the simulation inputs to get results and then evaluate the objective function. While decoding the chromosomes will give us the parameter value which is also called the “phenotype”.

d. Fitness Value

This value is a main factor in natural selection method since it relies on the survival of the fittest. So, we should know how the fitness is considered and calculated. Fitness means of how good the solution is, hence in case of maximization problems the highest objective function, that can used as reference, will lead into the highest fitness value equal to one for example and the other solutions will give less and less fitness values. These values are ranked in a decreasing order in case of maximization problems. The first iteration of the GA consists of the first three steps beforementioned that are encoding, decoding, and Fitness value calculation.

e. Propagation of the Algorithm

Initialization is the first step to be done so that we have initial solutions guessed that have been ranked according to their Fitness values. Know we move to the propagation of the algorithm that is done as described in the following section:

f. Selection

Selection step is dependent on the aim of the problem and then is related to the Fitness value of the previous solutions. It is defined as the procedure of electing two parents of the previous solutions that will mate to give new offspring of solutions by permutating and exchanging parts (called genes in genetics) of their chromosomes. The higher the fitness value of a solution the higher chance of being picked to mate and reproduce in the next step. In this context, the selection pressure is defined as the extent to which the best solutions are chosen. Note that there are many types of selection such as Roulette wheel, Rank selection, and Tournament selection, etc.

g. Crossover

After electing the best two individuals that will mate and reproduce to give birth to two new solution off-springs. This is done through three steps that are: Selection of individuals for the mating process, cross site and type (can be single or multiple points crossover) and swapping of the segments following the cross site randomly chosen.
Crossover is a stochastic procedure that is responsible of the diversity of solutions that can be progressively enhanced by continuously permutating best parent's multi-parameters that will lead to an amelioration in the populations' solutions.

h. Mutation

After crossover process new solutions are produced, however the configuration of new chromosomes will not be intact as the natural selection theory suggests. These new chromosomes will be subject to mutation that corresponds of three main processes i.e. Flipping, interchanging, and reversing.

Flipping consists of changing of a bit of the mutated chromosome, using a mutation chromosome, from 0 to 1 and vice-versa that will lead to the final child's chromosome.

Interchanging consists of randomly election of two or more positions of a chromosome that interchange their bits.

Reversing consists of a random selection of a bit position that is reversed with the bit next to it.

Mutation is a very important factor that can produce new solutions from out of the box by changing some parameters of the new solutions and this is very important to overcome the repetition of the same solutions and then overcoming local optima.

i. Evaluation and Replacement

The new solutions are then tested using the same procedures done in the initialization phase. The chromosomes of the offspring are decoded as they are already encoded. Then their corresponding parameters values (phenotypes) are used as simulation inputs. Sequentially the objective functions and the Fitness values are evaluated.

As for the replacement process the old generation is killed and replaced by the new one. This is done through two main methods: basic generational update (Nchildren are produced from N populations and the whole population is replaced) and the steady state update (every produced offspring is inserted to replace the worst existing member). In our work we used the steady state update as a replacement method.

j. Convergence Criteria

The convergence criteria are dependent on the problem constraints. If time is the main constraint then a maximum number of simulations will be the convergence criterion, however a certain hoped value may be considered as optimization goal and be index of convergence, or the propagation of the algorithm (e.g. Plateau formation) may be also used as a factor to stop or restart the simulation in this case. [13]

2. Particle Swarm Optimization

Particle Swarm optimization (PSO) is an evolutionary algorithm developed by Kennedy and Eberhart in 1995 [14]. It is inspired by a close behavioural review of schools of fish and flocks of birds. Normally, fish and birds travel in groups without colliding with one another by following the group and adjusting to its position and velocity based on the information provided by the group itself. Each “particle” represents a solution of the objective function and a “swarm” depicts the group of particles involved in the optimization workflow. These particles could be further grouped into subgroups (neighbourhood topologies) allowing the exchange of information from other particles in addition to their own.

The position of each particle in the search space is driven by a calculated velocity parameter based on previous iteration results. The velocity is updated by a mathematical formulation between the prior particle velocity, its distance to the position where the particle achieved its local best and its distance from the particle that achieved the global best. Each particle memorizes the best position or “solution” it attains during the entire optimization process (local best). The algorithm also memorizes the best position attained by any of its particles (global best)[15]. The velocity is truncated by a maximum value defined by the boundary of the search space. Figure 2 represents PSO solution vectors the position and the velocity relationship are obtained by the formula below and illustrated by Figure 3:

$$x_{i,j}(k+1) = x_{i,j}(k) + v_{i,j}(k+1) \tag{1}$$

$$v_{i,j}(k+1) = w \times v_{i,j}(k) + c_p r_1 \left(p_{best(i,j)} - x_{i,j}(k)\right) + c_g r_2 \left(g_{best(i,j)} - x_{i,j}(k)\right) \tag{2}$$

Where i refers to the particle, j refers to the optimization variable and k refers to the current iteration.

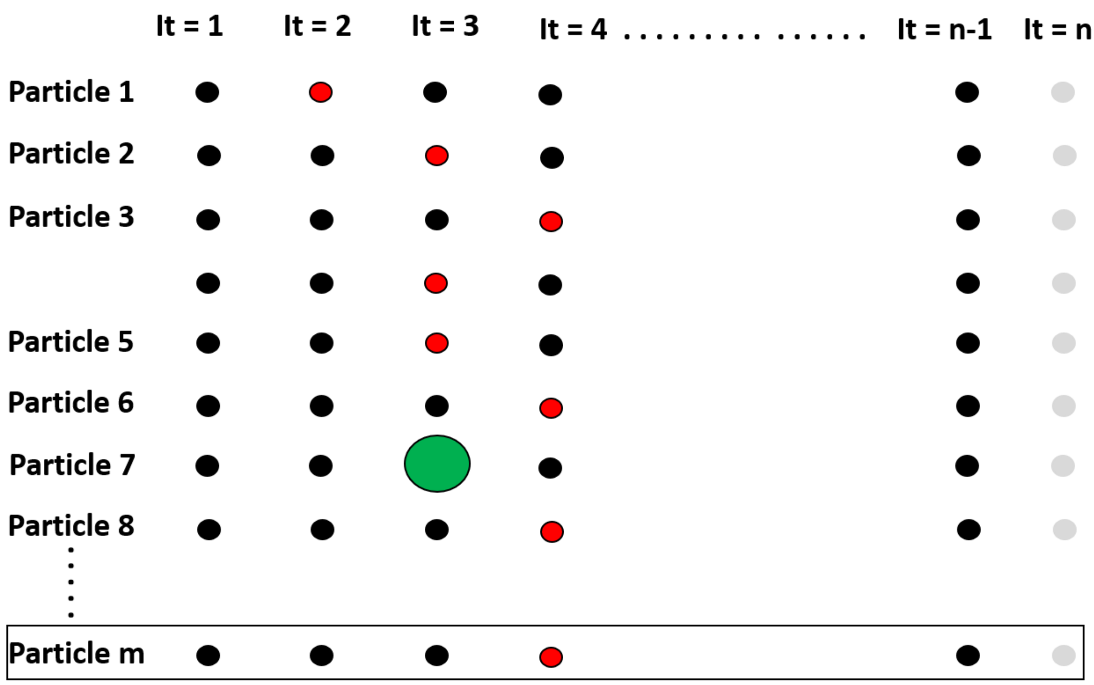


Figure 2- PSO solution vectors where • represents a particle • represents the solution vector with the best result in all iteration of a specific particle ● represents the solution vector with the best result through all iterations and among all particles

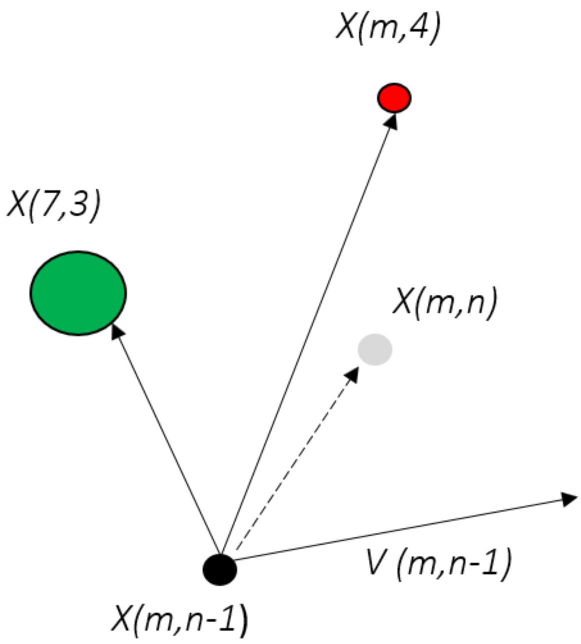


Figure 3- PSO position vector update

The velocity equation involves three main parameters; the inertia weight (w), the cognitive weight (c_p) and the social weight (c_g). The inertia weight defines the impact of the trend towards the previous particle velocity, the cognitive weight defines the impact of the trend towards the particle local best ($p_{best(i,j)}$) and the social weight defines the impact of the trend towards the swarm global best ($g_{best(i,j)}$). It also involves two independent uniform random variables r_1 and r_2 between 0 and 1. The main purpose of these variables is to make the overall process stochastically dependent which helps the optimizer avoiding local optima traps.

a. Propagation of the Algorithm

i. Initialization

As for stochastic population-based algorithms, initialization of the first population is important in the propagation of the optimization i.e. if we start from a good initial position this may help the algorithm by reducing the time and the number of simulations needed to reach optimal results. The initialization is mainly done randomly by choosing one location from the search space.

ii. Algorithmic Parameter Selection.

Many researchers have investigated the effect of PSO algorithmic parameters selection (w, c_p, and c_g) on the system convergence. Jiang et al. provided a sufficient condition after studying the convergence of the standard particle swarm system [17]. Jiang et al. suggested a set of parameters, $w = 0.715$ and $c_p = c_g = 1.7$. In addition, Zhang et al. found that up with a new process in selecting the parameters. The values of the parameters are found to be ($w = 0.724$, $c_p = c_g = 1.468$) and ($\omega = 0.785$, $c_p = c_g = 1.331$). [15]

Other researchers have also proposed other sets of parameters; that is for example: $\omega = 0.6$, $c = 1.7$ [18] and $\omega = 0.729$, $c = 1.494$ [19]. The velocity and position of each particle will move within a specified range. The maximum velocity was set at 0.1* X_{max} as per Zhang et al. work [20].

3. PSO Improvements

a. Inertia Improvements

Inertia weight is the main factor balancing exploration and exploitation in PSO optimization process. The contribution of the previous particle velocity to the updated velocity is determined by the inertia weight factor [21]. Regarding the importance of this factor in determining the velocity (step vector of each of the optimization parameters), many researchers studied the parametric effect of the inertia weight. Some adopted constant inertia weight, others considered adaptive inertia weight variation. The following table summarizes the main advancements of the inertia values.

Table 1- Summary of previous reports on inertia weight improvements

Inertia weight	**Formula**	**Reference**
Constant inertia weight	$\omega = constant = [0.9 - 1.2]$	[22]
Random inertia weight	$\omega = 0.5 + \frac{rand()}{2}$	[23]
Sigmoid increasing	$\omega_k = \frac{\omega_{start} - \omega_{end}}{1 + e^{-u\times(k-n\times gen)}} + \omega_{end}$ $u = 10^{(\log(gen)-2)}$ $gen = maximum\ number\ of\ generations$	[24]
Sigmoid decreasing	$\omega_k = \frac{\omega_{start} - \omega_{end}}{1 + e^{u\times(k-n\times gen)}} + \omega_{end}$ $\omega_{start} = initial\ limit$	[24]

	$\omega_{end} = final\ limit$ n =sigmoid constant = [0.25,0.5,0.75]	
Linear Decreasing	$\omega_k = \omega_{max} - \frac{\omega_{max} - \omega_{min}}{Max\_iterations}$ $\omega = [0.9 - 0.4]$	[25]
Linear Decreasing inertia weight	$w = w_{max} - (w_{max} - w_{min}) * \frac{MAXIter - iter}{MAXIter}$ $w_{max} = 0.9$ $w_{min} = 0.4$	[26]
Chaotic descending inertia weight	$z = rand(0,1)$ $z = 4 \times z \times (1 - z)$ $\omega = (\omega_1 - \omega_2) - \frac{Iter_{max} - Iter}{Max_{iterations}} + \omega_2 \times z$	[27]
Chaotic random inertia weight	$z = rand(0,1)$ $z = 4 \times z \times (1 - z)$ $\omega = 0.5 \times rand(0,1) + 0.5 \times z$	[27]
Global-local Best inertia weigh	$\omega_i = 1.1 - \frac{gbest_i}{pbest_i}$	[28]
Natural exponential inertia weight(e1)	$\omega_t = \omega_{min} + (\omega_{max} - \omega_{min}) \times e^{-\left[\frac{t}{\frac{Max_{iterations}}{10}}\right]}$	[29]
Natural exponential inertia weight(e2)	$\omega_t = \omega_{min} + (\omega_{max} - \omega_{min}) \times e^{-\left[\frac{t}{\frac{Max_{iterations}}{4}}\right]^2}$	[29]
Oscillating inertia weight	$\omega_t = \frac{\omega_{max} + \omega_{min}}{2} + \frac{\omega_{max} - \omega_{min}}{2} \times \cos\frac{2\pi t}{T}$ $T = \frac{2S_1}{3+2k}, S_1 = \frac{3}{4}S, S: total\ iterations\ number$	[30]

	For the remaining $S_2 = \frac{1}{4}S$, $\omega_t = \omega_{min}$ $\omega_{min} = 0.3$, $\omega_{max} = 0.9$, k can be varied [1-7]	
Simulated annealing	$w = w_{max} - (w_{max} - w_{min}) \times \lambda^{(k-1)}$ $\lambda = 0.95$	[31]
Logarithm Decreasing Inertia Weight	$w = w_{max} + (w_{min} - w_{max}) \log_{10}(a + \frac{10t}{Tmax})$ $a = 1 \quad w_{max} = 0.9 \quad w_{min} = 0.4$	[32]
Exponent Decreasing Inertia Weight	$w = (w_{max} - w_{min} - d_1) \times e^{\left[\frac{1}{1+\frac{d_2 t}{t_{max}}}\right]}$ $w_{max} = 0.95$, $w_{min} = 0.4$, $d_1 = 0.2$ $d_2 = 7$	[33]

b. Constriction Factor Improvement

Clerc introduced a new parameter instead of the inertia weight, a constriction factor was developed as a function of c1 and c2, which multiplies into the velocity equation [34]. The importance of the constriction factor is that it ensures the convergence of the system and allows the system to search in an efficient way avoiding premature convergence.

$$k = \frac{2}{|2 - \phi - \sqrt{\phi^2 - 4\phi}|}$$

$$v_{i+1} = k\left[\omega v_i + c_1 r_1 \left(x_{local_{best}} - x_i\right) + c_2 r_2 \left(x_{global_{best}} - x_i\right)\right]$$

$$x_{i+1} = v_{i+1} + x_i$$

Where ϕ = c1 + c2, $\phi > 4$ and both c1=c2=2.05 [35].

c. <u>Coefficient Tuning Improvement</u>

Ratnaweera et al. proposed three improvements to the core of PSO algorithm by addressing the coefficients tuning problem [36]:

Time Varying acceleration coefficients (TVAC): The acceleration constants c_1 and c_2 represent the weighting of the stochastic acceleration terms that pull each particle towards the pbest and gbest solutions. An important identification should be established for c_1 and c_2 as both play major roles in improving the performance of the PSO. c1 represents the "self-cognition" that pulls the particle to its own historical best position, assisting in exploring other solutions within that region. c2 represents the "social influence" that pushes the swarm to converge to the current globally best region, leading to faster conversions.

Thus, instead of having fixed values for c_1 and c_2 equal to 2 as initially proposed, a change in the coefficients values incrementally based on what stage the PSO solution is, improves the overall optimal solution of the PSO, as each stage (exploration, exploitations, convergence, jumping out) are individually optimized.

One modification is the Time-Varying Acceleration coefficients that can be represented as follows:

$$c_1 = (c_{1f} - c_{1i}) * \frac{iter}{MaxIter} + c_{1i} \tag{3}$$

$$c_2 = (c_{2f} - c_{2i}) * \frac{iter}{MaxIter} + c_{2i} \tag{4}$$

Where

$c_{1f} = 2.5$, $c_{1i} = 0.5$, $c_{2f} = 0.5$, and $c_{2i} = 2.5$

MPSO-TVAC (Mutated PSO with TVAC): This improvement has the very same idea of mutation introduced in the Genetic Algorithm, focusing on introducing new solutions in case the Global best in stuck into a local optimum. This improvement serves the introduction of diversity in the search space under study. In their work the mutated based PSO is applied with the Time-Varying acceleration coefficients beforementioned.

Self-organizing Hierarchical PSO with TVAC: In this approach the velocity update equation is no more dependent on the previous velocity term as proposed by Kennedy and Eberhart [37]. However, this change may lead to local optima solutions. To solve this problem, Ratnaweera et al. proposed HPSO with TVAC. This method keeps the velocity term from last iteration as zero and the velocity term is reinitialized using a reinitialization velocity proportional v_{max}.

d. APSO Improvement

Zhan et al. introduced a new improvement to the core of the PSO, called the adaptive particle swarm optimization (APSO). The PSO efficiency is studied through two main criteria: number of evaluations needed and premature convergence. In their work they proposed first an evolutionary state estimation (ESE) on which the APSO is based to enhance the algorithm performance. [38]

a. ESE: or evolutionary state estimation will study the population distribution in each generation. Based on this approach the adaptation of PSO is done. Following are the steps describing this clustering-based approach:

    1. Calculation of the mean distance of each particle i to all other particles:

$$d_i = \frac{1}{N-1} \sum_{j=1, j\neq i}^{N} \sqrt{\sum_{k=1}^{D} (x_i^k - x_j^k)^2} \tag{4}$$

$where$:

N: $population\ size$

D: $number\ of\ dimensions\ (number\ of\ optimization\ parameters)$

2. Compute the evolutionary factor:

$$f = \frac{d_g - d_{min}}{d_{max} - d_{min}} \in (0,1) \tag{5}$$

$where$:

d_g: $globally\ best\ particle\ distance$

d_{min}, d_{max}: $minimum\ and\ maximum\ distances\ respectively$

3. Classification of f into one of four states $S_1, S_2, S_3, and\ S_4$ representing Exploration, exploitation, convergence, and jumping out respectively. The membership to the evolution states is as follows $S_1 \rightarrow S_2 \rightarrow S_3 \rightarrow S_4$.

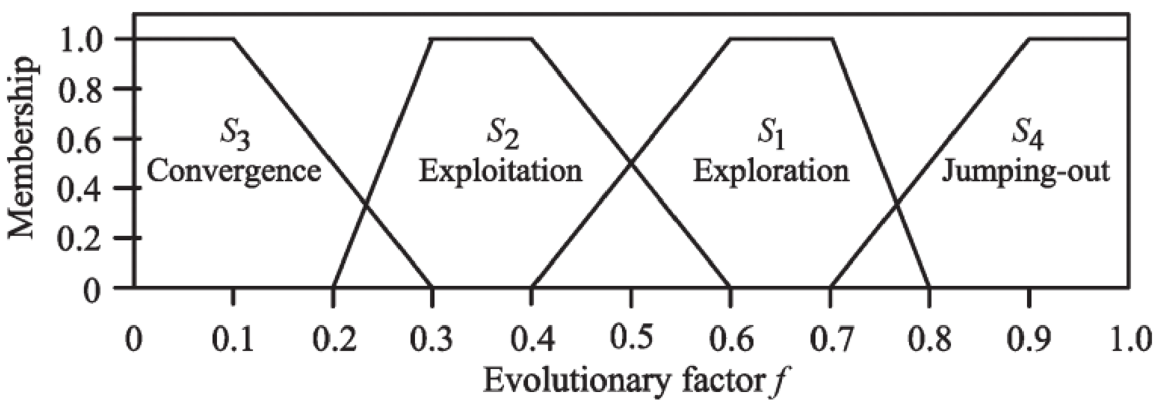


Figure 4-Classification of membership states with respect to the evolutionary factor f

b. APSO:

1. Inertia weight adaptation:

The inertia factor has an important role in balancing between global and local search. It is not seen efficient to just decrease ω with time. They propose that ω follows the evolutionary states using a sigmoid mapping:

$$\omega_f = \frac{1}{1 + 1.5e^{-2.6f}} \in [0.4,0.9] \qquad \forall f \in [0,1] \tag{6}$$

They considered $\omega_{initial} = 0.9$ in their work.

2. Acceleration coefficients:

Exploration (Strategy1): increasing c_1,and decreasing c_2 slightly since in this state the particles explore their own historical best positions rather than approaching the global best.

Exploitation (strategy2): increasing c_1,and decreasing c_2 slightly since in this state the particles are making use of their explored bests seeking more local niches for best solutions. Small c_2 may help in avoiding premature convergence by local optima deception.

Convergence state (strategy3): increasing both $c_1 and\, c_2$slightly to find the globally optimal solution by enhancing the global factor c_2 and then grabbing the particles to the possible global best.

Jumping out (strategy4): decreasing $c_1$and increasing c_2,when the global best jumps away to a better optimum. Its new position will be way far from the clustered particles.

$$\begin{gathered} |c_i(g+1) - c_i(g)| \leq \delta \qquad i = 1,2 \\ 0.05 \leq \delta \leq 0.1 \end{gathered} \tag{7}$$

CHAPTER III

METHODOLOGY

A. Problem Statement

Field development can be divided into subsurface and surface development. Subsurface development planning includes the optimization of the well count, type, trajectory, location and control. As for surface facility development planning, it includes the optimization of the production system layout design which involves the placement and allocation of drill centres, gathering centres, manifolds, flowlines, platforms and facilities. Regarding surface development planning, the production system constitutes a substantial portion of the overall cost of the field development hence engineers tend to come up with best design that minimize cost while providing an efficient development performance. In this course, the capacity, number, location and allocation of each element in the production system e.g. well head, manifolds, platforms and facilities must be optimized for obtaining the lowest cost possible.

The problem is represented by a multilayer node-segment where each node (i.e. manifolds, platforms, and processing facilities etc..) has its own capacity (or range of capacity) and a well-defined cost. Each segment will be joining a node from layer $i-1$ to another one from layer i. The main objective is to minimize the total cost of the production system which will require the election of the optimal number and positions of nodes in each layer, and the allocation of nodes between consecutive layers i.e. manifolds and wellheads. This problem is solved using a hybrid evolutionary algorithm that couples particle swarm optimization (PSO) with a clustering technique that makes the convergence faster and the solution more robust. The algorithm can start with a random guess within the search space or can be well defined initial guess (i.e. initial solutions from Clustering techniques such as Kmeans or can be the preliminary solutions of previous runs). Each particle (solution) represents a solution vector defining the optimization parameters (nodes positions, connection between nodes etc..).

Spatial search space is defined by an upper and lower bound defined by the location of the wellheads. The following is the schematic of multilayers of the production problem, it is represented by a tree-like problem as shown in Figure 5 for an offshore system.

