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

THE GRADUATE SCHOOL

A COVARIANCE MATRIX ADAPTATION ALGORITHM FOR SIMULTANEOUS ESTIMATION OF OPTIMAL PLACEMENT AND CONTROL OF PRODUCTION AND WATER INJECTION WELLS

by
Walter E. Poquioma

A thesis submitted in partial fulfillment of
the requirements for the degree of Master of Science
in the Discipline of Petroleum engineering

The Graduate School
The University of Tulsa

2015

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

THE GRADUATE SCHOOL

A COVARIANCE MATRIX ADAPTATION ALGORITHM FOR SIMULTANEOUS ESTIMATION OF OPTIMAL PLACEMENT AND CONTROL OF PRODUCTION AND WATER INJECTION WELLS

by
Walter E. Poquioma

A THESIS

APPROVED FOR THE DISCIPLINE OF

PETROLEUM ENGINEERING

By Thesis Committee

_____, Co-chair
Albert C. Reynolds

_____, Co-chair
Fahim Forouzanfar

_____
William Coberly

ABSTRACT

Walter E. Poquioma (Master of Science in Petroleum engineering)

A Covariance Matrix Adaptation Algorithm for Simultaneous Estimation of Optimal Placement and Control of Production and Water Injection Wells

Directed by Albert C. Reynolds and Fahim Forouzanfar

121 pp., Chapter 7: Discussion and Conclusions

(365 words)

In this thesis, both simultaneous and sequential algorithms for the joint optimization of well trajectories and their life-cycle controls are presented. The trajectory of a well is parameterized in terms of six variables which define a straight line in 3D. In the simultaneous joint optimization algorithm, the set of controls of a well throughout the life cycle of the reservoir is constructed as a linear combination of the left singular vectors which correspond to the largest singular values of a specified temporal covariance matrix. This covariance matrix is used to impose a temporal correlation on the controls at each well. In this approach, well controls are parameterized in terms of a few optimization parameters to reduce the dimension of the joint optimization problem. Moreover, the imposed smoothness on the well controls will result in temporally smooth well controls. An implementation of the Covariance Matrix Adaptation - Evolution Strategy (CMA-ES) optimization algorithm is used to solve the defined optimization problem. In the sequential optimization algorithm, first the trajectories of the wells are optimized using the CMA-ES optimization algorithm while the controls of the wells are pre-specified. Once the optimum trajectories of the wells are obtained, the life-cycle production optimization step is performed in order to find the optimal well controls for the specified well trajectories. For the production optimization step, a comparison is made between the performance of three production optimization algorithms

which are the standard EnOpt algorithm, the standard CMA-ES algorithm and a variant of the CMA-ES algorithm in which we set the initial covariance matrix equal to a pre-specified covariance which imposes a temporal correlation on the controls of each well. The performance of the proposed algorithms is tested for the joint optimization of well trajectories and controls of injectors and producers for the PUNQ and the egg reservoir models. The proposed simultaneous well placement-well control optimization algorithm obtained better results than the sequential optimization framework. The CMA-ES algorithm performed well for both well placement and production optimization purposes. Moreover, the CMA algorithm with a pre-specified covariance that imposes a temporal correlation on the well controls showed a superior performance compared to EnOpt for the life-cycle production optimization step of the sequential framework.

ACKNOWLEDGEMENTS

It is obvious that no good work can be done entirely alone and we continuously rely in others to achieve our objectives. This work is no different from others in that sense, therefore, I feel the need to thank many people for helping me to achieve this milestone in my life.

I want to start by thanking Dr. Reynolds for allowing me to form part of his research group, TUPREP, where I have met some of the most talented researchers. His input in the research work that I present here was key and his level of insight is unparalleled. I want to also thank him for teaching me how to express myself better and how to present my work adequately. I would also like to thank Dr. Forouzanfar, without whom this project would not have been possible. He guided me through every step of the process and gave his input from the very beginning. I am not overstating his contribution to this work. I also want to thank him for not only being a mentor but also for being a friend and for giving me advice on different matters during my stay at the university. Besides my advisors, I want to thank Dr. William Coberly for serving on my thesis committee and Dr. Kelkar for opening the doors of this university to me. I also want to thank Judy Teal from TUPREP for giving me a hand whenever I needed it.

I would like to express my gratitude to my girlfriend, Azraa Meyer, for always supporting me and telling me to go after my dreams. I believe she has an infinite amount of patience for continuing to care for me even when I am thousands of miles away from her. I would also like to thank my friends here at the university who have been there for me through the good times and especially the difficult times. Finally, I would like to thank my parents, Florangel Escorcia and Julio Poquioma, for being my guides through life in every aspect of it and always believing that I can achieve many great things.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

The advent of highly efficient computers in recent years has allowed numerical optimization algorithms to be applied to a wide range of real world problems to solve them relatively quickly. These problems involve finding an optimal set of parameters that maximize the profit (or minimize the cost) of a complex process that depends on these parameters. An instance of such a problem occurs in the field development planning of an oil reservoir which can be parametrized in terms of a large number of variables that will ultimately determine how much hydrocarbon is recovered from the reservoir and at what rate. The problem could include within its set of parameters the number of wells that should be drilled, their type (injector or producer), the schedule for the deployment of each well, their trajectory, the number of laterals each well has (if multi-lateral wells are allowed), and the controls of the wells throughout the stipulated reservoir life. To determine the cumulative oil produced from the beginning of the reservoir life until its end for a given combination of parameters, a computationally expensive reservoir simulation that models the physics of the reservoir must be performed. Therefore, an exhaustive search for an optimal set of parameters for a real world case is generally not an option. This is why numerical optimization algorithms are important because they offer a relatively rapid improvement of the cost function.

In the available literature of reservoir optimization, the proposed algorithms have generally focused on maximizing the net present value (NPV) obtained throughout the entire (pre-specified) reservoir life by performing well placement and/or well control optimization. In well placement optimization, the set of variables are usually the location of the wells and their trajectories while the well controls are assumed to be known a priori. On the other hand, well control (or production) optimization usually has the well controls (bottomhole pressures

or fluid rates) as the variables while the locations of the wells remain fixed. These two optimization problems have traditionally been treated separately because it has been shown that the NPV function exhibits very different behaviours depending on which of the two parametrizations is used. In particular, the surface of the NPV objective function for the well placement problem tends to be very rough [43, 18]. This has led to the use of gradient based (locally convergent) algorithms for the well controls optimization and non-gradient based optimization algorithms for the well placement problem. However, recently there has been an increasing trend towards solving the joint optimization of both well placement and well controls in simultaneous and sequential frameworks, e.g., [34, 2, 25, 17]. Most of the results obtained in the literature show that joint optimization outperforms both the well placement and production optimization algorithms because a bigger part of the parameter space can be explored using the joint approach. Furthermore, results have also shown that the joint simultaneous algorithms tend to do better than their sequential counterparts [34, 2, 17, 27]. Apparently, this is because in the sequential approach, optimal well locations are found for the fixed controls at every iteration of the algorithm while in the simultaneous approach, both the well controls and the well locations can vary freely allowing a more efficient sweep of the parameter space [2]. The main disadvantage of the joint optimization algorithms is that computational cost (number of simulation runs) needed for them to converge may be greater than that of sequential algorithms due to the large number of dimensions of the former. Generally, the number of simulation runs required to obtain convergence increases as the number of optimization parameters increases. Therefore one of the objectives in this thesis is to create a joint optimization framework in which the number of parameters that are being optimized is kept reasonably low. This is done by approximating the set of well controls with a smaller number of parameters that capture only the low frequency changes of the actual controls. This re-parametrization imposes temporal correlations between the well controls (of the same well) which actually has a smoothing effect on the function and aids the convergence of the algorithm. The Covariance Matrix Adaptation-Evolutionary Strategy (CMA-ES) algorithm is used to solve the well placement and the simultaneous

joint optimization problems. The CMA-ES is a stochastic algorithm that generates a number of multivariate Gaussian realizations at each generation at which the objective function is evaluated. The information gained by the evaluation of these random realizations is then used to update the parameters of the multivariate normal distribution until convergence criterion is met. CMA-ES was chosen as an optimization algorithm mainly due to its robust performance on non-smooth, non-convex functions and the versatility it offers in terms of switching from global to local search by the tuning of one parameter (population size) [5]. The Ensemble Optimization (EnOpt) [8] algorithm is implemented in this thesis to solve the well control optimization step within the joint sequential framework. The proposed framework is tested for two reservoir models: PUNQ [13] and the egg reservoir [29] models.

In the following, first a brief introduction to optimization and the previous work done in reservoir optimization are presented followed by the description of the parametrization for the trajectory and controls of a well. Secondly, the formulation of the simultaneous joint optimization problem of well trajectories and controls is presented. This is followed by a detailed explanation of the CMA-ES algorithm used to solve the presented optimization problem. Then the application of the presented algorithm to solve the joint optimization problem for the PUNQ and egg reservoir models is explained. In the computational results section, the sequential optimization framework is also investigated for optimizing the trajectories and controls of the wells. In this section, in addition to the investigation of the sequential optimization framework, a modified implementation of the CMA-ES algorithm for the life-cycle production optimization problem is presented and tested on the PUNQ reservoir model. Interestingly, the CMA-ES algorithm outperforms the EnOpt algorithm for the life-cycle production optimization of 3 injection and 6 production wells. Moreover, the proposed joint optimization algorithm outperformed the sequential optimization algorithm in all the tests that were performed on the PUNQ and the egg reservoir models.

1.1 Optimization

Often times it is necessary to find the minimum or maximum value of a scalar valued

function that depends on a set of parameters. The process of finding the set of parameters that yields the minimum (or maximum) function value is referred to as optimization. The optimization approach depends mainly on wether the function is linear or nonlinear in its parameters and the type of values the parameters can take on, e.g. discrete or continuous. There are other aspects of the function which also need to be taken into consideration in order to choose the most effective optimization method, such as continuity and smoothness of the function. Moreover, the parameters of the function could have some constraints on the values they can take on which will also affect the choice of the optimization algorithm.

If the function to be optimized is smooth, then information from the gradient and sometimes the Hessian (or approximated gradient and approximated Hessian) is used by the optimization algorithms to locate an extremum point [39], however, when no gradient information is available, many algorithms look for ingenious ways of exploiting the information gained only by the evaluation of the function at different points in the parameter space, e.g., [52, 8, 51, 32, 24]. Optimization algorithms can therefore be divided into two broad categories: gradient based and non-gradient based algorithms. In this thesis, the aim is to perform optimization on a nonlinear objective function subject to boundary and linear constraints on its parameters using a non-gradient based optimization algorithm. Hereinafter the term *objective function* will be used to denote the function that needs to be optimized and its parameters will be referred to either as *optimization variables* or *design parameters*.

1.2 General well placement optimization problem

Well placement optimization is the key element in the decision making process for preparing the optimal development plan for a reservoir. To solve a "general well placement optimization" problem, the number of wells, their trajectories and controls should be optimized in order to maximize a prescribed objective function, which is typically the net present value (NPV) of the life-cycle hydrocarbon production from the reservoir. Very few papers have addressed this general problem as it is extremely computationally expensive. Yeten et al. proposed in [57] an algorithm for optimizing the trajectory of a multi-lateral well

configuration (as well as its number of laterals) and the well controls but used only a single well control for the life cycle of the reservoir and, due to the relatively small number of parameters, were able to solve the problem using the genetic algorithm. They also investigated the effects that geological uncertainty could have on their overall solution and realized that the optimal number of laterals depended on whether one or multiple geological realizations were used. In [55] Wang et al. proposed a gradient-based optimization algorithm to simultaneously optimize the number of wells, their locations and controls for the rate-specified vertical wells. In their test cases, the controls and locations of the producers were assumed to be known while the number of injectors, their locations and their injection rates were optimized. Their algorithm starts by populating all the gridblocks that do not contain a producer with an injector while imposing a constraint on the specified total injection rate. The algorithm uses the steepest ascent method by computing the gradient of the NPV with respect to the injection rates of the wells, in this manner after some iterations of the steepest ascent algorithm certain wells are eliminated if their rates are reduced to zero. It should also be noted that in this algorithm the injection rates were considered to be constant throughout the reservoir life. Later, in [17], Forouzanfar and Reynolds improved the algorithm described in [55] to consider the bottomhole pressure constraints for the wells and also proposed an initialization step in order to mitigate the effect of the initial specified total injection and production rates for the wells. Isebor et al. proposed in [26] a generalized field-development optimization framework in which the number of wells, their types and drilling sequence, as well as their locations and controls subject to nonlinear constraints are optimized using derivative free optimization algorithms such as: particle swarm optimization (PSO) method, Branch and Bound (B&B), Mesh Adaptive Direct Search (MADS) and a PSO-MADS hybrid. They present this generalized approach as a mixed integer programming problem where the well controls are represented by continuous variables discretized in time, the well locations are represented by integer valued vectors and the decision of drilling a well and its type are defined by a categorical variable. However, the framework is computationally very expensive due to two reasons: the high dimension of the problem and the fact that solving mixed

integer programming problems is generally more difficult than solving problems where all the optimization parameters are real-valued variables.

1.3 Well trajectory description

The optimization of the placement of the wells could be defined in a general manner as the optimization of the location and trajectory of directional and multi-lateral wells, e.g., [57, 11, 54, 5, 17]. To parameterize the location and in general the trajectory of the wells, both discrete variables (simulation model gridblock indices), e.g., [44, 1, 11, 2, 34] and continuous real-valued variables, e.g., [43, 5, 17] are used. For example, in the well placement optimization method of Emerick et al. in [11], a directional well is a straight line and the gridblock indices, (i, j, k), of the two endpoints of the well are the optimization variables, whereas in the Bouzarkouna et al. [5] well placement optimization method, the trajectory of a multi-lateral well is parameterized in terms of real-valued variables. In the algorithm presented in [5], a multi-lateral well is composed of a multi-segment mainbore with multiple laterals. The "mainbore" is parameterized in a spherical coordinate system in terms of its length and two orientation angles which are measured relative to the spatial coordinates of its heel. A lateral, which stems from the mainbore, is defined in a similar manner with its reference point defined by its distance from the heel of the mainbore. In this thesis, we consider the optimization of the trajectory of a directional well which is represented as a line in 3D.

1.4 Global search vs local search algorithms

The heterogeneity of the underlying reservoir description makes the surface of the objective function for the well placement optimization non-smooth (gradient is not well defined) and multi-modal (multiple local optima) [43, 18], particularly when the well controls are bottomhole pressures. Due to the roughness of the objective function, the application of efficient local gradient-based or approximated gradient-based optimization algorithms to the well placement optimization problem may not be successful. In addition, simulators

do not commonly provide an efficient procedure to compute the gradient of the objective function which is needed to apply efficient gradient-based algorithms. For these reasons, heuristic global optimization algorithms are commonly implemented to solve the optimization problem. The genetic algorithm (GA) method, e.g., [11, 57, 44], particle swarm optimization (PSO) method, e.g., [43], the CMA-ES algorithm, e.g., [5] and hybrid algorithms, e.g., [40], have all been used. Although these heuristic methods are designed to obtain the global optimum solution of the problem, due to the limited available computational resources, the solutions obtained are usually far from the global optimum. On the other hand, the superior computational efficiency of some local search optimization algorithms compared to the heuristic optimization methods motivated their application to the well placement optimization problem. For example, gradient-based algorithms using adjoint gradient [20, 55, 54, 17], methods based on the Simultaneous Perturbation Stochastic Approximation (SPSA) algorithm [1, 34], derivative-free methods [16, 26] and direct search algorithms such as Hooke-Jeeves direct search (HJDS) and generalized pattern search (GPS) [2], have all been applied for the optimal well placement problem.

For the pure well placement optimization problem, the well controls are usually fixed at some specified values throughout the reservoir life, while the operational and economic constraints (e.g., maximum well watercut) are enforced as reactive constraints. Therefore, the obtained locations of the wells are optimal when the pre-specified well controls are incorporated for the life cycle of the reservoir. However, in real life, the controls of the wells are optimized in the life-cycle production optimization process of the reservoir management. The reservoir production optimization problem is concerned with optimizing the well controls within the life cycle of the reservoir in order to maximize the production from a reservoir, e.g., [6, 28, 38, 49, 53, 8, 7]. This optimization problem is usually defined by assigning a set of optimization variables for the controls of each well on a set of specified time intervals which span the life of the reservoir. Therefore, the number of optimization variables in the production optimization problem is usually large. However, numerous results in the literature (e.g., [53, 9]) imply that for the production optimization problem, there exist hyper-plateaus along

which the objective function is almost constant; hence, efficient local optimization algorithms could successfully be implemented to obtain a near-optimal solution. The existence of these hyper-plateaus for optimal well control problems is the obvious analogue of the existence of long flat valleys for history matching problems first noted in Oliver et al. [42]; see the discussion of Fig. 8.1 in Chapter 8 of the book.

As addressed above, for the well placement problem, global search optimization algorithms are desirable. However, the application of these algorithms for problems with a relatively high number of optimization variables, such as the joint optimization problem of well locations and controls, is generally prohibitive. Due to the above considerations, it is common in petroleum engineering to perform the well placement and production optimization steps separately. The well placement step is performed using fixed-specified well controls and once the optimal locations of the wells are obtained, one may perform the production optimization step to find the optimal well controls. However, it has been shown that the optimal locations of the wells depend on their specified controls [58, 15, 17] and this sequential optimization approach may result in a suboptimal solution [34, 2].

1.5 Joint optimization problem

Recently, there has been a growing interest in solving the joint optimization problem of well locations and controls in the reservoir optimization community, where both sequential and simultaneous optimization algorithms are proposed to find the optimum locations of the wells and their optimum controls. The sequential joint optimization of locations and trajectories of wells and their controls follows multiple steps. In one step, the set of well trajectory variables is optimized while the set of well controls is specified and fixed. After obtaining the optimized well trajectories, the set of well controls are optimized keeping the location of the wells fixed. One may iterate between these optimization steps in order to improve the optimum solutions. As mentioned above, this strategy could lead to suboptimal solutions.

Li and Jafarpour [34] presented a sequential algorithm where a coordinate descent

random search is used to do the well placement optimization step and a Gauss-Newton approach is used to perform the well control optimization step. In their algorithm, the well locations are parametrized in terms of their gridblock indices and the well controls of the injectors and producers are represented as real vectors as a result of the discretization of the controls in the time domain. The producers and injectors are controlled through their rates. Well distance constraints are considered for the well placement optimization step to avoid solutions where two or more wells are clustered together. A penalty term, that increases as the sum of all the distances between the wells decreases, is added to the objective function in the well placement optimization step, in order to handle these well distance constraints. It must also be added, that the penalty term used in their work could potentially generate a bias towards well placement solutions where the sum of the distances between wells is maximized. In their approach a series of measures are taken so that the problem of maximizing the NPV function is reduced to the minimization of the total cumulative water production. The measures taken to simplify the optimization problem are: incompressible fluid flow, constant total water injection rate (and liquid production rate due to the incompressible fluid flow assumption), constant discount factor of 1 and constant time steps. The simplified objective function is then augmented with the linear constraints imposed on the total liquid production rate and total injection rate to form the Lagrangian. This Lagrangian is then solved by using the Gauss-Newton method. Li and Jafarpour [34] proposed iterating with the aforementioned sequential process but they show their results only up until the second well placement optimization step. Their results show that using the sequential optimization algorithm provides a significant improvement of the objective function value when compared to performing a single well placement optimization step. In [27] and [35] a variant of the simultaneous perturbation stochastic approximation (SPSA) algorithm is used to solve the simultaneous optimization problem while the original SPSA algorithm is used to solve the sequential optimization problem. In the simultaneous joint optimization approach, the optimization variables corresponding to both well controls and well locations are concatenated to form a mixed integer optimization problem (well locations being described by discrete values

and well controls by continuous variables) which is then solved with a SPSA algorithm. The results show that the simultaneous joint optimization algorithm is able to generate better objective function values than the sequential approach. In their work, geologic uncertainty is also taken into account by performing robust optimization. In their approach, however, instead of evaluating the objective function on all the geological realizations generated (full ensemble) they propose to use a small number of these realizations during each optimization iteration of the algorithm. These realizations are randomly picked in a manner that guarantees that all the realizations in the ensemble have been used to compute the NPV after some number of iterations of the optimization algorithm have passed. The results obtained suggest high savings in computational time by using this robust optimization algorithm while still generating comparable results to the ones obtained using the full ensemble at each iteration.

Bellout et al. proposed in [2] a nested joint optimization algorithm where the upper level optimization problem is the well placement optimization and the lower level was the well controls optimization problem. Therefore, each function evaluation of the well placement optimization problem involves the full well control optimization at those particular well locations. In this formulation the wells are vertical and their locations are defined by their real valued coordinates while the well controls are represented by real valued vectors. Pattern search algorithms (HJDS, GPS and HOSPACK) are used to perform the well placement optimization part of the algorithm. They justified their preference of pattern search methods over stochastic search methods by the fact that the latter do not have a solid convergence theory and therefore need tuning of parameters which is often difficult [2]. However, it must be mentioned that Spall did give a proof of convergence for SPSA in [52]. For the production optimization step, an efficient sequential quadratic programming algorithm using adjoint gradient for the well control optimization is used. They were able to show that the nested joint optimization algorithm outperforms the sequential well placement approach. The main drawback of the algorithm is its high computational cost, as every function evaluation of the well placement optimization entails a full production optimization step and this leads to an increase in the number of function evaluations of the NPV function of about one order of

magnitude compared to sequential procedures. An interesting result found in this work was that the landscape for the well placement optimization in the joint optimization algorithm is smoother than that of the sequential case which allows for an easier global search in the well placement problem. They attribute the smoothing of the function to the fact that the performance of wells in less-promising locations can be significantly improved by the optimization of the controls [2].

Humphries et al. in [25] compared both the simultaneous and the sequential optimization approaches for unconstrained (in terms of production and injection rates) and constrained problems. In this algorithm the wells are vertical and therefore fully specified by two real valued coordinates (x, y) while the controls are represented as real valued vectors. A hybrid algorithm that combined the global search capabilities of PSO and the local search characteristic of GPS is used for the simultaneous optimization case. Advantage can be taken of the fact that both algorithms are highly parallelizable so parallelization can be used to save run time. In the sequential approach, PSO is used to solve the well placement problem with fixed well controls (pressure controlled) followed by the optimization of well controls with GPS. Their results suggest that in average the sequential case performs better than the simultaneous approach. They believe there could be two reasons for this. First, a favorable choice of initial well controls could have been chosen in the well placement step leading to a subspace conducive to good solutions, and second, by performing sequential optimization PSO is able to perform a more exhaustive search of the space due to the decrease in the dimension of the problem. In their algorithm, they also suggest methods for handling bound and general constraints. Bound constraints are handled in PSO by projecting back to the boundary any particle that travels outside the feasible region (area in the parameter space where constraints are satisfied) and then setting its velocity to zero so that the particle does not go in that direction in the future. A similar strategy is used for GPS. For other type of constraints (linear and nonlinear) they suggest rejecting infeasible points. The drawback in using this strategy is that the algorithm must always start inside a feasible region and it does not allow the algorithms to move across infeasible regions to find the optimum value of

the function [25].

In the formulation of the well placement optimization by Forouzanfar and Reynolds in [17], a gradient projection algorithm using adjoint gradients simultaneously optimizes the number of wells, their locations and a single control at each well. Although their method seems to be more efficient than other proposed joint optimization algorithms, their algorithm is applicable only for rate-specified wells.

As pointed out in the results of the previous studies, solving the joint optimization problem is computationally expensive, even for the case of vertical wells, e.g., the algorithm in [2] requires a few thousand reservoir simulation runs whereas more than ten thousand simulation runs are used in the algorithm described in [25]. In this thesis, the joint optimization of well trajectories and controls is investigated using simultaneous and sequential algorithms. An algorithm is proposed for parameterizing the problem of simultaneous joint optimization of well trajectories and controls in order to achieve the following objectives: 1- Improve the convergence of the algorithm by imposing a temporal correlation on the controls of the wells and 2- Reduce the dimension of the optimization problem. In the algorithms presented, the trajectory of a general directional well is represented as a line in 3D and it is defined in terms of six continuous variables. In order to define the simultaneous joint optimization problem, the controls of a well are parameterized in terms of a few optimization variables. Therefore, the total number of optimization variables per well in the joint optimization problem is kept relatively small. The joint optimization problem is solved using a Covariance Matrix Adaptation Evolution Strategy (CMA-ES) algorithm. The CMA-ES is a state of the art stochastic optimization algorithm that has been applied to solve several engineering problems, e.g., [37, 56, 22] and a great number of benchmark optimization functions [36]. In the sequential optimization algorithm, the well placement and production optimization steps are broken into two separate steps which are performed sequentially. For the well placement step, the CMA-ES optimization algorithm is implemented and, for the life-cycle production optimization step, both the Ensemble Optimization (EnOpt) and CMA-ES algorithms are considered. The EnOpt algorithm is a local optimization algorithm that approximates the

gradient of the function at a point by creating an ensemble of realizations around the point in question and using the information gained from their function evaluations. It has become popular due to its success in the production optimization problem and the fact that the adjoint gradient is not required.

To the best knowledge of the author, the CMA-ES algorithm was first applied by Bouzarkouna et al. [5] to the well placement optimization problem. As previously mentioned, in their work multi-lateral wells are parametrized in terms of a multi-segment mainbore with multiple laterals. The mainbore is parametrized in a spherical coordinate system in terms of its length and two orientation angles which are measured relative to the spatial coordinates of its heel. Laterals are also parametrized in a spherical coordinate system but in this case the reference point for the coordinate system is defined by the distance from the heel of the mainbore at which the lateral stems. In their test case the trajectories of one unilateral injector and one unilateral producer are optimized for the PUNQ reservoir model. They compared their results against those obtained by using the genetic algorithm, and found that the CMA-ES algorithm outperforms the GA in terms of computational efficiency and the final average NPV obtained. The computational savings can be attributed to the use of local meta-models (based on the quadratic approximation of the objective function). Moreover, a method is presented for handling constraints referred to as adaptive penalization with rejection, in which infeasible points are rejected if the point violating the constraint is far from the constraint boundary and the adaptive penalization method is used with infeasible points that are close to the boundary.

Fonseca et al. proposed a hybrid EnOpt-CMA algorithm in [14] for the reservoir production optimization problem. In their proposed algorithm, the covariance matrix used in the Enopt algorithm to generate the different multivariate normal distributed samples is updated (adapted) at every iteration with the covariance matrix update of the CMA-ES algorithm. Their improvement lies in the fact that this algorithm does not need the user to have prior knowledge of the covariance matrix as it will be adapted by the algorithm itself.

CHAPTER 2

WELL TRAJECTORY PARAMETRIZATION

The trajectory of a directional well is represented as a three dimensional (3D) line in this work. This is similar to the parametrization of the mainbore segment and laterals in the well placement algorithm of [5]. The schematic of the trajectory of a directional well as a function of well trajectory variables is shown in Fig. 2.1.

The trajectory of well w is parameterized in terms of 6 optimization variables which are given by

$$\boldsymbol{T}_w = [x_w, y_w, z_w, l_w, \theta_w, \phi_w]^T, \tag{2.1}$$

where (x_w, y_w, z_w) represent the spatial coordinate of the well trajectory center point; l_w is the length of the well; θ_w is the orientation angle of the well trajectory in the horizontal direction and ϕ_w is the inclination angle of the well trajectory in the vertical direction. Note that other specific trajectories of a vertical, horizontal or directional well can be derived by reducing the set of well trajectory parameters. For example, for the case of a fully-penetrating vertical well, the set of trajectory parameters will be reduced to $\boldsymbol{T}_w = [x_w, y_w]^T$. The bound constraints for the trajectory parameters, x_w, y_w, and z_w, are determined using the reservoir boundaries. The maximum and minimum lengths of the well are pre-specified. The bound constraints for θ_w and ϕ_w are defined as $0 \leq \theta_w \leq \pi$ and $0 \leq \phi_w \leq \pi$, respectively.

The determination of the perforations of a well and their productivity indices is as follows. For a given set of trajectory parameters for each well, the length of the well segments inside each penetrated gridblock and their respective orientations inside these gridblocks are determined through geometrical computations. The segment is then modelled by using the Peaceman [45] well model which leads to the following expression for the productivity index

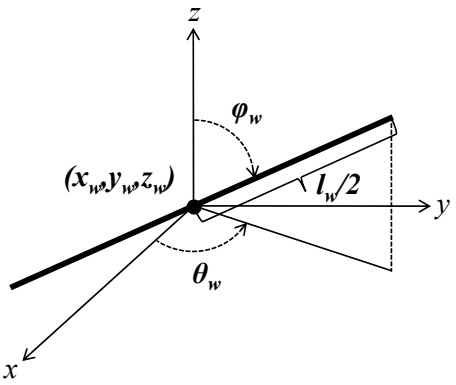


Figure 2.1: Schematic trajectory of a directional well [16].

of segment i:

$$PI_i = \frac{2\pi\alpha k_i h_i}{\ln \frac{r_{o_i}}{r_w}}, \tag{2.2}$$

where α is the unit conversion factor needed for consistency when field units are used ($\alpha = 1.127 \times 10^{-3}$), k_i in millidarcy is the "effective" permeability of the gridblock i in the radial direction, h_i in feet is the net length of the perforation interval in gridblock i, r_w is the wellbore radius in feet and r_{o_i} in feet is the "equivalent radius" of the perforation at gridblock i. The product $k_i h_i$ and the parameter r_{o_i} of a perforation are determined by using the procedure given in the Eclipse reservoir simulator manual [50]. To describe the procedure used in [50] it is convenient to first introduce the notation of $\tilde{x_i}\tilde{y_i}\tilde{z_i}$ to represent the local coordinate system for gridblock i with $\tilde{e_x}$, $\tilde{e_y}$, and $\tilde{e_z}$ as the unit vectors pointing in the positive $\tilde{x_i}$, $\tilde{y_i}$, and $\tilde{z_i}$ directions, respectively. With the aforementioned notation the axis (centerline) of the well located in griblock i can be represented by the vector:

$$\boldsymbol{h}_i = h_{x,i}\tilde{e_x} + h_{y,i}\tilde{e_y} + h_{z,i}\tilde{e_z}, \tag{2.3}$$

where $h_{x,i}$, $h_{y,i}$, and $h_{z,i}$ are the components of the segment length of the well in the local coordinate system of gridblock i.

In [50] the quantity $k_i h_i$ for perforation i is calculated as follows:

$$k_i h_i = \sqrt{\left(\sqrt{k_{y,i}k_{z,i}}h_{x,i}\right)^2 + \left(\sqrt{k_{x,i}k_{z,i}}h_{y,i}\right)^2 + \left(\sqrt{k_{x,i}k_{y,i}}h_{z,i}\right)^2}, \tag{2.4}$$

where $k_{x,i}$, $k_{y,i}$, and $k_{z,i}$ are the permeability of gridblock i in the $\tilde{x}_i$, $\tilde{y}_i$, and $\tilde{z}_i$ directions, respectively. In order to obtain the equivalent radius of the perforation, r_{o_i}, the method given in [50] first computes the productivity indices of the perforation in the $\tilde{x}_i$, $\tilde{y}_i$, and $\tilde{z}_i$ directions and then calculates a weighted average of these indices. As an example of how the individual productivity indices are computed in each direction the manner in which the productivity index in the $\tilde{x}_i$ direction is calculated is shown. The productivity index in the $\tilde{x}_i$ direction is given by

$$PI_{x,i} = \frac{2\pi\alpha\sqrt{k_{y,i}k_{z,i}}h_{x,i}}{\ln\frac{r_{o_{x,i}}}{r_w}}, \tag{2.5}$$

where $r_{o_{x,i}}$ is calculated by using the equation derived by Peaceman in [46],

$$r_{o_{x,i}} = 0.28\left(\frac{\left[\left(\frac{k_{y,i}}{k_{z,i}}\right)^{1/2}\Delta z_i^2 + \left(\frac{k_{z,i}}{k_{y,i}}\right)^{1/2}\Delta y_i^2\right]^{1/2}}{\left[\left(\frac{k_{y,i}}{k_{z,i}}\right)^{1/4} + \left(\frac{k_{z,i}}{k_{y,i}}\right)^{1/4}\right]}\right). \tag{2.6}$$

The average productivity index of the perforation is calculated by

$$\overline{PI_i} = \sqrt{PI_{x,i}^2 + PI_{y,i}^2 + PI_{z,i}^2}. \tag{2.7}$$

Finally $r_{o_{x,i}}$ can be calculated by solving for it in

$$\overline{PI_i} = \frac{2\pi\alpha k_i h_i}{\ln\frac{r_{o_i}}{r_w}}, \tag{2.8}$$

which results in

$$r_{o_i} = e^{\left(2\pi\alpha k_i h_i / \overline{PI_i}\right)} \times r_w. \tag{2.9}$$

It should mentioned that using this well model may result in discontinuities in the rate of the wells (which reflect as discontinuities in the objective function value) when vertical or horizontal wells move across the boundary of two gridblocks. The reason for this is that when a well is moved from one gridblock to another one next to it the productivity index of the well may change in a discontinuous manner due to the heterogenous permeability field.

However the CMA-ES algorithm, like many other random search algorithms, is designed to be robust even when such discontinuities exist in the objective function. In [17], a method for obtaining a relatively smooth NPV function by distributing the rate of a well among its neighbouring blocks is presented by Forouzanfar and Reynolds.

CHAPTER 3

WELL CONTROL PARAMETRIZATION IN THE SIMULTANEOUS JOINT OPTIMIZATION ALGORITHM

The set of well controls for well w can be represented by

$$\boldsymbol{u}_w = [u_{w,1}, \cdots, u_{w,n_c}]^T, \tag{3.1}$$

where $u_{w,j}$ is the well control during the jth control step and n_c is the total number of controls steps. The well controls are usually defined within some bound constraints given by

$$u_w^{\text{low}} \leq u_{w,j} \leq u_w^{\text{up}}, \qquad j = 1, \cdots, n_c, \tag{3.2}$$

where u_w^{low} and u_w^{up} are the lower and upper bound values for the controls of well w, respectively. Defining one optimization variable corresponding to the control of a well at every control step (Eq. 3.1) will result in a large number of optimization variables. As mentioned previously, due to the possibility of numerous local optima, a global search optimization algorithm is desired for the joint optimization of well locations and controls. However, the implementation of such optimization algorithms for solving the joint optimization problem is computationally impractical when a large number of control steps are considered for each well, i.e., when n_c is large. One approach to deal with this problem is to define a very small number of control steps (see e.g., [57]) in order to keep the number of optimization variables small, but this can lead to a suboptimal solution.

In this work, a second approach to parameterize the set of controls of a well in terms of an arbitrary number of optimization variables is proposed. To do so, it is assumed that the controls of a well are temporally correlated in time. To quantify this temporal correlation

of the controls of the well w, a temporal correlation on the controls of each well is imposed. This is done by specifying a covariance matrix for the controls at each well. In this thesis, a spherical covariance function is used to form the covariance matrix for the controls of well w. This covariance matrix is denoted by $\boldsymbol{C}_U^w$ and is given by

$$\boldsymbol{C}_U^w = [c_{U_{i,j}}^w], \tag{3.3}$$

where

$$c_{U_{i,j}}^w = \begin{cases} \sigma_w^2 \left[1 - \frac{3|i-j|}{2N_s} + \frac{1}{2}\left(\frac{|i-j|}{N_s}\right)^3\right] & , \quad |i-j| \leq N_s, \\ 0 & , \quad \text{otherwise}, \end{cases} \tag{3.4}$$

where σ_w is the standard deviation for the well control; i and j refer to the ith and jth control steps, respectively; and N_s is the correlation length, i.e., the number of control steps over which the controls at well w are correlated. Here, it is considered that any set of controls of a well can be represented as an outcome of a multi-variate Gaussian distribution with a mean of $\bar{\boldsymbol{u}}_w$ and a covariance matrix of $\boldsymbol{C}_U^w$. The vector of mean well controls is defined by

$$\bar{u}_{w,j} = \frac{u_w^{\text{up}} + u_w^{\text{low}}}{2}, \qquad j = 1, \cdots, n_c. \tag{3.5}$$

Since the well controls are considered to be normally distributed, the standard deviation of the controls of the well, w, is defined as

$$\sigma_w = \frac{u_w^{\text{up}} - u_w^{\text{low}}}{6}, \tag{3.6}$$

which is based on the fact that if we sample Gaussian random variable, 99.7% of the time, we will obtain a sample within three standard deviations of its mean value. Given the temporal covariance matrix of the well controls, a random realization for the set of well controls is obtained by

$$\boldsymbol{u}_w = \bar{\boldsymbol{u}}_w + (\boldsymbol{C}_U^w)^{\frac{1}{2}}\, \boldsymbol{z}, \tag{3.7}$$

where the vector $\boldsymbol{z}$ is a vector of independent standard random normal deviates, i.e., $\boldsymbol{z} = \mathcal{N}(\boldsymbol{0}, \boldsymbol{I})$.

Following Eq. 3.7, any set for the controls of well w can be parameterized as:

$$\boldsymbol{u}_w = \bar{\boldsymbol{u}}_w + (\boldsymbol{C}_U^w)^{\frac{1}{2}} \boldsymbol{P}_w, \tag{3.8}$$

where $\boldsymbol{P}_w$ is the vector of parameterized optimization variables of well w, i.e., the vector of optimization variables, $\boldsymbol{u}_w$, is replaced by the parametrization vector $\boldsymbol{P}_w$. Note that the dimension of the vector of parameterized variables, $\boldsymbol{P}_w$, is equal to the dimension of the vector of control variables, $\boldsymbol{u}_w$.

The covariance matrix, $\boldsymbol{C}_U^w$, is real symmetric positive definite. To solve for the term $(\boldsymbol{C}_U^w)^{\frac{1}{2}}$ the singular value decomposition (SVD) of the covariance matrix is used in order to obtain

$$\boldsymbol{C}_U^w = \boldsymbol{U}^w \boldsymbol{\Gamma}^w (\boldsymbol{U}^w)^T, \tag{3.9}$$

therefore in Eqs. 3.7 and 3.8 the following expression can be used

$$(\boldsymbol{C}_U^w)^{\frac{1}{2}} = \boldsymbol{U}^w (\boldsymbol{\Gamma}^w)^{\frac{1}{2}}. \tag{3.10}$$

By substituting Eq. 3.10 into Eq. 3.8, the following is obtained

$$\boldsymbol{u}_w = \bar{\boldsymbol{u}}_w + \boldsymbol{U}^w (\boldsymbol{\Gamma}^w)^{\frac{1}{2}} \boldsymbol{P}_w. \tag{3.11}$$

Note that generating the vector of well controls using Eq. 3.8 imposes some degree of temporal smoothness on the control variables of the well depending on the value of N_s in Eq. 3.4. The singular vectors corresponding to the largest singular values represent the low frequency changes in the values of the well controls through the control steps, and those corresponding to the smaller singular values represent the high frequency changes in the values of the well controls. Specifying the values for the components of the vector $\boldsymbol{P}_w$ gives

a specific vector of the well controls. However, since the dimension of the vector $\boldsymbol{P}_w$ is the same as the dimension of the vector of control variables, the optimization problem would still be large. In order to reduce the dimension of the variables representing the vector of well controls, only the n_{P} largest singular values in the SVD of the temporal covariance matrix, $\boldsymbol{C}_U^w$, in Eq. 3.8 are used. This effectively results in maintaining the low frequency changes in the set of well control variables while the high frequency changes will be deleted. Therefore, the vector of well controls will be represented by

$$\boldsymbol{u}_w = \bar{\boldsymbol{u}}_w + \boldsymbol{U}_{\mathrm{P}}^w \left(\boldsymbol{\Gamma}_{\mathrm{P}}^w\right)^{\frac{1}{2}} \boldsymbol{P}_{\mathrm{P},w}, \tag{3.12}$$

where $\boldsymbol{U}_{\mathrm{P}}^w$ is the reduced matrix of left singular vectors and $\boldsymbol{\Gamma}_{\mathrm{P}}^w$ is the reduced diagonal matrix of the singular values. In Eq. 3.12, the vector of well control variables is parameterized in terms of the vector $\boldsymbol{P}_{\mathrm{P},w}$ which is of a dimension n_{P}. By retaining only a small number of singular values, the dimension of the production optimization part of the joint well placement optimization problem is significantly reduced. Moreover, imposing the temporal smoothness on the control variables of the wells results in obtaining smooth controls for the wells. Due to the choice of the correlation length in practice (for example half of the reservoir life) the value of σ_w may result in rough well controls when a significant number of singular values are retained because $\boldsymbol{C}_U^{w\,\frac{1}{2}}$ may not provide sufficient smoothness to damp the high frequency changes, see Eq. 3.7 and 3.6. It should also be mentioned that even though each entry of the vector $\boldsymbol{P}_{\mathrm{P},w}$ can take on any value, by analogy to sampling from a multi-variate Gaussian distribution, the individual elements are truncated to stay within the range $[-6, 6]$. The truncation does not guarantee that the values of all components of the vector $\boldsymbol{u}_w$ will be between u_w^{up} and u_w^{low}. Therefore, when evaluating the objective function, if the value of a well control is outside its specified bounds, it is truncated back to its nearest bound value.

CHAPTER 4

SIMULTANEOUS JOINT WELL PLACEMENT OPTIMIZATION PROBLEM

The joint optimization problem considered here pertains to the estimation of the optimal well trajectories and the optimal well controls for water flooding of an oil reservoir which maximizes the net present value of production, NPV, from the reservoir given by

$$NPV(\boldsymbol{v}, \boldsymbol{y}(\boldsymbol{v})) = \sum_{n=1}^{N} \frac{\left[\sum_{i=1}^{N_{\text{prod}}} (r_{\text{o}} q_{\text{o},i}^n - r_{\text{w}} q_{\text{w},i}^n) - \sum_{i=1}^{N_{\text{winj}}} r_{\text{winj}} q_{\text{winj},i}^n\right] \Delta t^n}{(1+b)^{t^n/365}}. \tag{4.1}$$

Throughout, $\boldsymbol{v}$ is the vector of all optimization variables which includes the optimization variables corresponding to the trajectory and controls of all wells; $\boldsymbol{y}$ is the vector of simulation state variables at all simulation time steps; N is the total number of reservoir simulation time steps; N_{prod} is the total number of producers; N_{winj} is the total number of water injection wells; r_{o}, r_{w} and r_{winj}, all in \$/STB, are the oil revenue, the water injection cost and the water disposal cost, respectively; $q_{\text{o},i}^n$, $q_{\text{w},i}^n$ and $q_{\text{winj},i}^n$, all in STB/day, respectively are the average oil production rate of the i^{th} producer, the average water production rate of the i^{th} producer and the average water injection rate of the i^{th} injection well during the n^{th} time step, respectively; b is the annual discount rate; t^n is the cumulative time (days) up to the n^{th} time step; Δt^n in days is the length of n^{th} simulation time step. Note that the reservoir description is fixed in the problems considered here, hence, the reservoir model parameters do not appear in Eq. 4.1.

The problem of joint optimization of well trajectories and controls are then defined by

$$\max_{\boldsymbol{v}} NPV, \tag{4.2}$$

subject to

$$\boldsymbol{v}^{\text{low}} \leq \boldsymbol{v} \leq \boldsymbol{v}^{\text{up}}, \tag{4.3}$$

where

$$\boldsymbol{v} = [\boldsymbol{v}_1^T, \cdots, \boldsymbol{v}_{n_{\text{W}}}^T]^T, \tag{4.4}$$

$$\boldsymbol{v}_w^T = [\boldsymbol{T}_w^T, \boldsymbol{P}_{\text{P},w}^T], \qquad w = 1, \cdots, n_{\text{W}}, \tag{4.5}$$

where n_{W} is the number of wells subject to the optimization; $\boldsymbol{T}_w$ is the vector of well trajectory parameters for well w given in Eq. 2.1 and $\boldsymbol{P}_{\text{P,w}}$ is the vector of parameterized well control variables for well w (Eq. 3.12), i.e., $\boldsymbol{P}_{\text{P,w}} = [\text{p}_{w,i}, \cdots, \text{p}_{w,n_{\text{P}}}]$. The dimension of the joint optimization problem is equal to $n_{\text{W}} \times (n_{\text{T}} + n_{\text{P}})$, where n_{T} is the number of optimization variables for parameterizing the trajectory of a well and n_{P} is the number of optimization variables for parameterizing the controls of a well. If in the joint optimization problem, the full trajectory of a directional well is considered for the optimization then $n_{\text{T}} = 6$. However, when the wells are fully-penetrating vertical wells, $n_{\text{T}} = 2$.

CHAPTER 5

COVARIANCE MATRIX ADAPTATION - EVOLUTION STRATEGY (CMA-ES) OPTIMIZATION ALGORITHM

The Covariance Matrix Adaptation Evolution Strategy (CMA-ES) is a stochastic search algorithm that maximizes a real valued objective function $f(\boldsymbol{v}) : \mathbb{R}^n \to \mathbb{R}$ by continuously adapting the parameters that define the multivariate normal distribution (MVND) from which different candidate solutions are sampled [23]. The main steps in the algorithm are shown in Algorithm 1 while the full algorithm is given in Algorithm 2. The following

Algorithm 1 Covariance Matrix Adaptation-Evolution Strategy (general algorithm)

Set parameters

- Set variables used in the covariance matrix adaptation and step size adaptation: $c_c, c_\mu, c_1, c_\sigma, d_\sigma$.
- Set the population size λ and the offspring size μ.

Initialize distribution $\mathcal{N}\left(\boldsymbol{m}^{(0)}, {\sigma^{(0)}}^2 \boldsymbol{C}^{(0)}\right)$

- Set the initial covariance matrix: $\boldsymbol{C}^{(0)} = \boldsymbol{I}$
- Chose the initial mean and step size of the MVND, i.e. $\boldsymbol{m}^{(0)}$ and $\sigma^{(0)}$.

For generation $g = 0, 1, 2, ...$

- Sample λ independent search points from the MVND, evaluate them and rank them based on their objective function value from best to worst.
- Update the mean by using weighted intermediate recombination of the best μ points.
- Adapt the Covariance matrix of the distribution
- Adapt the step size of the distribution

Break if one or multiple stopping criteria are met

sections describe each of the steps performed by the algorithm in more detail and also explain the reasoning behind the updates.

5.1 Sampling of Population

The process of sampling is referred to as *mutation* in evolution strategies and is given by

$$\boldsymbol{v}_k^{(g+1)} = \boldsymbol{m}^{(g)} + \sigma^{(g)} \mathcal{N}\left(\mathbf{0}, \boldsymbol{C}^{(g)}\right), \qquad k = 1, ..., \lambda, \tag{5.1}$$

where $\boldsymbol{v}_k^{(g+1)}$ represents the k^{th} candidate solution (a real-valued n dimensional vector) of generation (iteration) $g+1$; $\boldsymbol{m}^{(g)}$, $\sigma^{(g)}$ and $\boldsymbol{C}^{(g)}$ represent the mean, the step size and the covariance matrix of the normal distribution at generation (iteration) g, respectively; λ pertains to the population size in the CMA-ES algorithm. Hansen [21] recommends using

$$\lambda = 4 + \text{floor}(3\ \ln(n)), \tag{5.2}$$

where n represents the dimension of the problem, i.e., the dimension of the vector of optimization variables $\boldsymbol{v}$. The formulation in Eq. 5.2 has to its advantage that the population size scales with the logarithm of the dimension of the problem; therefore, the population size increases slowly as the dimension of the problem increases.