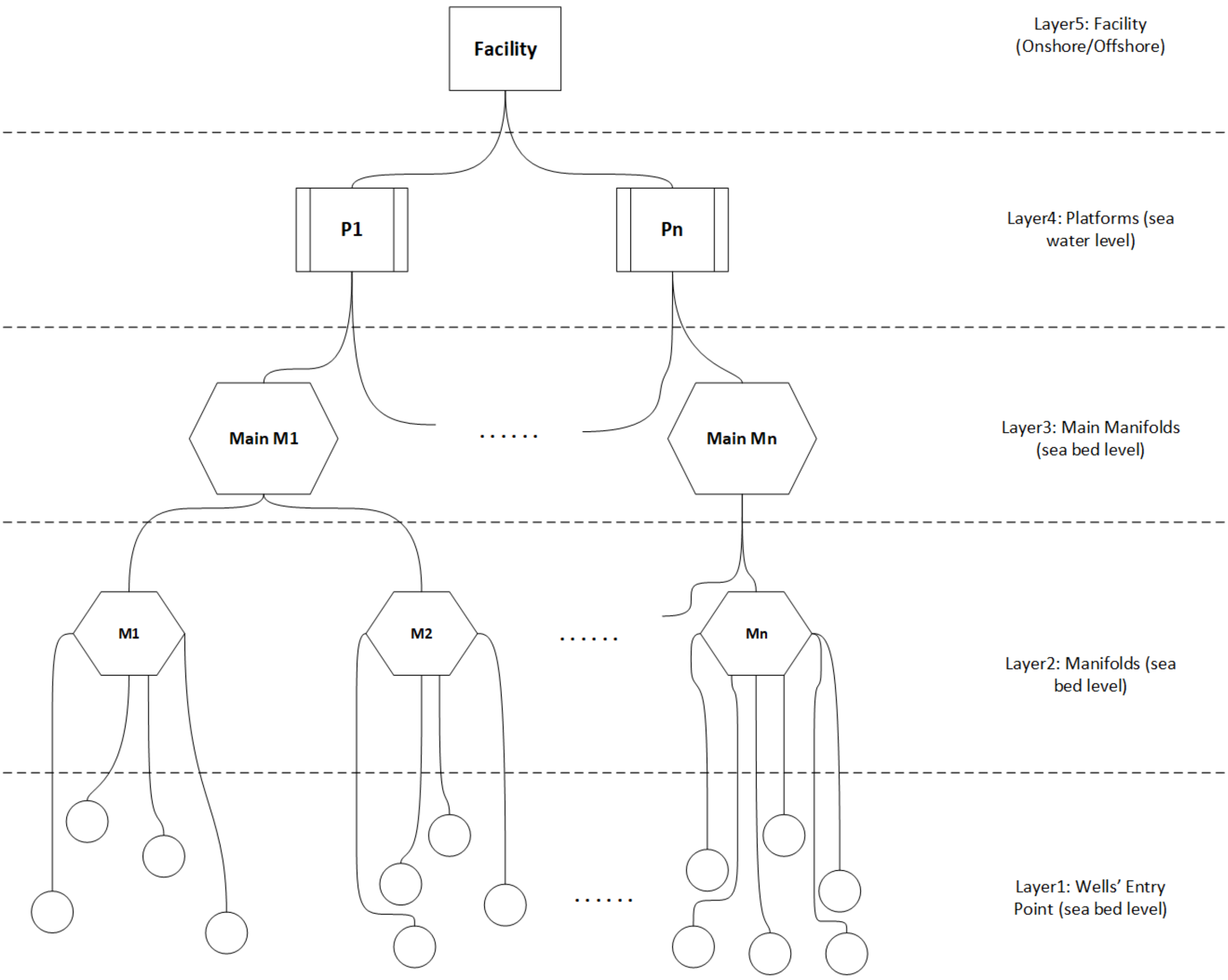


Figure 5- Schematic of the problem for an offshore field

1. Input Parameters

The input parameters define the number of layers of the problem and this can be changed depending on the predefined problem i.e. onshore production systems can be represented as two or three layers of optimization that may be the drilling sites / centers, gathering centers and processing facility, for instance. However, offshore problems may be much more complicated so it can be represented by three or more layers of optimization i.e. entry points, manifolds, platforms, facilities etc.…

Another input parameter that should be defined prior to solving the optimization problem is the map data. This data can be entered as an imported map i.e. using maps exported from Petrel or any other software. The other way is to define the lower and upper bounds of the map and the resolution (that defines the number of grids to which this map is divided) of the map. This two-dimensional developed map is used as the search space where nodes are to be put. The elevation of each layer is predefined as a constant to all nodes or can be defined as a three-dimensional map, however the elevation is not that important in the general flow of optimization. An important aspect of this methodology is that every layer can have its own map where prohibited areas can be excluded from the search space of this specific layer.

Well data such as the coordinates and the type of these wells are the starting points of this problem. These parameters are the output of well placement optimization processes that have been studied by many researchers. The type of the wells, i.e. horizontal or vertical, also has an impact on the algorithmic path for example in case of vertical wells only one entry point is possible however in case of horizontal wells two possible entry points should be considered when solving the problem.

Another input parameter is the nodes' number or range in each layer. This input can be predefined based on engineering best practices or can be tuned to get the optimal number of nodes within the range. This factor is related to another important input which is the capacity of nodes. In other words, how many nodes from layer $i-1$ can be assigned to each node in layer i. The nodes capacity input parameter can be defined also as a constant value or can be varied within a range. These two inputs affect the clustering process and the optimization process since the number of nodes in upper layer will define the number of clusters in the previous layer. In addition, the capacity of each node will also affect the number of elements in each cluster. Thus, these two factors have an impact on the whole optimization results as the number of clusters and the elements of each cluster dictates the total distance per layer $TD_{\ i}$ and then will affect the Total Cost.

The pricing for the nodes in each layer and for the segments between each two layers is also an important input variable. This factor will also affect the number and the capacity of the nodes. In addition, the positioning of the nodes will be affected by the pricing of the segment connecting it with the layers below and above it. For instance, if all segments had the same pricing, we expect that the node connected to a cluster of nodes in the previous layer should be located near the mean/ geometric median of the elements of this cluster. In contrary, if the pricing was not uniform the position of the node should be shifted towards the layer having the highest segment pricing. Assuming that the pricing of segments per unit length joining nodes of layer $i-1$ and layer i be $Cs_{\ i}$; Then, if $Cs_{\ i+1} > Cs_{\ i}$ this will lead to shifting nodes in layer i to their connected nodes in layer $i+1$ in order to minimize the total cost. Thus, the whole problem outcome is highly dependent on the estimated pricing of nodes and segments in each layer. Enlisted below are the input parameters that constitutes our optimization problem:

1. Number of optimization layers “N”, i.e. if we have in the problem (wells-manifolds-main manifolds-platforms) in this example we have $\mathrm{N} = 4$.
2. Search space boundaries and resolution
3. Wellheads’ data (i.e. wells’ number, coordinates, and types)
4. Range/ maximum number of nodes at every layer of optimization (manifolds, platforms, and facility).
5. Maximum Capacity (number/range) of each of the optimization nodes.
6. Pricing of each segment/node according to their capacities.

2. Optimization Parameters (OP)

The optimization parameters are the variables that should be varied towards optimizing the objective function of the problem (Total Cost, CAPEX, or NPV etc...). These parameters are seen to be critical in the defined problem, i.e. they are not fixed values and changing any of these will affect the objective function value.

The position of nodes in layers starting from entry point which is the first layer since wells’ position is an input parameter. Moving nodes as optimization parameters will dictates other parameters such as the connection between nodes. The positioning of nodes in layer n should take into consideration many variables such as the clustering of the connected nodes in layers $\mathrm{n} - 1$ and $\mathrm{n} + 1$ in order to minimize the TC. The election of these positions can be done using several methods such as choosing the positions of all nodes and then assign to them nodes from lower layer in this case positioning is prior to clustering and then reiterate until convergence this can be named PSO based Clustering. The other method is that nodes in each layer can be chosen based on clustering of the nodes of the lower layer this is a “Bottom up Clustering”. The first type of clustering considers the Total Cost and this due to its randomness and its dictation to PSO updates that considers the Best Total cost value. However, the Bottom up method may not fully consider the impact of positions of nodes in the upper layer on the positioning of nodes in any intermediate layer (Last layer is not affected by this factor). This will be comprehensively explicated in the Optimization Flow Section.

The connection between nodes, as mentioned, is dependent on all the other OPs. Positions, capacities, and number of nodes in each layer in addition to the pricing weights of segments between layers affect the connection of nodes between layers. For instance, if pricing of segments is uniform, a node in layer $\mathrm{n}-1$ must be connected to the nearest node in layer n until this latter's capacity is saturated. Moreover, if a specific node in layer n is connected to nodes in the upper and lower layers but the pricing of segments to the upper layer $Cs_{i+1} > Cs_i$. In this case, the node in layer n is shifted towards the upper layer $(\mathrm{n}+1)$ node and a node from the lower layer$(\mathrm{n}-1)$ may be connected to another neighbor, nearest node in order to minimize the total cost.

Number of nodes can be considered as an input or an optimization parameter according to the problem in hand. Pricing method (i.e. if the pricing is fixed for a specific node or may change with the capacity) and the capacity of nodes affect the election of the optimal number of nodes in each layer. The degree wells are spread on the map also affect the decision of how many clusters should be defined in the upper layer which is the number of nodes in this layer. Yet, the optimality is a trade-off between the cost of nodes in each layer and the segments price joining each two layers. Then, engineering assumptions or problem definition dictates whether nodes' number is presented as an input or as a range where it is considered as an optimization variable and is changed until optimality is met.

Another optimization parameter is the nodes' capacity that is related to the availability of each types of nodes in the market. For example, if present manifolds have a well-defined capacity in terms of number of connections or flow rate capacity in this case a physical limitation is present as an upper bound in terms of defining the capacity of this specific type of nodes. The same applies for all other layer's elements. Also, in this case it is problem dependent to assign a specific value of capacity for each node or use a range where this value can be optimized. The capacity of each node is highly dependent on the number of nodes and the degree of spread of the nodes in lower layer. In this problem, the capacity is optimized within a range. Yet, if any node exceeds its upper bound capacity, the excess in allocated nodes from the lower layer are assigned to other not saturated nodes in a way that optimality or sub-optimality of the run is conserved. Other ways may be applied to meet physical limitations such as penalty on outliers' solutions.

Following is the list summarizing the optimization parameters under study in this specific problem:

1. Position of nodes in each layer.
2. The connection between nodes of layer $\mathrm{n}-1$ and layer n.
3. Number of nodes at each level of optimization
4. Capacity of each node

3. Objective Function

The objective function is the total cost (TC) according to the pricing of all nodes and segments. The main aim in this work is to minimize the total cost which forms a part of the CAPEX. So, minimizing the total cost will lead to minimizing the CAPEX and then maximizing the NPV. The following is the mathematical representation of the cost function.

$$TC = \sum_{i=1}^{N} [(Cn_i \times n_i) + (Cs_i \times TD_i)] \tag{8}$$

Where:

$TC: is\ the\ total\ cost$ (\$)

$N: is\ the\ number\ of\ optimization\ layers$

$Cn_i: is\ the\ cost\ nodes\ in\ layer\ i (\$/node)$

$n_i: is\ the\ number\ of\ nodes\ in\ layer\ i$

$Cs_i:\ is\ the\ cost\ of\ segments\ (per\ unit\ length) between\ layer\ i - 1\ and\ layer\ i\ \left(\frac{\$}{m}\right)$

$TD_i: is\ the\ Total\ distance\ between\ nodes\ of\ layer\ i - 1\ and\ layer\ i\ (m)$

B. Proposed Optimization Algorithm

1. Optimization Workflow

In this section, the general workflow of the proposed algorithm is described. This workflow involves seven sequential steps that include input datafile preparation, search space setting, algorithmic parameter initialization, nodes assignment, objective function evaluation, convergence checking and algorithmic parameters update. Below is a detailed description of each of these steps in the proposed optimization algorithm.

Step 1: Read Data file

The input parameters defining the problem such as wells' data, search space data, ranges of optimization parameters, and algorithmic parameters related to the PSO algorithm (population size, cognitive and social coefficients, inertia weight etc..).All these parameters are stored in and read from a predefined excel spreadsheet which represents the datafile of the problem.

<u>Step 2: Set Search Space</u>

Upper and lower bounds of platforms and manifolds locations coordinates are set in the datafile. These limits can be set to narrow the search space for faster convergence and can be considered as the search space since neither a platform nor a manifold should, basically, be located outside the field's boundaries. Typical boundaries can be defined as:

$$X_{lower} = \min(x_{wells}), X_{upper} = \max(x_{wells})$$

$$Y_{lower} = \min(y_{wells}), Y_{upper} = \max(y_{wells})$$

$$X_{lower} < X_p(i) < X_{upper}$$

$$Y_{lower} < Y_p(i) < Y_{upper}$$

<u>Step 3: Initialization/Update of Optimization Parameters</u>

The main optimizer used in this problem is the particle swarm optimization algorithm. This population-based algorithm as described in Chapter 2starts by initializing a predefined number of particles/solutions inside the swarm. These particles represented by vectors defining all the optimization variables involved in the problem. An initial guess is randomly obtained within the search space defined by the upper and lower boundary positions, and specific ranges of other optimization parameters such as nodes capacity.

Initialization can be also an input from previous optimization runs or any other suboptimal solutions that may help in getting faster to the optimal solution.

Step 4: Nodes Connection Assignment Using Clustering method

Assignment of nodes between nodes of layer i and nodes of layer $i + 1$ is done based on two main criteria i.e. the proximity of nodes in layer i to all nodes in layer $i + 1$ and the maximum capacities of nodes in layer $i + 1$.

Starting from layer of $i = 1$ and using a bottom up assignment method, distances of all nodes in layer i to all nodes of layer $i + 1$ are calculated and stored in a matrix. Note that, distances are calculated according to the nodes in layer i to those in layer $i + 1$. In this sense, the pivot is the distance from nodes of previous layer to the nodes of next layer. In other words, if the pivot was the nodes in the higher level the clustering is then based on the proximity of nodes in layer $i + 1$ to the nodes in layer i. Then, if we consider wells-manifolds level, in case of up-bottom clustering each manifold is seen to which wells is near and assigned accordingly. However, if the bottom up method is applied, wells are seen to which manifold are near and the assignment is done accordingly. In this way, the least distance can be achieved and in the same time if one manifold is not assigned to any of the wells it can be eliminated. Thus, the number of nodes can also be optimized in parallel to the distance optimization.

After calculating the bottom up distances, these distances are ranked in an increasing way in order to choose in an optimized way the connectivity between nodes of different layers.

Each node in layer i is assigned to the nearest node in layer $i + 1$ taking into consideration the maximum capacity of these latter nodes. In other words, nodes are assigned until the nodes in the higher level are saturated. Saturated nodes cannot support any additional connection.

In this Step, used nodes are identified and a check of the assignments of nodes is done. In case not all nodes are assigned the algorithm will proceed to Step 5. Unless, meaning that all nodes of the considered layer are assigned, layer index is updated layer ($i = i + 1$). Sequentially, another check of whether the layer index $i < total\ number\ of\ layers$ or not. If this criterion is met the algorithm will continue to Step 2 to assign the nodes of the next level. Unless, the process will be ended and all nodes in all layers are then assigned.

In this step, used nodes in the lower layer are eliminated and the saturated nodes of the higher level are also eliminated. In addition, the maximum capacity of each node in the higher level is also updated ($\max Capacity_{j+1} = \max Capacity_j - nodes\ assigned_j$). Where: j is the number reiterations until all nodes of layer i are assigned.

After eliminating the used elements in the lower layer and the saturated nodes in the upper layer. Lower and upper layers are updated accordingly to undergo a new assignment process according to their updated attributes. Thus, the process will go to the Step 2 again with updated pools of the same layers' indices.

Using this method of clustering two important features were added. In one hand, controlling nodes' capacity in an efficient way by only defining the maximum capacity so that a more room to consider the optimized distance is available. In the other hand, some nodes will not be connected to any of the lower layer nodes, and then can be eliminated. Thus, number of nodes is also controlled in parallel with the total distance reduction.

Step 5: Objective Function Evaluation

After deciding number of nodes in each layer and assigning connections between nodes. The evaluation of the Cost Function (Total cost) of all the swarms/solutions is done using the defined Objective Function “TC”. This factor affects, as aforementioned, both the CAPEX, the productivity of wells, and, hence, the NPV.

Step 6: Checking Convergence

Convergence of the algorithm is the key factor of its efficiency. Stochastic methods such as PSO and GA handle complex problems, though no clear criterion is set for their convergence. They mainly rely on the number of iterations or some criteria that should be set according to the problem under study. Following is the function that checks convergence of the optimizer after all the parameters are set at each iteration. Convergence criteria is considered as the maximum number of iterations that is predefined as an input.

Step 7: PSO Parameters Update

The results of all solution in the run are stored and ranked in order to define the Global Best and the particle Local Best for all the swarms/solutions of the population under study. After determining the Global Best and Local Best. The Velocity term is updated using its definition from the PSO algorithm. The Velocity is a vector defining the change in each element of the position vector which represents the particle/solution. The position vector for each solution is updated in **Step 3.**

Continue from Step3.

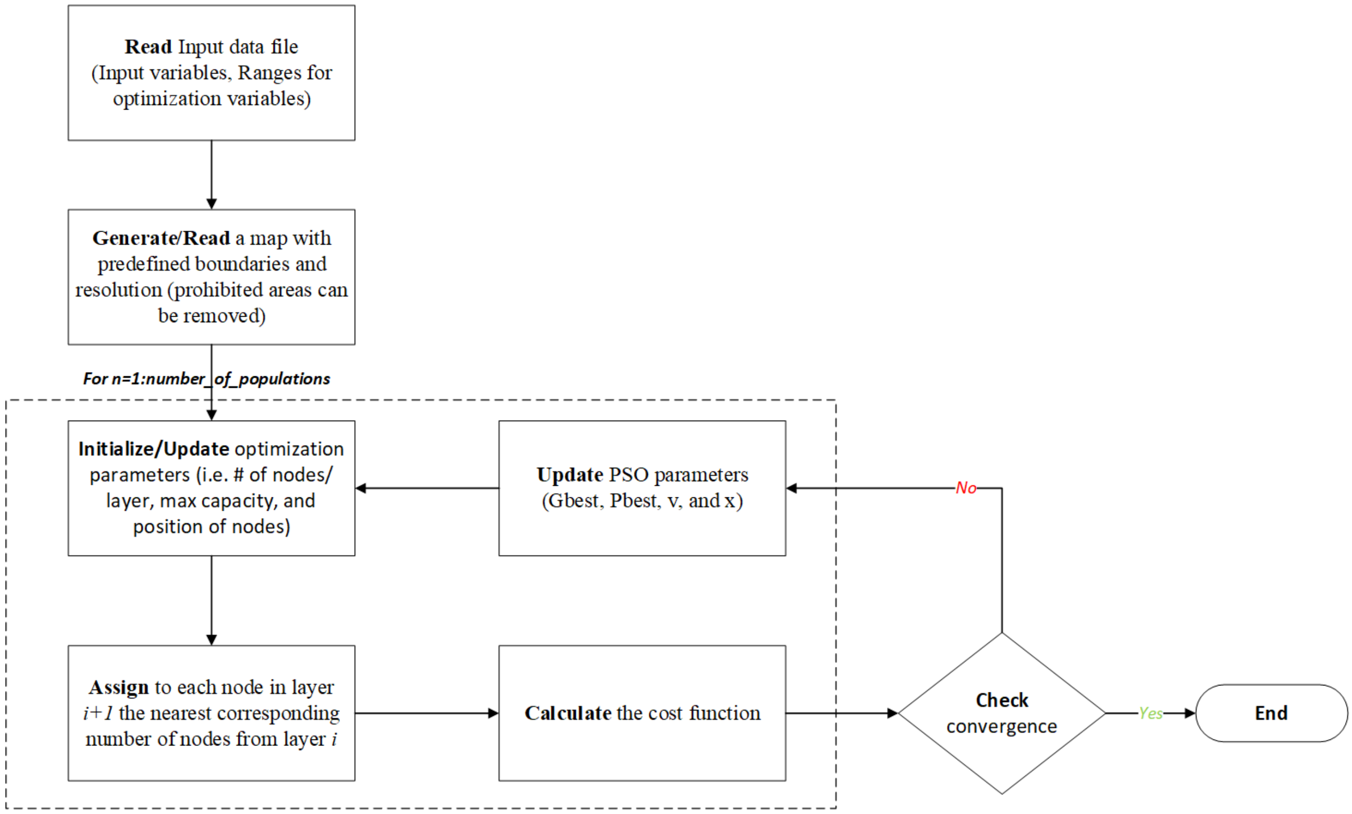


Figure 6– PSO General Workflow

a. <u>Clustering Function (Step 4)</u>

Assignment of nodes between nodes of layer i and nodes of layer $i+1$ is done based on two main criteria i.e. the proximity of nodes in layer i to all nodes in layer $i+1$ and the maximum capacities of nodes in layer $i+1$.

1. Starting from layer of $i=1$ and using a bottom up assignment method, distances of all nodes in layer i to all nodes of layer $i+1$ are calculated and stored in a matrix. Note that, distances are calculated accroding to the nodes in layer i to those in layer $i+1$. In this sense, the pivot is the distance from nodes of previous layer to the nodes of next layer. In other words, if the pivot was the nodes in the higher level the clustering is then based on the proximity of nodes in layer $i+1$ to the nodes in layer i. Then, if we consider wells-manifolds level, in case of up-

bottom clustering each manifold is seen to which wells is near and assigned accorindgly. However, if the bottom up method is applied, then the wells are seen to which manifold are proxim and the assignment is done accordingly. In this way, the least distance can be achieved and in the same time if one manifold is not assigned to any of the wells it can be eliminated. Thus, the number of nodes can also be optimized in parallel to the distance optimization.

2. After calculating the bottom up distances, these distances are ranked in an increasing way in order to choose in an optimized way the connectivity between nodes of different layers.
3. Each node in layer i is assigned to the nearest node in layer $i + 1$ taking into consideration the maximum capacity of these latter nodes. In other words, nodes are assigned until the nodes in the higher level are saturated. Saturated nodes cannot support any additional connection.
4. In this Step, used nodes are identified and a check of the assignments of nodes is done. In case not all nodes are assigned the algorithm will proceed to step 5. Unless, meaning that all nodes of the considered layer are assigned, layer index is updated layer($i = i + 1$). Sequentially, another check of whether the layer index $i < total\ number\ of\ layers$ or not. If this criterion is met the algorithm will continue to Step 2 to assign the nodes of the next level. Unless, the process will be ended and all nodes in all layers are then assigned.
5. In this step, used nodes in the lower layer are eliminated and the saturated nodes of the higher level are also eliminated. In addition, the maximum capacity of each node in the higher level is also updated ($\max Capacity_{j+1} = \max Capacity_j -$

$nodes\ assigned_j$). Where: j is the number reiterations until all nodes of layer i are assigned.

6. After eliminating the used elements in the lower layer and the saturated nodes in the upper layer. Lower and upper layers are updated accordingly to undergo a new assignment process according to their updated attributes. Thus, the process will go to the Step 2 again with updated pools of the same layers' indices.

Using this method of clustering, two important features were added. In one hand, controlling nodes' capacity in an efficient way by only defining the maximum capacity so that a more room to consider the optimized distance is available. In the other hand, some nodes will not be connected to any of the lower layer nodes, and then can be eliminated. Thus, number of nodes is also controlled in parallel with the total distance reduction. Figure 7 represents the workflow of the proposed clustering method.

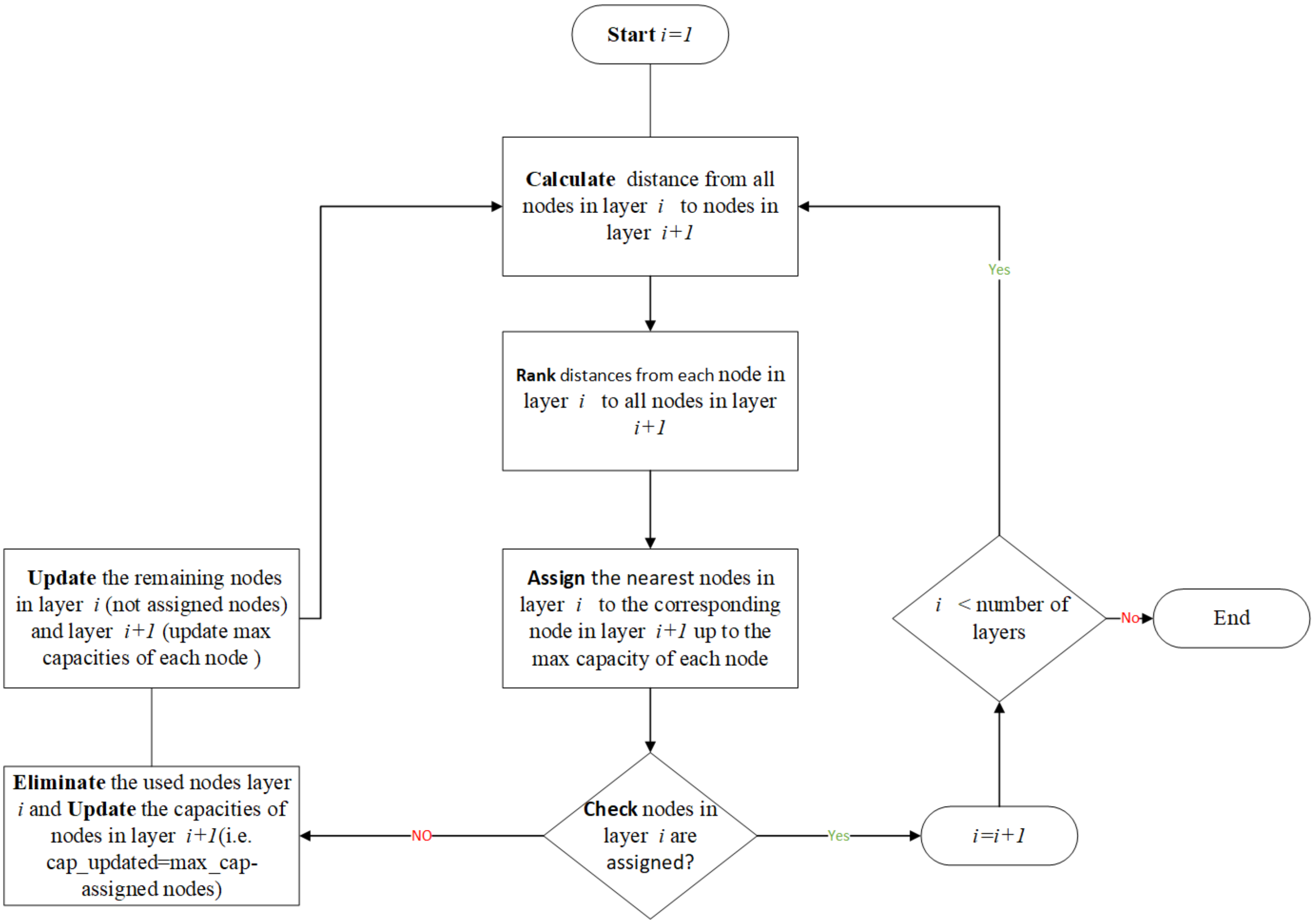


Figure 7- Clustering function workflow

2. Improvements on Standard PSO

Based on the superiority of the PSO algorithm compared with GA, PSO was considered as the main optimization tool in this work. Several improvements were introduced in the literature, as described in literature review section, to enhance the efficiency in terms of convergence and time.

a. <u>Adaptive Particle Swarm Optimization (APSO)</u>

Adaptive Particle Swarm Optimization (APSO) showed superior to the Standard PSO. Thus, APSO is considered as the main optimizer. APSO, as described in the literature review, is an improved version of PSO that focuses on determining the evolutionary state of the solutions. After determining the evolutionary state of the problem, algorithmic parameters are adapted accordingly. This will ensure a better and smooth convergence of the problem. Figure **8** represents the steps of the APSO algorithm applied to the defined multilayer problem.