In order to generate a sample from the distribution $\mathcal{N}\left(\mathbf{0}, \boldsymbol{C}^{(g)}\right)$, used in Eq. 5.1, the following equivalent expression is used:

$$\mathcal{N}\left(\mathbf{0}, \boldsymbol{C}^{(g)}\right) \sim \boldsymbol{C}^{(g)^{\frac{1}{2}}} \mathcal{N}\left(\mathbf{0}, \boldsymbol{I}\right), \tag{5.3}$$

where $\boldsymbol{I}$ is nxn the identity matrix. Sampling a point from the distribution $\mathcal{N}\left(\mathbf{0}, \boldsymbol{I}\right)$ is equivalent to sampling n independent standard normal deviates, with n being the dimension of the problem. The square root of the covariance matrix, $\boldsymbol{C}^{(g)^{\frac{1}{2}}}$, is computed by using its eigendecomposition, which is given by

$$\boldsymbol{C}^{(g)} = \boldsymbol{B}^{(g)} \boldsymbol{D}^{(g)} \boldsymbol{B}^{(g)^T} = \boldsymbol{B}^{(g)} \boldsymbol{D}^{(g)^{\frac{1}{2}}} \boldsymbol{D}^{(g)^{\frac{1}{2}^T}} \boldsymbol{B}^{(g)^T} = \left(\boldsymbol{B}^{(g)} \boldsymbol{D}^{(g)^{\frac{1}{2}}}\right) \left(\boldsymbol{B}^{(g)} \boldsymbol{D}^{(g)^{\frac{1}{2}}}\right)^T, \tag{5.4}$$

hence

$$\boldsymbol{C}^{(g)^{\frac{1}{2}}} = \boldsymbol{B}^{(g)}\boldsymbol{D}^{(g)^{\frac{1}{2}}}, \tag{5.5}$$

where $\boldsymbol{D}^{(g)}$ is a diagonal matrix with its diagonal elements corresponding to eigenvalues of $\boldsymbol{C}^{(g)}$ and $\boldsymbol{B}^{(g)}$ is a matrix with columns that form an orthonormal basis of eigenvectors corresponding to the eigenvalues. Since $\boldsymbol{D}^{(g)}$ is a diagonal matrix obtaining its square root is very cheap because it only involves computing the square roots of the elements in its main diagonal. The eigendecomposition of $\boldsymbol{C}^{(g)}$ is preferred over other types of factorizations because the eigenvalues of $\boldsymbol{C}^{(g)}$ are used later in the algorithm to check the condition number of the matrix and determine whether it has degenerated or not.

5.2 Mean Update

The initial value of the mean, $\boldsymbol{m}^0$, is chosen by the user based on his best guess of the optimal solution. If there is no reasonable guess for $\boldsymbol{m}^0$ then it can be drawn randomly from the uniform distribution on the hypercube constructed from the upper and lower bounds on the optimization variables [21].

The mean of the MVND is updated at every generation as a weighted sum of the best μ candidates from a pool of λ candidates. For a maximization problem, a candidate is considered better than another one if it yields a higher objective function value than the other candidate when it is evaluated, e.g. $\boldsymbol{v}_{1:\lambda}$ is considered to be better than $\boldsymbol{v}_{2:\lambda}$ if $f(\boldsymbol{v}_{1:\lambda}) > f(\boldsymbol{v}_{2:\lambda})$. The weighted sum, called the weighted intermediate recombination in ES, is itself a process of *selection* (as μ individuals are chosen from λ and different weights are assigned to each) and *recombination* (as the selected individuals are added together) [21]. At generation $g+1$, the mean of the MVND, $\boldsymbol{m}^{(g+1)}$, is computed by

$$\boldsymbol{m}^{(g+1)} = \sum_{i=1}^{\mu} w_i \boldsymbol{v}_{i:\lambda}^{(g+1)}, \tag{5.6}$$

where

$$w_i = \frac{w_i^{'}}{\sum_{j=1}^{\mu} w_j^{'}}, \qquad w_1 \geq w_2 \geq \ldots \geq w_\mu > 0, \tag{5.7}$$

and

$$w_i^{'} = \ln(\mu + 0.5) - \ln(\mathrm{i}) \qquad \text{for,} \quad \mathrm{i} = 1, ..., \mu. \tag{5.8}$$

w_i is the weight assigned to $\boldsymbol{v}_{i:\lambda}^{(g+1)}$, which represents the i^{th} best individual out of λ individuals of generation $g+1$. The weights of Eq. 5.8 are those suggested by [21].The parameter μ can be chosen by the user but Hansen [21] strongly recommends $\mu = \text{floor}\left(\frac{\lambda}{2}\right)$. The setting of weights through Eq. 5.8 generates weights that decay in an exponential fashion with the highest value corresponding to the best individual ($i = 1$) and the lowest to the μ^{th} best value ($i = \mu$).

A metric defined as variance effective selection mass, μ_{eff}, is used to check if the weight values are reasonable. It is defined as

$$\mu_{\text{eff}} = (\sum_{i=1}^{\mu} w_i^2)^{-1}. \tag{5.9}$$

Hansen mentioned that $\mu_{\text{eff}} \approx \frac{\lambda}{4} \approx \frac{\mu}{2}$ is a desirable value [21]. Note that this value is roughly an average value for the two limiting cases: one where all the candidates are given the same importance in the update of the mean, $w_i = 1/\mu \; \forall \, i = 1, ..., \mu$ ($\mu_{\text{eff}} = \mu$); and one where only the best candidate is considered for the update of the mean ($w_1 = 1$ and the rest of the weights are set to zero and $\mu_{\text{eff}} = 1$). As explained in the following, μ_{eff} is also used in other parts of the algorithm.

The purpose of updating the mean by using Eq. 5.6 is to move the center of the distribution to locations where better solutions can be found. The weighting gives a higher importance to the best individuals, therefore it is different to the sample mean which would

assign the same importance (weight) to all the candidates regardless of their fitness value.

Note that if the samples generated by Eq. 5.1 are interpreted as independent and identically distributed (i.i.d) random variables with distribution $\mathcal{N}\left(\boldsymbol{m}^{(g)}, {\sigma^{(g)}}^2 \boldsymbol{C}^{(g)}\right)$ then $\boldsymbol{m}^{(g+1)}$ can be seen as sample taken from the mixture distribution which is obtained by performing the convex combination of μ identical distributions of the form $\mathcal{N}\left(\boldsymbol{m}^{(g)}, {\sigma^{(g)}}^2 \boldsymbol{C}^{(g)}\right)$. This result can be obtained by replacing the sample $\boldsymbol{v}_{i:\lambda}^{(g+1)}$ with the distribution $\mathcal{N}\left(\boldsymbol{m}^{(g)}, {\sigma^{(g)}}^2 \boldsymbol{C}^{(g)}\right)$ in Eq. 5.6 which yields

$$\boldsymbol{m}^{(g+1)} \sim \sum_{i=1}^{\mu} w_i \mathcal{N}\left(\boldsymbol{m}^{(g)}, {\sigma^{(g)}}^2 \boldsymbol{C}^{(g)}\right). \tag{5.10}$$

The interpretation of $\boldsymbol{m}^{(g+1)}$ as a sample from the mixture distribution shown in Eq. 5.10 will be useful later in the chapter.

5.3 Covariance Matrix Adaptation

The adaptation of the MVND is further improved by the adaptation of the covariance matrix through the iterations of the algorithm. This update aims at shaping the contours of equal probability of the distribution so that they mimic the local shape of the iso-fitness contours of the objective function. However, the update does not control the overall standard deviation, σ, of the MVND (called step size in this context) which is controlled by the step size adaptation.

The covariance matrix adaptation combines two mechanisms to adapt the iso-density contours of the MVND so that they mimic the local shape of the iso-fitness contours of the objective function: the rank-μ and the rank-one updates. The rank-μ update tries to exploit the information obtained from one generation to the next. The rank-one update looks at the paths taken by the mean of the distribution through all the generations in an effort to incorporate information from the general trend that is being followed by the distribution. The update equation for the covariance matrix is given by

$$\boldsymbol{C}^{(g+1)} = (1 - c_1 - c_\mu)\boldsymbol{C}^{(g)} + c_1 \boldsymbol{p}_c^{(g+1)} \boldsymbol{p}_c^{(g+1)^T} + c_\mu \frac{\boldsymbol{C}_\mu^{(g+1)}}{{\sigma^{(g)}}^2}, \tag{5.11}$$

where $\boldsymbol{C}_{\mu}^{(g+1)}$ is a matrix with a rank of $\min(\mu, n)$, $\boldsymbol{p}_{c}^{(g+1)}$ is called the evolution path (with its self outer product giving a rank one matrix), c_μ is the learning rate of the rank-μ update and c_1 is the learning rate of the rank-one update. In the following sections, the two update mechanisms are explained separately to give the reader a better understanding of what each of them controls.

5.3.1 Rank-μ Update

The rank-μ update is derived from Eq. 5.11, with $c_1 = 0$ and $c_\mu \neq 0$:

$$\boldsymbol{C}^{(g+1)} = (1 - c_\mu)\boldsymbol{C}^{(g)} + c_\mu \frac{\boldsymbol{C}_{\mu}^{(g+1)}}{\sigma^{(g)^2}}. \tag{5.12}$$

The covariance matrix $\boldsymbol{C}_{\mu}^{(g+1)}$ has the following form:

$$\boldsymbol{C}_{\mu}^{(g+1)} = \sum_{i=1}^{\mu} w_i \left(\boldsymbol{v}_{i:\lambda}^{(g+1)} - \boldsymbol{m}^{(g)}\right)\left(\boldsymbol{v}_{i:\lambda}^{(g+1)} - \boldsymbol{m}^{(g)}\right)^T. \tag{5.13}$$

The covariance matrix $\boldsymbol{C}_{\mu}^{(g+1)}$ can be thought of as an estimator of the distribution variance of the best μ selected steps [21] (as opposed to the distribution variance of the selected population). This can be better understood by comparing Eq. 5.13 with the well known empirical (sample) covariance:

$$\boldsymbol{C}_{emp}^{(g+1)} = \frac{1}{\lambda - 1}\sum_{i=1}^{\lambda}\left(\boldsymbol{v}_i^{(g+1)} - \frac{1}{\lambda}\sum_{j=1}^{\lambda}\boldsymbol{v}_j^{(g+1)}\right)\left(\boldsymbol{v}_i^{(g+1)} - \frac{1}{\lambda}\sum_{j=1}^{\lambda}\boldsymbol{v}_j^{(g+1)}\right)^T. \tag{5.14}$$

There are two main differences between the covariance matrix estimators given by Eq. 5.14 and Eq. 5.13. The first and most important difference lies in the reference means used by each of the estimators. In Eq. 5.14 the reference mean used is the sample mean, therefore this is an estimator of the covariance matrix based on the sampled points. On the other hand, Eq. 5.13 uses the true mean of the distribution and this leads to a very different interpretation. If the vector $\boldsymbol{v}_{i:\lambda}^{(g+1)} - \boldsymbol{m}^{(g)}$ is seen as a sampled step from the mean, then Eq. 5.13 can in fact be perceived as an estimator of the covariance matrix of

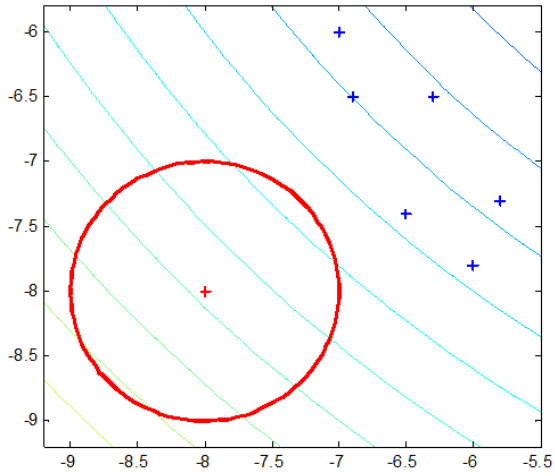
the distribution of a portion of the sampled steps. The second relevant difference between these two estimators is that only the best μ points are used to compute $\boldsymbol{C}_{\mu}^{(g+1)}$ while all the sampled points are used to obtain the empirical covariance. The objective of doing this is to increase the variance of $\boldsymbol{C}_{\mu}^{(g+1)}$ in the direction of the selected steps. Therefore, adding $\boldsymbol{C}_{\mu}^{(g+1)}$ to $\boldsymbol{C}^{(g)}$ will alter the shape of the distribution by increasing its variance more in previously successful directions than in any others so that the probability of generating these steps is higher [21]. The fact that different weights are used to give different importance to the outer products added in Eq. 5.13, whereas equal weights $(1/(\lambda-1))$ are used in Eq.5.14 is not a crucial difference between the two estimators.

To better understand why Eq. 5.13 increases the variance in the general direction of the selected steps, one should picture how the covariance matrix dictates the shape of equal probability contours in a normal distribution. Usually, the effect of covariance matrices in normal distributions is depicted as hyperellipsoids of equal probability that have their major axes aligned with the eigenvectors of the covariance matrix. These hyperellipsoids give an indication of the directions in which the variance has its highest value (eigenvector corresponding to largest eigenvalue) and its lowest one (eigenvector corresponding to smallest eigenvalue) in a distribution. The squared values of the lengths of the major axes of the hyperellipsoid correspond to the eigenvalues of the covariance matrix. With this mind, the other piece of information needed is that the outer product of a vector with itself generates a matrix of rank one whose only nonzero eigenvalue corresponds to the eigenvector that is aligned in the same direction as the vector used to create the matrix. Therefore it is easy to see how adding many matrices of the type $\left(\boldsymbol{v}_{i:\lambda}^{(g+1)}-\boldsymbol{m}^{(g)}\right)\left(\boldsymbol{v}_{i:\lambda}^{(g+1)}-\boldsymbol{m}^{(g)}\right)^{T}$ where most of the vectors point in a similar direction of improvement would generate a covariance matrix with a high variance in the same direction. The reader is referred to [21] for details about the representation of covariance matrices as equiprobability hyperellipsoids in space.

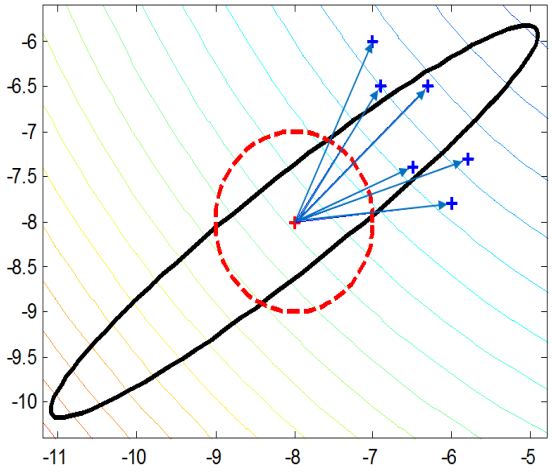
The effect of adding $\boldsymbol{C}_{\mu}^{(g+1)}$ to the covariance matrix of the distribution is depicted in Fig. 5.1. In the figure, the contours belong to the two-dimensional spherical function $\left(f(\boldsymbol{x})=\sum_{i=1}^{2}\boldsymbol{x}_i^2\right)$ that is being optimized by a particular version of the CMA-ES algorithm.

In this version the mean is updated as in the original CMA-ES algorithm and the covariance matrix is adapted by only adding $\boldsymbol{C}_{\mu}^{(g+1)}$ to it without using any weights. From Fig. 5.1 it can be seen that the variance of the initial distribution, $\mathcal{N}\left([-8,-8]^T, \boldsymbol{I}\right)$, is elongated in the direction of the selected steps after adding $\boldsymbol{C}_{\mu}^{(g+1)}$ to its covariance matrix. Note that this also has the effect of aligning the main axis of the ellipse (representing the covariance matrix) in a direction that is almost perpendicular to the plane that is tangent to the contour (which is a surface in higher dimensions) that goes through the mean ($\boldsymbol{m}^{(g)}$). If one of the two main factors that differentiate $\boldsymbol{C}_{\mu}^{(g+1)}$ from $\boldsymbol{C}_{emp}^{(g+1)}$ is not present in the calculation of $\boldsymbol{C}_{\mu}^{(g+1)}$ then the desired effect would not be possible. As an example consider using the best μ steps and their sample mean for the calculation instead. In this case the direction with the highest variance of the hypothetical $\boldsymbol{C}_{\mu}^{(g+1)}$ would not point in the direction of the successful steps. This is because the sample mean of the best μ steps would be around the center of the selected samples, therefore the vectors $\boldsymbol{v}_i^{(g+1)} - (1/\mu)\sum_{j=1}^{\mu} \boldsymbol{v}_j^{(g+1)}$ will point in very different directions to the vectors calculated using the original mean $\left(\boldsymbol{v}_{i:\lambda}^{(g+1)} - \boldsymbol{m}^{(g)}\right)$ as a reference. This type of covariance matrix would most likely have its lowest variance in the direction of improvement of the objective function and the highest variance in a perpendicular direction to the improving one (making it parallel to tangent hyperplane of the contour at the mean).

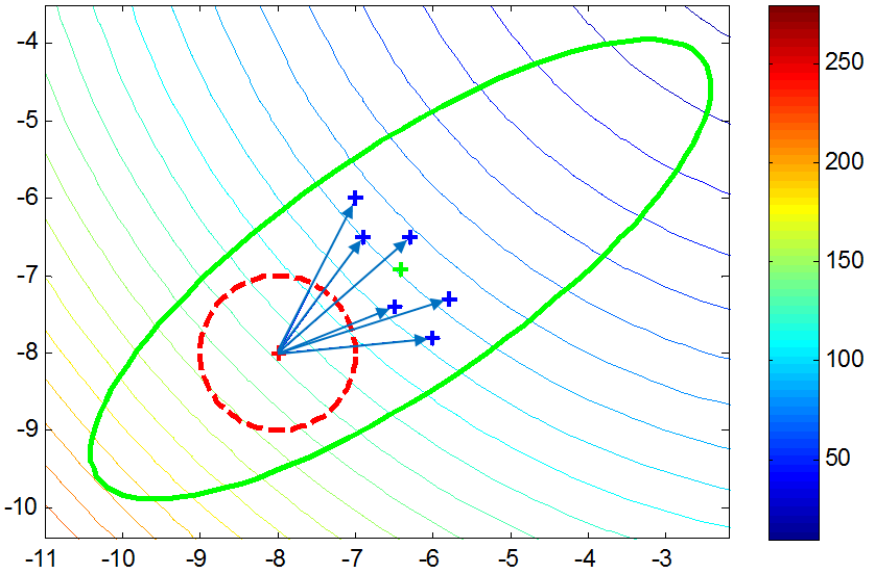
Another important aspect that needs to be discussed about the rank-μ update given in Eq. 5.12 is how $\boldsymbol{C}_{\mu}^{(g+1)}$ is added to the covariance of the present distribution, $\boldsymbol{C}^{(g)}$, to obtain the new covariance, $\boldsymbol{C}^{(g+1)}$. At first sight, Eq. 5.12 looks like a simple convex combination of $\boldsymbol{C}_{\mu}^{(g+1)}$ and $\boldsymbol{C}^{(g)}$, but a more thorough consideration of the equation shows that $\boldsymbol{C}^{(g)}$ has information of all the previous $\boldsymbol{C}_{\mu}$ matrices that were added together through the iterations.



(a) μ best samples (blue crosses) taken from $\mathcal{N}([-8,-8],\boldsymbol{I})$. The red cross represents the point $[-8,-8]$.



(b) Calculation of the best μ steps (blue vectors) and the point computation of C_μ (black ellipsoid)



(c) New distribution after updating the mean (green cross) and updating the covariance matrix by $\boldsymbol{C}^{(g+1)} = \boldsymbol{C}^{(g)} + \boldsymbol{C}_\mu^{(g+1)}$ (green ellipsoid)

Figure 5.1: Rank-μ update of the Covariance matrix of the distribution $\mathcal{N}([-8,-8],\boldsymbol{I})$ in the first iteration of the optimization of the 2D spherical function.

The expansion of Eq. 5.12 in terms of all the previously added $\boldsymbol{C}_\mu$ matrices is given by

$$\begin{aligned}
\boldsymbol{C}^{(g+1)} &= (1-c_\mu)\boldsymbol{C}^{(g)} + c_\mu \frac{\boldsymbol{C}_\mu^{(g+1)}}{\sigma^{(g)^2}} \\
&= (1-c_\mu)^{g+1}\boldsymbol{C}^{(0)} + \\
&c_\mu \left((1-c_\mu)^g \frac{\boldsymbol{C}_\mu^{(1)}}{\sigma^{(0)^2}} + (1-c_\mu)^{g-1} \frac{\boldsymbol{C}_\mu^{(2)}}{\sigma^{(1)^2}} + ... + (1-c_\mu) \frac{\boldsymbol{C}_\mu^{(g)}}{\sigma^{(g-1)^2}} \right) + c_\mu \frac{\boldsymbol{C}_\mu^{(g+1)}}{\sigma^{(g)^2}} \qquad (5.15) \\
&= (1-c_\mu)^{g+1}\boldsymbol{C}^{(0)} + \\
&c_\mu \left((1-c_\mu)^g \frac{\boldsymbol{C}_\mu^{(1)}}{\sigma^{(0)^2}} + (1-c_\mu)^{g-1} \frac{\boldsymbol{C}_\mu^{(2)}}{\sigma^{(1)^2}} + ... + (1-c_\mu) \frac{\boldsymbol{C}_\mu^{(g)}}{\sigma^{(g-1)^2}} + (1-c_\mu)^0 \frac{\boldsymbol{C}_\mu^{(g+1)}}{\sigma^{(g)^2}} \right)
\end{aligned}$$

and is called exponential smoothing. This is a popular averaging technique commonly used on time series data to smooth out fluctuations and obtain the general trend the data is following so that predictions can be made. It is called exponential smoothing because it assigns exponentially decaying weights to the points (in this case $\boldsymbol{C}_\mu$ matrices) so that the most recent ones have the higher weights. In this manner after a certain amount of generations some of the older matrices, e.g. $\boldsymbol{C}^{(0)}$, $\boldsymbol{C}_\mu^{(1)}$ at $g = 50$ with $c_\mu = 0.1$, have negligible influence on the whole sum. The main reason Hansen used this was to damp the changes of the covariance matrix of the distribution and allow enough time for the adaptation so that matrix degeneration was avoided [21]. The other reason is that information from previous generations is used to update the covariance matrix.