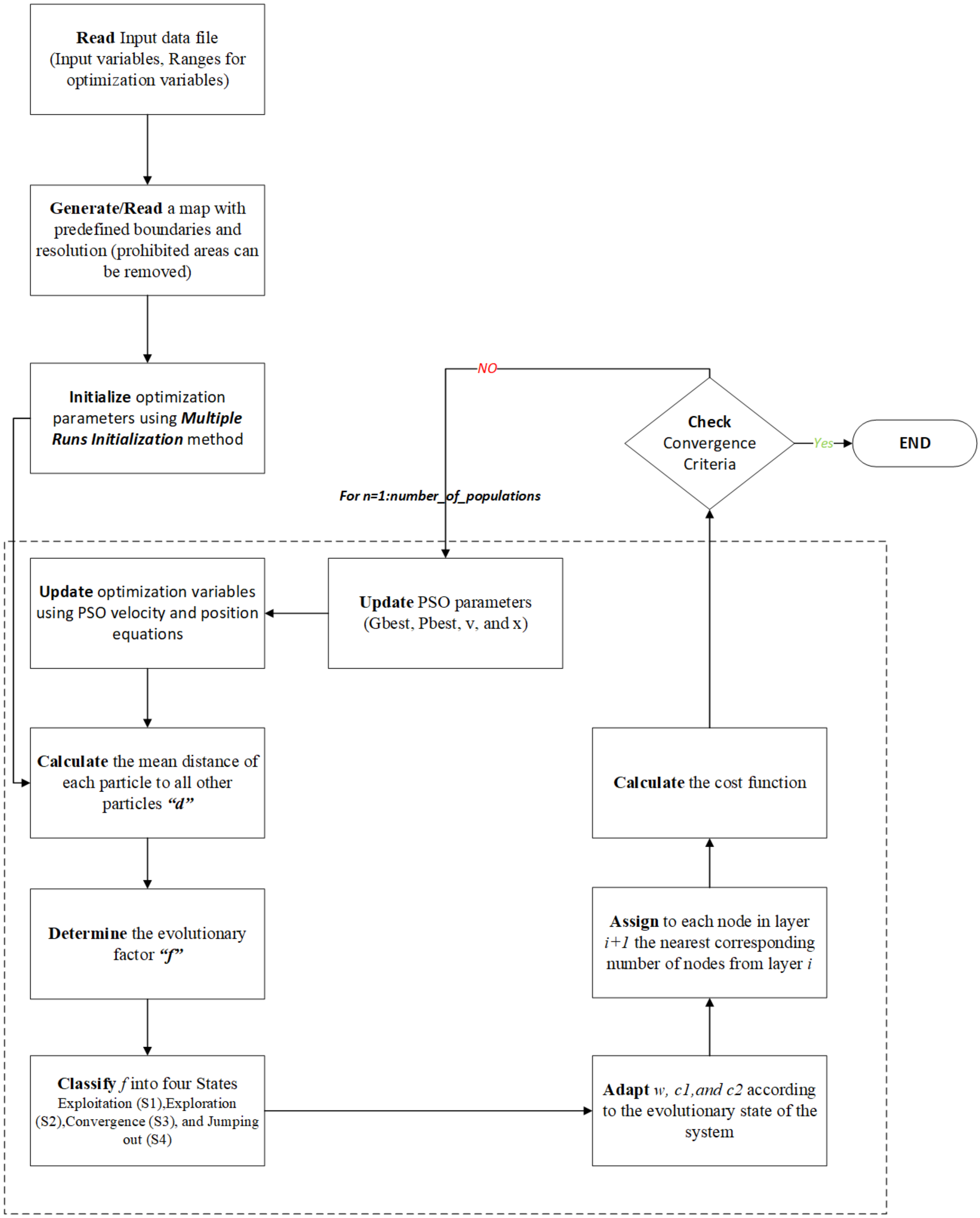


Figure 8- APSO algorithm workflow

b. Multiple Runs Initialization

Initialization of the initial guess in the PSO algorithm is considered as a main factor affecting the results of the algorithm. Hence, initialization improvement was introduced in this context the “Multiple Runs Initialization”. Generally, PSO consists of a population of solutions that are initialized randomly. These initial guesses are updated using the PSO velocity and position equations.

This introduced improvement consists of the running APSO algorithm multiple times but for few generations (i.e. 200 generations). The best result of each run is stored as one initial solution. The process is repeated until all initial particles (solutions) are initialized. Based on this initialization, the main optimization loop starts from a “relatively good” initial guess.

Figure 9 describes the workflow of multiple runs initialization that is applied prior to the main optimization loop:

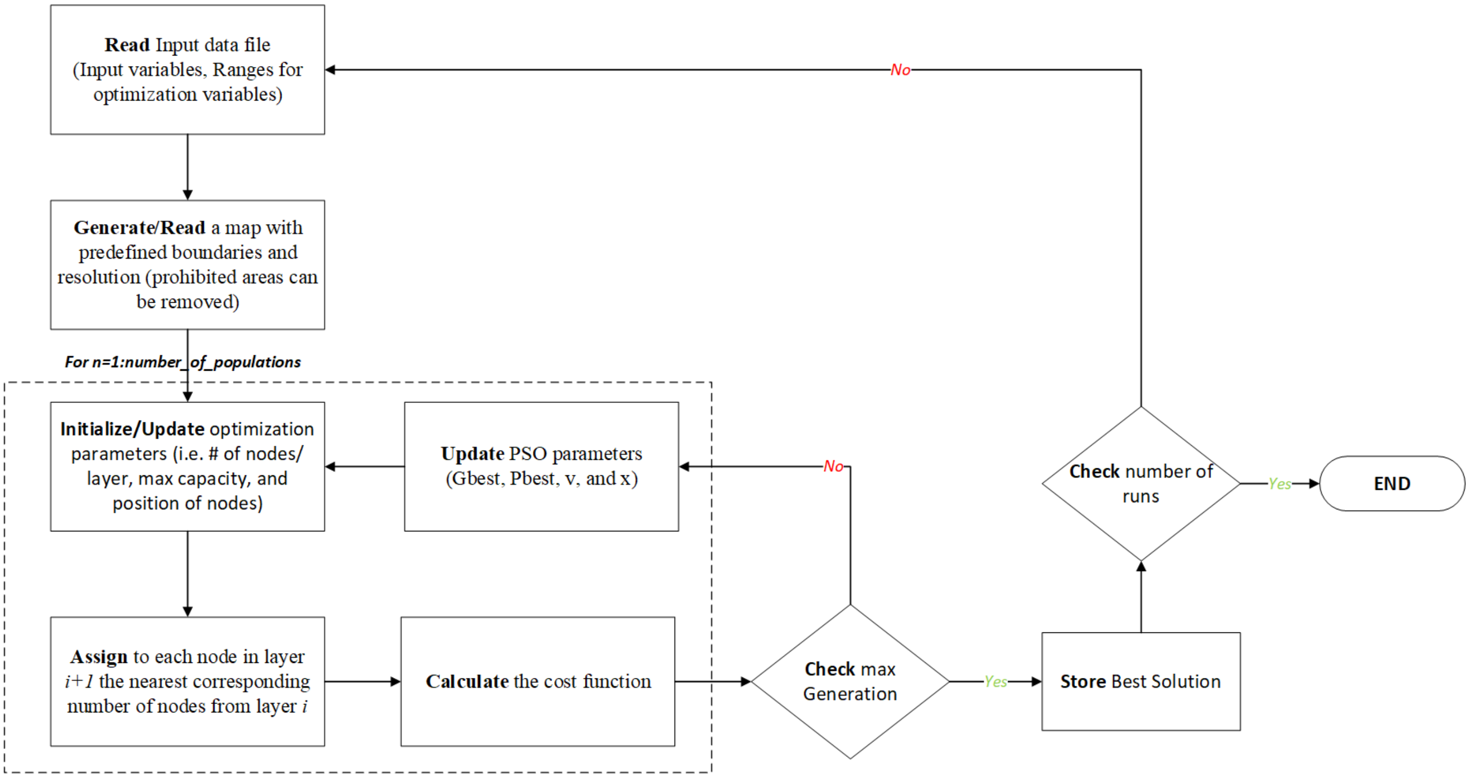


Figure 9- Multiple runs initialization workflow

c. <u>Restart</u>

Another improvement was added to the APSO algorithm the "Restart" step. Restarting the particles of a population is done after facing the pre-convergence plateau. This method was introduced to mutate the present solutions by reinitializing the population randomly, but, the history of the particle's best and Global best. The new mutation to the PSO solution will lead to a better convergence by introducing new random solutions to the problem after having a constant solution for a certain number of generations. This number of generations where the solution is not converging is predefined as an input to the problem. This parameter could be further tuned depending on the problem under study.

Figure 10 presents the complete proposed algorithm that is introduced in this work, this algorithm is compared with other algorithms and its superiority was verified.

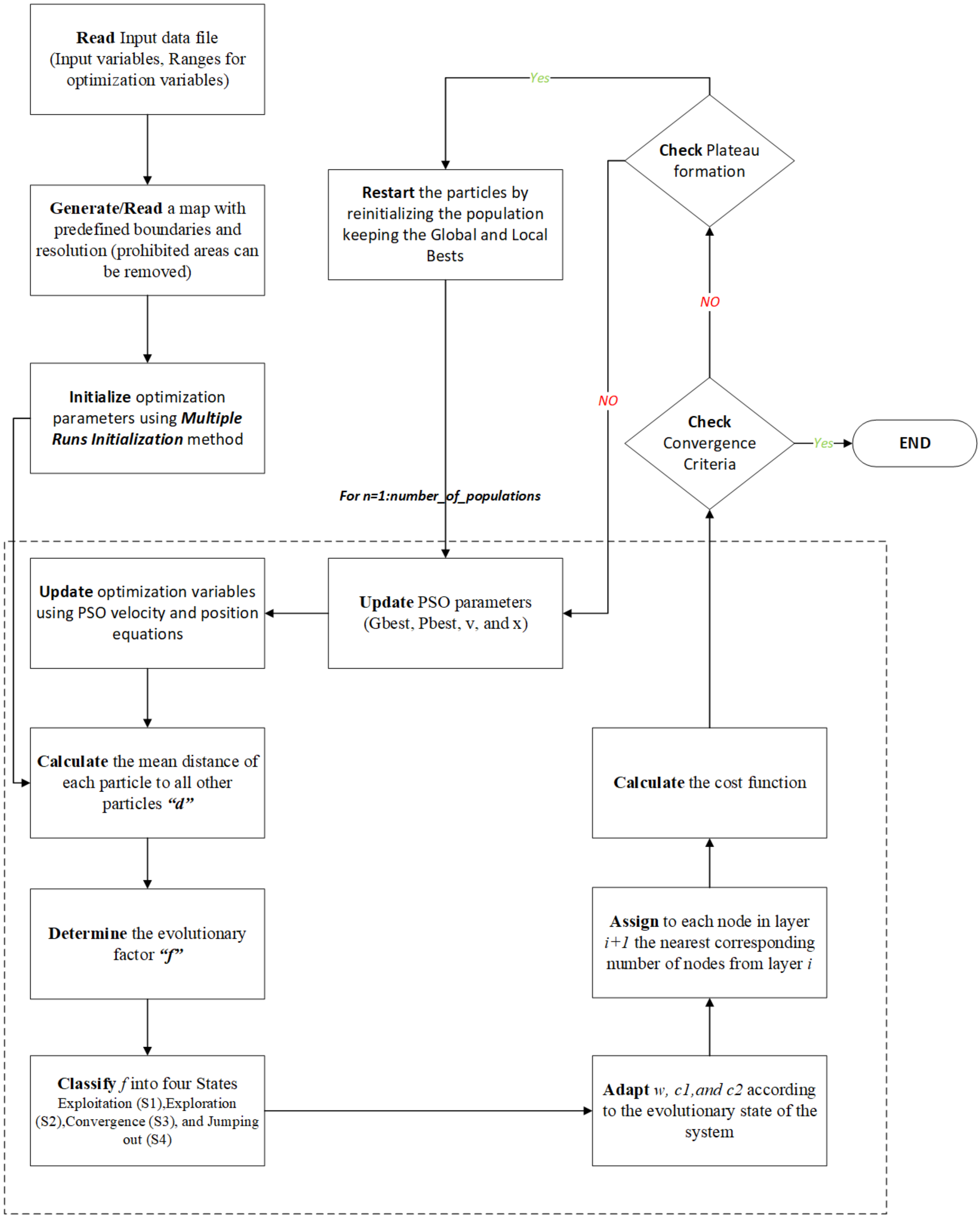


Figure 10- Proposed Algorithm complete workflow

CHAPTER IV

RESULTS AND DISCUSSION

In this section, we compare the performance of three different optimization algorithms; Deterministic Optimization Algorithm (DOA), Genetic Algorithm (GA) and Particle Swarm Optimization (PSO) on 2 layered production system problem where the location of platform was altered while fixing the location of a predefined number of wells. DOA is a method that considers all possible combinations and evaluates the Objective Function of each. The OLYMPUS case is used for illustration for many cases [39]. In order to compare the performance and limitations of each of the employed methods, problems with different levels of complexities are considered.

A. GA Vs. PSO

In this section, we compare the performance of three different optimization algorithms; Deterministic Optimization Algorithm (DOA), Genetic Algorithm (GA) and Particle Swarm Optimization (PSO) on 2 layered production system problem where the location of platform was altered while fixing the location of a predefined number of wells. DOA is a method that considers all possible combinations and evaluates the Objective Function of each. The OLYMPUS case is used for illustration for many cases [39]. In order to compare the performance and limitations of each of the employed methods, problems with different levels of complexities are considered.

1. Case 1: Varied Number of Wells, 1 Platforms

We optimized the location of one platform with different numbers of wells for validation purpose. For this example, the three optimization methods yielded the same results regardless of the number of wells (up to 20 wells) as illustrated in Figure 11 for eight wells.

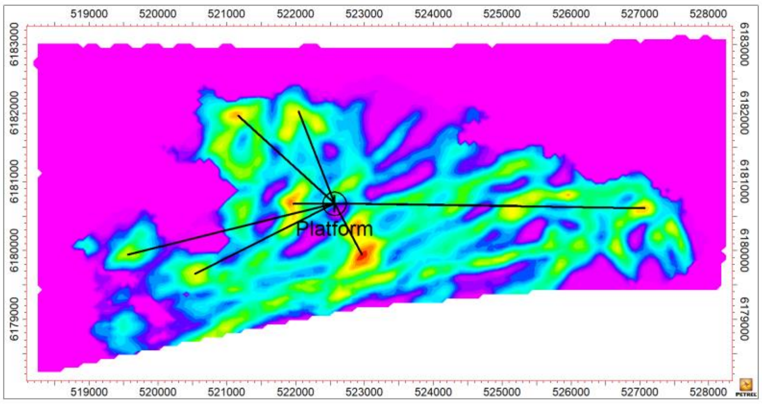


Figure 11– Single Platform location optimization using the three optimization methods (8 wells)

A major limitation in terms of CPU and memory was noticed when applying the DOA which made it incompatible to the proposed evolutionary algorithms. The time consumed by the deterministic method increases exponentially with the number of wells and exceeded 650 minutes for no more than 26 wells as illustrated in Figure 12. For a larger well number memory limitation is faced, and this method is no longer valid. On the other hand, PSO and GA are not affected by the number of parameters (number of wells and platforms) as illustrated in Figure 13.

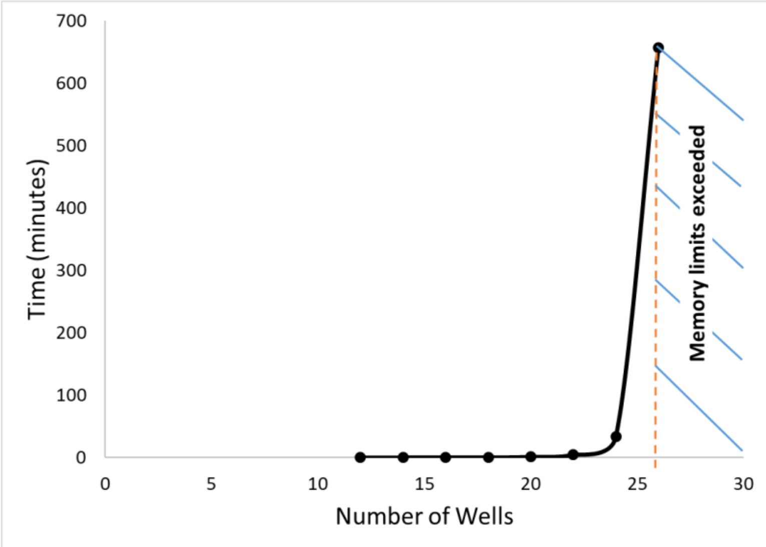


Figure 12– Deterministic method time performance

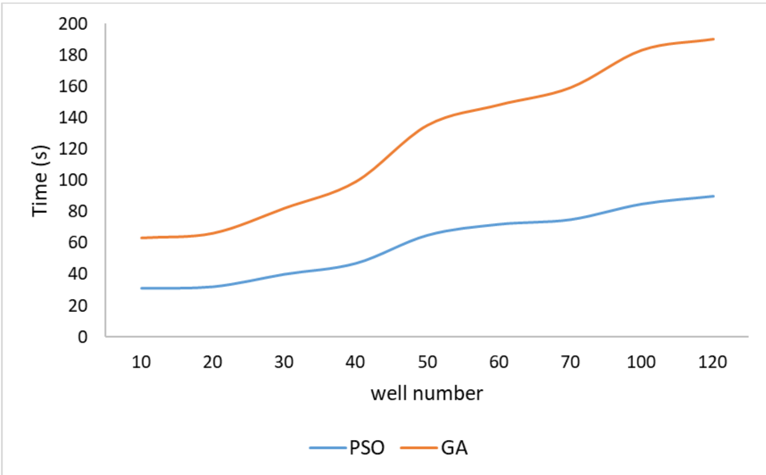
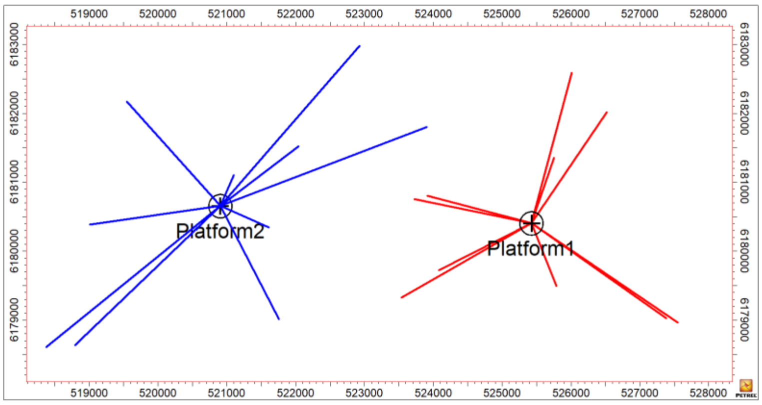


Figure 13– GA vs. PSO performance

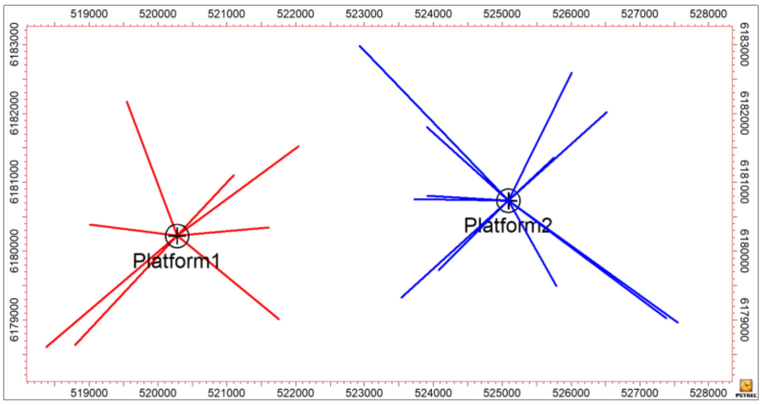
After excluding the deterministic methods from the comparison, we analyzed the performance of both evolutionary algorithms, PSO and GA in terms of consistency and convergence time at different level of complexity.

2. Case 2: Vertical Wells – 20 Wells, 2 Platforms.

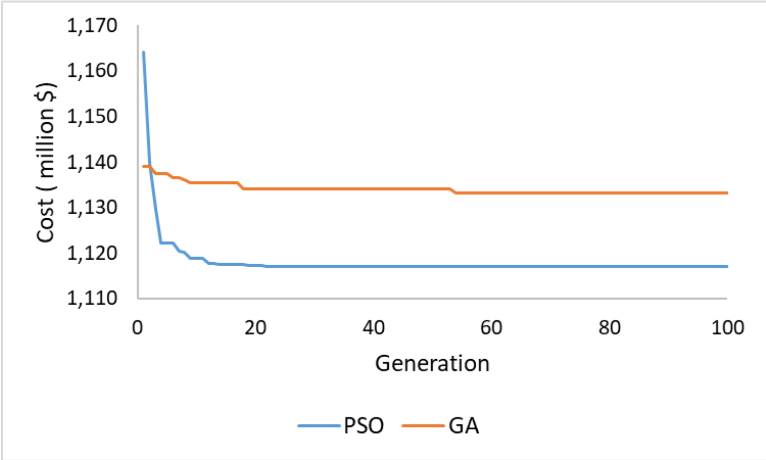
PSO showed superiority over GA. PSO converged to its best solution that minimizes the objective function; the total cost. GA converges to a sub-optimal solution; 15 million $ larger than PSO. PSO more efficient than GA in optimizing the well number corresponding to each platform. PSO assigns 8 wells for Platform1 and 12 wells for Platform 2 (Figure 14–B) while GA just spitted the wells equally between the two platforms (Figure 14–A).



A – Solution using GA



B – Solution using PSO

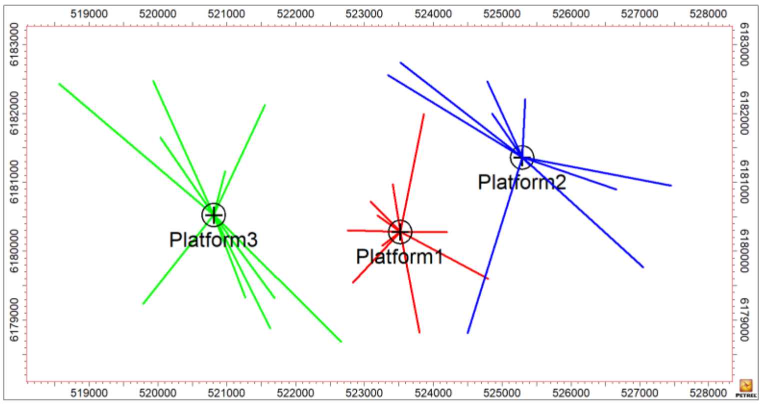


C – GA vs. PSO Performance

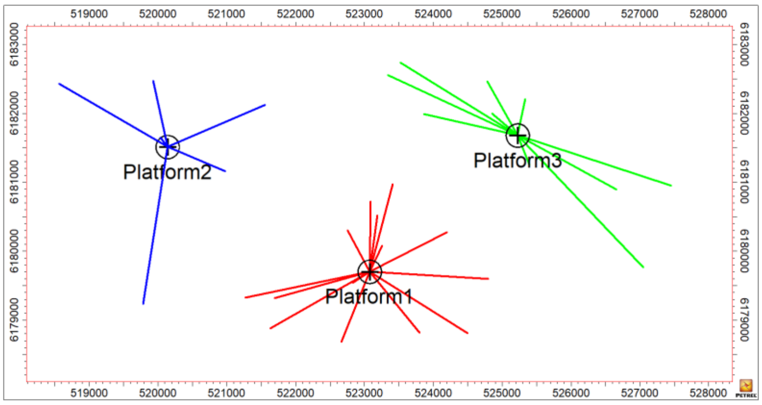
Figure 14 – Case 1, Vertical Wells: 20 Wells, 2 Platforms

3. Case 3, Vertical Wells – 30 Wells, 3 Platforms.

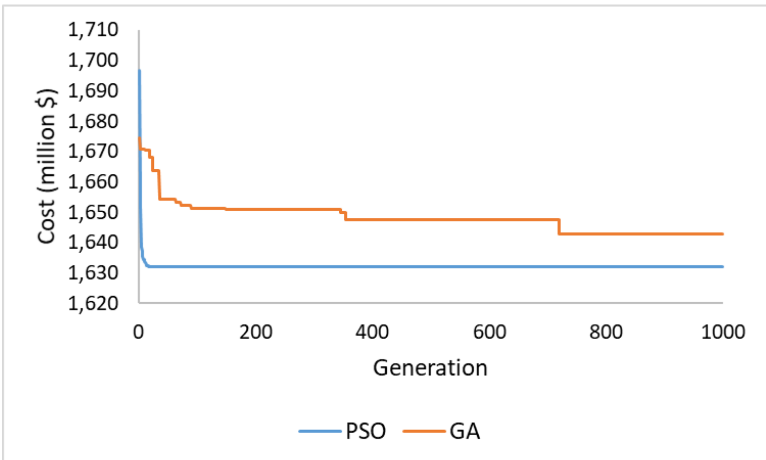
The level of complexity is upgraded so that we can compare both methods' performance. In this case we consider the presence of 30 defined wells subject to three platforms that should be optimized in terms of location and number of wells assigned to each of them. Results are shown in Figure 15. GA assigns equal number of wells to both platforms as illustrated in Figure 15–A while PSO assigns 14 wells to Platform 1, 10 to Platform 2 and 6 to Platform 3. PSO shows a better performance than GA: PSO converged in less than 50 generations while GA did not converge even after 700 generations. Furthermore, the optimal solution reached by PSO clearly outperforms that reached by GA.



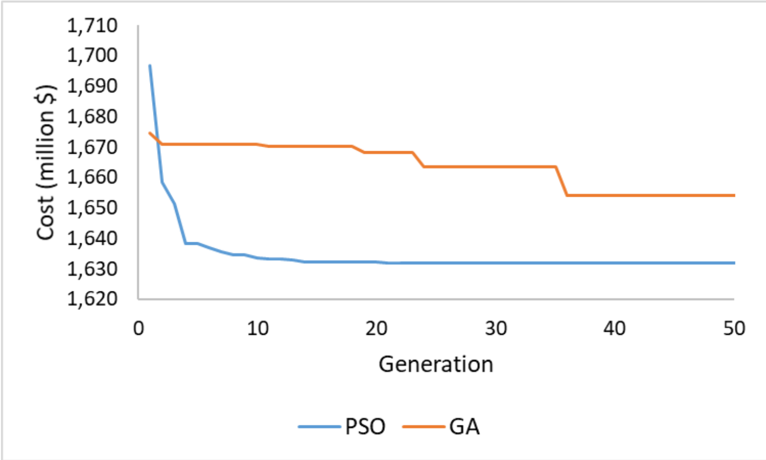
A – Solution using GA



B – Solution using PSO



C – GA vs. PSO Performance

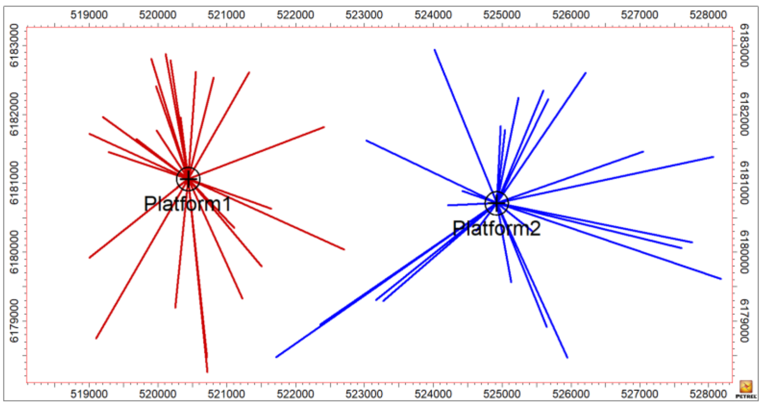


D – GA vs. PSO Performance. Zoom on the first 50 iterations.

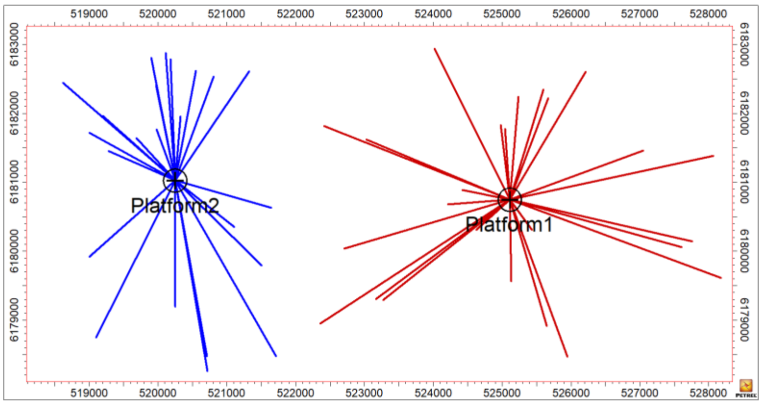
Figure 15 – Case 2, Vertical Wells: 30 Wells, 3 Platforms

4. Case 4, Vertical Wells – 50 Wells, 2 Platforms.

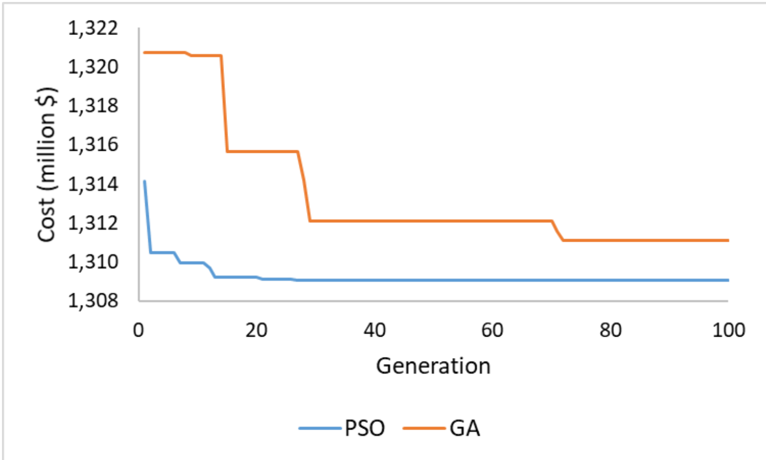
As beforementioned PSO vs. GA optimization performance in terms of the total wells distances is now compared at a higher level of complexity. In this case 50 wells are considered that should be assigned to 2 platforms. Results are shown in Figure 16 and illustrate the higher performance of PSO compared to GA both in terms of the optimality of the reached solution and the required number of iterations to reach this solution.



A – Solution using GA



B – Solution using PSO

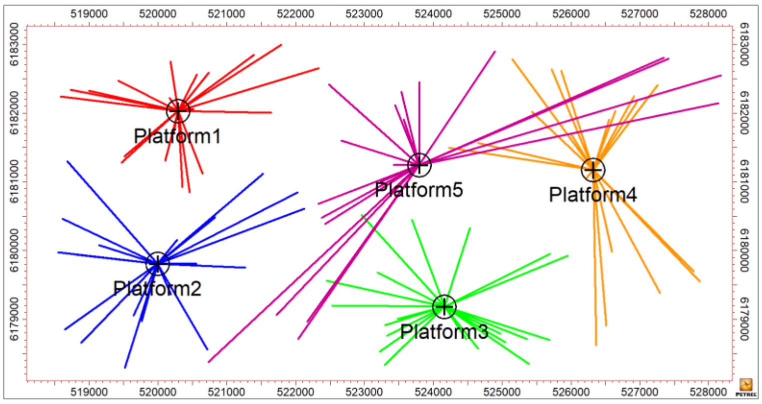


C – GA vs. PSO Performance

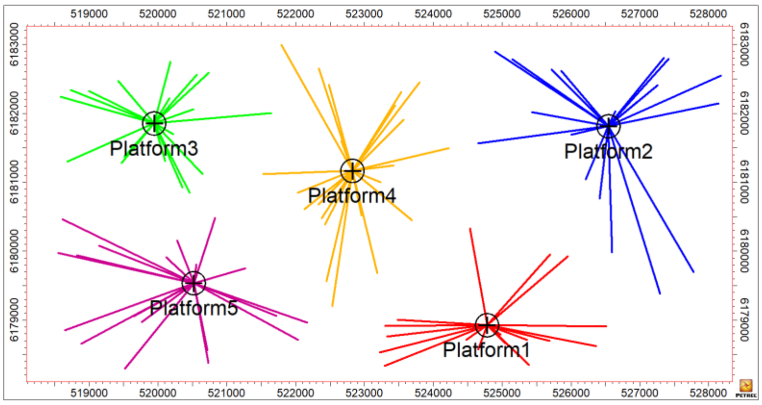
Figure 16 – Case 3, Vertical Wells: 50 Wells, 2 Platforms

5. Case 5, Vertical Wells – 100 Wells, 5 Platforms.

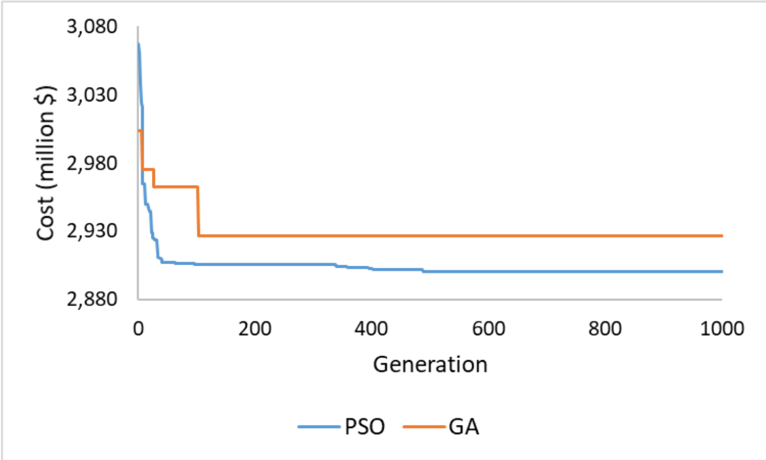
In this case we consider a highest level of complexity by considering 100 defined wells that should be clustered and connected to 5 different platforms. The number of decision variables exceeds 100 in this case which requires more generations by both methods to reach optimal solutions. As shown in Figure 17, PSO reached optimal solution after about 600 generations (iterations) however the GA is stuck in a local optimum that makes it far from convergence in this exercise. From Figure 17–A, we can see that GA is far from an optimal solution as depicted by wells sub-optimally connected to platforms. This is not the case for PSO where "visual" verification does not show this issue; wells appear to be connected to the right platform.



A – Solution using GA



B – Solution using PSO

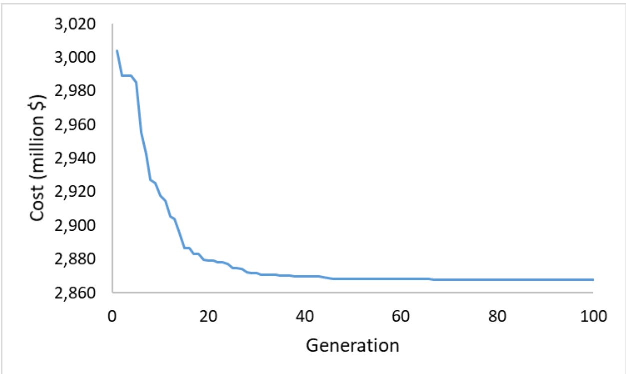


C – GA vs. PSO Performance

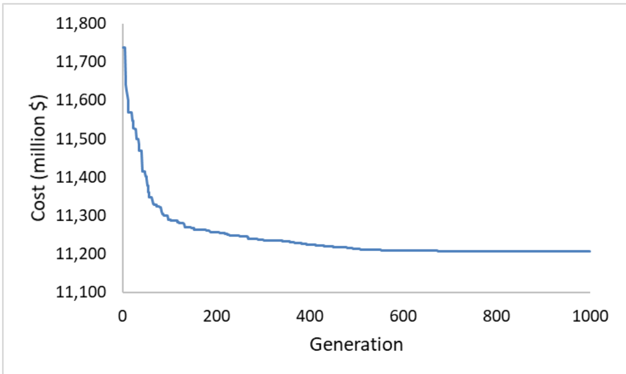
Figure 17 – Case 4, Vertical Wells: 100 Wells, 5 Platforms.

6. Stress Test of the Proposed Algorithm.

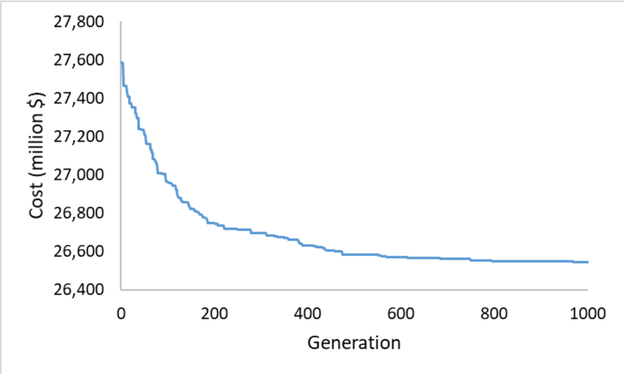
Unlike the limitations encountered and reported in the literature on the feasibility and efficiency of the optimization problem [1, 7-12], our proposed algorithm converged regardless of the level of complexity up to the 5000 wells, 100 platforms limit case (Case 9) that we have considered. Results are presented in Figure 18 and Table 2. These results confirm the robustness of the algorithm and its suitability to be used for any practical platform placement optimization task.

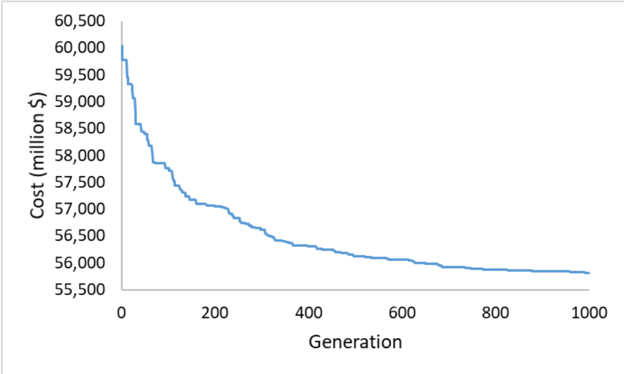


A – Case 4: 100 wells, 5 Platforms



B – Case 5: 500 wells, 20 Platforms





C – Case 6: 1000 wells, 50 Platforms

D – Case 7: 5000 wells, 100 Platforms

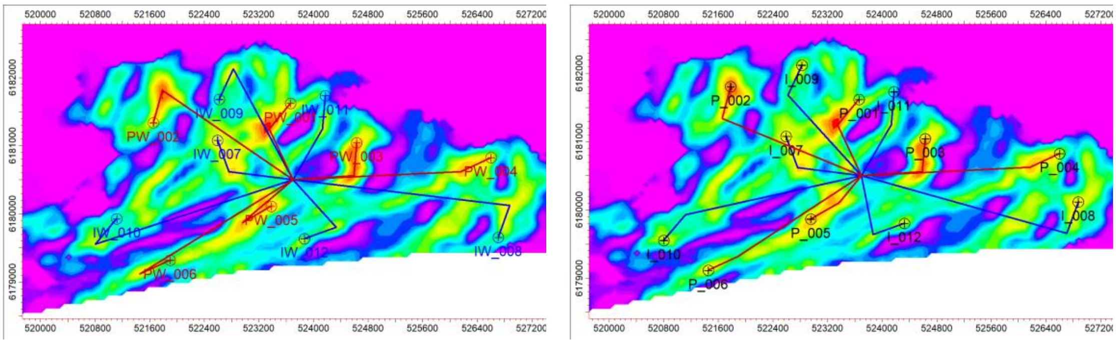
Figure 18 – Stress test of the proposed platform optimization algorithm, Results of PSO optimizer for high number of platforms and wells.

Table 2—Summary of the proposed algorithm stress test results (vertical wells).

Case/Performance	**Time/Generation (s)**	**Number of generations**	**CPU Time(s)**
Case 4: 100 wells, 5 platforms	0.1	400	40
Case 5: 500 wells, 20 platforms	1.1	600	660
Case 6: 1000 wells, 50 platforms	5	800	4000
Case 7: 5000 wells, 100 platforms	20.657	1000	20657

7. Horizontal Wells.

Platform location optimization in the case of horizontal wells is to ensure a minimum platform-to-heel measured depth. That is, the optimizer needs to select, among other parameters, the toe and heel of the well based on that criteria (see Figure 19 for illustration). One extra parameter, by well, is added to the optimization problem which adds, in turn, to the complexity of the problem. The PSO algorithm is modified to account for the horizontal wells and shows very good convergence and consistently smooth results. The efficiency, robustness and consistency of the used algorithm are illustrated on Figure 20 and Figure 21 for 50 wells, 5 platforms (Case 8) and 100 wells, 5 platforms (Case 9), respectively.



A – 6 producers, 6 injectors – Original well direction

B – 6 producers, 6 injectors – Switched well direction

Figure 19 – Illustration of the extra parameter associated to platform placement optimization in the case of horizontal wells: Decision of the heal and toe of the well to minimize the platform-to-toe measured depth.

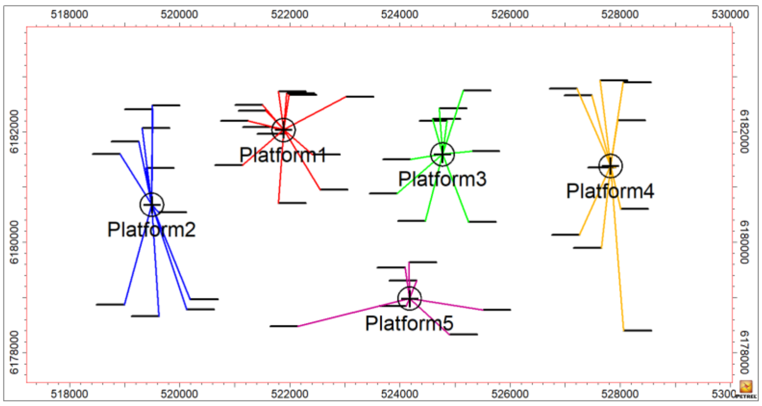


Figure 20 – Case 8, Horizontal wells: 50 wells, 5 Platforms.

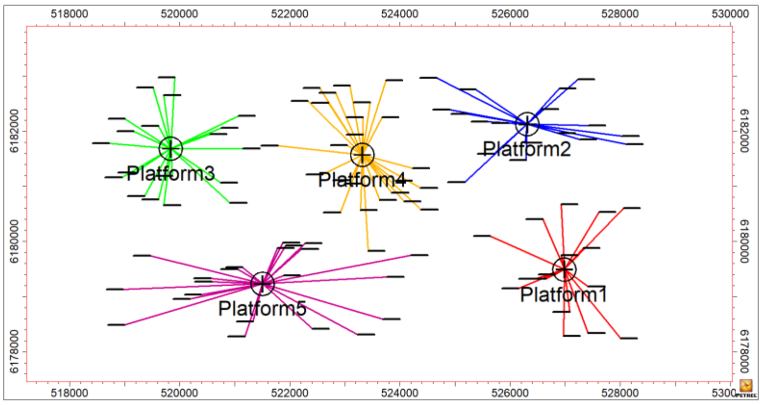


Figure 21 – Case 9, Horizontal wells: 100 wells, 5 Platforms.

8. *Conclusion*

Optimization parameters include platform locations, the number of wells per platform, and the assignment of wells to platforms; that is, which wells are connected to which platform. Furthermore, one extra optimization parameter is added in case of horizontal well: the platform-to-heel measured depth. We have presented an efficient optimization method that can robustly solve this problem regardless of the number of wells and platforms. That has been illustrated up to a case comprising 5000 wells and 100 platforms.

We compared three different methods: 1) a deterministic approach that clusters the wells considering all the possible combinations, 2) the GA algorithm assisted with internal distance and 3) well-assignment optimizer applied as assistant in the PSO method which is used for the first time for platform placement optimization in field development planning. Unlike what is claimed in the literature (Sales et al. 2018), GA cannot address this problem; it converges to a sub-optimal solution.

Different levels of complexity were considered starting from a simple one platform case as a validation problem. Several levels of complexity were then added by increasing the number of wells and/or the number of platforms. In all cases, including the nine cases presented in this paper, PSO systematically outperformed GA in terms of both the optimality of the reached solution and the number of iterations needed to reach that solution. GA failed to converge to a solution and got stuck in local optimum in Case 4 (100 wells, 5 platforms) and above.

In this context, we can conclude that the PSO assisted with the distance optimizer is the most robust and efficient methodology to address this optimization problem in terms of convergence time and objective function optimal values; regardless of the complexity of the problem.

B. Standard PSO vs. APSO

Based on the results of the previous section PSO has shown its superiority compared to GA. In this section we compare the efficiency of PSO with our Proposed Algorithm "Improved APSO" algorithm. Multilayer nodes and segments problem is considered with an increased levels of complexity. Complexity is defined by two main factor the number of nodes and the number of layers in the defined example. Simple cases of two layers optimization problems were studied first with an increasing number of nodes in each layer. Then, we moved to three-layers and four-layers examples in parallel with the increase in the nodes' number in each layer. This systematic approach was applied in order to define the limitations of PSO and the advantages of the newly proposed algorithm in terms of convergence and time (i.e. number of generations to reach plateau).

1. *Case1: 20 wells- 1 manifolds*

Starting from a simple case study 20 wells that should be assigned to 1 manifold whose location should be optimized only in this case. These 20 wells were defined within the map boundaries. As Figure 22 shows for this simple problem both PSO and APSO converged quickly to the optimal solution. This optimal solution is the same as the result obtained while searching for the geometric median, which is mathematically the closest point to all points of a cluster. Thus, having only one cluster of wells in this case the only effective factor that should be considered is the position of the manifold which represents in this case the geometric median of the wells.