The learning rate c_μ controls the relative importance given to the $\boldsymbol{C}_\mu$ matrices calculated in the most recent generations over the ones calculated in older generations. To be precise the number $1/c_\mu$ is approximately equal to the number of recent generations that contain 63% of the total weight (see Eq. 5.15). Therefore, as c_μ increases, the number of recent generations that contain 63% of the total weight decreases. This can be shown by using the fact that the weights of the $\boldsymbol{C}_\mu$ matrices form a geometric progression with a scale factor c_μ and a common ratio $(1 - c_\mu)$. The sum of the elements of a geometric progression

is given by

$$s = a + ar + ar^2 + ... + ar^{n-1} = a\frac{1-r^n}{1-r}, \qquad , r \neq 1, \tag{5.16}$$

where a represents the scale factor and r represents the common ratio. Applying this equation to the geometric series given by

$$s = c_\mu \sum_{i=g+1-\Delta g}^{g} (1-c_\mu)^{g-i} = c_\mu + c_\mu(1-c_\mu) + c_\mu(1-c_\mu)^2 + ... + c_\mu(1-c_\mu)^{\Delta g-1}, \tag{5.17}$$

yields

$$s = c_\mu \sum_{i=g+1-\Delta g}^{g} (1-c_\mu)^{g-i} = c_\mu \frac{1-(1-c_\mu)^{\Delta g}}{1-(1-c_\mu)} = c_\mu \frac{1-(1-c_\mu)^{\Delta g}}{c_\mu} = 1-(1-c_\mu)^{\Delta g}, \tag{5.18}$$

where Δg is the number of generations for which the percentage of the total weight needs to be determined. Looking at the term $(1-c_\mu)^{\Delta g}$ in Eq. 5.18 it is easy to see that as Δg increases, s takes on different values along an inverse exponential decay curve that approaches 1 asymptotically. The following approximation is used to determine the value for Δg which contains around 63% of the total weight:

$$1-(1-c_\mu)^{\Delta g} \approx 0.63 \approx 1-\tfrac{1}{e}. \tag{5.19}$$

It follows that

$$(1-c_\mu)^{\Delta g} \approx \tfrac{1}{e}. \tag{5.20}$$

And taking the natural log of Eq. 5.20,

$$\Delta g \ln(1-c_\mu) \approx -1. \tag{5.21}$$

By taking a first order Mercator series around 0,

$$\Delta g(-c_\mu) \approx -1 \tag{5.22}$$

and, thus,

$$\Delta g \approx \tfrac{1}{c_\mu}. \tag{5.23}$$

Eq. 5.23 means that 63% of the information comes from the $1/c_\mu$ most recent generations (see Eq. 5.17). The learning rate is in fact controlling how fast the covariance matrix is allowed to change. If $c_\mu = 1$ the rank-μ update only uses the information of the current $\boldsymbol{C}_\mu$, whereas if $c_\mu = 0$ no new information is gained and the covariance matrix will remain unchanged through the iterations. As previously mentioned, changes in the covariance matrix need to occur slowly to avoid the danger of degeneration, i.e., low values of c_μ are needed, but on the other hand, if the learning is too slow the distribution would not be adapted in a reasonable number of function evaluations. In [21] Hansen suggests that a good setting of the learning rate for the rank-μ update is

$$c_\mu = \min\left(1 - c_1, 2\frac{\mu_{\text{eff}} - 2 + 1/\mu_{\text{eff}}}{(n+2)^2 + \mu_{\text{eff}}}\right), \tag{5.24}$$

where c_1 is given by Eq. 5.26.

The last point to cover about the rank-μ update is the division of $\boldsymbol{C}_\mu^{(g+1)}$ by $\sigma^{(g)^2}$. Dividing $\boldsymbol{C}_\mu^{(g+1)}$ by $\sigma^{(g)^2}$ removes the effect of the different σ's used to generate the sampled steps, $\boldsymbol{v}_{i:\lambda}^{(g+1)} - \boldsymbol{m}^{(g)}$, in different generations; see Eq. 5.1. This allows the independent adaptation of $\boldsymbol{C}$ and σ, which respectively control the shape of the distribution (orientation of the major axes of the hyperellipsoids of equal probability and the ratios between the lengths of these axes) and the scale of the distribution (as it the factor by which the major axes of the hyperellipsoid defined by $\boldsymbol{C}$ are expanded or shrunk). The separate adaptation of $\boldsymbol{C}$ and σ is necessary because the optimal size of the overall standard deviation of the

distribution, σ, cannot be adapted quickly enough (in terms of iterations) by the covariance matrix adaptation. The rank-μ and rank-one updates are designed to adapt the shape of the distribution alone. Note also that by dividing the sampled steps, $\boldsymbol{v}_{i:\lambda}^{(g+1)} - \boldsymbol{m}^{(g)}$, by $\sigma^{(g)}$ these steps could be interpreted as samples of a random variable with distribution $\mathcal{N}\left(\mathbf{0}, \boldsymbol{C}^{(g)}\right)$ instead of $\mathcal{N}\left(\boldsymbol{m}^{(g)}, \sigma^{(g)^2}\boldsymbol{C}^{(g)}\right)$. This can be derived easily by solving for $\mathcal{N}\left(\mathbf{0}, \boldsymbol{C}^{(g)}\right)$ in Eq. 5.1. The condition of steps being drawn from $\mathcal{N}\left(\mathbf{0}, \boldsymbol{C}^{(g)}\right)$ is needed to obtain an unbiased update. By an unbiased update it is meant that $\mathrm{E}\left[\boldsymbol{C}^{(g+1)}|\boldsymbol{C}^{(g)}\right] = \boldsymbol{C}^{(g)}$ which is needed in order to avoid problems such as premature convergence due to the overuse of fitness information. This has to do with the fact that in evolutionary algorithms the parameters of the distribution are updated using two operators: selection and variation. The selection operator uses fitness information in order to select the parents of the next population while the variation operator recombines the parameters of the parents and generates new candidates. The purpose of the variation operator is to explore the solution space and not to exploit the information given by the fitness values because this is already done by the selection operator. The updates of the mean, the covariance matrix and, the step size (overall standard deviation) of the distribution use the already selected candidates and recombine them (they are variation operators); therefore, they need to satisfy the unbiasedness condition to avoid an overuse of the information gained by the fitness information that could lead to an imbalance between exploration of the search space and exploitation of old information. In this manner, if the candidates were selected without using any fitness information (randomly for example), the updates will have zero bias towards any change in the current values of the parameters of the distribution [4]. See Appendix A for proof of the unbiasedness of the three updates.

5.3.2 Rank-one Update

The rank-one update is obtained from Eq. 5.11, with $c_\mu = 0$ and $c_1 \neq 0$:

$$\boldsymbol{C}^{(g+1)} = (1 - c_1)\boldsymbol{C}^{(g)} + c_1 \boldsymbol{p}_c^{(g+1)} \boldsymbol{p}_c^{(g+1)^T}. \tag{5.25}$$

This update increases the likelihood of generating mutation steps in the direction

defined by the *evolution path*. The reason behind this is similar to the explanation given in the previous section about how adding $\boldsymbol{C}_u^{(g+1)}$ to the covariance matrix of the distribution increases the variance in the direction of selected steps. The outer product $\boldsymbol{p}_c^{(g+1)}\boldsymbol{p}_c^{(g+1)^T}$ creates a covariance matrix that has zero variance in any other direction except the one given by $\boldsymbol{p}_c^{(g+1)}$. To visualize why this happens, it is easier to go back to the interpretation of a covariance matrix as a hyperellipsoid (explained in [21]) where the eigenvectors of the matrix represent the major axes of the hyperellipsoid. Since the matrix $\boldsymbol{p}_c^{(g+1)}\boldsymbol{p}_c^{(g+1)^T}$ only has one nonzero eigenvalue that corresponds to the eigenvector which points in the same direction as $\boldsymbol{p}_c^{(g+1)}$, the hyperellipsoid collapses to a single line. Adding this line distribution to the present distribution will then increase the variance of the latter only in the direction given by $\boldsymbol{p}_c^{(g+1)}$.

The rank-one update uses exponential smoothing for similar reasons to the ones discussed in the rank-μ update. The learning rate c_1 controls the amount of information used from previous rank one matrices generated from older evolution paths to update the variance of the current distribution. According to Hansen [21] an adequate setting of c_1 is given by

$$c_1 = \frac{2}{(n+1.3)^2 + \mu_{\text{eff}}}. \tag{5.26}$$

The *evolution path*, $\boldsymbol{p}_c$, is basically a weighted sum of the steps taken by the mean of the distribution through all the generations [21]. The sum uses a process defined as *cumulation* which is essentially exponential smoothing (described in the previous section). The objective of creating this vector is to use sign information that would be otherwise lost if the outer product $\left(\frac{\boldsymbol{m}^{(g+1)}-\boldsymbol{m}^{(g)}}{\sigma^{(g)}}\right)\left(\frac{\boldsymbol{m}^{(g+1)}-\boldsymbol{m}^{(g)}}{\sigma^{(g)}}\right)^T$ is used directly for the rank-one update. This can be seen by careful examination of the following equation:

$$\boldsymbol{p}_c^{(g+1)} = (1-c_c)\boldsymbol{p}_c^{(g)} + \sqrt{c_c(2-c_c)\mu_{\text{eff}}}\left(\frac{\boldsymbol{m}^{(g+1)}-\boldsymbol{m}^{(g)}}{\sigma^{(g)}}\right), \tag{5.27}$$

where c_c is the learning rate for the cumulation of $\boldsymbol{p}_c$ ($0 \leq\ c_c \leq 1$) which controls how much

information from the evolution paths of previous generations is used. It has an analogous role to that of c_μ in the rank-μ update. Note that if $c_c = 1$ then no information is used from previous generations and if $c_c = 0$ no information from the new generation is incorporated into $\boldsymbol{p}_c^{(g+1)}$. The initial value of the evolution path $\boldsymbol{p}_c^{(0)}$ is taken as zero. It is recommended by Hansen [21] that c_c be computed by

$$c_c = \frac{4\mu_{\text{eff}}/n}{n + 4 + 2\mu_{\text{eff}}/n}. \tag{5.28}$$

The term $\sqrt{c_c(2 - c_c)\mu_{\text{eff}}}$ in Eq. 5.27 is used to normalize the variance of $\boldsymbol{p}_c^{(g+1)}$ so that $\boldsymbol{p}_c^{(g+1)} \sim \mathcal{N}\left(0, \boldsymbol{C}^{(g)}\right)$ [21]. To show this, it is necessary to interpret $\boldsymbol{m}^{(g+1)}$ as a sample from the mixture distribution shown in Eq. 5.10 and perform the following steps:

$$\begin{aligned}
\boldsymbol{m}^{(g+1)} &\sim \sum_{i=1}^{\mu} w_i \mathcal{N}\left(\boldsymbol{m}^{(g)}, {\sigma^{(g)}}^2 \boldsymbol{C}^{(g)}\right), \\
&\sim \mathcal{N}\left(w_1 \boldsymbol{m}^{(g)}, w_1^2 {\sigma^{(g)}}^2 \boldsymbol{C}^{(g)}\right) + \mathcal{N}\left(w_2 \boldsymbol{m}^{(g)}, w_2^2 {\sigma^{(g)}}^2 \boldsymbol{C}^{(g)}\right) \\
&\quad + ... + \mathcal{N}\left(w_\mu \boldsymbol{m}^{(g)}, w_\mu^2 {\sigma^{(g)}}^2 \boldsymbol{C}^{(g)}\right), \\
\boldsymbol{m}^{(g+1)} &\sim \mathcal{N}\left(\sum_{i=1}^{\mu} w_i \boldsymbol{m}^{(g)}, \sum_{i=1}^{\mu} w_i^2 {\sigma^{(g)}}^2 \boldsymbol{C}^{(g)}\right),
\end{aligned} \tag{5.29}$$

because $\sum_{i=1}^{\mu} w_i = 1$ and $\sum_{i=1}^{\mu} w_i^2 = 1/\mu_{\text{eff}}$, it follows from Eq. 5.29 that

$$\boldsymbol{m}^{(g+1)} \sim \mathcal{N}\left(\boldsymbol{m}^{(g)}, (1/\mu_{\text{eff}}) {\sigma^{(g)}}^2 \boldsymbol{C}^{(g)}\right). \tag{5.30}$$

Therefore,

$$\begin{aligned}
\boldsymbol{m}^{(g+1)} - \boldsymbol{m}^{(g)} &\sim \mathcal{N}\left(\boldsymbol{m}^{(g)} - \boldsymbol{m}^{(g)}, (1/\mu_{\text{eff}}) {\sigma^{(g)}}^2 \boldsymbol{C}^{(g)}\right), \\
&\sim \mathcal{N}\left(\boldsymbol{0}, (1/\mu_{\text{eff}}) {\sigma^{(g)}}^2 \boldsymbol{C}^{(g)}\right),
\end{aligned} \tag{5.31}$$

and

$$\begin{aligned}\frac{\sqrt{\mu_{\text{eff}}}}{\sigma^{(g)}}\left(\boldsymbol{m}^{(g+1)}-\boldsymbol{m}^{(g)}\right) &\sim \mathcal{N}\left(\mathbf{0}, \frac{\sqrt{\mu_{\text{eff}}}^2}{{\sigma^{(g)}}^2}(1/\mu_{\text{eff}}){\sigma^{(g)}}^2\boldsymbol{C}^{(g)}\right), \\ &\sim \mathcal{N}\left(\mathbf{0}, \boldsymbol{C}^{(g)}\right).\end{aligned} \tag{5.32}$$

The last step needed to show that $\boldsymbol{p}_c^{(g+1)} \sim \mathcal{N}\left(0, \boldsymbol{C}^{(g)}\right)$ would be to add $(1-c_c)\boldsymbol{p}_c^{(g)}$ and $\sqrt{c_c(2-c_c)}\sqrt{\mu_{\text{eff}}}\left(\frac{\boldsymbol{m}^{(g+1)}-\boldsymbol{m}^{(g)}}{\sigma^{(g)}}\right)$, as shown in Eq. 5.27, while assuming that $\boldsymbol{p}_c^{(g)} \sim \mathcal{N}\left(0, \boldsymbol{C}^{(g)}\right)$; the result is

$$\begin{aligned}(1-c_c)\boldsymbol{p}_c^{(g)} + \sqrt{c_c(2-c_c)}\sqrt{\mu_{\text{eff}}}\left(\frac{\boldsymbol{m}^{(g+1)}-\boldsymbol{m}^{(g)}}{\sigma^{(g)}}\right) &\sim \mathcal{N}\left(\mathbf{0}, (1-c_c)^2\boldsymbol{C}^{(g)}\right) \\ &+ \mathcal{N}\left(\mathbf{0}, c_c(2-c_c)\boldsymbol{C}^{(g)}\right),\end{aligned} \tag{5.33}$$

so

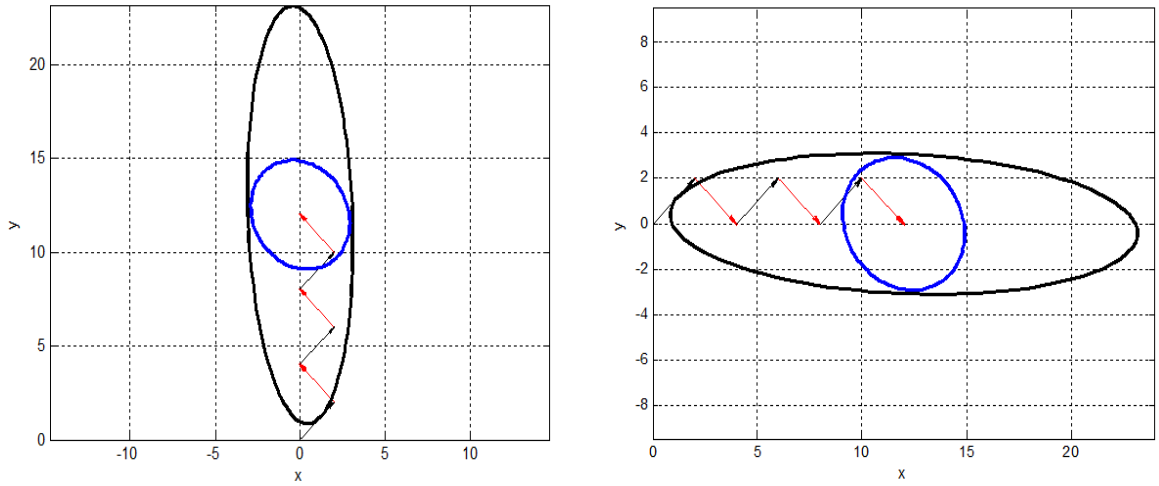
$$(1-c_c)\boldsymbol{p}_c^{(g)} + \sqrt{c_c(2-c_c)\mu_{\text{eff}}}\left(\frac{\boldsymbol{m}^{(g+1)}-\boldsymbol{m}^{(g)}}{\sigma^{(g)}}\right) \sim \mathcal{N}\left(\mathbf{0}, \boldsymbol{C}^{(g)}\right). \tag{5.34}$$

From Eqs. 5.33 and 5.34 it follows that

$$\boldsymbol{p}_c^{(g+1)} \sim \mathcal{N}\left(\mathbf{0}, \boldsymbol{C}^{(g)}\right). \tag{5.35}$$

The result shown in Eq. 5.35 can be used to show that the rank-one update is an unbiased update by following a similar to procedure the one used to show that the rank-μ update is unbiased (in Appendix A).

The evolution path is designed to use the information of the previous generations effectively by approximating the trend followed by the mean. It is easy to show this by an example. Consider a hypothetical rank-one update (given in Eq. 5.25) in which the rank-one matrix $\left(\frac{\boldsymbol{m}^{(g+1)}-\boldsymbol{m}^{(g)}}{\sigma^{(g)}}\right)\left(\frac{\boldsymbol{m}^{(g+1)}-\boldsymbol{m}^{(g)}}{\sigma^{(g)}}\right)^T$ is used instead of $\boldsymbol{p}_c^{(g+1)}\boldsymbol{p}_c^{(g+1)^T}$, which does not seem like a bad idea since the information gained from the step taken by the mean from generation g to generation $g+1$, $\boldsymbol{m}^{(g+1)}-\boldsymbol{m}^{(g)}$, is still being added to the information obtained from steps taken by the mean in previous generations. The update would actually be very ineffective if



(a) Rank-one update used on vertical zigzag path followed by the center of the distribution

(b) Rank-one update used on horizontal zigzag path followed by the center of the distribution

Figure 5.2: Rank-one update of the Covariance matrix of the MVND using two different rank-one matrices. The black ellipsoids represent the iso-density contours of the distribution with a covariance matrix adapted with $\left(\boldsymbol{p}_c^{(g+1)}\right)\left(\boldsymbol{p}_c^{(g+1)}\right)^T$ while the blue ellipsoids represent the iso-density contours of the distribution with a covariance matrix adapted with $\left(\frac{\boldsymbol{m}^{(g+1)}-\boldsymbol{m}^{(g)}}{\sigma^{(g)}}\right)\left(\frac{\boldsymbol{m}^{(g+1)}-\boldsymbol{m}^{(g)}}{\sigma^{(g)}}\right)^T$.

the aforementioned measure is taken because when the outer product is computed the sign information of the vector $\boldsymbol{m}^{(g+1)}-\boldsymbol{m}^{(g)}$ is lost as $\boldsymbol{y}\boldsymbol{y}^T=(-\boldsymbol{y})\left(-\boldsymbol{y}^T\right)$ [21]. This hypothetical update is merely the weighted sum of rank-one matrices generated by the self outer product of the individual steps taking by the mean at each generation. Therefore, if the mean is following a clear trend but it does so by taking zigzagging steps, then the successive line distributions would be almost perpendicular to each other and the resulting covariance matrix would not adapt to the trend. The rank-one update, given by Eq. 5.25, avoids this problem by first calculating a vector, $\boldsymbol{p}_c^{(g+1)}$, that points in the general direction that the mean has followed in recent generations (the evolution path), and then computes its self outer product to generate a line distribution that is aligned with this vector. In Fig. 5.2 the rank-one update is illustrated for two cases where the mean has followed two different zigzagging paths. If the update is done by using $\boldsymbol{C}^{(g+1)}=\boldsymbol{C}^{(g)}+\boldsymbol{p}_c^{(g+1)}\boldsymbol{p}_c^{(g+1)^T}$ both covariance matrices are adapted successfully and the direction of the highest variance is aligned with the general trend followed by the mean. However, when the rank-one matrix $\left(\frac{\boldsymbol{m}^{(g+1)}-\boldsymbol{m}^{(g)}}{\sigma^{(g)}}\right)\left(\frac{\boldsymbol{m}^{(g+1)}-\boldsymbol{m}^{(g)}}{\sigma^{(g)}}\right)^T$ is used for

the rank-one update, the two final covariance matrices have the same shape and alignment even though the trends followed by their means are perpendicular.

5.4 Step Size Adaptation

The adaptation of the covariance matrix through the use of Eq. 5.11 does not guarantee that the distribution will achieve an optimal scale (overall standard deviation) in a reasonable number of generations (relative to the general evolution strategy). For this reason a more explicit way of controlling the scale of the distribution is required and this is achieved separately through the step size adaptation. To control the step size, $\sigma^{(g+1)}$, an evolution path called the conjugate evolution path, $\boldsymbol{p}_\sigma$, is used. The updating processes for the conjugate evolution path and the step size are respectively given by

$$\boldsymbol{p}_\sigma^{(g+1)} = (1 - c_\sigma)\boldsymbol{p}_\sigma^{(g)} + \sqrt{c_\sigma(2 - c_\sigma)\mu_{\text{eff}}}\boldsymbol{C}^{(g)^{-\frac{1}{2}}}\left(\frac{\boldsymbol{m}^{(g+1)} - \boldsymbol{m}^{(g)}}{\sigma^{(g)}}\right), \tag{5.36}$$

and

$$\sigma^{(g+1)} = \sigma^{(g)} \exp\left(\frac{c_\sigma}{d_\sigma}\left(\frac{\|\boldsymbol{p}_\sigma^{(g+1)}\|}{\mathrm{E}[\|\boldsymbol{z}\|]} - 1\right)\right), \tag{5.37}$$

where c_σ is the learning rate for the conjugate evolution path, d_σ is the damping parameter for the step size variation between successive generations and $\boldsymbol{z}$ is a random variable with distribution $\mathcal{N}(0, \boldsymbol{I})$. The initial conjugate evolution path, $\boldsymbol{p}_\sigma^{(0)}$, is taken as zero while the initial step size, $\sigma^{(0)}$, can be chosen by the user according to the problem; however, Hansen [21] suggests, if it is suspected that the solution lies within the hypercube $[a, b]^n$,

$$\sigma^{(0)} = 0.3(b - a). \tag{5.38}$$

The learning rate of the conjugate evolution path, c_σ, and the damping coefficient for

the step size variation, d_σ, are

$$c_\sigma = \frac{\mu_{\text{eff}} + 2}{n + \mu_{\text{eff}} + 5} \tag{5.39}$$

and

$$d_\sigma = 1 + 2 \max\left(0, \sqrt{\frac{\mu_{\text{eff}} - 1}{n+1}} - 1\right) + c_\sigma \tag{5.40}$$

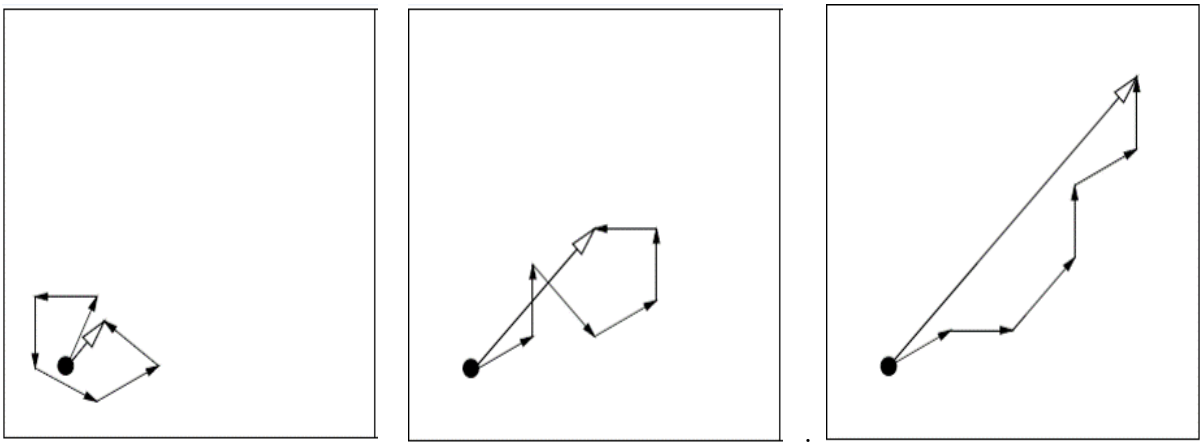
by [21].