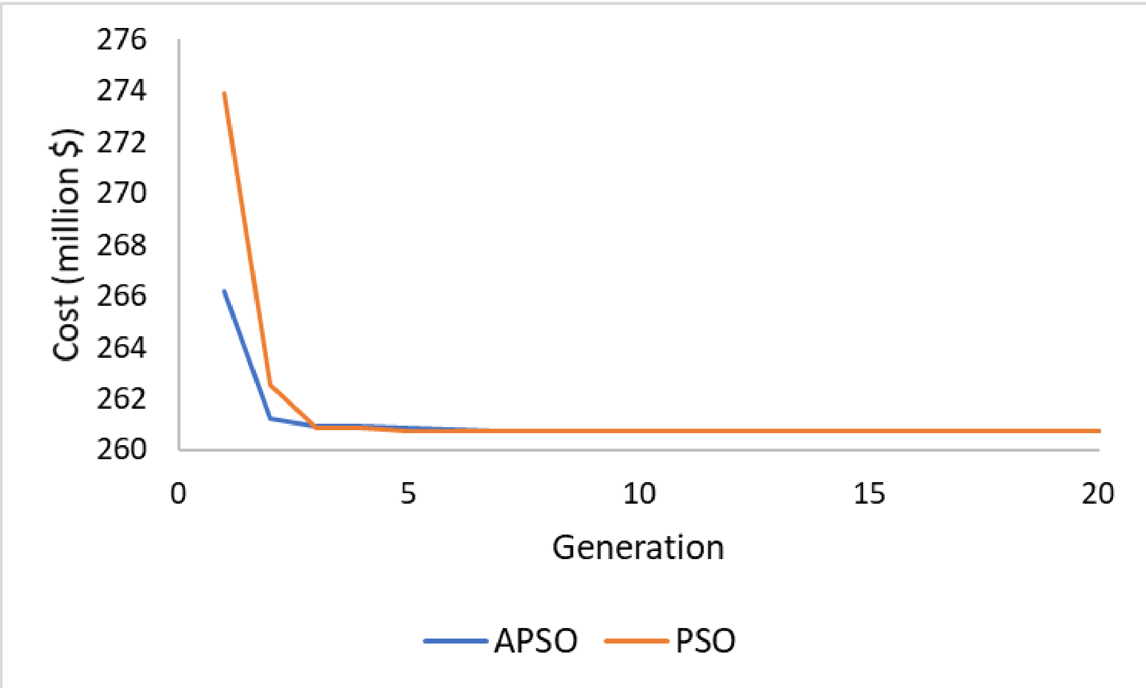


Figure 22- PSO vs. APSO (20 wells- 1 manifold)

2. *Case2: 100 wells- 1 manifolds*

Now increasing the number of nodes in the first layer (i.e. the wells) to 100 wells while keeping 1 manifold to be placed in this case also. *Figure 23* shows also the very same result for both algorithms in a very fast convergence rate. This also is the deterministic solution defined by the geometric median of the 100 wells under study. Thus, until this level of complexity both algorithms showed a high efficiency in terms of convergence value and time.

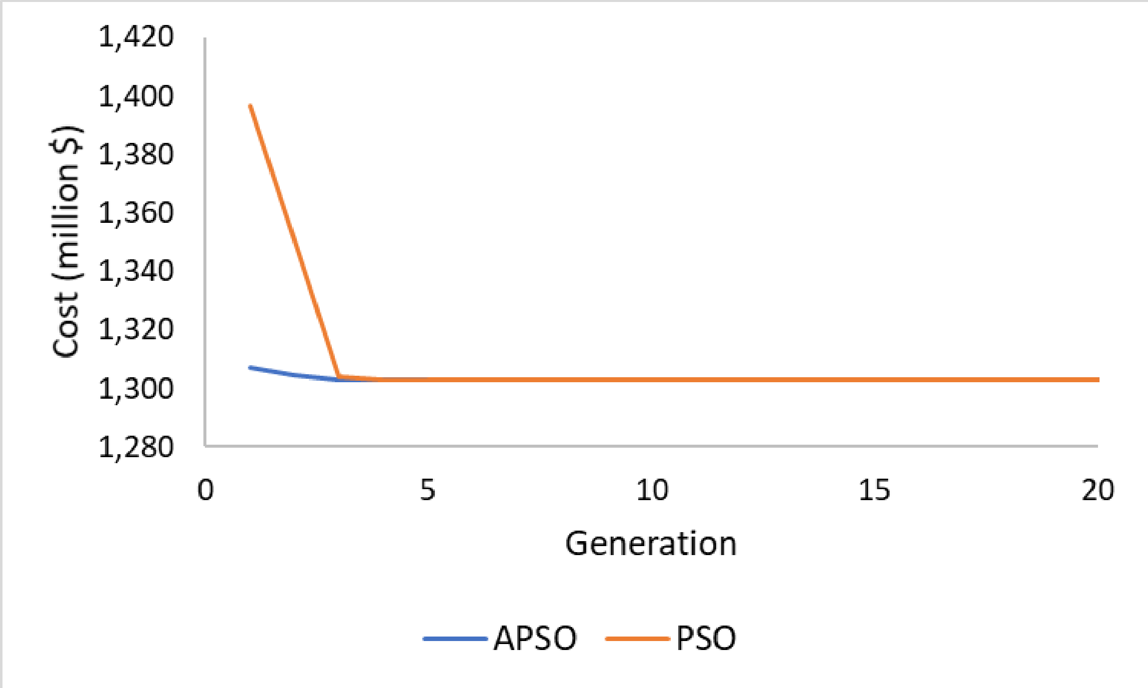


Figure 23- PSO vs. APSO (100 wells- 1 manifold)

3. Case3: 100wells- 5 manifolds

In the third case, another level of complexity is added: the number of manifolds is increased. Hence, now both positions of manifolds and assignment of wells to manifolds is introduced at this level. As *Figure 24* shows that both algorithms also converged to the very same optimal solution however ASPO started from a better initial guess and reached the optimal solution slightly faster than PSO. However, obtained results are similar and quick to be reached and this can be returned to the relatively low complexity of the defined example.

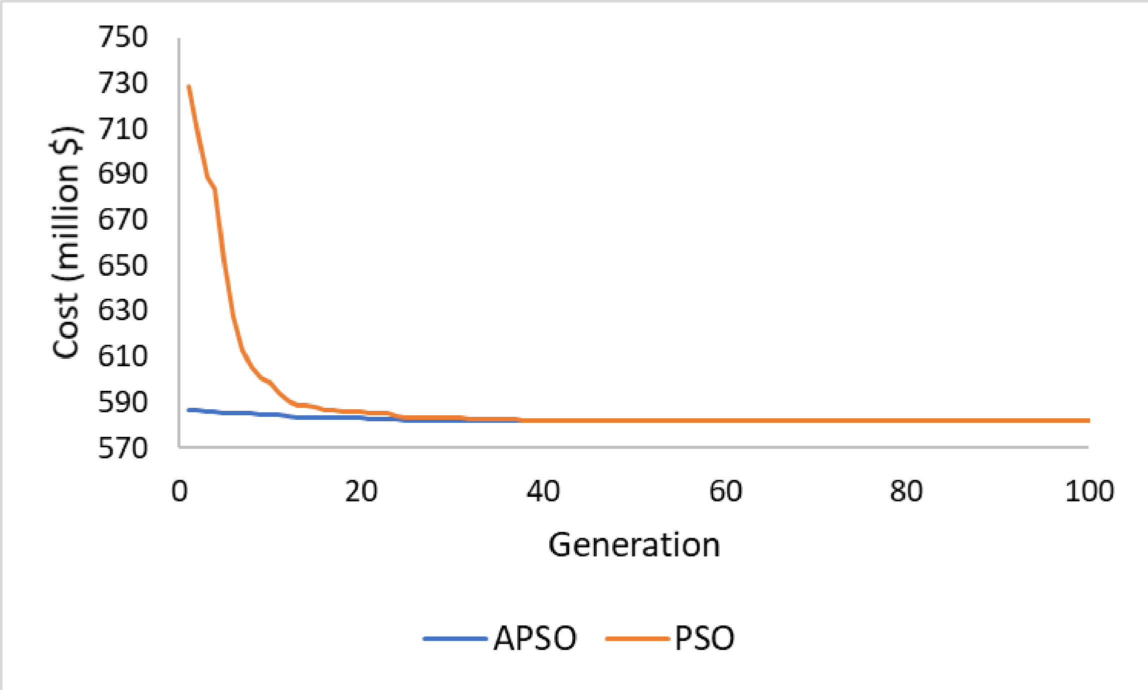


Figure 24- PSO vs. APSO (100wells- 5 manifold)

4. Case: 100 wells- 15 manifolds

Moving to higher complexity level by increasing layer two nodes to 15 manifolds, hence, increasing the positions of nodes to be optimized and then the number of clusters to be formed to 15 cluster. Another factor is the assignment of wells to each of these manifolds' clusters. As *Figure 25* shows that improved APSO started at this level of complexity to show its superiority over PSO standard algorithm. The proposed algorithm started from a better initial guess and converged to the optimal solution with few generations' number (200 generations). On the other hand, PSO didn't converge to the optimal value even after 1000 generations. This indicates that the proposed algorithm is overcoming early convergence problem (local optimal solutions) and is moving toward the Global optimal solution benefiting from the improvements that have been introduced to the standard PSO.

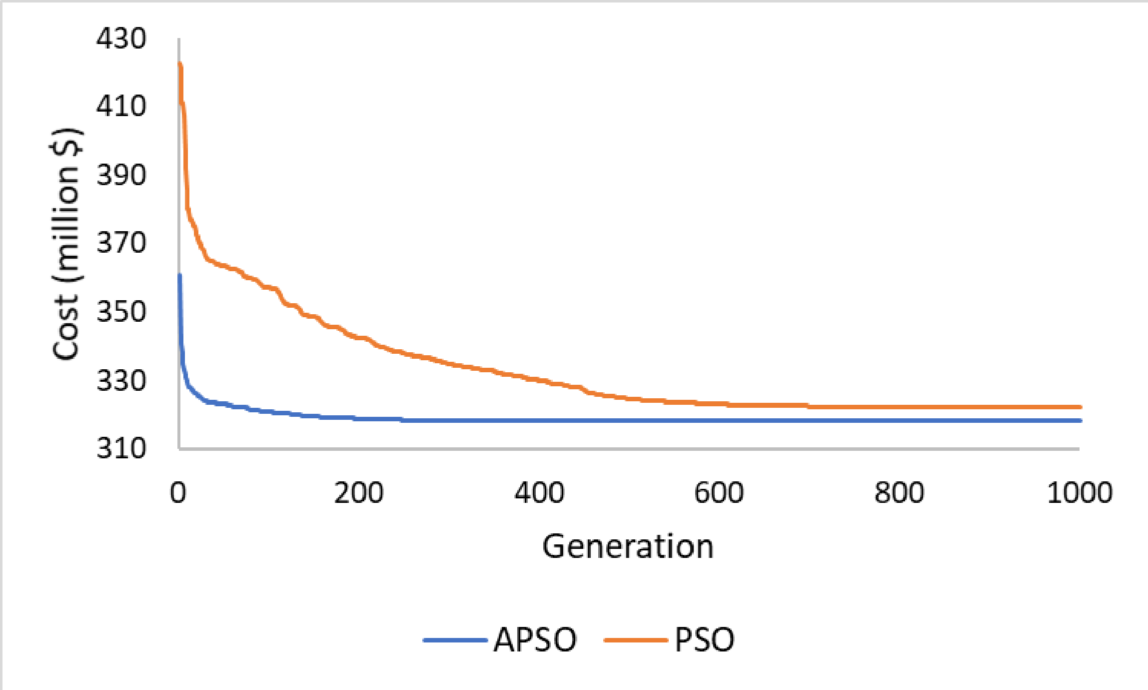


Figure 25- PSO vs. APSO (100 wells- 15 manifold)

5. Case5: 100wells- 15 manifolds-5 Platforms

In this case, another layer of optimization is added to the problem which become a 3-layers optimization problem containing wells, manifolds, and platforms. This higher level imposes additional efforts on the optimization course i.e. the positions platforms that represent a new clustering level of the manifolds coming from the previous layer. It is a layer by layer clustering; however, the results of the previous layer clustering will affect the assignment decision taken by the introduced clustering function. Figure 26 shows also the superiority of the improved APSO over the PSO also in terms of convergence value and rate. In one hand, the proposed algorithm outperformed PSO result by about 30 million $. On the other hand, proposed method reached its optimal value in about 400 generations; however, PSO didn't converge even after 2000 generations. So, increase of the complexity is giving advantage to the improved APSO over the PSO algorithm.

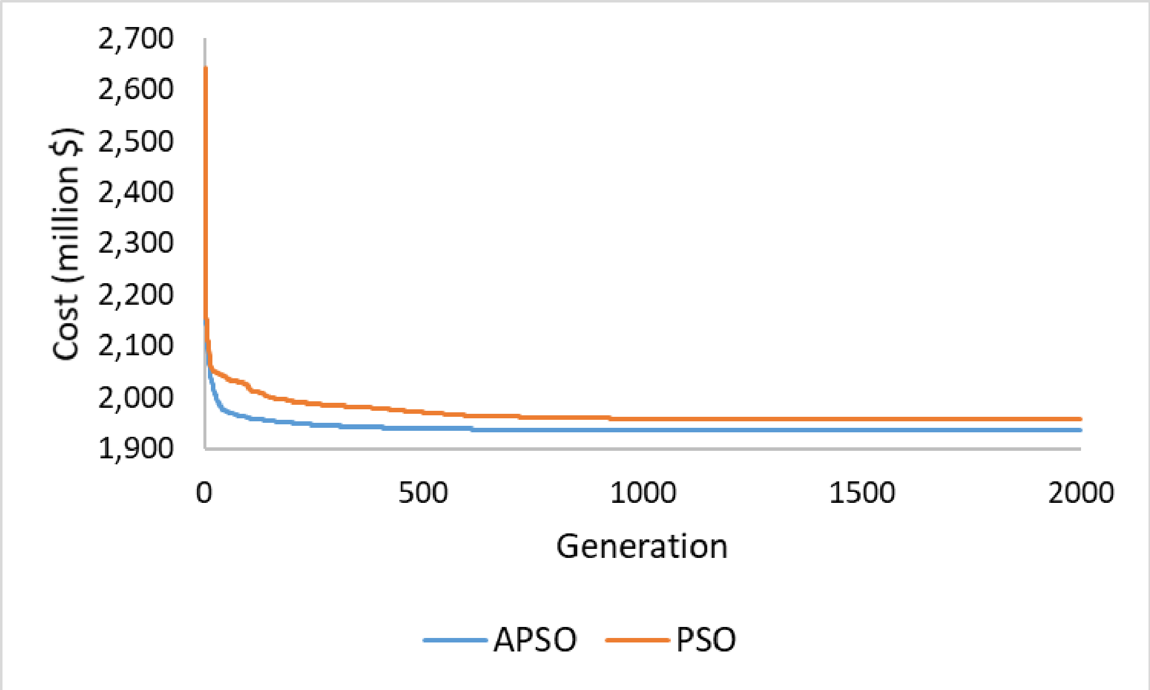


Figure 26- PSO vs. APSO (100 wells- 15 manifolds- 5 platforms)

6. Case6: 100 wells -15 platforms -5 manifolds -1 facility

Finally, a 4-layers optimization example is defined i.e. 100 wells, 15 manifolds, 5 platforms, and a facility. This high complexity level is used to compare the proposed algorithm with the standard PSO. This tree-like 4 layers example confirmed the superiority of the proposed algorithm over the standard PSO. Figure 27 shows that proposed APSO algorithm outperformed the standard PSO also in terms of the convergence value and time. Standard PSO, though after 2000 generations, is about 100 million $ far from the optimal value reached in about 500 generations by the improved APSO. Proposed algorithm shows, then, advantage on the standard PSO while the complexity of the problem is increased which makes it a more robust and efficient approach to deal with these problems of high complexity.

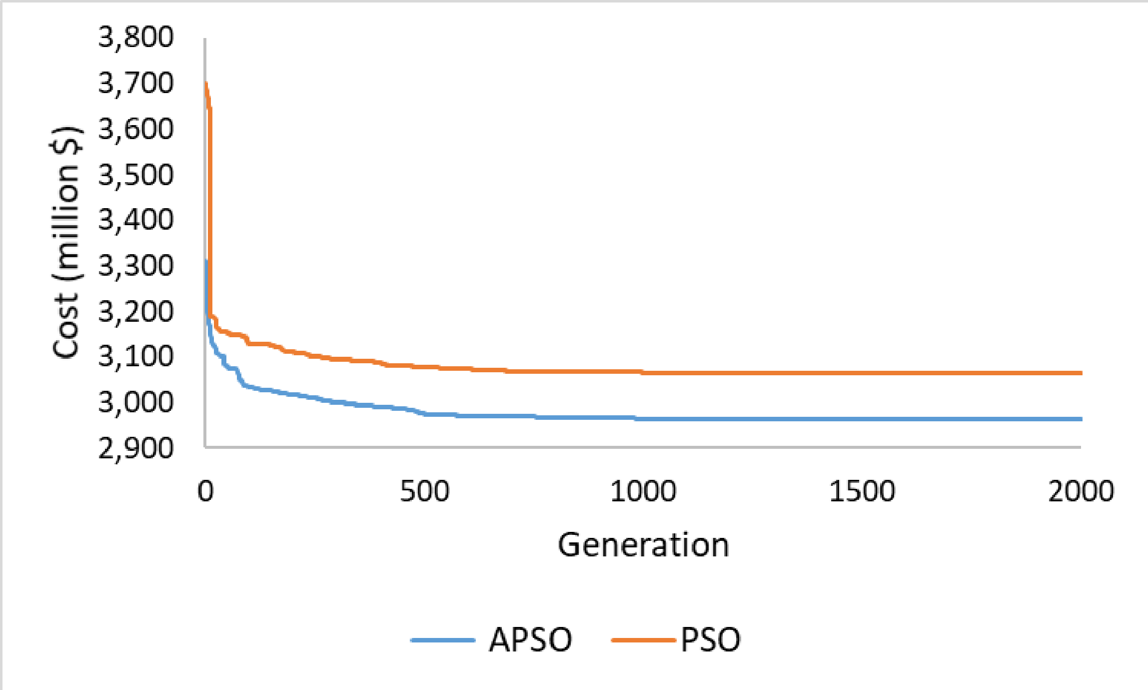


Figure 27- PSO vs. APSO (100 wells- 15 manifolds- 5 platforms- 1 facility)

7. Conclusion

In this section, standard PSO is compared with adaptive particle swarm optimization introduced by Zhan et al. [38]. This improved method focuses on analysing the evolutionary state of the problem and adapt algorithmic parameters i.e. inertia weight, social and global cognitive factors accordingly. Four evolutionary states were defined: exploration, exploitation, convergence, and jumping out. Each of these states has its own special alteration on the algorithmic parameters.

Comparison of standard PSO and APSO showed a superiority of the latter improved method on different layers of complexity. Starting from very simple examples (i.e. two layers problems) and reaching complex multiple layers problems. Thus, APSO is chosen to be used as the main optimizer to which new improvements are applied in the next section

C. Proposed Algorithm

In the following section, we present the results of the optimization using the proposed algorithm i.e. Improved APSO algorithm. This algorithm that can applied for multilayers optimization problems is a generic solution of any of these problems. In the following two, three, and four layers examples are considered. Results of optimization are represented in terms of the Objective function for multiple runs to ensure that the proposed method is consistent. Positions and clusters are also represented using PETREL software in 2D and 3D maps. Also, in this section the results are represented in order of complexity. In this section all the three improvements are applied to the standard PSO algorithm:

1. Parameters adaptation (APSO)
2. Smart Initialization (multiple runs initialization)
3. Mutation (Restart of the solutions)

1. Two layers

a. <u>Case1: 20 Wells-1 manifolds</u>

Starting from a simple two-layers example i.e. 20 wells-1 manifold. This problem is very simple since the assignment of nodes between first and second layers is predefined by default since we have only one node the upper layer. Thus, only the position of the manifold is the optimization parameter in this case. Figure 30 shows that even when the problem is repeated for 10 independent runs, the objective function is fast converging (2-3 generations) to the very same optimal value. Figure 28 and Figure 29 represent respectively the 3D and 2D representation of the solution. The position of the manifold in this case represents exactly the geometric median position which represents the nearest point to all the cluster's elements.

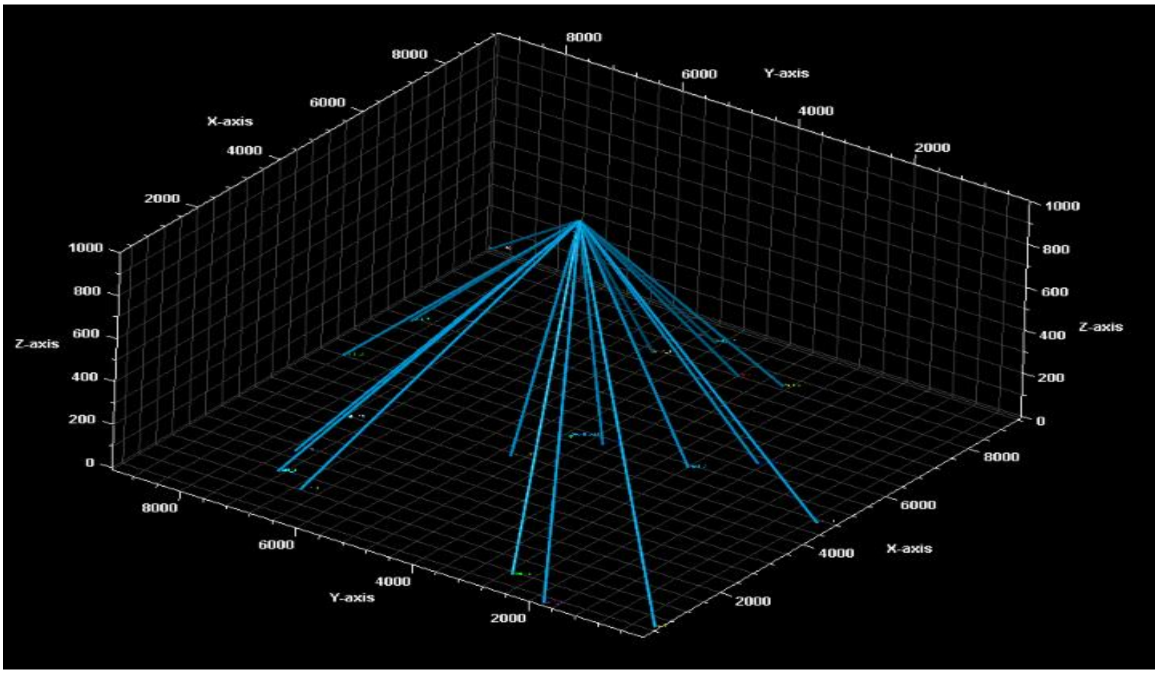


Figure 28- 3D plot (20 wells- 1 manifold)

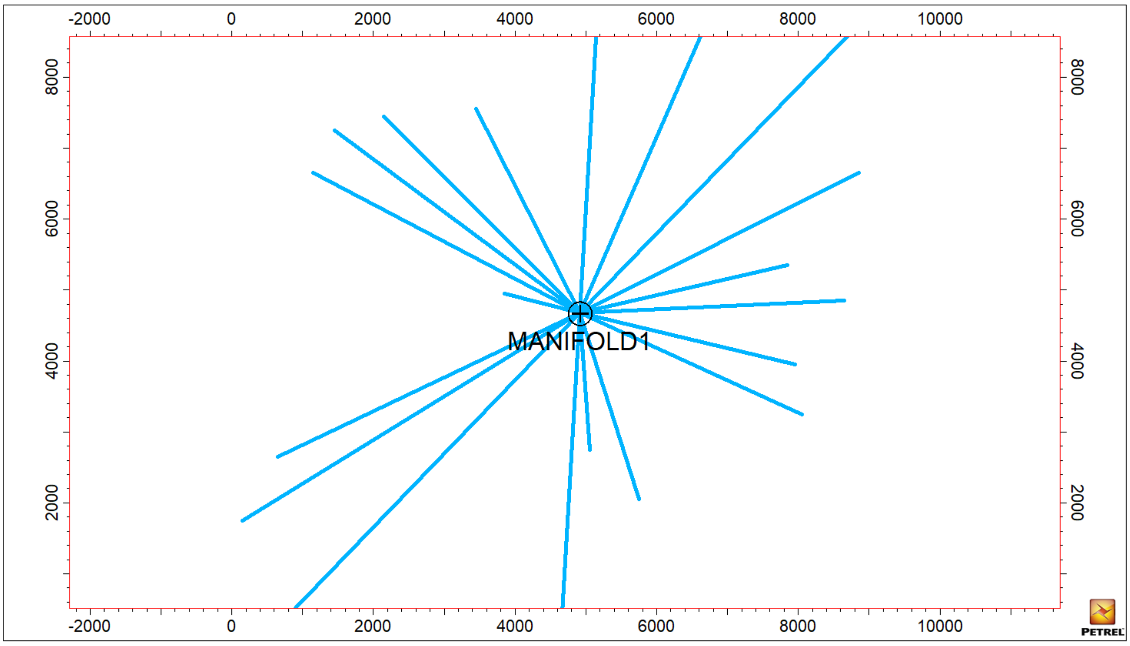


Figure 29- 2D plot (20 wells- 1 manifold)

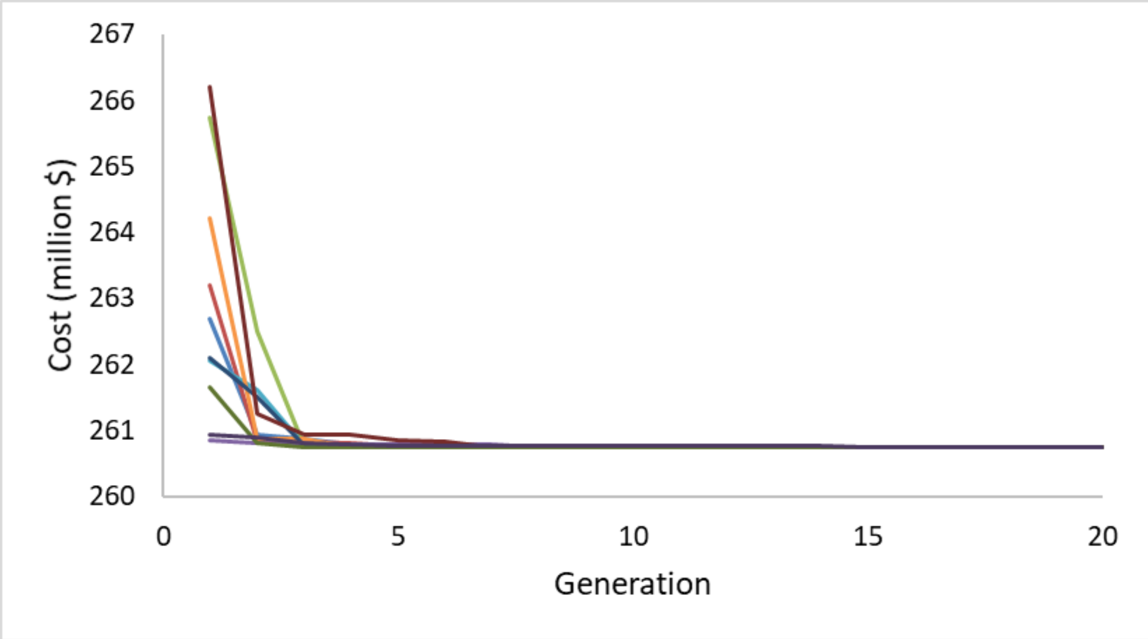


Figure 30- Objective Function 10 runs (20 wells- 1 manifold)

b. <u>Case2: 100 Wells-1 manifolds</u>

Increasing the number of nodes in the first layer to 100 wells while keeping the use of 1 manifold is another level of complexity introduced in this example. Though, 10 independent runs of the same case resulted the very same optimal solution (i.e. the position of the manifold is the same as the geometric median of the wells). *Figure 33* shows that the 10 independent runs converged in few numbers of generations to their optimal solution. *Figure 31* represents the 3D sketch of the 100 wells-1 manifold positions and connections. In fact, manifolds and wells are in the same level i.e. in case of offshore production system optimization both are on the seabed level, but the sketch presents an altitude difference just for readability and clarity of the sketch. Figure 32 presents a 2D map sketch of the case in hand, this map defines the position of the manifold and the connection between wells and manifold.Figure *32*

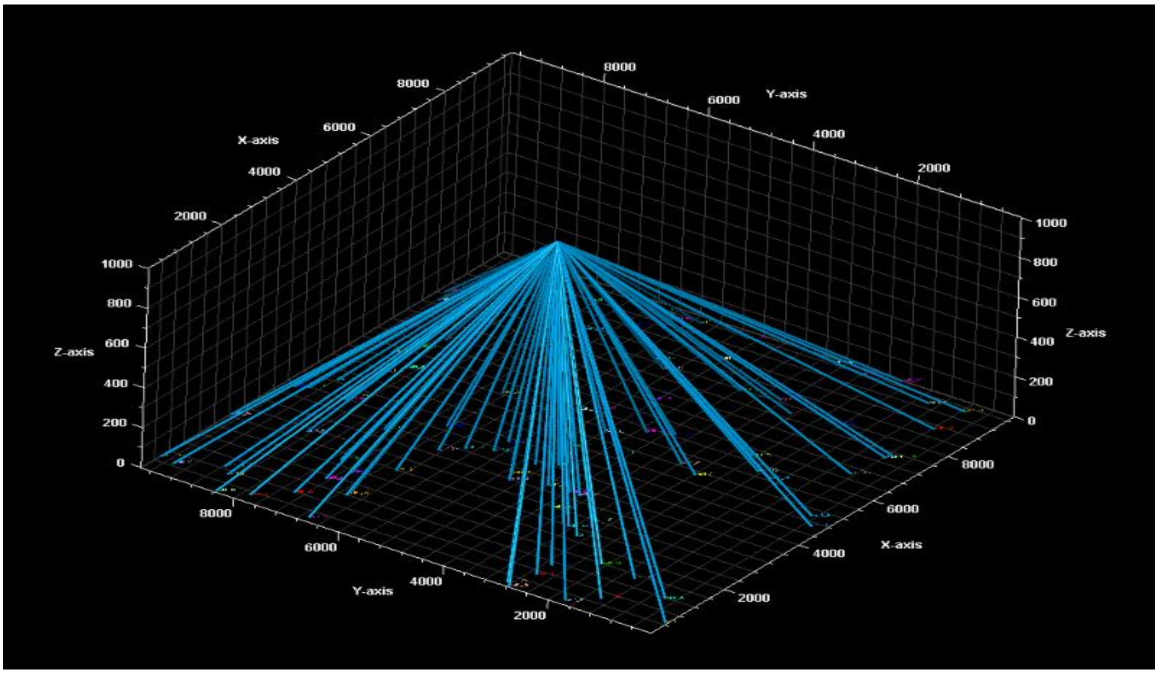


Figure 31- 3D plot (100 wells- 1 manifold)

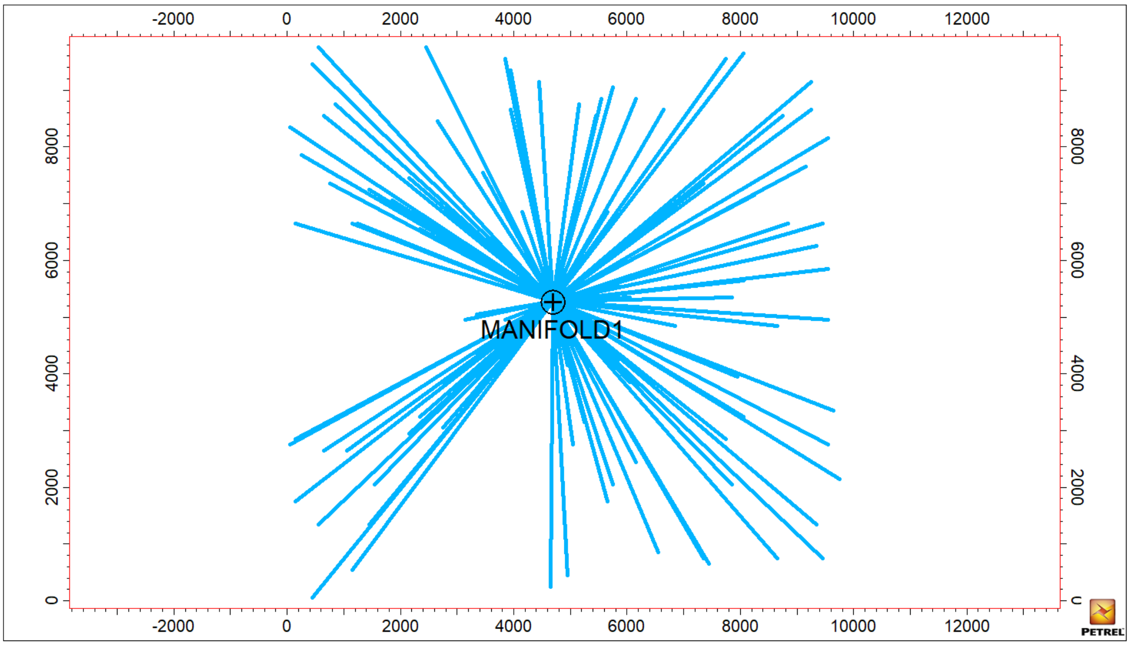


Figure 32- 2D plot (100 wells- 1 manifold)

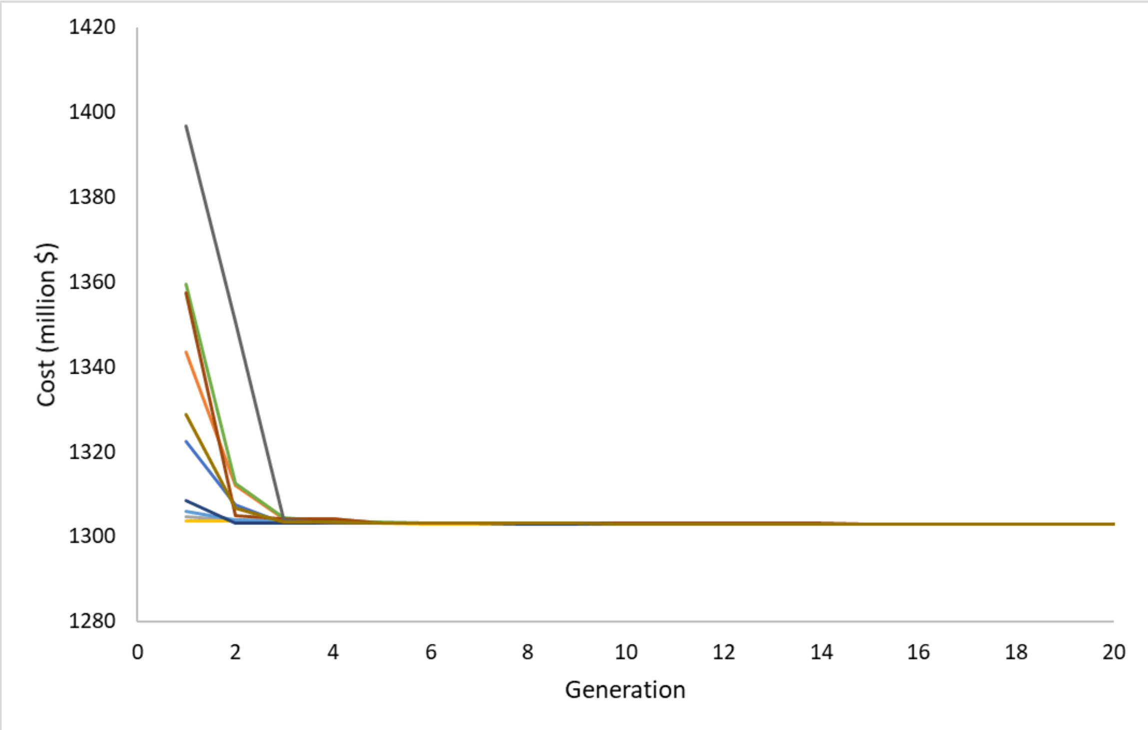


Figure 33- Objective Function 10 runs (100 wells- 1 manifold)

c. <u>Case3: 100 Wells-5 manifolds</u>

In this 2-layers example the number of nodes in the second layer is increased i.e. the number of manifolds that represents the clusters' number is increased. Two types of complexities were added in this case: the positions of the manifolds, and the wells to clusters assignment using the introduced clustering method. *Figure 36* shows the objective function results for 10 different runs of the same problem. The repetition of the run is to make sure that the algorithm in use is consistent and robust. The results show that in the 10 different runs, started from the very same initialization, the solutions of all runs were found to be similar with a slight standard deviation of 0.005%.*Figure 34* is a sketch of the 3D configuration of the solution. The figure showed 5 well established clusters where each well is connected to the nearest manifold taking into consideration the maximum capacity of each manifold. Also, in this example the difference in altitude between manifolds and well heads is just for readability and clarity purposes and the problem formulation considered them being at the seabed level. *Figure 35* represents a 2D sketch of the wells-manifolds positions and connectivity.

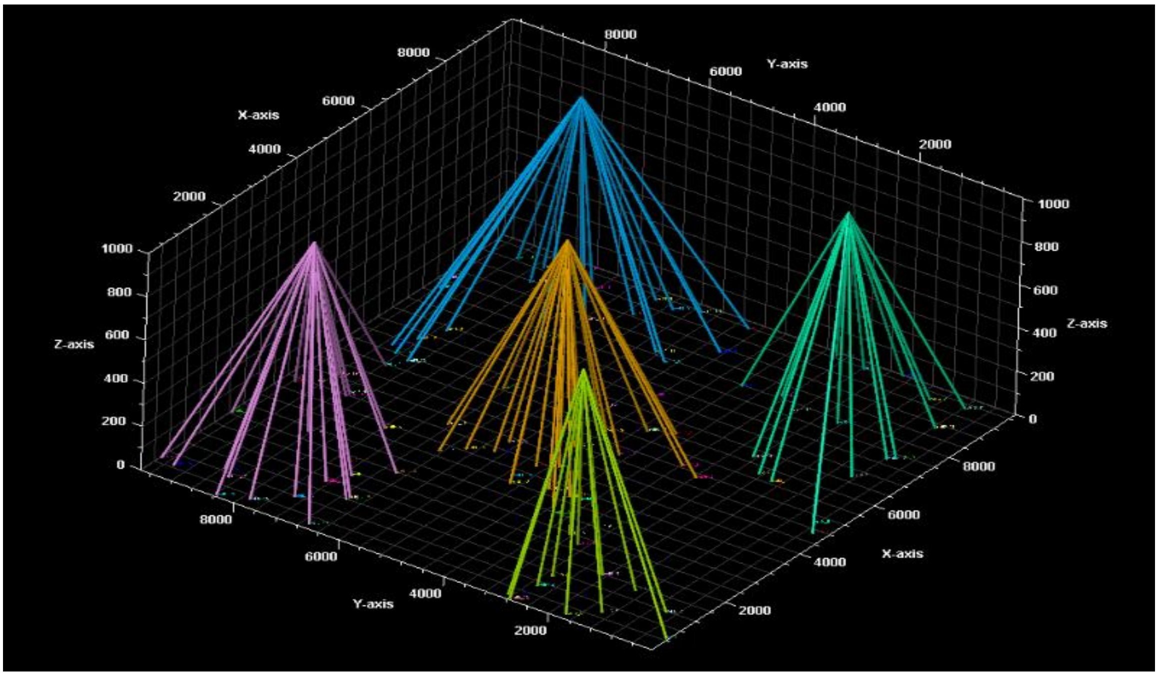


Figure 34- 3D plot (100 wells- 5 manifolds)

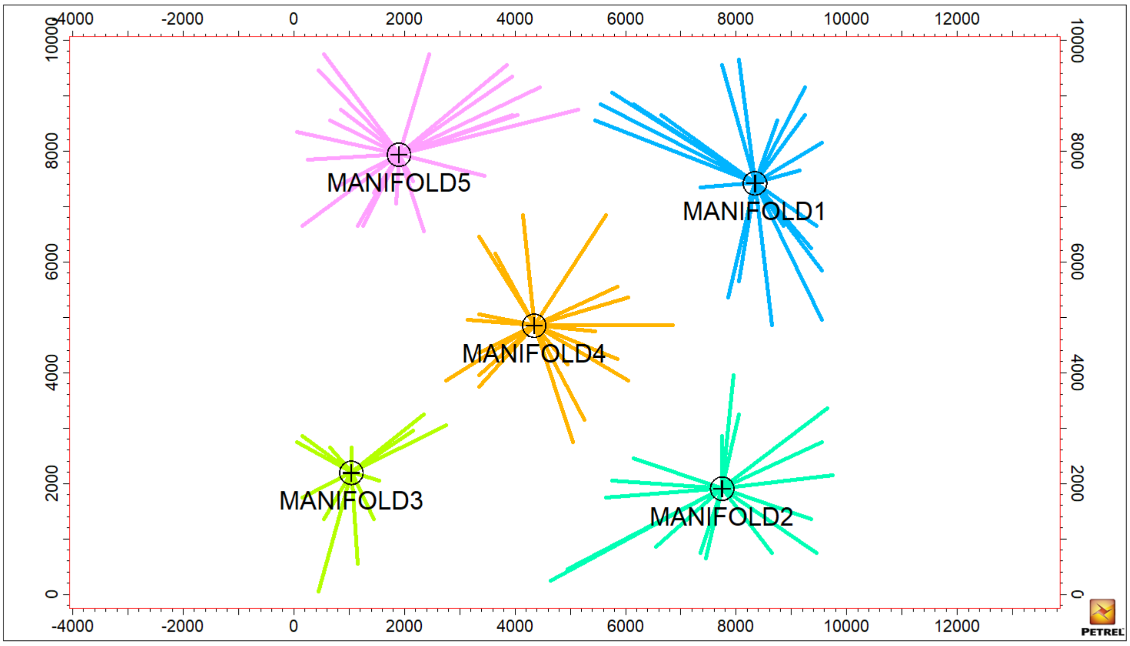


Figure 35- 2D plot (100 wells- 5 manifolds)

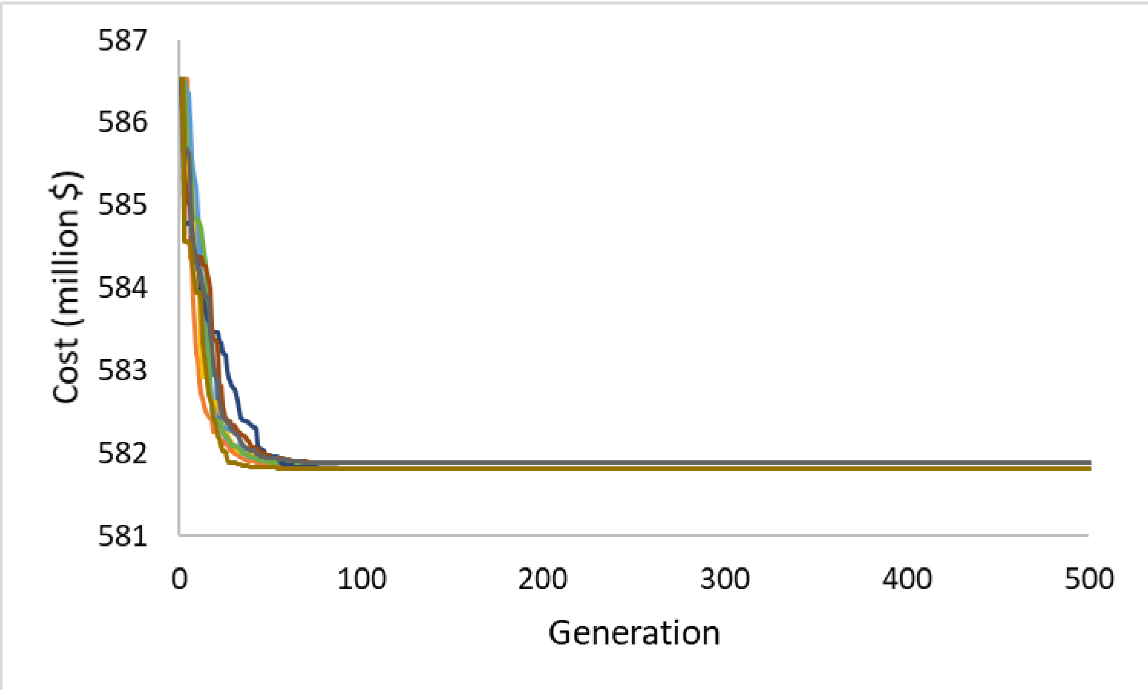


Figure 36- Objective Function 10 runs (100 wells- 5 manifolds)

d. Case4: 100 Wells-15 manifolds

Further increase in the number of nodes in layer 2 i.e. increase the number of manifolds that should be positioned first and, then, assigned to the corresponding wells in the lower layer. From this level of complexity, the need to start using the smart initialization technique arises. *Figure 39* represents the objective function results that can be divided into two main steps:

1. The smart initialization phase where about 50 runs were conducted. Each run represents an optimization trial starting from a random initialization. The number of initialization runs is chosen based on the number of particles in the population of the main optimization loop i.e. in this case the size of the population is 50 particles. These initialization runs are conducted just for 200 generations, after which the best solution in each run is stored as one initial solution of the initialization for the main optimization loop.
2. Main optimization phase where the process starts from the stored best results from the multiple run initialization phase. In this phase, 10 independent runs (50 particles each as the size of the population) are conducted to test the consistency of the proposed method.

Figure 39 shows the complete optimization process starting by the smart initialization and ending with the main optimization loop results. *Figure 40* shows the main loop optimization results where the 10 independent runs converged to the same optimal value. In addition, Figure 41 narrows down the figure to the results of the first 200 generations where all the runs have converged to their optimal solution with no deviation in this case.

Figure 37and Figure 38 represents respectively the 3D and 2D sketches of the optimal solution i.e. the optimal manifolds positioning and clustering. Also, the manifolds are considered on the same plan of the well heads (seabed); however, visualization of the results imposed to vary the altitude while sketching the 3D plot.

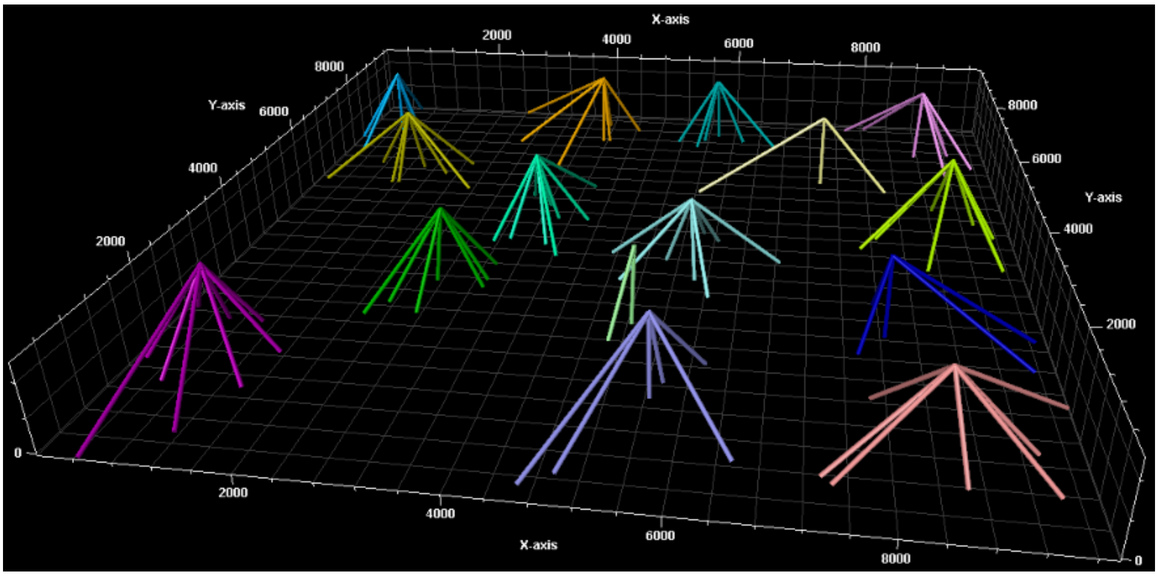


Figure 37- 3D plot (100 wells- 15 manifolds)

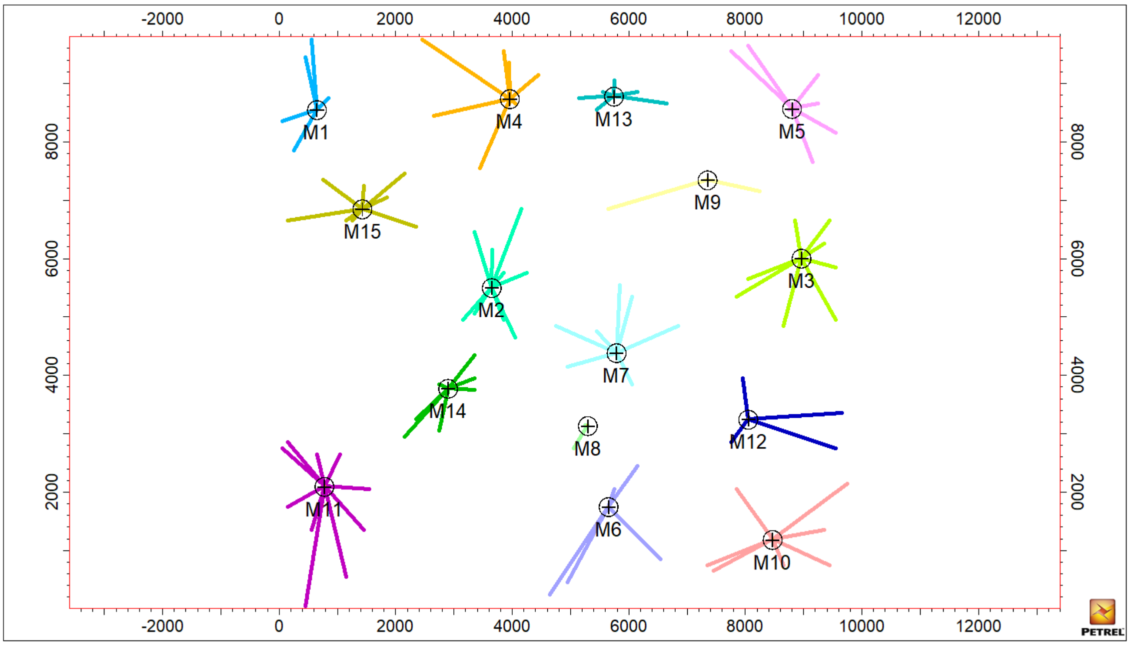


Figure 38- 2D plot (100 wells- 15 manifolds)