In Eq. 5.36, the step taken by the mean from generation g to generation $g+1$, $\boldsymbol{m}^{(g+1)} - \boldsymbol{m}^{(g)}$, is normalized by the factor $\frac{\sqrt{\mu_{\text{eff}}}\boldsymbol{C}^{(g)^{-\frac{1}{2}}}}{\sigma^{(g)}}$ so that $\sqrt{\mu_{\text{eff}}}\boldsymbol{C}^{(g)^{-\frac{1}{2}}}\left(\frac{\boldsymbol{m}^{(g+1)} - \boldsymbol{m}^{(g)}}{\sigma^{(g)}}\right) \sim \mathcal{N}(0, \mathrm{I})$ [21]. This can be seen by removing the effect of $\boldsymbol{C}^{(g)}$ from the distribution shown in Eq. 5.32 as follows:

$$\begin{aligned}
\frac{\sqrt{\mu_{\text{eff}}}}{\sigma^{(g)}}\left(\boldsymbol{m}^{(g+1)} - \boldsymbol{m}^{(g)}\right)\boldsymbol{C}^{(g)^{-\frac{1}{2}}} &\sim \boldsymbol{C}^{(g)^{-\frac{1}{2}}}\mathcal{N}\left(\boldsymbol{0}, \boldsymbol{C}^{(g)}\right), \\
&\sim \mathcal{N}\left(\boldsymbol{0}, \boldsymbol{C}^{(g)^{-\frac{1}{2}}}\boldsymbol{C}^{(g)^{\frac{1}{2}}}\left(\boldsymbol{C}^{(g)^{\frac{1}{2}}}\right)^T\left(\boldsymbol{C}^{(g)^{-\frac{1}{2}}}\right)^T\right), \\
\frac{\sqrt{\mu_{\text{eff}}}}{\sigma^{(g)}}\left(\boldsymbol{m}^{(g+1)} - \boldsymbol{m}^{(g)}\right)\boldsymbol{C}^{(g)^{-\frac{1}{2}}} &\sim \mathcal{N}\left(\boldsymbol{0}, \boldsymbol{I}\right).
\end{aligned} \tag{5.41}$$

Removing the effect of $\boldsymbol{C}^{(g)}$ is important because as it was mentioned previously the optimal adaptations of σ and $\boldsymbol{C}^{(g)}$ occur in different time scales (in terms of iterations of the algorithm). By assuming that $\boldsymbol{p}_\sigma^{(g)} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I})$ and following the same steps seen in Eqs. 5.33 and 5.34, it can be shown, that using Eq. 5.36 to add the distributions of $\boldsymbol{p}_\sigma^{(g)}$ and $\sqrt{\mu_{\text{eff}}}\boldsymbol{C}^{(g)^{-\frac{1}{2}}}\left(\frac{\boldsymbol{m}^{(g+1)} - \boldsymbol{m}^{(g)}}{\sigma^{(g)}}\right)$ results in $\boldsymbol{p}_\sigma^{(g+1)} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I})$. The fact that $\boldsymbol{p}_\sigma^{(g+1)} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I})$ allows the length of the evolution path to be compared to its expected length under random selection (random selection means that the index $i : \lambda$ in Eq. 5.6 is independent of the fitness value of $\boldsymbol{v}_{i:\lambda}^{(g+1)} \, \forall \, i = 1, ..., \lambda$) which is needed for the adaptation of the step size. According to Eq. 5.37, the step size is increased if the actual length of the conjugate evolution path ($\|\boldsymbol{p}_\sigma^{(g+1)}\|$) is bigger than its expected length under random selection ($\mathrm{E}[\|\boldsymbol{z}\|]$) and is decreased if the opposite is true [21]. The reasoning behind this rule is that when

the successive steps taken by the mean of the distribution (where the step from generation g to generation $g+1$ is given by $\boldsymbol{m}^{(g+1)} - \boldsymbol{m}^{(g)}$) are positively correlated (meaning their dot products are bigger than zero), their combined effect is to increase the length of the evolution path (becomes bigger than its expected length) therefore the step size is increased in an attempt to cover a similar distance with fewer steps. In the case where the steps are anti-correlated (dot product is smaller than zero), their effect is to cancel each other out which makes the actual conjugate evolution path shorter than its expected length; therefore the step size is decreased as the same distance could be covered with a smaller step size. The logic behind the control of the step size is better understood by looking at the three scenarios shown in Fig. 5.3. The expected length of the conjugate evolution path is equivalent to the



(a) Anti-correlated steps
$\|\boldsymbol{p}_\sigma^{(g+1)}\| < \mathrm{E}[\|\boldsymbol{z}\|]$

(b) Uncorrelated steps
$\|\boldsymbol{p}_\sigma^{(g+1)}\| \approx \mathrm{E}[\|\boldsymbol{z}\|]$

(c) Correlated steps
$\|\boldsymbol{p}_\sigma^{(g+1)}\| > \mathrm{E}[\|\boldsymbol{z}\|]$

Figure 5.3: Conjugate evolution paths with the same number of steps. The lengths of the steps are normalized so that they are comparable. Depending on the correlation (or lack of) of the steps the step size will shrink, expand or remain unchanged. [21]

expected value of the positive square root of a random variable $\boldsymbol{x}$ that has a χ^2_n distribution with n degrees of freedom [3]. Let $\boldsymbol{z} \sim \mathcal{N}(0, \boldsymbol{I})$. The length of $\boldsymbol{z}$,

$$\|\boldsymbol{z}\| = \sqrt{\sum_{i=1}^{n} z_i^2}; \tag{5.42}$$

then, from [12],

$$\|\boldsymbol{z}\| \sim \chi_n \tag{5.43}$$

and the expected value of a variable with a χ_n distribution, which can be found in [12], is

$$\mathrm{E}\left[\|\boldsymbol{z}\|\right] = \sqrt{2}\frac{\Gamma\left(\frac{n+1}{2}\right)}{\Gamma\left(\frac{n}{2}\right)}. \tag{5.44}$$

However, the following approximation to Eq. 5.44 is suggested by [21] for a more practical application in the algorithm:

$$\mathrm{E}[\|\boldsymbol{z}\|] \approx \sqrt{n}\left(1 - \frac{1}{4n} + \frac{1}{21n^2}\right). \tag{5.45}$$

The update of the step size satisfies the unbiasedness requirement discussed in the previous sections which is an important condition if the algorithm is to avoid issues such as premature convergence. The proof is shown in Appendix A.

5.5 Practical considerations and tuning of learning rates

To apply the CMA-ES optimization algorithm, the optimization variables corresponding to the well trajectories (components of vector $\boldsymbol{T}_w$) are normalized because these optimization variables have different orders of magnitude. In this work, the lower bounds of the normalized domains of the trajectory parameters are defined by

$$\widetilde{\boldsymbol{T}}_{w,i}^{low} = 0, \qquad i = x_w, y_w, z_w, l_w, \theta_w, \phi_w. \tag{5.46}$$

Choosing the same lower bound for all the domains of the trajectory parameters means that the only parameters that need to be adjusted in order to achieve similar orders of magnitudes amongst the trajectory parameters are the upper bounds of the normalized domains. The choice of zero as the lower bound of the normalized domains was done so that all the normalized variables are always positive which is a standard practice when it comes

to the normalization of variables.

The upper bounds of the x_w, y_w, and z_w trajectory parameters in the normalized domain are defined as follows:

$$\widetilde{\boldsymbol{T}}^{up}_{w,i} = \frac{\boldsymbol{T}^{up}_{w,i} - \boldsymbol{T}^{low}_{w,i}}{D_i}, \quad i = x_w, y_w, z_w, \tag{5.47}$$

where D_{x_w}, D_{y_w}, and D_{z_w} represent the average size of the gridblocks in the x_w, y_w, and z_w directions, respectively. In the case where the gridblocks are uniform then D_{x_w}, D_{y_w}, and D_{z_w} are simply the dimensions of the gridblocks in the x_w, y_w, and z_w directions, respectively. Based on the upper bound values of the x_w, y_w, and z_w in the normalized domain an appropriate value was chosen for the $\widetilde{\boldsymbol{T}}^{up}_{w,i}$ corresponding to the rest of the well trajectory parameters (l_w, θ_w, and ϕ_w) so that all the normalized parameters have the same order of magnitude. For all the test cases pertaining the PUNQ and egg reservoir models the chosen values for the $\widetilde{\boldsymbol{T}}^{up}_{w,i}$'s corresponding to the l_w, θ_w, and ϕ_w variables were all equal to 5. It was considered a plausible choice based on the values of $\widetilde{\boldsymbol{T}}^{up}_{w,z_w}$ (corresponding to z_w) for the PUNQ and the egg reservoir models which were 5 and 7 respectively. The lowest upper bound amongst the normalized domains corresponding to the x_w, y_w, and z_w parameters was the one corresponding to z_w for both the PUNQ and egg reservoir models. Since the parameters l_w, θ_w, and ϕ_w originally have smaller domains compared to those of the x_w and y_w parameters for both the PUNQ and egg reservoir models choosing an upper bound that honored that relationship seemed appropriate. The normalization of the trajectory parameters is defined by

$$\widetilde{\boldsymbol{T}}_{w,i} = \frac{\boldsymbol{T}_{w,i} - \boldsymbol{T}^{low}_{w,i}}{\boldsymbol{T}^{up}_{w,i} - \boldsymbol{T}^{low}_{w,i}} \left(\widetilde{\boldsymbol{T}}^{up}_{w,i} - \widetilde{\boldsymbol{T}}^{low}_{w,i} \right), \quad i = x_w, y_w, z_w, l_w, \theta_w, \phi_w, \tag{5.48}$$

where $\widetilde{\boldsymbol{T}}_{w,i}$ is the normalized parameter.

The initial step size for the algorithm is set to $\sigma^0 = 1$. The bound constraints in the normalized domain are enforced using the truncation approach, i.e., the value of the

normalized optimization variable is truncated to have a value within its bounds at every optimization iteration. Note that this normalization of the bounds is not used for the well control parameters where their values are constrained to the range $[-6, 6]$ (see the well control parametrization section).

In this work, the learning rates of the CMA-ES algorithm (c_1 and c_μ) are tuned so that a reasonable level of learning is achieved by the covariance matrix within the total number of simulations performed. The total default learning rate of the covariance matrix (the sum $c_1 + c_\mu$) obtained when Eq. 5.26 and Eq. 5.24 are applied to the joint optimization problem defined in this thesis is around 1×10^{-3} ($c_\mu \approx 8 \times 10^{-4}$ and $c_1 \approx 2 \times 10^{-4}$). Based on [21], this effectively means that after the first $1/(1 \times 10^{-3})$ generations, around 37% of the new covariance matrix will be composed of the initial matrix (i.e., the identity matrix for the default CMA case) which could be considered as a very slow change. Therefore, a study was done on the performance of CMA-ES on the minimization of the Rosenbrock function (a popular benchmarking function used to test optimization algorithms first presented by [48]) to observe the effect of the total learning rate on the performance of the CMA-ES algorithm for a fixed number of function evaluations. The Rosenbrock function used for the tests is given by

$$f(\boldsymbol{v}) = \sum_{i=1}^{N-1} \left[(1 - \boldsymbol{v}_i)^2 + 100 \left(\boldsymbol{v}_{i+1} - \boldsymbol{v}_i^2 \right)^2 \right] \tag{5.49}$$

where N is the dimension of the problem. The global minimum of the Rosenbrock function described here is located at $\boldsymbol{v} = [1, 1, ..., 1]^T$ and has an objective function value of 0. Note that the dimension of the Rosenbrock function used for the tests was the same as the dimension of the first joint optimization problem presented in the computational results section (i.e., $n = 90$), where the CMA-ES algorithm requires 17 simulation runs per generation.

In the series of tests performed, different total learning rates were tested as seen in Figs. 5.4 and 5.5. For each of the total learning rates a set of 40 different optimization runs were performed with each of these optimization runs having an initial guess different from the other 39. The minimum values for each of these 40 optimization runs were recorded and

averaged after a certain number of function evaluations. The results obtained after 5,000 and 100,000 function evaluations are shown and discussed. The algorithm was terminated early if the condition number of the covariance matrix was bigger than 1×10^{10}, which is an indication of a degenerate matrix. Fig. 5.4 shows the values obtained after averaging

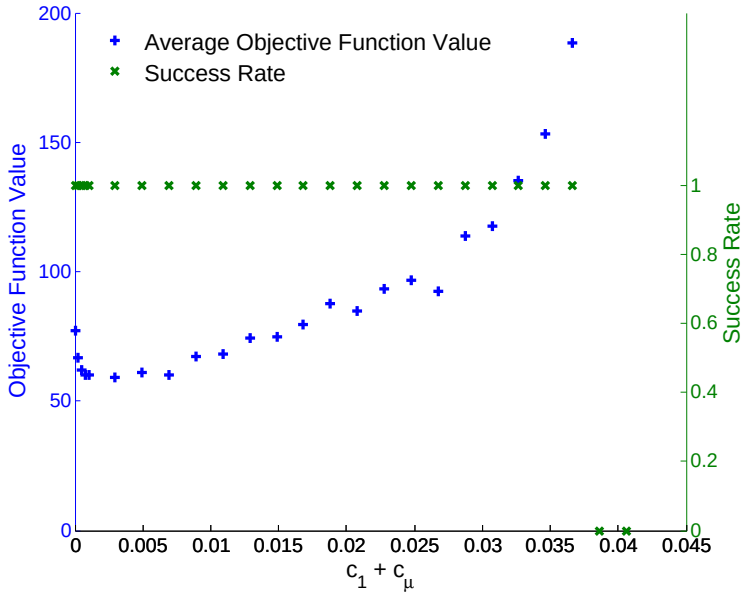


Figure 5.4: Average objective function value for different total learning rates at 100,000 function evaluations in high dimensional Rosenbrock function.

the minimum objective function values corresponding to the 40 different initial guesses after 100,000 function evaluations (blue crosses). It can be seen that a high total learning rate results in a high average objective function value mainly because the covariance matrix starts to degenerate. This means that the normal deviates, generated in the mutation step of the algorithm by perturbing the mean of the distribution, will be constrained to the subspace spanned by the eigenvectors corresponding to the biggest eigenvalues of the covariance matrix; see Eq. 5.1. On the other hand, when the learning rates are very close to zero, almost no learning occurs. Therefore, the algorithm takes little or no advantage of the information gained through the iterations, which could explain the suboptimality of these points. The success rate shown in the figure measures the fraction of cases that were not terminated early (i.e., before reaching the stipulated number of function evaluations which in Fig. 5.4 is 100,000) due to the high condition number of the covariance matrix (covariance matrix

degeneration). For the last two cases (total learning rates of 0.038 and 0.041), the algorithm is terminated early due to covariance matrix degeneration for all 40 cases. Therefore, it is concluded that those learning rates were too high for this problem. Note that the lowest function value is obtained at a total learning rate of about 3×10^{-3}.

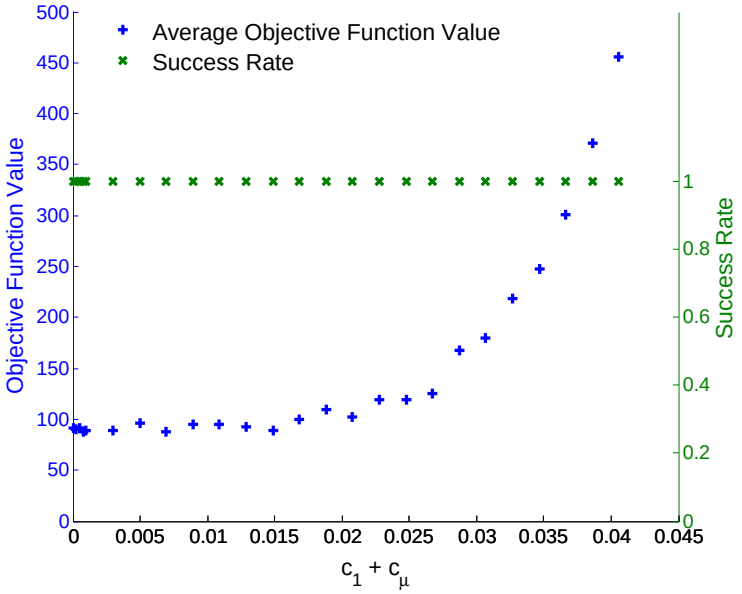


Figure 5.5: Average objective function Value for different total learning rates at 5,000 function evaluations in high dimensional Rosenbrock function.

For the case of 5,000 function evaluations, presented in Fig. 5.5, the trend seems to be very similar to that shown in Fig. 5.4 except in this case the average function value remains relatively low for a longer range of total learning rate values. The reason for having a long range of total learning rates with similar average function value in this case could be due to the fact that at 5,000 function evaluations moderately high values of the total learning rate have not been given enough time (iterations) to degenerate the covariance matrix. Therefore, one could be bold in the choice of the total learning rate when the number of function evaluation is low in the hope that the covariance matrix incorporates new information faster without degenerating during the stipulated maximum number of function evaluations. In Fig. 5.4 the average function value starts to increase after a total learning rate value of about 0.015. After this point the increase in the function value is almost monotonic as the total learning rate is incremented. Based on this result and the fact that the joint optimization

algorithm would have a similar maximum number of simulations (perhaps even lower), the total learning rate was set to 0.01 ($c_\mu \approx 8 \times 10^{-3}$ and $c_1 \approx 2 \times 10^{-3}$) in our implementation of the CMA-ES algorithm. This total learning rate is about ten times the default learning rate for the joint optimization problem, which would guarantee that considerable learning occurs after a few hundred optimization iterations without the danger of degenerating the covariance matrix. It is also important to highlight that when comparing the minimum values obtained at $5,000$ and $100,000$ simulation runs there does not seem to be a great improvement in the objective function value (30% decrease for $95,000$ more runs), therefore it may not be worth the computational expense to run the CMA-ES algorithm for too long on this problem.

The last point to discuss is the stopping criteria used in the algorithm. For the optimization runs performed in the problems described below, the following three different stopping criteria were used. The first criterion is the relative error of the mean:

$$\frac{\|\boldsymbol{m}^{(g+1)} - \boldsymbol{m}^{(g)}\|}{\|\boldsymbol{m}^{(g)}\|} \leq \epsilon, \tag{5.50}$$

for which a tolerance, ϵ, value of 0.01 (1%) was used.

The second criterion uses the step size, σ, to check if the algorithm has reached a zone where no further improvement is taking place. The value is dependent on the scale of the problem but for this problem a value of $\sigma = 0.01$ can be used.

The third criterion is given by the maximum number of function evaluations (simulation runs). This las criterion is in fact the criterion that usually stops the algorithm so special care needs to be taken when setting this number as it will most likely be the cause for the termination of the optimization run. Different numbers were used for the limit of this criterion as it was set by taking into consideration what would be a reasonable number of simulation runs for a particular optimization problem.

It should be mentioned that another stopping criterion was enforced in all the optimization runs, specifically, if the condition number of the covariance matrix becomes greater

than 1×10^6, it is assumed that the covariance matrix has degenerated. If the algorithm stops due to exceeding the tolerance (1×10^6) the optimization run is stopped and a new optimization run is performed starting with the same initial guess but using a different initial seed for the random number generator. It should be mentioned that the tolerance used was never violated by the tests performed in this work.

A summary of the CMA-ES algorithm applied in this thesis is given in Algorithm 2.

5.6 Handling Linear Inequality Constraints by sampling from a truncated MVND

The linear inequality constraints considered in this work are related to the physical bounds of the egg reservoir model (described further below) as vertical planes are used to separate the active gridblocks from the inactive ones in the optimization algorithm. The feasible region defined by these constraints create a convex domain therefore truncating the MVND to this feasible region seemed like a plausible choice. To sample from this truncated MVND an efficient Gibbs sampler, first described by Rodriguez-Yam et al. in [47], was used. Credit should also be given to Muller [31] who used this method along with CMA-ES to solve some linearly constrained benchmark problems. This method has advantages over other popular ones such as penalty functions and rejection sampling when applied to problems with convex domains. In the case of penalty functions the penalty parameter has to be tuned to the problem and this could prove difficult. In the case of rejection sampling, when infeasible points have a high probability of being generated, too many samples are often rejected [31]. Sampling from the truncated MVND guarantees that all the samples are generated within the feasible region and no tuning of parameters is required.

To understand how to sample from a truncated MVND it is necessary to explain how to sample from a random variable that has a truncated one dimensional gaussian distribution. Given a random variable X, where

$$X \sim \mathcal{N}_\tau(m, \sigma) \qquad \tau = \{x \in \mathbb{R} : g \leq x \leq h\}, \tag{5.51}$$

Algorithm 2 Covariance Matrix Adaptation-Evolution Strategy algorithm

Set parameters

- Set variables used in the covariance matrix adaptation and step size adaptation: $c_c, c_\mu, c_1, c_\sigma, d_\sigma$.
- Set the population size λ, the offspring size μ and determine the weights $w_{i=1...\mu}$.
- Set tolerances for the stopping criteria: $\epsilon_1, \epsilon_2, \epsilon_3$ and γ.

Initialization

- Set evolution paths $\boldsymbol{p}_\sigma^{(0)} = \mathbf{0}$, $\boldsymbol{p}_c^{(0)} = \mathbf{0}$.
- Set initial covariance matrix: $\boldsymbol{C}^{(0)} = \boldsymbol{I}$.
- Chose the initial mean and step size of the MVND, i.e. $\boldsymbol{m}^{(0)}$ and $\sigma^{(0)}$.

For generation $g = 0, 1, 2, ...$

- Sample new population, for $k = 1, ..., \lambda$
  - $\boldsymbol{v}_k^{(g+1)} = \boldsymbol{m}^{(g)} + \sigma^g \mathcal{N}\left(\mathbf{0}, \boldsymbol{C}^{(g)}\right)$ 5.1
- Update the mean
  - $\boldsymbol{m}^{(g+1)} = \sum_{i=1}^{\mu} w_i \boldsymbol{v}_{i:\lambda}^{(g+1)}$ 5.6
- Adapt the Covariance matrix of the distribution
  - $\boldsymbol{C}_\mu^{(g+1)} = \sum_{i=1}^{\mu} w_i \left(\boldsymbol{v}_{i:\lambda}^{(g+1)} - \boldsymbol{m}^{(g)}\right)\left(\boldsymbol{v}_{i:\lambda}^{(g+1)} - \boldsymbol{m}^{(g)}\right)^T$ 5.13
  - $\boldsymbol{p}_c^{(g+1)} = (1 - c_c)\boldsymbol{p}_c^{(g)} + \sqrt{c_c(2 - c_c)\mu_{\text{eff}}}\left(\frac{\boldsymbol{m}^{(g+1)} - \boldsymbol{m}^{(g)}}{\sigma^{(g)}}\right)$ 5.27
  - $\boldsymbol{C}^{(g+1)} = (1 - c_1 - c_\mu)\boldsymbol{C}^{(g)} + c_1 \boldsymbol{p}_c^{(g+1)} \boldsymbol{p}_c^{(g+1)^T} + c_\mu \frac{\boldsymbol{C}_\mu^{(g+1)}}{\sigma^{(g)^2}}$ 5.11
- Adapt the step size of the distribution
  - $\boldsymbol{p}_\sigma^{(g+1)} = (1 - c_\sigma)\boldsymbol{p}_\sigma^{(g)} + \sqrt{c_\sigma(2 - c_\sigma)\mu_{\text{eff}}}\boldsymbol{C}^{(g)^{-\frac{1}{2}}}\left(\frac{\boldsymbol{m}^{(g+1)} - \boldsymbol{m}^{(g)}}{\sigma^{(g)}}\right)$ 5.36
  - $\sigma^{(g+1)} = \sigma^{(g)} \exp\left(\frac{c_\sigma}{d_\sigma}\left(\frac{\|\boldsymbol{p}_\sigma^{(g+1)}\|}{\mathrm{E}[\|\boldsymbol{z}\|]} - 1\right)\right)$ 5.37

Break if one or multiple stopping criteria are met

- $\frac{\|\boldsymbol{m}^{(g+1)} - \boldsymbol{m}^{(g)}\|}{\|\boldsymbol{m}^{(g)}\|} \leq \epsilon_1$
- $\sigma < \epsilon_2$
- $\kappa\left(\boldsymbol{C}^{(g+1)}\right) > \epsilon_3$
- $g > \gamma$

it is possible to generate a realization of X, x, by taking the following steps. First, the bounds

of the convex set τ are standardized by calculating their z scores:

$$g_{\text{std}} = \frac{(g - m)}{\sigma}, \tag{5.52}$$

$$h_{\text{std}} = \frac{(h - m)}{\sigma}, \tag{5.53}$$

then the cumulative probability density function (CDF) values of g_{std} and h_{std}, $\Phi(g_{\text{std}})$ and $\Phi(h_{\text{std}})$, are computed by using the CDF of the untruncated standard gaussian distribution $\mathcal{N}(0, 1)$. Note that $\Phi(.)$ represents the CDF of $\mathcal{N}(0, 1)$. After this step, a number is drawn randomly from the uniform distribution defined on the interval $(\Phi(g_{\text{std}}), \Phi(h_{\text{std}}))$ as follows

$$r = \Phi(g_{\text{std}}) + U(\Phi(g_{\text{std}}) - \Phi(h_{\text{std}})), \tag{5.54}$$

where U is a number drawn randomly from the uniform distribution defined on $(0, 1)$. Then the value in the domain of the distribution $\mathcal{N}(0, 1)$ corresponding to r is computed by using the inverse CDF of $\mathcal{N}(0, 1)$. Finally, the value corresponding to r in the domain of $\mathcal{N}(0, 1)$, $\Phi^{-1}(r)$, is transformed to the domain of the distribution $\mathcal{N}_{\tau}(m, \sigma)$ by reversing the z-score transformation

$$x = \Phi^{-1}(r)\,\sigma + m. \tag{5.55}$$

In order to understand how the algorithm described in [47] works the Gibbs sampling algorithm needs to be explained. Gibbs sampling (first presented by Geman and Geman [19]) is a Markov Chain Monte Carlo method used to sample from multidimensional joint distributions when the conditional distributions are known and easy to sample from. The algorithm starts by picking a feasible point in the domain. The point is then used to generate a new sample by sampling only in one of the dimensions of the joint distribution while keeping the values of the original point in the other dimensions fixed. To sample in the selected dimension, the

conditional probability of this dimension given the fixed values in the other dimensions is used. This results in a new point (sample) that differs from the previous one in only one of the dimensions. To generate the next sample a new dimension is picked from the latest generated point and sampling in this newly-selected dimension is performed. The algorithm cycles through the dimensions as many times as samples are required. The samples form a Markov chain whose stationary distribution is that of the joint probability distribution [31]. Due to the fact that the last generated sample (point) is used to get a new one there is a small degree of correlation between them. To "thin" out this effect it is common practice to take only every n^{th} generated sample (point), with n being the number of dimensions of the multivariate distribution. A summary of the algorithm is shown in Algorithm 3.

The Gibbs sampling algorithm facilitates sampling from a truncated MVND by reducing the

Algorithm 3 Gibbs Sampling

Initialize

- Choose an initial point, $\boldsymbol{x}^0 = (x_1^0, x_2^0, ..., x_n^0)$.

For $i = 1, 2, ..., k$ **samples**

- Sample in every dimension using the conditional distribution of the dimension in question given fixed values in the other dimensions; i.e. obtain x_j^i from $p\left(x_j^i | x_1^i, ..., x_{j-1}^i, x_{j+1}^{i-1}, ..., x_n^{i-1}\right)$.
- Accept the vector $\boldsymbol{x}^i$ as a new sample once sampling has been done in all the dimensions (thinning).

problem to that of sampling from a single truncated one-dimensional Gaussian distribution. This is because the conditional one-dimensional distributions that represent the dimensions of a MVND (given fixed values for the other dimensions) are also normally distributed. Therefore, one can start by picking a feasible point (the mean for example) and then apply the Gibbs sampling algorithm, but in this case when each dimension is visited by the algorithm the bounds on each dimension are determined based on the coordinates of the current point and only then the sampling is performed by following the steps given in Eqs. 5.52, 5.53, 5.54 and 5.55. The sampling process can be simplified further using the method of Rodriguez-Yam et al. [47], who apply a linear transformation to the random vector $\boldsymbol{X}$ that

has a truncated normal distribution. The transformation is as follows:

Given the random vector $\boldsymbol{X}$, where

$$\boldsymbol{X} \sim \mathcal{N}_T\left(\boldsymbol{m}, \sigma^2 \boldsymbol{C}\right) \qquad T = \{\boldsymbol{x} \in \mathbb{R}^n : \boldsymbol{E}\boldsymbol{x} \leq \boldsymbol{b}\} \tag{5.56}$$

and T is a convex set. Given the linear mapping $\boldsymbol{A}$ with

$$\boldsymbol{A}\boldsymbol{C}\boldsymbol{A}^T = \boldsymbol{I} \tag{5.57}$$

it follows that

$$\boldsymbol{A}\boldsymbol{X} \sim \mathcal{N}_T\left(\boldsymbol{A}\boldsymbol{m}, \sigma^2 \boldsymbol{A}\boldsymbol{C}\boldsymbol{A}^T\right) \tag{5.58}$$

or

$$\boldsymbol{A}\boldsymbol{X} \sim \mathcal{N}_S\left(\boldsymbol{A}\boldsymbol{m}, \sigma^2 \boldsymbol{I}\right) \qquad S = \{\boldsymbol{A}\boldsymbol{x} : \boldsymbol{x} \in \mathbb{R}^n \text{ and } \boldsymbol{E}\boldsymbol{x} \leq \boldsymbol{b}\}, \tag{5.59}$$

where $\boldsymbol{E}$ is the $m \times n$ matrix that contains the coefficients of the linear inequality constraints and $\boldsymbol{b}$ is the m-dimensional vector containing the bounds of the constraints. The advantage of the transformation is that all the variances of the one-dimensional distributions of $\boldsymbol{A}\boldsymbol{X}$ are equal to σ^2. The matrix $\boldsymbol{A}$ is found by using the eigendecomposition of $\boldsymbol{C}$ (computed in the CMA-ES algorithm and is given by Eq. 5.4) which is given by $\boldsymbol{C} = \boldsymbol{B}\boldsymbol{D}\boldsymbol{B}^T$. Substituting $\boldsymbol{B}\boldsymbol{D}\boldsymbol{B}^T$ for $\boldsymbol{C}$ in Eq. 5.57 yields

$$\begin{aligned} \boldsymbol{A}\boldsymbol{B}\boldsymbol{D}^{\frac{1}{2}}\boldsymbol{D}^{\frac{1}{2}^T}\boldsymbol{B}^T\boldsymbol{A}^T &= \boldsymbol{I} \\ \boldsymbol{A}\boldsymbol{B}\boldsymbol{D}^{\frac{1}{2}} &= \boldsymbol{I} \\ \boldsymbol{A} &= \boldsymbol{D}^{-\frac{1}{2}}\boldsymbol{B}^T. \end{aligned} \tag{5.60}$$

To simplify the expressions in Eq. 5.59 the following definitions are made:

$$\boldsymbol{\alpha} = \boldsymbol{A}\boldsymbol{m}, \tag{5.61}$$

$$\boldsymbol{Z} = \boldsymbol{A}\boldsymbol{X}, \tag{5.62}$$

$$\boldsymbol{F} = \boldsymbol{E}\boldsymbol{A}^{-1}; \tag{5.63}$$

therefore,

$$S = \{\boldsymbol{z} \in \mathbb{R}^n : \boldsymbol{F}\boldsymbol{z} \leq \boldsymbol{b}\}. \tag{5.64}$$

With this notation the conditional distribution for the j^{th} dimension is given by

$$p(z_j|\boldsymbol{z}_{-j}) = \mathcal{N}_{S_j}\left(\alpha_j, \sigma^2\right), \tag{5.65}$$

where $\boldsymbol{z}_{-j} = [z_1, ..., z_{j-1}, z_j, ..., z_n]^T$. Let $\boldsymbol{f}_j$ denote the j^{th} column of the matrix $\boldsymbol{F} = \boldsymbol{E}\boldsymbol{A}^{-1}$. The feasible domain, S_j, is obtained by finding the solution to the following system of linear inequalities:

$$\boldsymbol{f}_j z_j \leq \boldsymbol{b} - \boldsymbol{F}_{-j}\boldsymbol{z}_{-j}, \tag{5.66}$$

where $\boldsymbol{F}_{-j} = [\boldsymbol{f}_1, ..., \boldsymbol{f}_{j-1}, \boldsymbol{f}_{j+1}, ..., \boldsymbol{f}_n]$ and $\boldsymbol{f}_j \in \mathbb{R}^m$. The solution to the set of inequalities given in Eq. 5.66 is a one-dimensional convex set. In order to obtain the solution to the set of linear inequalities given in Eq. 5.66 the algorithm first solves for z_j in each of these inequalities. In this manner, there will be a set of inequalities that will be in the form of upper bounds for z_j and another set of inequalities that will be in the form of lower bounds for z_j. By taking the minimum value in the set that contains the upper bounds and the maximum value in the set that contains the lower bounds as the bounds of S_j it is ensured that all the other upper and lower bounds are satisfied. In the event that none of the inequalities in Eq. 5.66 yields an upper bound then S_j is unbounded above. On the other hand if none of

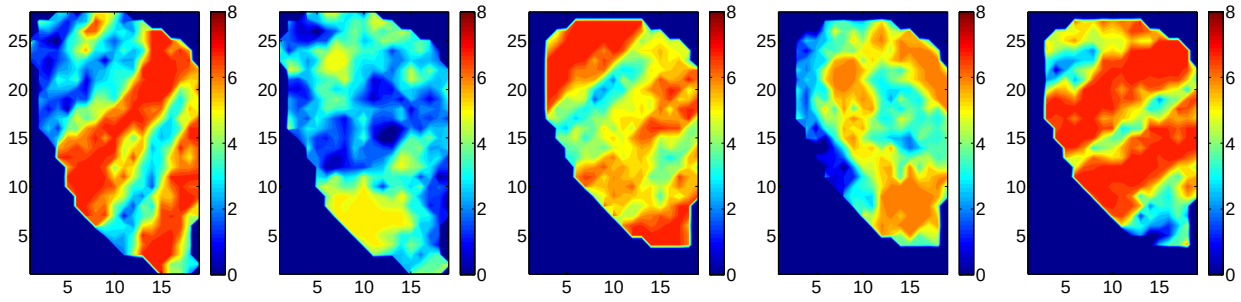
the inequalities yields a lower bound then S_j is unbounded below. After defining the bounds of S_j the next step is to sample from the one-dimensional truncated Gaussian distribution $\mathcal{N}_{S_j}(\alpha_j, \sigma^2)$.

CHAPTER 6

COMPUTATIONAL RESULTS

6.1 PUNQ reservoir model

In this section, the results for the application of the proposed algorithm to solve the joint optimization of well trajectories and controls for the PUNQ reservoir model are presented. PUNQ is a three phase three dimensional reservoir model and has a $19 \times 28 \times 5$ grid. The horizontal log-permeability distribution of reservoir simulation layers are shown in Fig. 6.1.



(a) Layer 1 (b) Layer 2 (c) Layer 3 (d) Layer 4 (e) Layer 5

Figure 6.1: Horizontal log-permeability distribution of reservoir simulation layers of PUNQ model.

The reservoir model is bounded to the east and north by a fault and initially there is a small gas cap located at the center of the dome-shaped structure. The initial reservoir pressure is 3400.4 psi at the datum depth of 7726.4 ft. The original PUNQ model has strong aquifers connected in the north and west edges of the reservoir. However, in our example these aquifers are removed with the purpose of including some injection wells in the reservoir. The joint optimization problem considered here is to optimize the trajectories and well controls for 6 production wells and 3 injection wells. The reservoir life is considered to

be 10,800 days. The controls for production wells are specified bottomhole pressures (BHP) at the different control steps with a maximum and minimum pressures of 3500 and 1500 psi, respectively. The water injection well controls are the water injection rates at the different control steps and the maximum and minimum water injection rates for an injection well at each control step are set to 6000 and 0 STB/day, respectively. The maximum length of a well is set to 500 ft. The values for economic parameters r_{o}, r_{w} and r_{winj} are set to 70, 10 and 10 \$/STB, respectively, and $b = 0.05$.

In order to investigate the performance of the proposed parametrization, three cases are considered for setting up the joint optimization problem. In Case 1, the reservoir life is partitioned into 4 equally-sized control steps of 2700 days. In this case, the proposed parametrization for the well controls (the controls of a well are not temporally correlated) is not used, instead, the well controls of each well are added to the set of optimization variables of each well, i.e., the number of optimization variables for each well is 10, where 6 of them pertain to the well trajectory parameters and the remaining 4 correspond to the values of the well control variables at the specified control steps. Thus, the total number of optimization (design) variables is 90. For the results of this case to be comparable with the results obtained with the cases that use parametrization, the well control variables are normalized to the range $[-6, 6]$; see discussion in section 3.

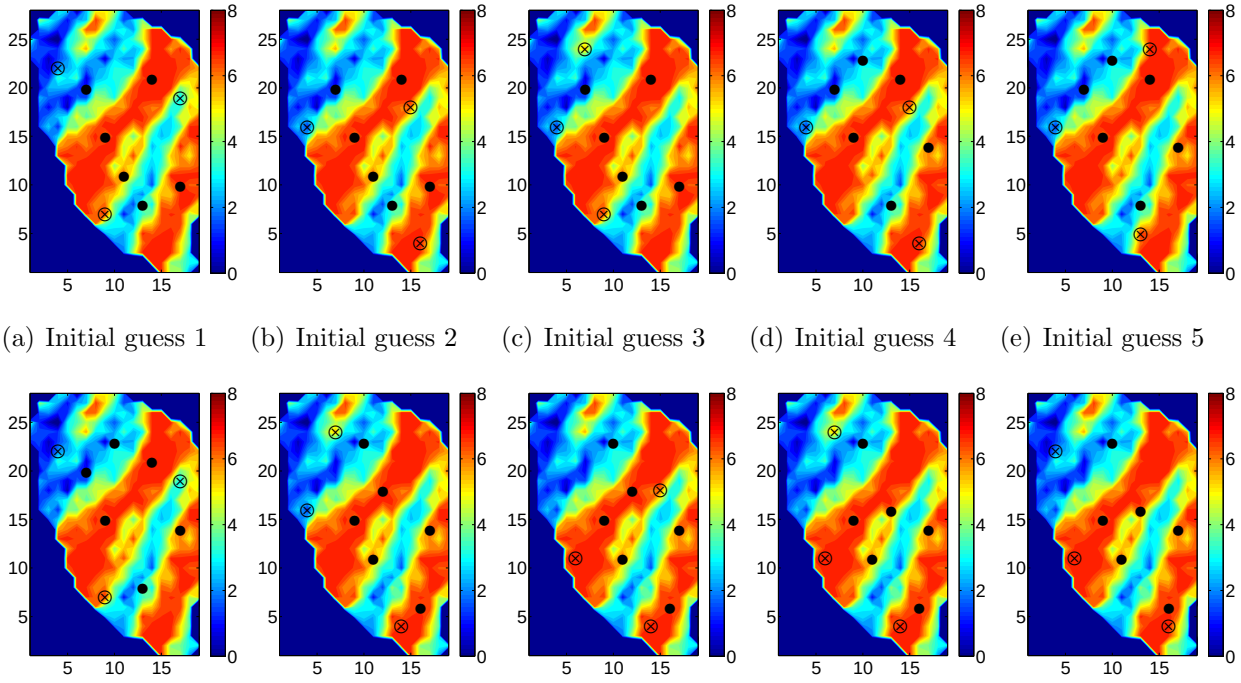
In Case 2, similar to Case 1, the reservoir life cycle is partitioned into 4 equally-sized control steps. In this case, Eq. 3.8 is used to parameterize the well controls, where in order to construct the covariance matrix $\boldsymbol{C}_U^w$, two sub-cases are defined as follows: Case 2a - $N_s = 3$ and Case 2b - $N_s = 2$, i.e., in Case 2a and Case 2b, the correlation lengths are 8100 and 5400 days, respectively. However, in Case 2 it is important to note that the full set of singular values of the SVD of the covariance matrix is used. The parametrization gives us a way to impose the temporal correlation on the well controls. Similar to Case 1, the number of optimization variables is 10 for each well.

In Case 3, the reservoir life cycle is partitioned into 100 control steps of 108 days, i.e., the dimension of the vector of control variables of each well is 100. The equation

given in Eq. 3.12 is used to parameterize the set of well controls in terms of 4 optimization parameters, i.e., the four largest singular values of the SVD of the covariance matrix are retained in Eq. 3.12, $n_{\mathrm{P}} = 4$. This will result in a same number of optimization variables per well as in Cases 1 and 2. Similar to Case 2, two sub-Cases are defined as follows: Case 3a: $N_s = 75$, i.e., the correlation length is 8100 days and Case 3b: $N_s = 50$, i.e., the correlation length is 5400 days.

In order to compare the results of the three Cases, the optimization run of each case and sub-case is run 10 times and the NPV results are averaged. The initial value for each well control is selected to be the average of its upper and lower bounds, at each control step, i.e., the initial injection rate of the injectors is set to 3000 STB/day and the initial bottomhole pressure of the producers is set equal to 2500 psi for all control steps. For each optimization run, the initial locations of the wells are selected based on engineering judgement, and are kept the same for all cases. The initial locations of the center points of the wells are shown in Fig. 6.2 for all 10 initial guesses. The solid black circles show the well center points of the producers and the circles with a cross show the initial well center points of the injectors. The locations in Fig. 6.2 are shown on the log permeability field of the simulation layer 1. Except for the spatial coordinate direction of the well center point, the rest of the parameters for the well trajectories are arbitrarily set to their mean values, i.e., the initial length of the wells are set to 250 ft and the initial orientation angles are set to $\theta_{w,\mathrm{init}} = \frac{\pi}{2}$ and $\phi_{w,\mathrm{init}} = \frac{\pi}{2}$.

The optimization runs are terminated after about 4012 simulation runs (236 iterations of the CMA algorithm for joint optimization problem). The average NPV value for all optimization runs is shown in Fig. 6.3 and the summary of optimized NPV at 2000 and 4012 simulation runs is presented in Table 6.1. As an example, the plot of NPV value versus the number of the simulation runs for Case 2b and Case 3b are shown in Fig. 6.4. Comparing the NPV values for the optimization runs corresponding to Case 2 and Case 3 with the optimization run for Case 1 (Fig. 6.3) shows that the joint optimization runs that implemented the proposed parametrization for the well controls obtained higher NPV values than the optimization run without well control parametrization. This is mostly due to the



(a) Initial guess 1 (b) Initial guess 2 (c) Initial guess 3 (d) Initial guess 4 (e) Initial guess 5

(f) Initial guess 6 (g) Initial Guess 7 (h) Initial Guess 8 (i) Initial guess 9 (j) Initial guess 10

Figure 6.2: The initial location of the well center points for the different initial guesses used for the PUNQ model test cases.

temporal smoothness that is imposed on the controls of the wells. As one may expect, the results of Case 2 and Case 3 optimization runs are very close, because the degree of freedom for optimizing the well controls are essentially the same in both cases; i.e., $n_{\mathrm{P}} = 4$. However, on average, the results for Case 3b are better than the corresponding results for Case 2b, i.e., increasing the number of control steps for each well to 100 in this particular situation resulted in a slightly better NPV value compared to the case with 4 control steps, probably because the well controls could change more smoothly throughout the reservoir life. The maximum NPV obtained for Case 3b was also higher than the maximum NPV obtained for Case 2b as shown in Fig. 6.4(b) and Table 6.1. For Case 2a and Case 3a, which share the same temporal correlation length, Case 2a with a lower number of control steps yielded slightly higher NPVs. This could be due to the fact that preserving more energy of the covariance matrix after SVD truncation (in Case 2a) overweighs the improvement gained by the smoother change of the well controls resulted from increasing the number of control

Table 6.1: The summary of the optimized NPV values at 2000 and 4012 simulation runs for the different joint optimization runs using the PUNQ model.

	at 2000 Sim runs					at 4012 Sim runs				
NPV, \$ $\times 10^9$	Case 1	Case 2a	Case 2b	Case 3a	Case 3b	Case 1	Case 2a	Case 2b	Case 3a	Case 3b
Mean	2.405	2.545	2.541	2.505	2.584	2.531	2.658	2.657	2.631	2.724
Min	2.298	2.355	2.283	2.315	2.331	2.388	2.483	2.389	2.446	2.443
Max	2.600	2.762	2.741	2.670	2.716	2.737	2.827	2.867	2.802	2.870
SD	0.099	0.132	0.138	0.131	0.128	0.111	0.126	0.142	0.130	0.145

steps (in Case 3a). For covariance matrices with similar correlation lengths, more energy is being preserved when the number of retained singular values is closer to the dimension of the matrix. This could explain the better solutions obtained in Case 2a where all the singular values are retained compared to the Case 3a, where only 4 singular values are retained out of 100. This effect becomes less pronounced for covariance matrices with smaller correlation lengths. Hence, it could not outweigh the advantage gained by increasing the number of control steps in the cases with $N_s = 50$ (compare Case 2b and Case 3b).

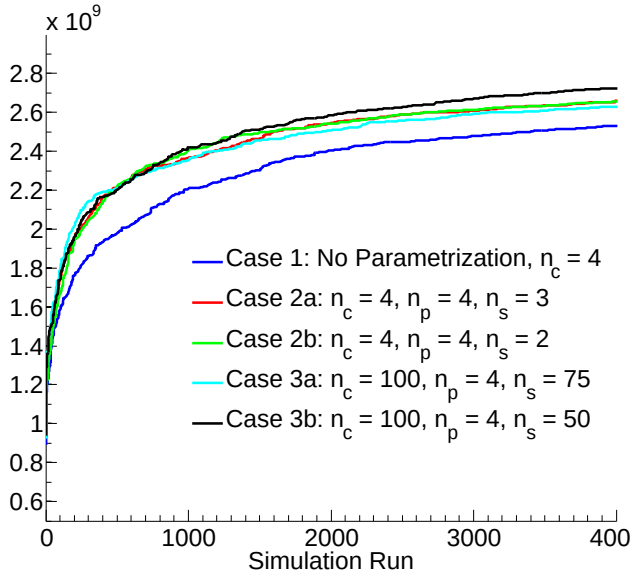
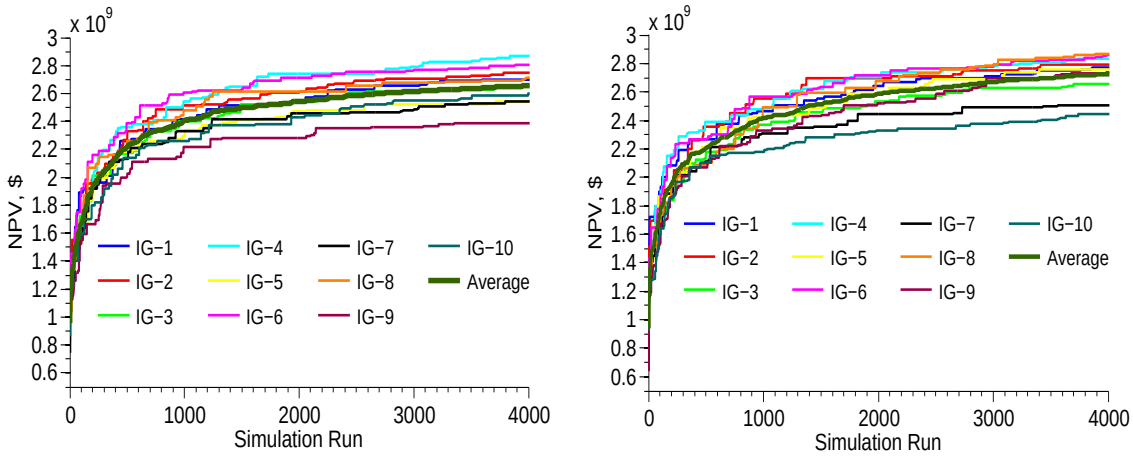


Figure 6.3: The average NPV value vs. the number of simulation runs for the joint optimization runs using the PUNQ model.