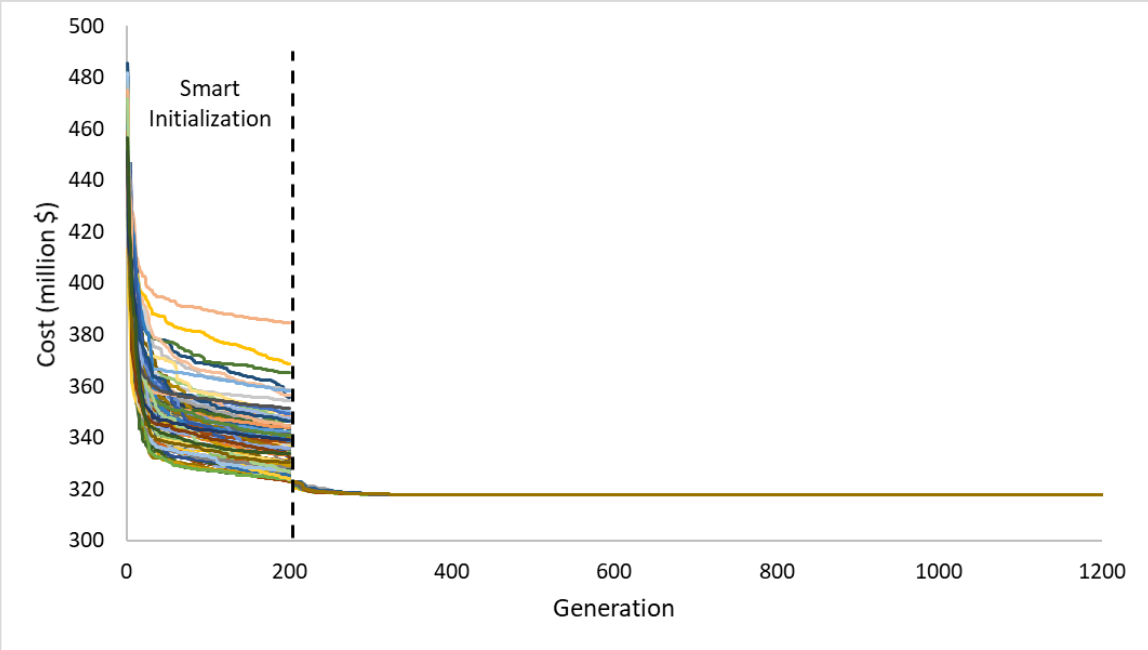


Figure 39- Objective Function Initialization and main optimization loop (100 wells- 15 manifolds)

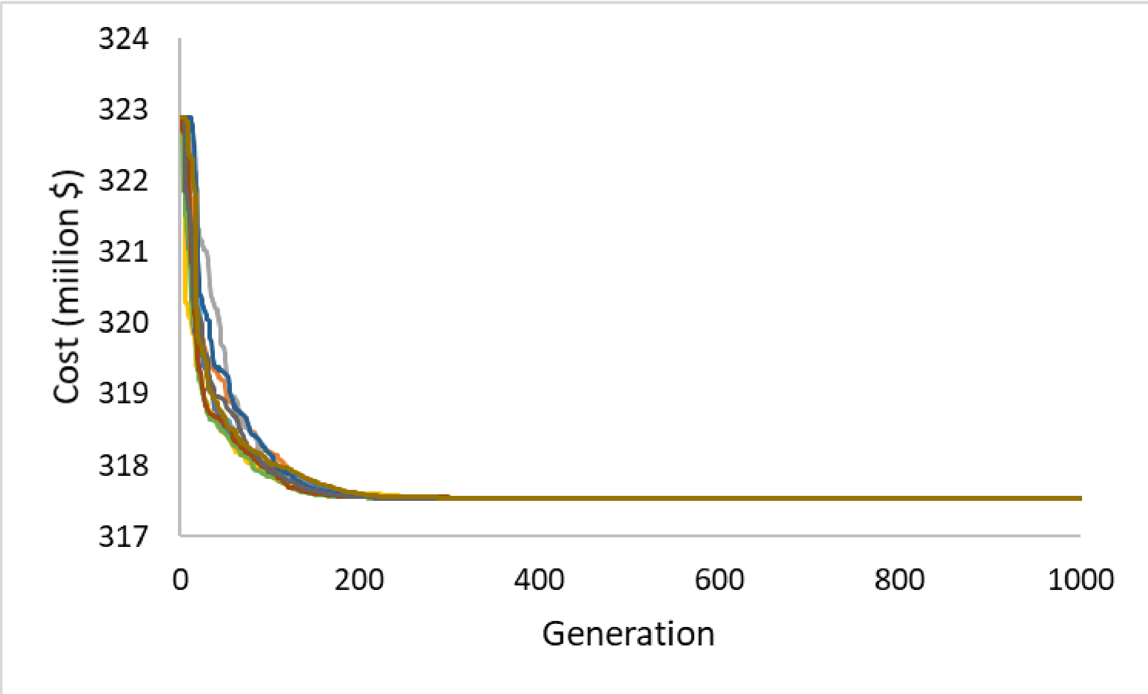


Figure 40- Objective Function of the main loop 10 runs (100 wells- 15 manifolds)

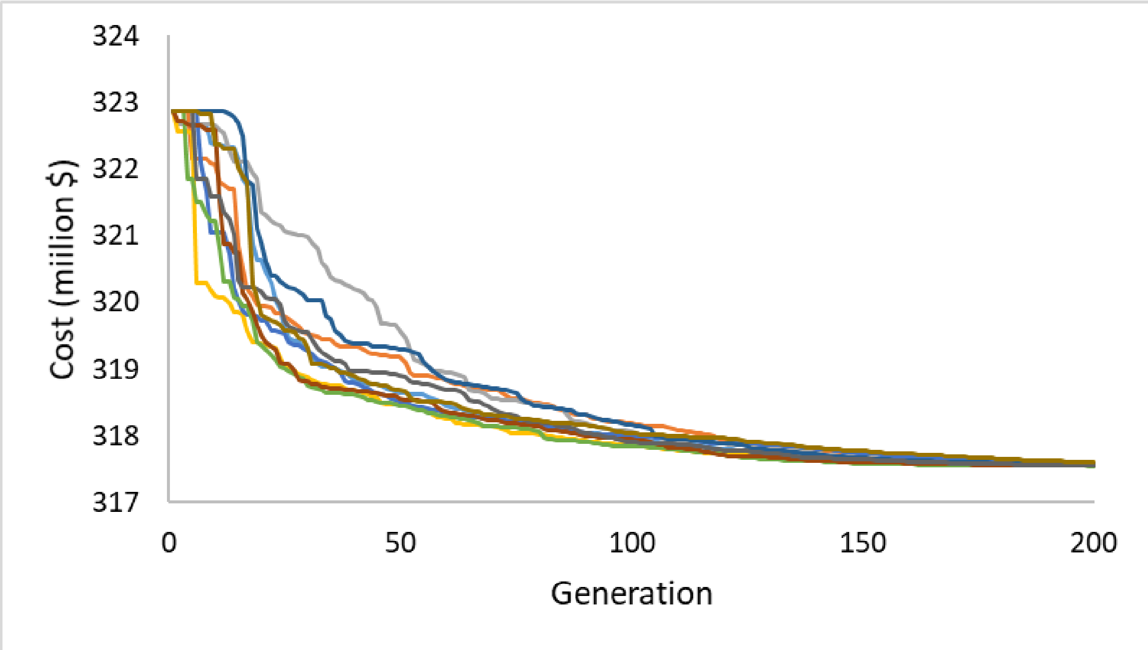


Figure 41-Zoom on the first 200 generations of the objective Function of the main loop 10 runs (100 wells- 15 manifolds)

2. Three layers

a. <u>Case5: 100 Wells-15 manifolds-5 Platforms</u>

At this stage, three layers example is considered to test the proposed algorithm. As described in the methodology section, the capacity input parameter represents the maximum capacity of the nodes. In addition, the number of nodes is just an upper limit input for the problem. This flexibility of the number of nodes and their capacity in parallel with the clustering function that controls these two factors according to nodes' positions that is defined by the main optimizer (APSO) makes the optimization problem more practical. As seen in *Figure 42* and *Figure 43*, that despite the input of 5 platforms as a maximum number of platforms (initial number of platforms), the solution ended up with only three platforms that were placed in optimal locations and assigned in an optimal way to the nodes in the lower layer (manifolds).

Figure 44 represents the complete optimization process using smart initialization (multiple run initialization) for 200 generations. The best results of each of the 50 initialization runs are stored and used as the initial guess for the main loop optimization. The results of the main optimization loop as shown in Figure *45* represents a smooth convergence in the 10 independent conducted runs reaching the very near optimal solution with a standard deviation of 0.009%. This fact shows the consistency in the results obtained using the proposed method.

Figure 42 presents the 3D sketch of this 3-layers case, where manifolds are the clusters' centres of the wells and, in their turn, the platforms are the clusters' centres of the manifolds. Manifolds and wells heads are on the same altitude level (seabed) but for the sake of visualization of the case they are separated in the sketch. However, the platforms are usually in offshore fields placed on the sea water level.

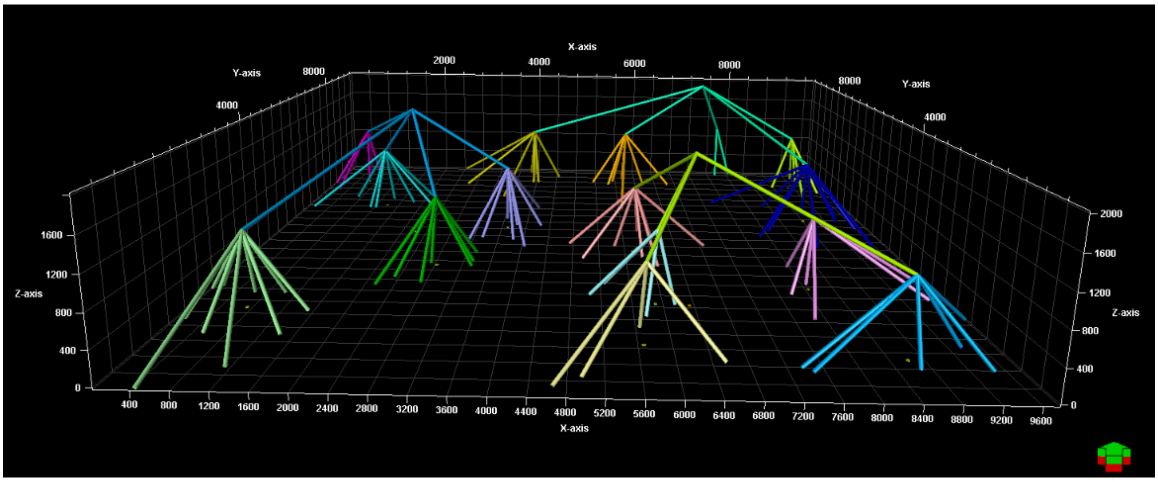
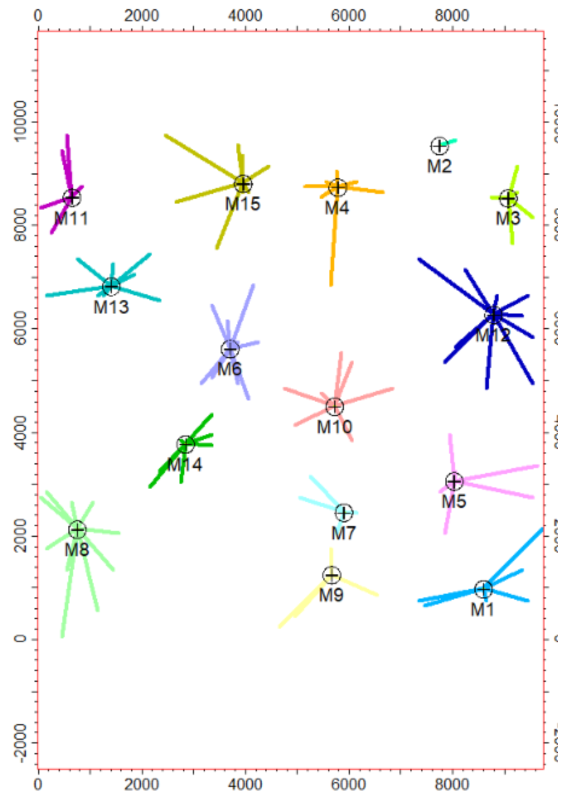
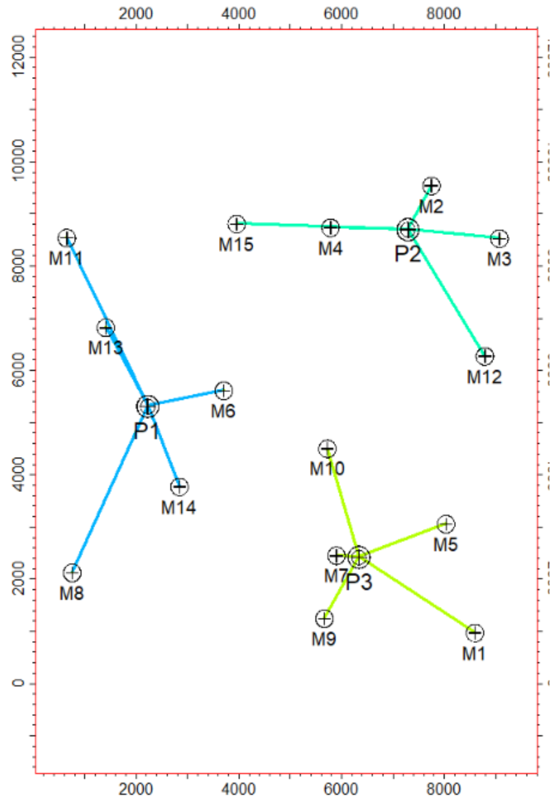


Figure 42-3D plot (100 wells- 15 manifolds-5 platforms)

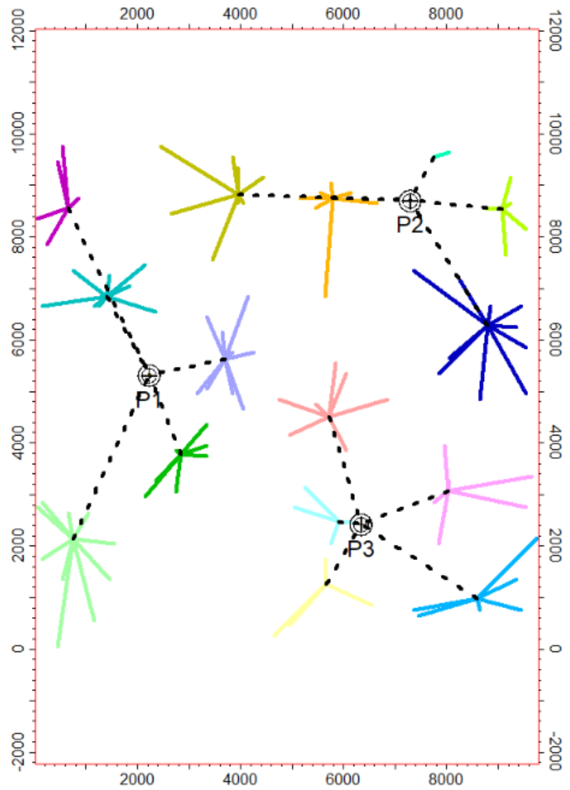
Figure 43 represents the 2D sketch of the solution of this 3-layers case. Figure 43-A describes layers 1-2; it shows the positioning of the manifolds in layer 2 and the well assignment to each manifold cluster. It is important to remark that while optimizing nodes of an intermediate layer (layer falling between two other layers) the positioning of these nodes is dependent on both assigned nodes in the lower layer and the node in upper layer to which it will be assigned. Figure 43-B shows the 2D result between layers 2 and 3 i.e. manifolds- platforms layers. Although, the input number of platforms is initially 5, the solution found that it is more optimal to use just 3 platforms that are positioned optimally based on the introduced clustering function. Figure 43-C represents the best solution of the overall 3-layers nodes positions and connectivity i.e. represents well heads, manifolds, and platforms.



A- 2D plot of layers 1-2 (wells-manifolds)



B- 2D plot of layers 2-3 (manifolds-platforms)



C- 2D plot of layers 1-2-3 (wells-manifolds-platforms)

Figure 43- 2D plots (100 wells- 15 manifolds-5 platforms)

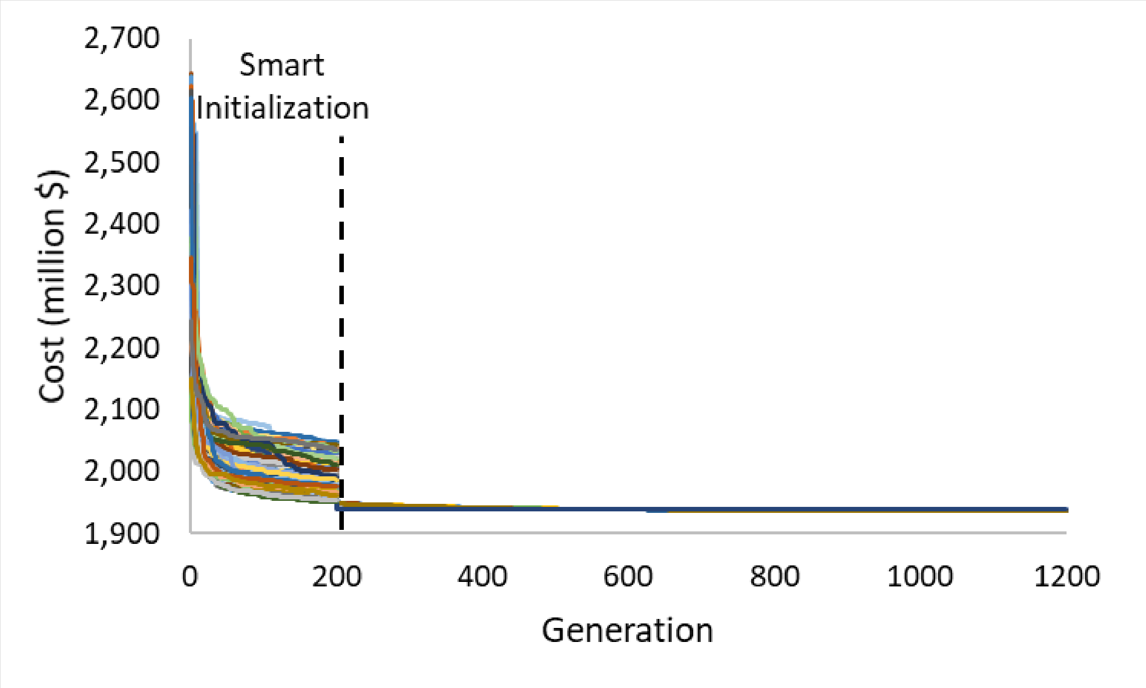


Figure 44- Objective Function Initialization and main optimization loop (100 wells- 15 manifolds-5 platforms)

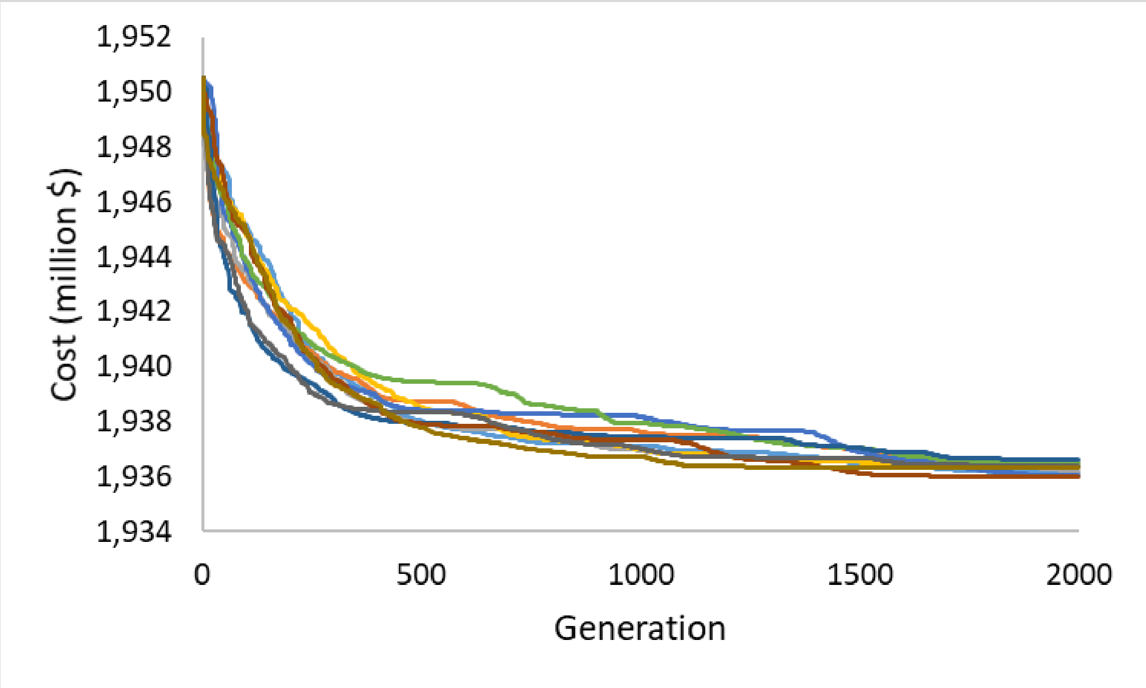


Figure 45- Objective Function of the main loop 10 runs (100 wells- 15 manifolds-5 platforms)

3. Four layers

a. <u>Case6: 100 Wells-15 manifolds-5 Platforms-1 Facility</u>

Finally, considering the highest complexity example: 4 layers problem having 100 nodes in the first layer (100 wells), 15 nodes in the second layer (manifolds), 5 nodes in the third layer (platforms), and 1 node in the fourth layer (facility). In this high complexity case, the number, positions, and assignments of nodes in each layer is considered. As the 3D sketch in Figure 46 shows, the wells are clustered to match the connect to the 15 well-placed manifolds in layer 2. However, these 15 manifolds are clustered into only 3 optimized -location platforms in layer3. This decision taken by the clustering function is important since the cost of these additional platforms is found to be higher than the additional connections' costs in case of reduction of nodes' number. This is another important aspect of the proposed algorithm compared with other optimizer such as standard PSO, GA etc…