Among all the optimization runs performed, the best NPV values correspond to the optimization run 8 in Case 3b with the optimized NPV value equal to $\$2.870 \times 10^9$ and the



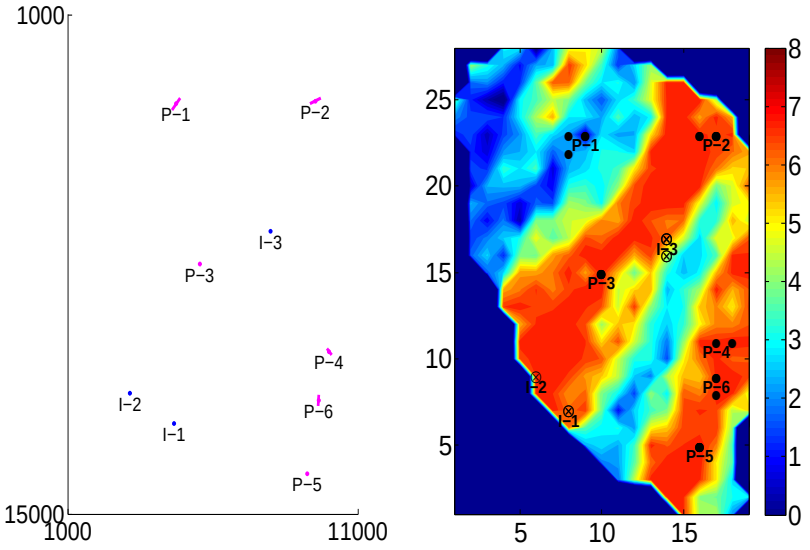
(a) Case 2b (b) Case 3b

Figure 6.4: The plot of NPV values vs. the number of simulation runs for the individual optimization runs of cases 2b and 3b.

optimization run 4 in Case 2b with the optimized NPV value equal to $\$2.867 \times 10^9$. In the following, the results obtained for these optimization runs are presented as an example of the results for the joint optimization algorithm.

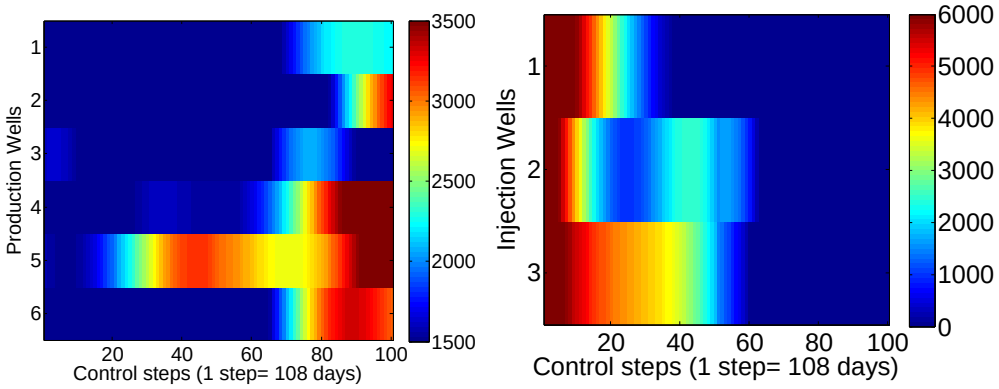
The optimized trajectories of the wells for the best optimization run of Case 3b, which also obtained the best NPV value among all optimization runs, (NPV= $\$2.870 \times 10^9$) are shown in Fig. 6.5(a). The perforations corresponding to these well trajectories are shown in Fig. 6.5(b). Note that although all wells have a 3D trajectory, for the purpose of presentation, the trajectories are shown in Fig. 6.5(a) as the projection of the well trajectories on a horizontal plane and the perforations of the wells are shown on the permeability field of simulation layer 1. The points shown in Fig. 6.5(b) are only vertical projections of the gridblocks perforated therefore it should be kept in mind that the perforations may occur in many more gridblocks located in different layers of the reservoir model. For example, the two points associated with the production well P-2 only show the two different sets of x and y coordinates shared amongst the gridblocks that are perforated by the well; the well actually perforates six gridblocks with at least one perforation in each of the five layers.

The final well controls corresponding to the optimization run 8 in Case 3b are shown in Fig. 6.6. The liquid production rate and water cuts of the production wells for this case



(a) Trajectories (b) Perforations

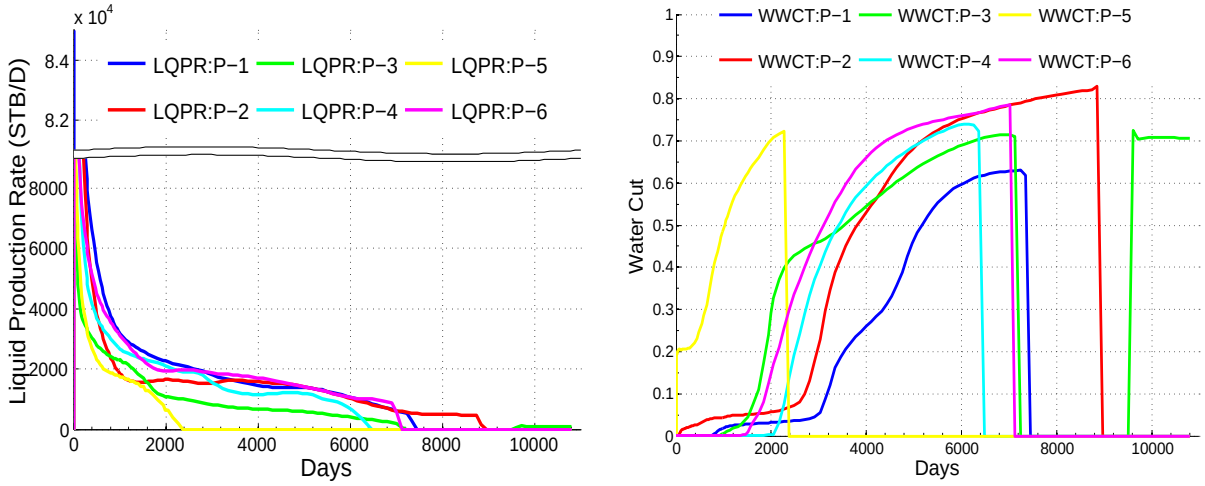
Figure 6.5: The well trajectories and perforations corresponding to the solution with the highest NPV value for the optimization runs of Case 3b.



(a) Production wells (b) Injection wells

Figure 6.6: The final well controls corresponding to the solution with the highest NPV value for the optimization runs of Case 3b.

are shown in Fig. 6.7. It should be mentioned that for all the results presented in this section even when the bottomhole pressures of the producers seem relatively low in order to check whether a well is open or shut the reader should refer to the figures where the liquid production rates and the water cuts of the wells are shown. If a producer is closed and the bottomhole pressure of its control is relatively low it means that the reservoir pressure around the well is lower than bottomhole pressure of the control of the well. As expected, the well controls shown in Fig. 6.6 are very smooth temporarily because a temporal correlation



(a) Liquid production rate for producers

(b) Water cut for producers

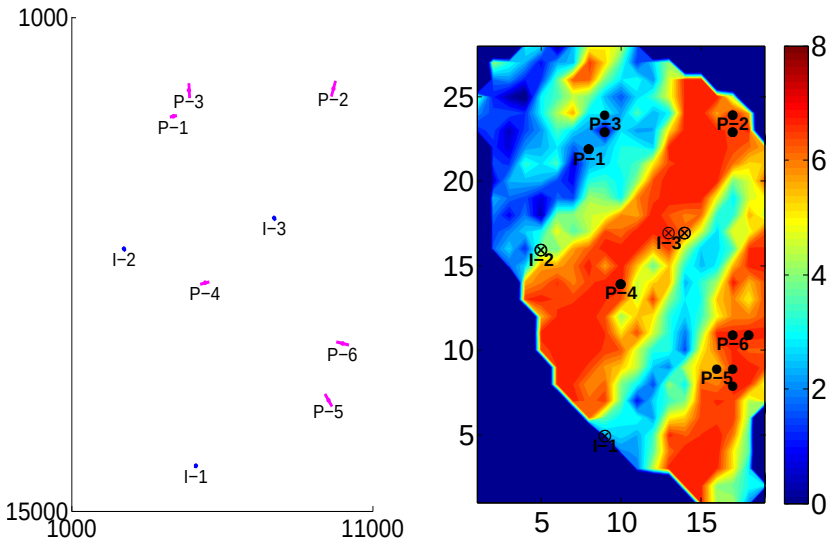
Figure 6.7: The liquid production rates and water cuts corresponding to the producers in the solution with the highest NPV value for the optimization runs of Case 3b.

is imposed on the well controls through the parametrization. Looking at Figs. 6.6 and 6.7 it can be seen that wells P-1, P-4 and P-6 produce close to the lower bound on the BHP (1500 psi) for roughly three quarters of the total reservoir life and then fully closed in the last quarter. These three wells are possibly closed for the last quarter due to their relatively high water cuts, although, in the case of P-1 it could be argued that the well could remain open for a longer period of time instead of keeping P-2 open for longer as its water cut is lower than that of P-2 and they both seem to have very similar liquid production rates. The production well P-5 is closed after the first quarter of the reservoir life most likely due to its high water cut. In the case of P-3 the well remains open for around three quarters of the reservoir life, then it is closed for around $2,500$ days and is finally opened again until the end of the reservoir life; for this late time period the water cut remains constant instead of increasing (unlike the first three quarters of the reservoir life). It is interesting to see that all the wells are closed below the water cut value above which the revenue from oil production would be less than the cost of water disposal. For the given economic parameters the water cut above which there is an economic loss in production is 0.875 (economic limit). Perhaps the solution could be improved further but the controls give a rough guideline of a production strategy that could be followed in practice. One of the possible reasons that could explain why the

wells are closed before their economic limit is reached in this solution is that due to the high level of smoothing imposed on the controls of the wells (due to the small number of retained singular values of the SVD representation of the temporal covariance matrix) it is not possible to maintain the control of the producer close to its lower bound and then increase suddenly the control to its upper bound (ban-bang control) when the economic limit of production has been reached. For example, in Fig. 6.6 it can be seen that the controls of P-4 and P-6 only reach their upper bounds after 9180 days (85 control steps) but it could be argued that these wells were closed earlier than the stipulated time because in order to reach the upper bounds of their controls these producers were forced to increase their bottomhole pressures slowly (due to the imposed temporal correlation) reaching the reservoir pressure along the process. The injection wells are injecting at high rates at the beginning of production and are slowly closed with all injectors shut in for the last 40 % of the reservoir life. Injection well I-3 injects at higher rates and for a longer period compared to the other two injection wells. This solution may take precedence over other possible solutions because I-3 is closer to more of the producing wells than the other two and by injecting at high rates at the beginning it ensures a high early production which lessens the reducing effect that the discount rate has on the NPV.

The optimized trajectories of the wells for the best optimization run of Case 2b are shown in Fig. 6.8(a) and the perforations corresponding to these well trajectories are shown in Fig. 6.8(b). In these figures, it can be seen that production wells 1 and 3 are very close to each other. To avoid this type of solutions, a constraint on the distance between the wells can be imposed by following a similar approach to that described in [34].

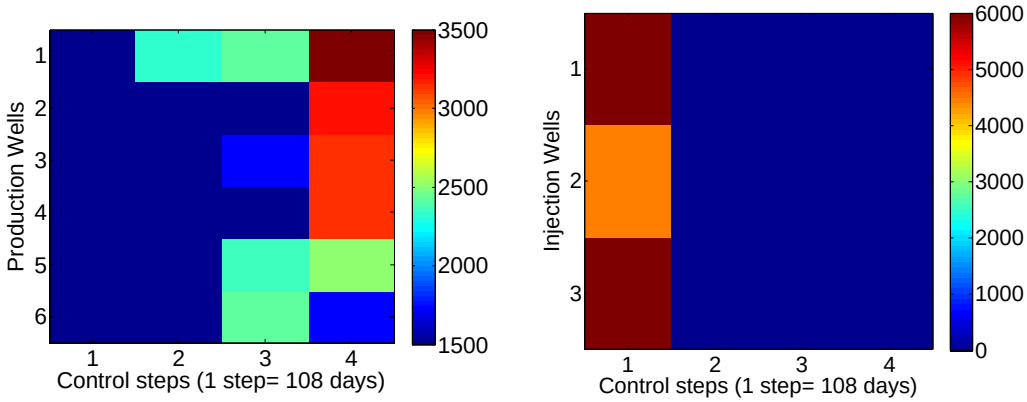
The final well controls corresponding to the optimization run 4 in Case 2b are shown in Fig. 6.9. The liquid production rates and water cuts of the production wells are shown in Fig. 6.10. Most of the production wells are open only for the first half of the reservoir life except for producers wells P-2 and P-4 which are open for the first three quarters of the reservoir life. The well P-1 is closed after the first quarter of the reservoir possibly to avoid a high water production early in the reservoir life. The three injection wells are injecting at



(a) Trajectories (b) Perforations

Figure 6.8: The well trajectories and perforations corresponding to the solution with the highest NPV value for the optimization runs of Case 2b.

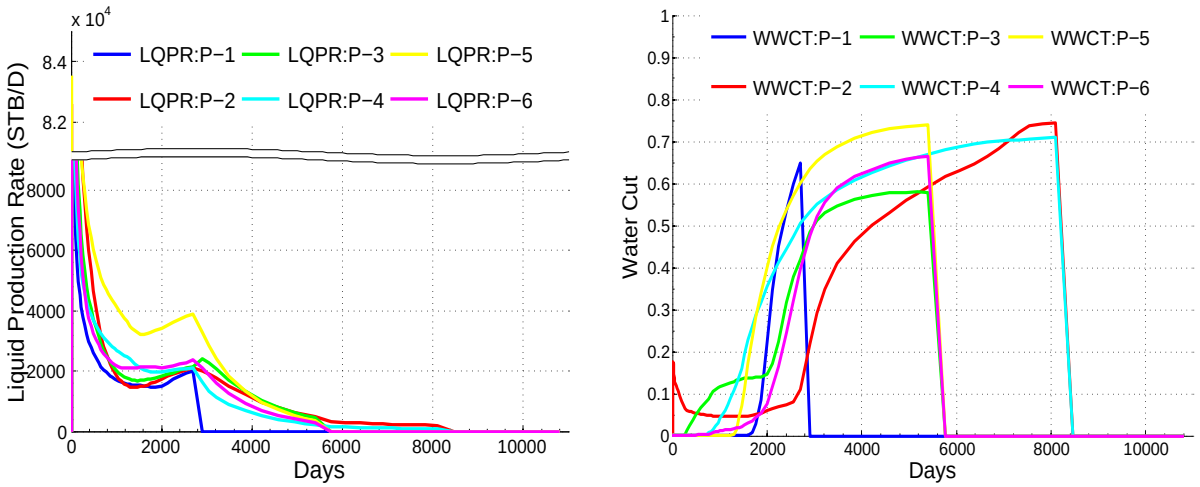
high rates, in fact I-1 and I-3 are injecting at the maximum allowable rate (6000 STB/D), at the beginning of the reservoir life. After 2,400 days all the injections wells are closed.



(a) Production wells (b) Injection wells

Figure 6.9: The final well controls corresponding to the solution with the highest NPV value for the optimization runs of Case 2b.

It should be noted that, although the solutions obtained using the joint optimization algorithm look reasonable, due to the imposition of a temporal covariance matrix on the controls of the wells and the reduction of the degrees of freedom of the production optimization problem, the solution obtained for the well controls may be suboptimal considering their



(a) Liquid production rate for producers

(b) Water cut for producers

Figure 6.10: The liquid production rates and water cuts corresponding to the producers in the solution with the highest NPV value for the optimization runs of Case 2b.

corresponding well locations. In order to benchmark the final well controls (see Figs. 6.9 and 6.6), the locations and trajectories of the wells are fixed at the positions obtained by the joint optimization algorithm (Figs. 6.8 and 6.5) and then life-cycle production optimization is performed to obtain the optimum controls of the wells. To perform the life-cycle production optimization, an implementation of the ensemble-based optimization algorithm (EnOpt) [8] is used. The implementation of EnOpt used in this study follows the one addressed in [59] and an ensemble size of 20 is used for the computations of the stochastic gradient for the EnOpt algorithm. As shown by [10], EnOpt may be considered as a special variance of G-SPSA [33]. Similar to the case of the joint optimization problem definition, 100 control steps each of a length equal to 108 days is used for the life-cycle production optimization of the reservoir. The EnOpt algorithm was applied with the temporal correlation lengths of 5400 days, i.e., $N_s = 50$. The production optimization is performed with two initial guesses. For the initial guess 1, the average of the upper and lower bounds of the controls is used, i.e., the initial injection rate of the water injection wells is set to 3000 STB/day and the initial bottomhole pressure of the producing wells is set equal to 2500 psi for all control steps. For the second initial guess, the well controls obtained from solving the joint optimization problem are used (Figs. 6.9 and 6.6) as the initial guess for the controls of the wells. Note that

for Case 2b, the well controls of the solution obtained from the joint optimization algorithm with 4 control steps of size 2700 days are projected onto 100 controls steps of size 108 days. The results of EnOpt algorithm for the life-cycle production optimization using the obtained well locations shown in Figs. 6.8 and 6.5 are shown in Fig. 6.11.

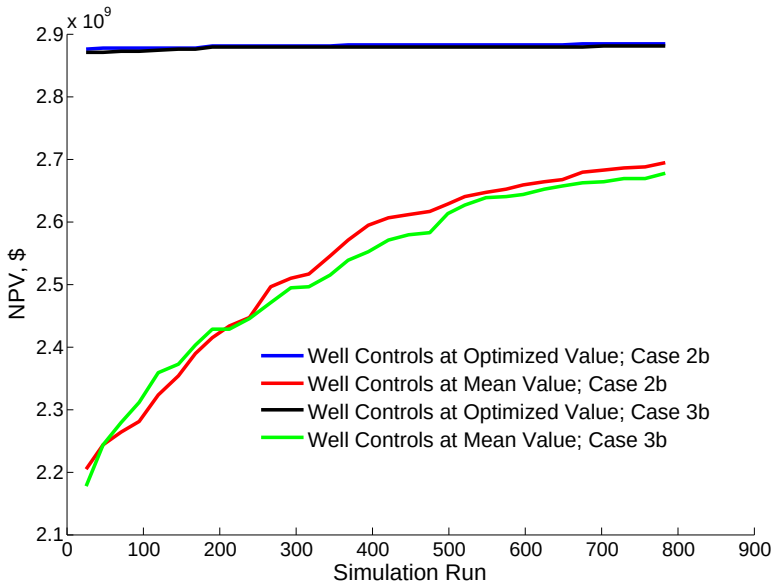
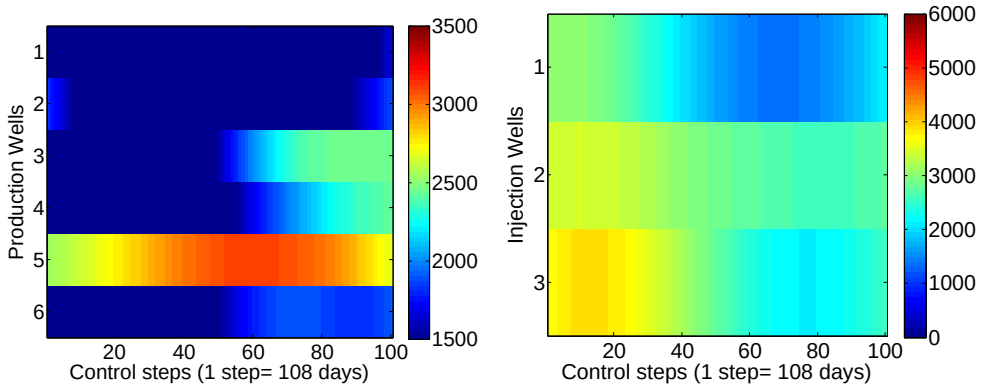


Figure 6.11: NPV vs. simulation runs for the life-cycle optimization of the reservoir with the optimized well trajectories. Blue and black curves correspond to the cases where the initial values of the controls were at their optimized values, and red and green curves correspond to the cases where the controls were initially set at their mean values.

From Fig. 6.11, it can be observed that applying the EnOpt algorithm for the life-cycle production optimization did not result in significant improvements in the NPVs obtained from the optimization of the joint problem (the results corresponding to the initial guess 4 in Case 2b and the initial guess 8 for Case 3b); see the top lines in Fig. 6.11 which represent the optimization runs corresponding to Case 2b and Case 3b with the optimized well controls used for the initial guess. The EnOpt production optimization algorithm improved the final NPV values insignificantly from $\$2.867 \times 10^9$ to $\$2.883 \times 10^9$ for the Case 2b and from $\$2.870 \times 10^9$ to $\$2.880 \times 10^9$ for Case 3b. Although not shown here, the solution of EnOpt algorithm with the optimized well controls as the initial guess are very close to the ones shown in Figs. 6.9 and 6.6. The final NPV values by the EnOpt algorithm when the well controls are initially set at their mean values and the well trajectories are taken from the

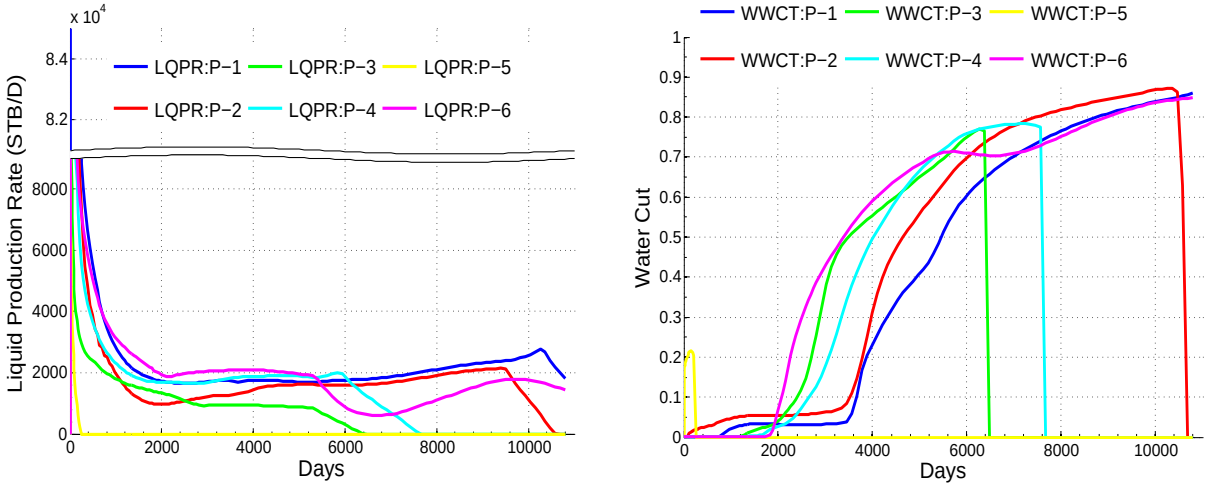
solutions of Case 2b and Case 3b are equal to $\$2.693 \times 10^9$ and $\$2.678 \times 10^9$, respectively. The optimal well controls found by the EnOpt algorithm when using the mean value of the controls as the initial values gave lower NPV values than the ones obtained from the implementation of the joint optimization algorithm; compare the final NPV from the joint optimization algorithm, $\$2.867 \times 10^9$ with the one obtained with EnOpt, $\$2.693 \times 10^9$ for Case 2b and the NPV value $\$2.870 \times 10^9$ from the joint optimization algorithm compared with $\$2.678 \times 10^9$ from EnOpt algorithm for Case 3b. It should however be noted that, (1) similar to the joint optimization algorithm presented, EnOpt also imposes a temporal smoothness on the controls of the wells, and (2) EnOpt algorithm utilizes stochastic gradients which are less robust than adjoint gradients. Moreover, EnOpt is a stochastic method and only one EnOpt optimization was performed. The well controls corresponding to the solutions obtained with EnOpt after starting with the mean values of the well controls are shown in Figs. 6.12 and 6.14. The liquid production rates and water cuts corresponding to the controls shown in Figs. 6.12 and 6.14 are shown in Figs. 6.13 and 6.15 respectively. It should be mentioned that the well controls presented in Fig. 6.12 are comparable with the well controls presented in Fig. 6.6 as they both correspond to the same set of well trajectories, where, the well controls in Fig. 6.6 are obtained by the joint optimization algorithm and the well controls in Fig. 6.12 are obtained by EnOpt algorithm. A similar comment applies to Figs. 6.14 and 6.9.