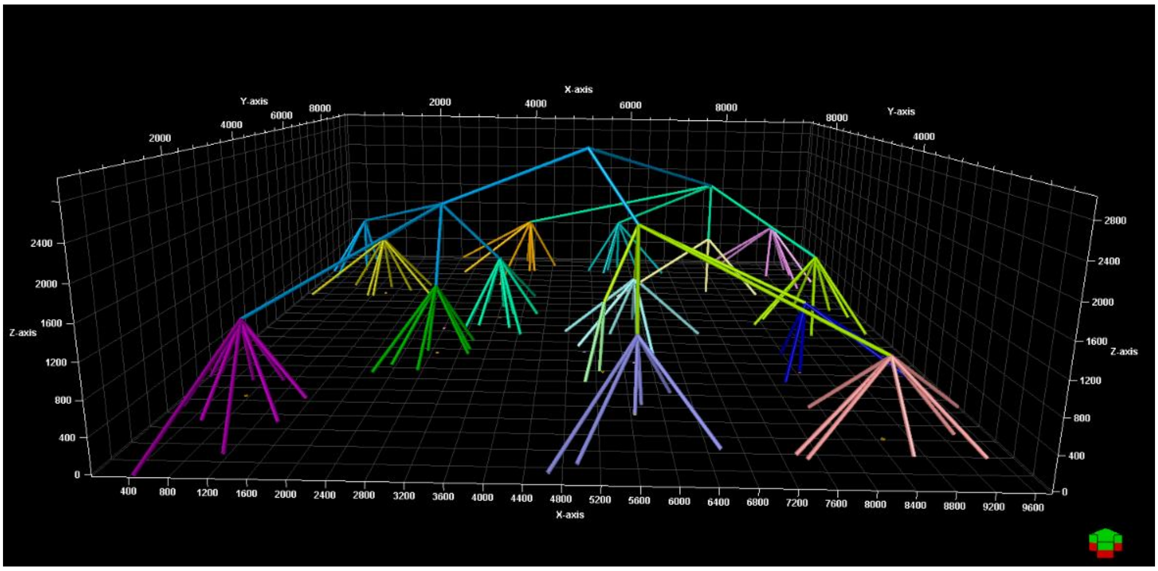
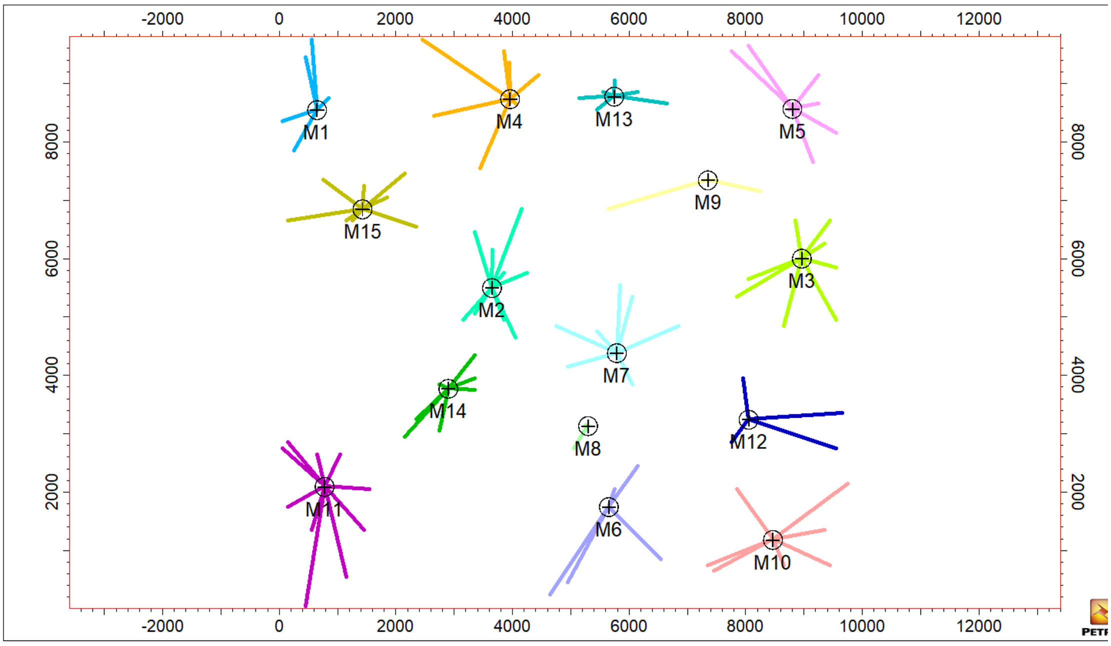


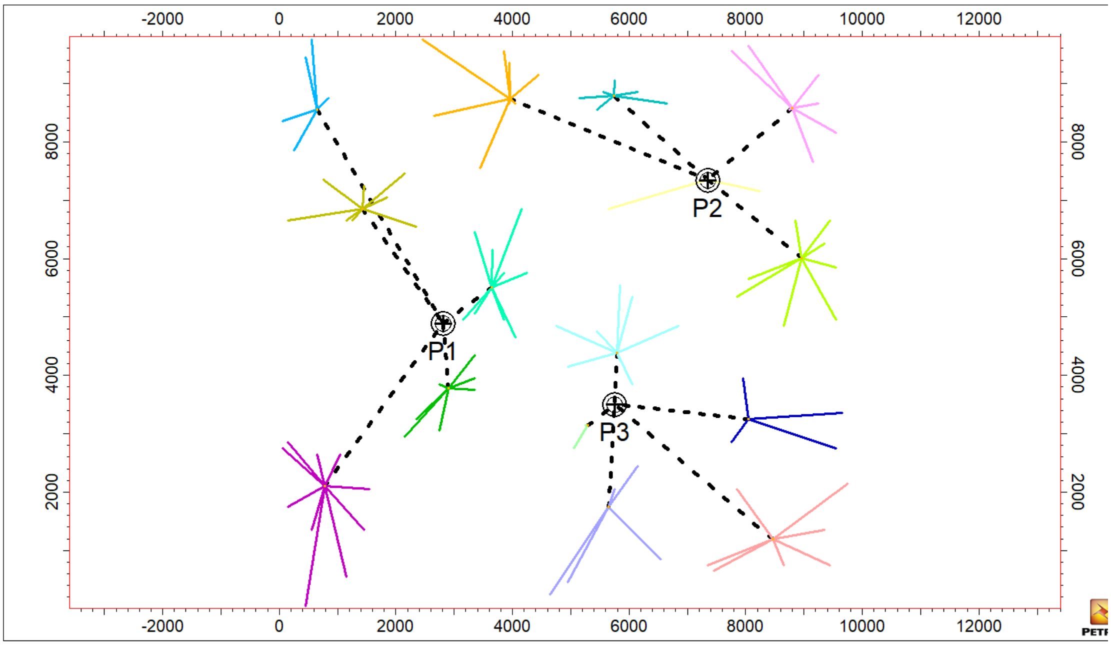
Figure 46-3D plot (100 wells- 15 manifolds-5 platforms-1 facility)

Figure 48 presents the whole optimization process using smart initialization and all improvements added to the standard algorithm. Due to the complexity of the problem, 100 particles are chosen to constitute the population. Hence, 100 runs were conducted in the multiple run initialization process for 500 generations in this case. After initializing the first guess of the main optimization loop, another 10 independent runs were conducted for 2000 generations each. Figure 49 shows the results of the objective function (total cost) for the main optimization loop. Convergence is reached after about 1500 generations. The results of the 10 independent runs show a very near optimal value with a standard deviation of 0.019%.

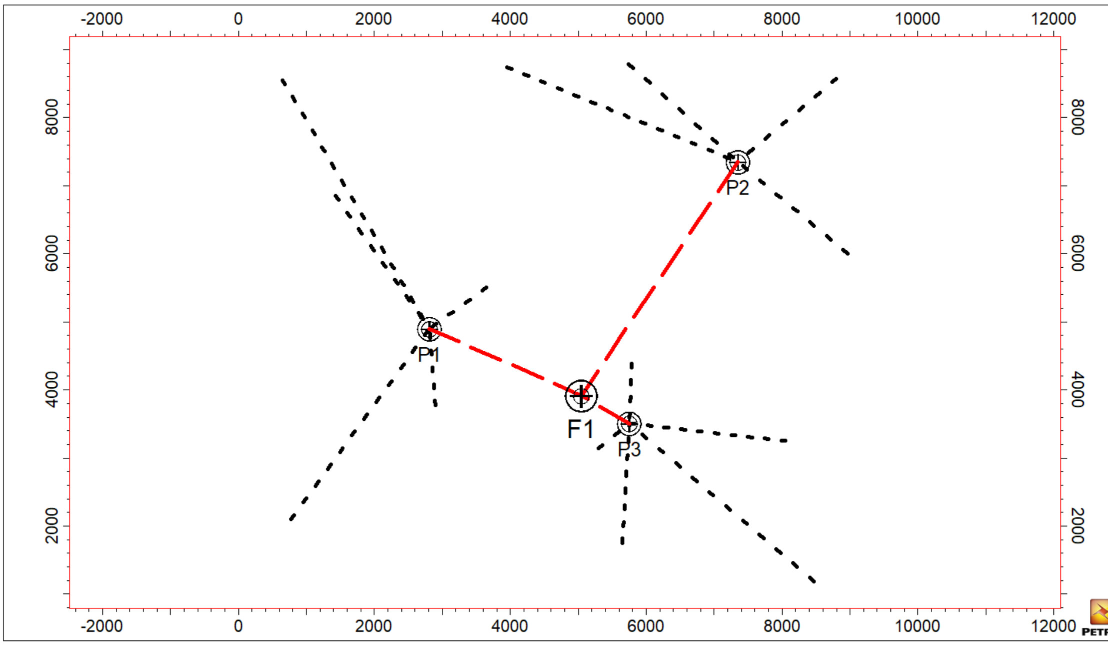
Figure 47-A presents a 2D plot that describes the positions and assignment of nodes of layers 1 and 2 (well heads and manifolds). Figure 47-B is the sketch representing nodes and connectivity between layers 1-2-3 (well heads, manifolds, and platforms). In layer 3 the input number of platforms was initialized to 5, however, the optimizer decided to use just 3 platforms in this layer. So, the number of nodes is being indirectly optimized to minimize the total cost. Figure 47-C represents the results of nodes positions and clustering of layers 3 and 4 i.e. the positions and connectivity of platforms to the facility in the higher layer. The position of the facility is assumed to be unconstrainted in this case just for the sake of illustration. However, the facility usually in case of offshore is placed onshore mainly and this will be considered in the further work.



A-2D plot of layers 1-2 (wells-manifolds)



B-2D plot of layers 1-2-3 (wells-manifolds-platforms)



C-2D plot of layers 2-3-4 (manifolds-platforms-facility)

Figure 47- 2D plots (100 wells- 15 manifolds- 5 platforms- 1facility)

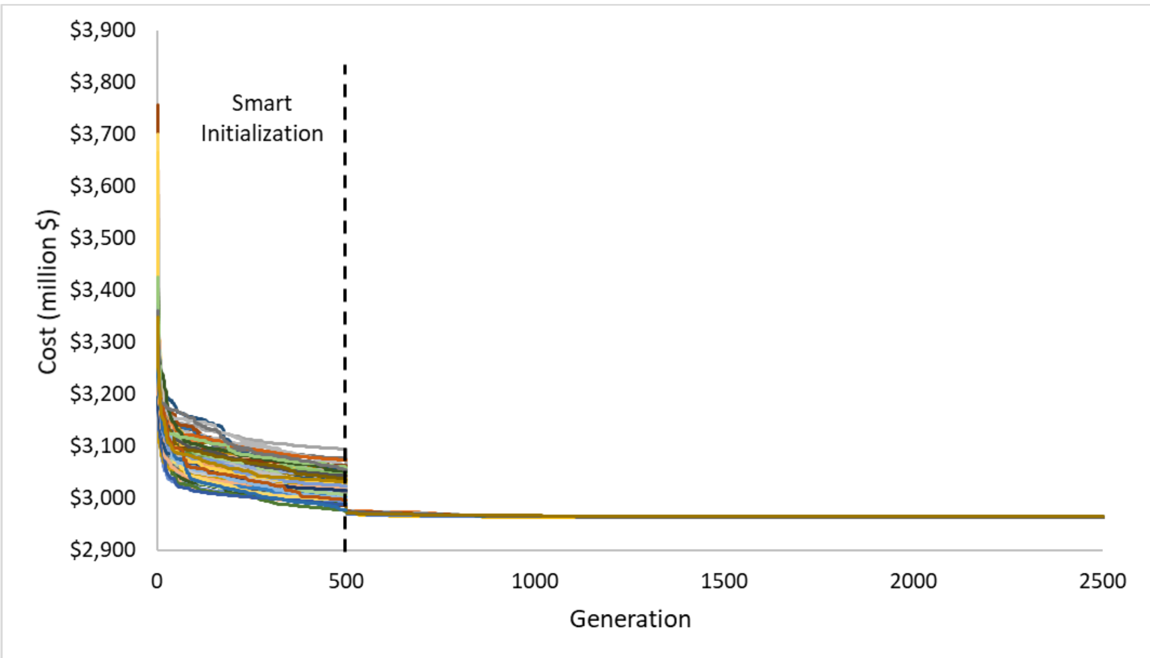


Figure 48- Objective Function Initialization and main optimization loop (100 wells- 15 manifolds-5 platforms-1 facility)

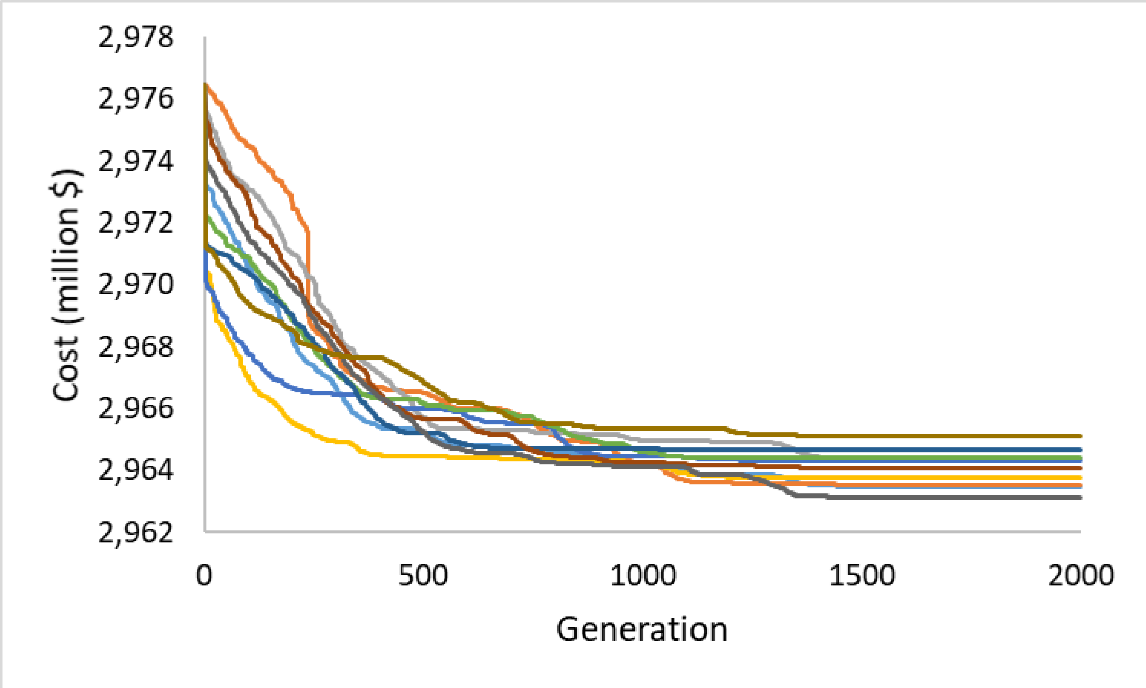


Figure 49- Objective Function of the main loop 10 runs (100 wells- 15 manifolds- 5 platforms- 1 facility)

D. Limitations and Proposed Way Forward

Production system optimization problem under study is a very important task that forms a major part of field development planning. Many assumptions were considered in this work for the sake of simplicity and to make sure that the proposed mathematical algorithm is tested. This work can be considered as a mathematical foundation for the next Engineering decision optimization.

Factors that should be considered in future work to address some limiting assumptions we have taken:

- Consider horizontal wells in the production system optimization
- Consider a detailed costing strategy for every component including drilling costs

- Consider prohibited areas for the placement of nodes and while routing the segments
- Consider the dogleg severity while choosing the entry point to the wells and other constraints while connecting nodes in different layer i.e. risers that connect platforms to PLEMs should have a base on a well-defined constrained distance from platform.
- Consider the topology of the seabed while optimizing production system.

CHAPTER V

CONCLUSION

Optimization of production system is at the core of the economic development of oil and gas reservoirs. Offshore production system optimization is a challenging problem that have been extensively studied. In this work, we represented an extensive literature review of the methods that are used in optimizing production system problems. However, proposed methods have limitations 1) in terms of the number of layers; they considered only two layers problems 2) in terms of the number of nodes in each layer. In this work, we used a recent applied method in this field of research i.e. genetic algorithm GA, and we compared this method to another population-based method the particle swarm optimization on a two layers example. PSO outperformed GA on all levels of complexity.

In addition, our defined problem is not restricted to a specific number of layers; it is a generic mathematical solution that can handle any number of layers in an optimization problem. We first used PSO as the main optimizer assisted with other functions i.e. clustering function. PSO algorithm that showed good results was compared to an improved version of PSO that overcomes some convergence problems that have been faced by the standard algorithm. Three main improvements were added to the standard algorithm: 1. Adaptive PSO (literature), 2. Smart initialization using Multiple runs initialization, and 3. Restart Function. These improvements led to better solutions by the proposed algorithm compared to the Standard PSO, especially when the complexity of the example in hand is increased. Another aspect of the proposed algorithm is the consistency of the obtained results while comparing the results of different independent runs starting from the same initialization.

The proposed algorithm shows a superiority over standard PSO, in parallel with the robustness that has been verified by the consistency of the results.

REFERENCES

[1] P. Hansen, E. de Luna Pedrosa Filho, and C. C. Ribeiro, "Location and sizing of offshore platforms for oil exploration," *European Journal of Operational Research*, vol. 58, no. 2, pp. 202-214, 1992.

[2] C. Panahli, "Implementation of Particle Swarm Optimization Algorithm within FieldOpt Optimization Framework-Application of the algorithm to well placement optimization," NTNU, 2017.

[3] O. J. Isebor, "Derivative-free optimization for generalized oil field development," Stanford University, 2013.

[4] Y. A. Ahmed, "Optimization of Well Design and Location in a Real Field," PhD thesis, Stanford University, Department of Energy Resources Engineering, 2009. ga09aAYAbukhamsin, 2009.

[5] J. Onwunalu, "Optimization of field development using particle swarm optimization and new well pattern descriptions," Stanford University, 2010.

[6] O. J. Isebor, D. Echeverría Ciaurri, and L. J. Durlofsky, "Generalized field-development optimization with derivative-free procedures," *SPE Journal,* vol. 19, no. 05, pp. 891-908, 2014.

[7] V. R. Rosa and V. J. M. Ferreira Filho, "Optimizing the location of platforms and manifolds," in *ASME 2012 31st International Conference on Ocean, Offshore and Arctic Engineering*, 2012, pp. 813-818: American Society of Mechanical Engineers.

[8] W. Watson Jr, D. Mahaffey, J. Still, and R. Taylor, "PLATLOC: A program for optimizing offshore platform locations," in *Petroleum Computer Conference*, 1989: Society of Petroleum Engineers.

[9] S. Dogru, "Selection of optimal platform locations," *SPE Drilling Engineering,* vol. 2, no. 04, pp. 382-386, 1987.

[10] L. d. P. A. Sales, A. R. Pitombeira-Neto, and B. de Athayde Prata, "A genetic algorithm integrated with Monte Carlo simulation for the field layout design problem," *Oil & Gas Sciences and Technology–Revue d'IFP Energies nouvelles,* vol. 73, p. 24, 2018.

[11] F. P. Campozana, R. L. Dos Santos, M. G. Madeira, S. H. G. Sousa, and M. Spinola, "Optimization of Surface Network and Platform Location using a Next Generation Reservoir Simulator Coupled with an Integrated Asset Optimizer-An Application to an Offshore Deep Water Oil Field in Brazil," in *International Petroleum Technology Conference*, 2008: International Petroleum Technology Conference.

[12] H. Rodrigues, B. Prata, and T. Bonates, "Integrated optimization model for location and sizing of offshore platforms and location of oil wells," *Journal of Petroleum Science and Engineering,* vol. 145, pp. 734-741, 2016.

[13] S. Sivanandam and S. Deepa, *Introduction to genetic algorithms*. Springer Science & Business Media, 2007.

[14] R. Eberhart and J. Kennedy, "A new optimizer using particle swarm theory," in *MHS'95. Proceedings of the Sixth International Symposium on Micro Machine and Human Science*, 1995, pp. 39-43: Ieee.

[15] W. Zhang, D. Ma, J.-j. Wei, and H.-f. Liang, "A parameter selection strategy for particle swarm optimization based on particle positions," *Expert Systems with Applications,* vol. 41, no. 7, pp. 3576-3584, 2014.

[16] M. Jiang, Y. P. Luo, and S. Y. Yang, "Stochastic convergence analysis and parameter selection of the standard particle swarm optimization algorithm," *Information Processing Letters,* vol. 102, no. 1, pp. 8-16, 2007.
[17] I. C. Trelea, "The particle swarm optimization algorithm: convergence analysis and parameter selection," *Information processing letters,* vol. 85, no. 6, pp. 317-325, 2003.
[18] R. C. Eberhart and Y. Shi, "Comparing inertia weights and constriction factors in particle swarm optimization," in *Evolutionary Computation, 2000. Proceedings of the 2000 Congress on*, 2000, vol. 1, pp. 84-88: IEEE.
[19] A. Harb, H. Kassem, and K. Ghorayeb, "The Blank Hole Particle Swarm Optimization for Well Placement Optimization," *Computational Geoscience,* 2018.
[20] J. C. Bansal, P. Singh, M. Saraswat, A. Verma, S. S. Jadon, and A. Abraham, "Inertia weight strategies in particle swarm optimization," in *Nature and Biologically Inspired Computing (NaBIC), 2011 Third World Congress on*, 2011, pp. 633-640: IEEE.
[21] Y. Shi and R. Eberhart, "A modified particle swarm optimizer," in *Evolutionary Computation Proceedings, 1998. IEEE World Congress on Computational Intelligence., The 1998 IEEE International Conference on*, 1998, pp. 69-73: IEEE.
[22] R. C. Eberhart and Y. Shi, "Tracking and optimizing dynamic systems with particle swarms," in *Evolutionary Computation, 2001. Proceedings of the 2001 Congress on*, 2001, vol. 1, pp. 94-100: IEEE.
[23] A. Nikabadi and M. Ebadzadeh, "Particle swarm optimization algorithms with adaptive Inertia Weight: A survey of the state of the art and a Novel method," *IEEE journal of evolutionary computation*, 2008.
[24] J. Xin, G. Chen, and Y. Hai, "A particle swarm optimizer with multi-stage linearly-decreasing inertia weight," in *Computational Sciences and Optimization, 2009. CSO 2009. International Joint Conference on*, 2009, vol. 1, pp. 505-508: IEEE.
[25] R. C. E. Yuhui Shi, "Parameter Selection in Particle Swarm Optimization," *Evolutionary Programming,* vol. VII, 1998.
[26] Y. Feng, G.-F. Teng, A.-X. Wang, and Y.-M. Yao, "Chaotic inertia weight in particle swarm optimization," in *Innovative Computing, Information and Control, 2007. ICICIC'07. Second International Conference on*, 2007, pp. 475-475: IEEE.
[27] M. S. Arumugam and M. Rao, "On the performance of the particle swarm optimization algorithm with various inertia weight variants for computing optimal control of a class of hybrid systems," *Discrete Dynamics in Nature and Society*, vol. 2006, 2006.
[28] G. Chen, X. Huang, J. Jia, and Z. Min, "Natural exponential inertia weight strategy in particle swarm optimization," in *Intelligent Control and Automation, 2006. WCICA 2006. The Sixth World Congress on*, 2006, vol. 1, pp. 3672-3675: IEEE.
[29] K. Kentzoglanakis and M. Poole, "Particle swarm optimization with an oscillating inertia weight," presented at the Proceedings of the 11th Annual conference on Genetic and evolutionary computation, Montreal, Québec, Canada, 2009.

[30] W. Al-Hassan, M. Fayek, and S. Shaheen, "Psosa: An optimized particle swarm technique for solving the urban planning problem," in *Computer Engineering and Systems, The 2006 International Conference on*, 2006, pp. 401-405: IEEE.
[31] Y.-l. Gao, X.-h. An, and J.-m. Liu, "A particle swarm optimization algorithm with logarithm decreasing inertia weight and chaos mutation," in *Computational Intelligence and Security, 2008. CIS'08. International Conference on*, 2008, vol. 1, pp. 61-65: IEEE.
[32] H.-R. Li and Y.-L. Gao, "Particle swarm optimization algorithm with exponent decreasing inertia weight and stochastic mutation," in *Information and Computing Science, 2009. ICIC'09. Second International Conference on*, 2009, vol. 1, pp. 66-69: IEEE.
[33] R. C. Eberhart, Y. Shi, and J. Kennedy, *Swarm intelligence*. Elsevier, 2001.
[34] M. Clerc, "The swarm and the queen: towards a deterministic and adaptive particle swarm optimization," 1999.
[35] A. Ratnaweera, S. K. Halgamuge, and H. C. Watson, "Self-organizing hierarchical particle swarm optimizer with time-varying acceleration coefficients," *IEEE Transactions on evolutionary computation*, vol. 8, no. 3, pp. 240-255, 2004.
[36] J. Kennedy, "Particle swarm optimization," in *Encyclopedia of machine learning*: Springer, 2011, pp. 760-766.
[37] Z.-H. Zhan, J. Zhang, Y. Li, and Y.-H. Shi, "Orthogonal learning particle swarm optimization," *IEEE transactions on evolutionary computation*, vol. 15, no. 6, pp. 832-847, 2011.
[38] R. Fonseca, C. Geel, and O. Leeuwenburgh, "Description of OLYMPUS reservoir model for optimization challenge," *Integrated Systems Approach to Petroleum Production,* 2017.