In addition to the joint optimization Cases 1, 2 and 3 which were presented above, here we investigate the case where, the joint optimization problem is broken into two steps and the resulting procedure is referred to in this work as the sequential optimization framework. In the first step, the optimal well trajectory problem is solved using the CMA-ES optimization algorithm with fixed well controls. To investigate the effect of using different sets of well controls in the well placement step, two different sets of well controls are used. In the first set, the controls of both injectors and producers are kept fixed at their mean values between their upper and lower bounds and in the second case, the controls of the injectors are kept fixed at their upper bound while the producers are kept fixed at their lower bound. Once



(a) Production wells (b) Injection wells

Figure 6.12: The final well controls obtained by the EnOpt algorithm where the well trajectories are from the solution of Case 3b (Fig. 6.5) and the well controls are initially set at their mean values.

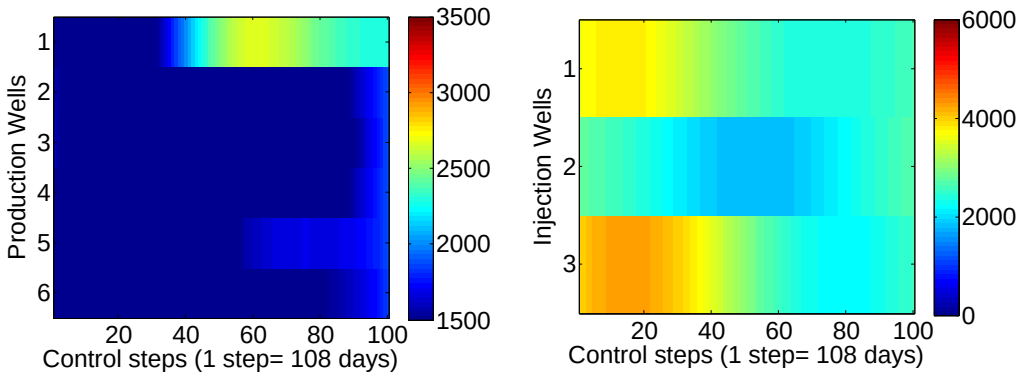


(a) Liquid production rate for producers (b) Water cut for producers

Figure 6.13: The liquid production rates and water cuts obtained by applying the well controls shown in Fig. 6.12 to the wells shown in Fig. 6.5.

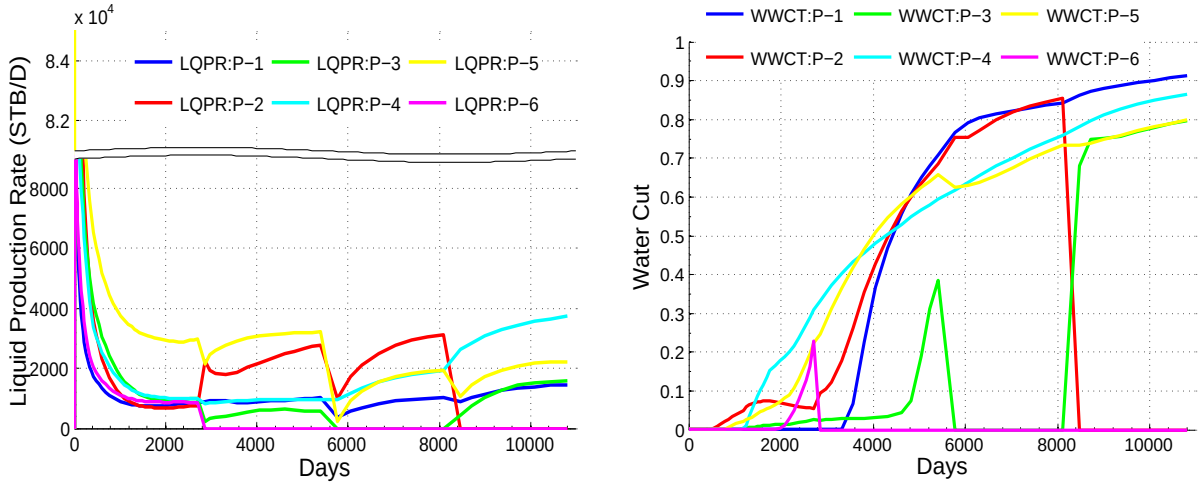
the optimal trajectory of the wells are estimated for both cases, the life-cycle production optimization step is performed, in which, the well locations are fixed and the controls of the wells are optimized. The life-cycle production optimization step is performed using the standard EnOpt algorithm. The name CMA-EnOpt sequential well placement algorithm is used in this work to refer to the sequential optimization algorithm that uses CMA-ES for the well placement step and the EnOpt algorithm for the production optimization step.

The well placement optimization step of the sequential optimization algorithm starts



(a) Production wells (b) Injection wells

Figure 6.14: The final well controls obtained by the EnOpt algorithm where the well trajectories are from the solution of Case 2b (Fig. 6.8) and the well controls are initially set at their mean values.



(a) Liquid production rate for producers (b) Water cut for producers

Figure 6.15: The liquid production rates and water cuts obtained by applying the well controls shown in Fig. 6.14 to the wells shown in Fig. 6.8.

with the 10 different initial locations for the well center points given in Fig. 6.2. The well placement step is performed for 3540 simulation runs (236 iterations of CMA-ES algorithm for the well placement optimization). Fig. 6.16 presents the plot of the average NPV for the ten initial guesses versus the simulation runs for the CMA-ES well placement optimization step corresponding to the case where the controls are kept fixed at their mean values and the case where producers are kept at their lower bound and injectors at their upper bound. The average final NPV for the well placement optimization step with mean well controls

is $\$2.044 \times 10^9$ while in the case where the maximum injection rates for the injectors and minimum BHP are kept fixed, the average final value is $\$2.4246 \times 10^9$. Among all optimization runs with different initial well trajectories, the optimization run with initial guess 4 yielded the highest final NPV value, NPV=$\$2.160 \times 10^9$, for the case where the mean well controls are used. In the well placement step that used maximum injection rates and minimum BHP, the initial guess 6 generated the highest final NPV value, NPV=$\$2.539 \times 10^9$.

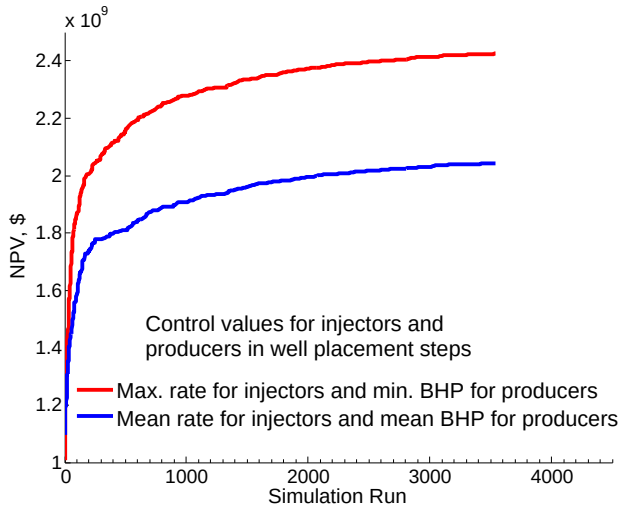


Figure 6.16: Average NPV for ten initial guesses vs. simulation runs for the well placement step of the sequential joint optimization algorithm, where CMA-ES algorithm is implemented for well placement optimization with two different sets of fixed well controls.

For the production optimization step, the EnOpt algorithm is used with an ensemble size equal to 20 and a temporal correlation length of 50 control steps (5400 days). The EnOpt algorithm is performed for 34 iterations. In Fig. 6.17, the average NPV curves obtained after applying the production optimization step are shown. The curves represent the average NPV corresponding to the two sets of solutions obtained from the well placement steps whose average NPV curves are shown in Fig. 6.16. The average final NPV after the production step that corresponds to the well placement step with fixed mean controls is 2.481×10^9. The final average NPV obtained in the production optimization step performed after the well placement step that uses maximum injection rates and minimum BHP for the producers is the same as the one obtained in the well placement step. The highest final NPV value after the well placement and production optimization step for the case where the mean

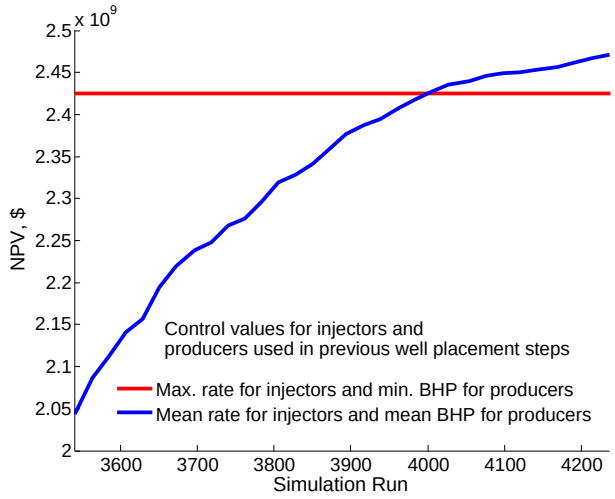
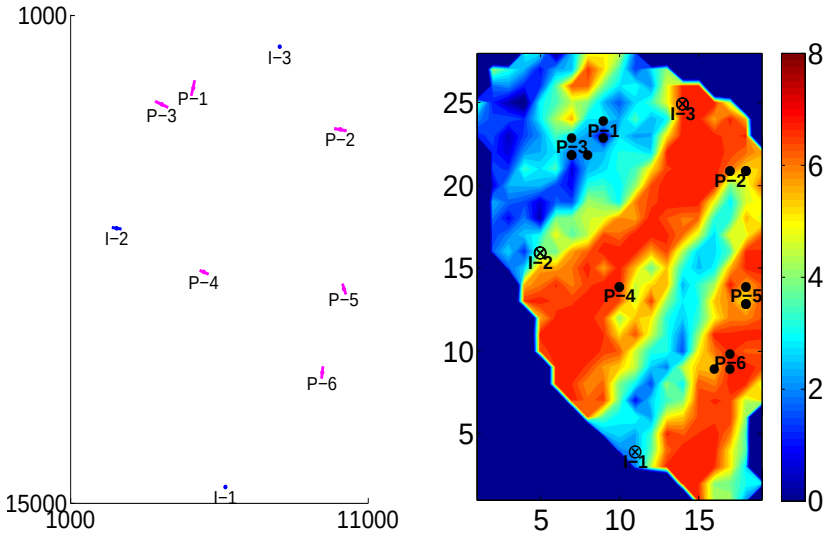


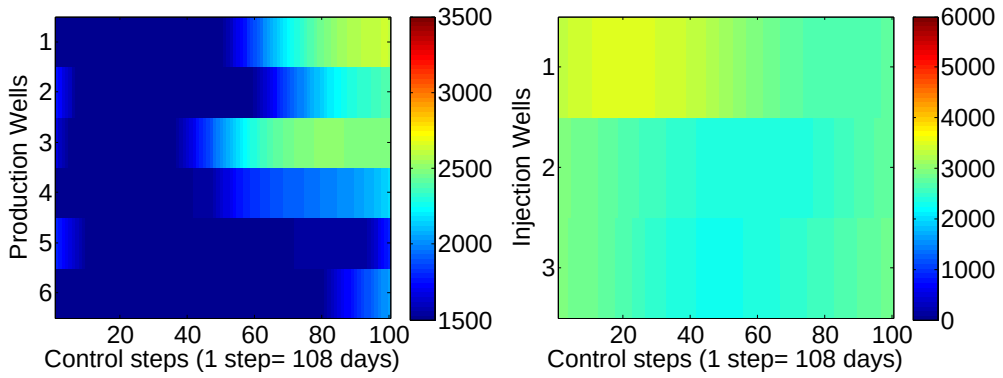
Figure 6.17: NPV vs. simulation runs for the production optimization step of the sequential joint optimization algorithm, where the EnOpt algorithm is implemented for production optimization. The production optimization step uses the well locations previously found in a well placement step that uses fixed well controls.



(a) Trajectories (b) Perforations

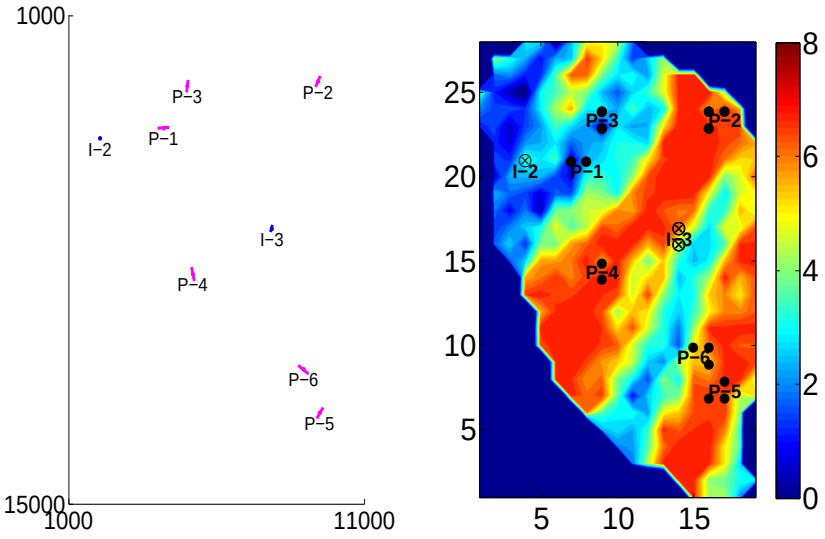
Figure 6.18: The well trajectories and perforations corresponding to the solution with the highest NPV value for the optimization runs of the CMA-EnOpt sequential algorithm that uses fixed mean controls for the well placement step (optimization run 5).

controls are used in the well placement step is 2.601×10^9 which corresponds to initial guess 5 for the well trajectories. The final well locations and trajectories corresponding to case 5 are shown in Fig. 6.18 and the corresponding well controls are shown in Fig. 6.19. The highest NPV value for the production optimization step corresponding to well placement with fixed



(a) Production wells (b) Injection wells

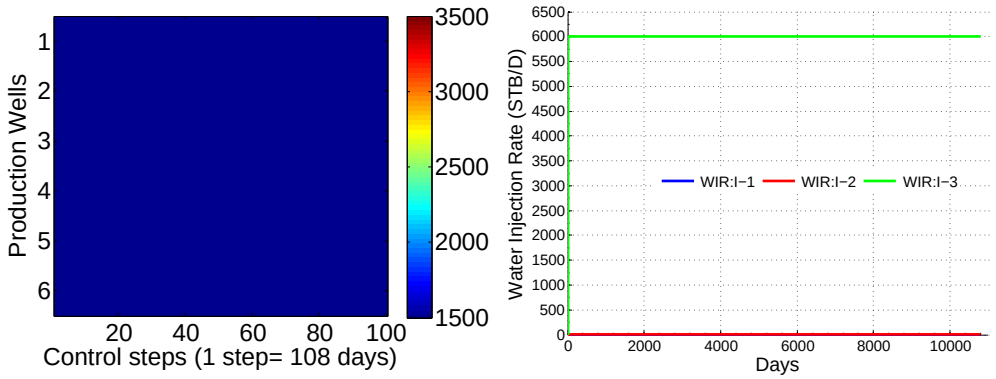
Figure 6.19: The final well controls corresponding to the solution with the highest NPV value for the CMA-EnOpt sequential algorithm that uses fixed mean controls for well placement (optimization run 5).



(a) Trajectories (b) Perforations

Figure 6.20: The well trajectories and perforations corresponding to the solution with the highest NPV value for the optimization runs of the CMA-EnOpt sequential algorithm that uses maximum injection rates for the injectors and minimum bottom hole pressures for the producers for the well placement step (optimization run 6).

maximum injection rates for injectors and minimum BHP for producers is the same as the one obtained in the well placement step and corresponds to the well trajectories of case 6. The final well locations and trajectories corresponding to case 6 are shown in Fig. 6.20 and its well controls are shown in Fig. 6.21. The lack of improvement after applying the production optimization step to the case where the well placement is performed using the injectors at



(a) Production wells

(b) Injection wells

Figure 6.21: The final well controls corresponding to the solution with the highest NPV value for the CMA-EnOpt sequential algorithm that uses maximum injection rates for the injectors and minimum bottom hole pressures for the producers for the well placement step (optimization run 6).

their upper bound and the producers at their lower bound shows that pursuing this route might lead to suboptimal results when compared to the case where the mean controls are used in the sequential algorithm. However, it is interesting to consider the solution for the best case from the well placement step using maximum injection rates for injectors and minimum BHP for producers which is shown in Figs. 6.20 and 6.21. In the solution (which is given by case 6) I-1 is eliminated (placed in inactive gridblocks) and I-2 is moved to an area of low permeability so that for the given maximum injecting pressure (7000 psi) the injection rate is very close to zero. Taking these two measures avoids the over-injection of water in the field throughout the reservoir life and increases the NPV.

In order to test further the sequential algorithm another two production optimization algorithms are implemented after the well placement step. The first algorithm used for the production optimization is the standard CMA-ES algorithm and for this algorithm similar to the EnOpt case, the number of optimization variables is $N = 100 \times 9 = 900$. The CMA-ES algorithm is run for 31 optimization iterations where each iteration requires 24 simulation runs, see Eq. 5.2. In the standard CMA-ES algorithm, the initial covariance matrix is set equal to the identity matrix. In the following this algorithm is simply referred to as CMA. The second algorithm uses the CMA-ES algorithm with an initial covariance matrix which

imposes temporal smoothness on the set of controls of each well. In this case, the covariance matrix at iteration (generation) 0 is set to

$$\boldsymbol{C}_U^{(0)} = \begin{bmatrix} \boldsymbol{C}_U^1 & & & & \\ & \ddots & & & \\ & & \boldsymbol{C}_U^w & & \\ & & & \ddots & \\ & & & & \boldsymbol{C}_U^{n_W} \end{bmatrix} \tag{6.1}$$

where $\boldsymbol{C}_U^w$ is the covariance matrix corresponding to the well controls of well w given in Eq. 3.3. Similar to the EnOpt algorithm, the temporal correlation length is set equal to 50 control steps (5400 days) in order to form the initial covariance matrix for the CMA-ES production optimization runs. In this CMA variant, a temporal correlation is imposed on the initial covariance matrix of the CMA-ES algorithm; in this work, this algorithm is referred to as smoothed CMA-ES (SCMA) algorithm. However, as previously mentioned, this covariance matrix is updated through the CMA-ES algorithm steps. In the following, the sequential algorithms corresponding to the different life-cycle optimization algorithms listed above are referred to as CMA-CMA sequential and CMA-SCMA sequential well placement algorithms.

The life-cycle production optimization step is performed for 31 optimization iterations for the CMA and SCMA optimization algorithms (so that the total number of simulations for the two new optimization algorithms is comparable to the number of simulations performed by the EnOpt algorithm which is run for 34 iterations). Note that for both production optimization algorithms, the same set of well locations and trajectories (perforations) obtained from the CMA-ES well placement optimization step that had the controls set at their mean values is used (the same locations and trajectories used by EnOpt).

The plot of average NPV versus the number of optimization iterations for the life-cycle production optimization step of the sequential algorithm using the two different CMA production algorithms is shown in Fig. 6.22 along with the average NPV curve obtained by

the CMA-EnOpt sequential algorithm. The average final NPV for the production optimization step using CMA and SCMA algorithms are $\$2.359 \times 10^9$ and $\$2.492 \times 10^9$, respectively. It should be probably noted that each iteration of the CMA and SCMA production optimization algorithms required 24 simulation runs and it is considered here that each iteration of the EnOpt algorithm with the ensemble size of 20 requires in average about 22 simulation runs.

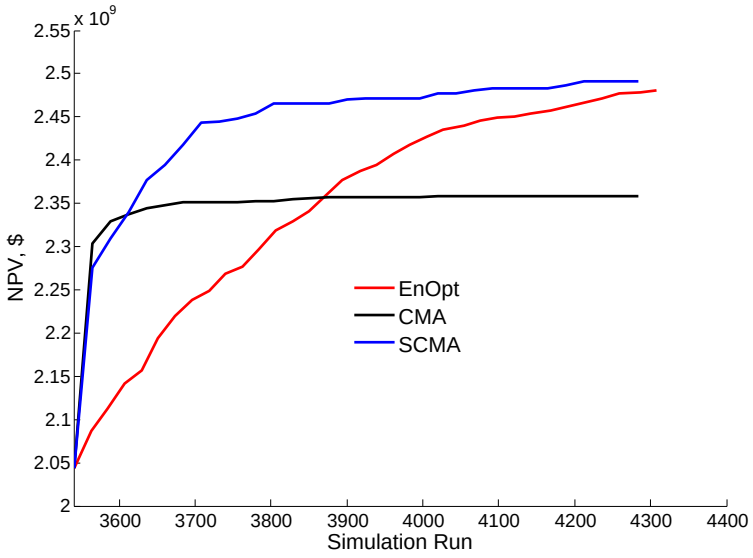
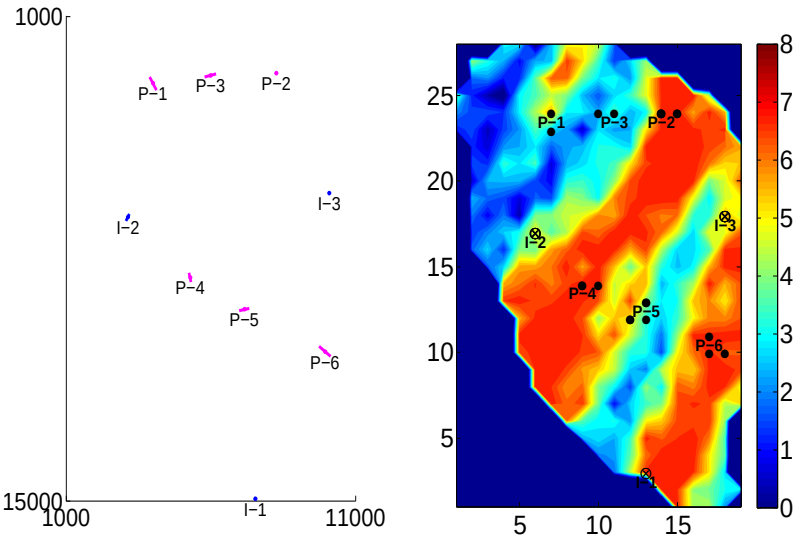


Figure 6.22: Average NPV for ten initial guesses vs. simulation run for the life-cycle production optimization step of the sequential joint optimization algorithm using EnOpt, CMA and SCMA algorithms.

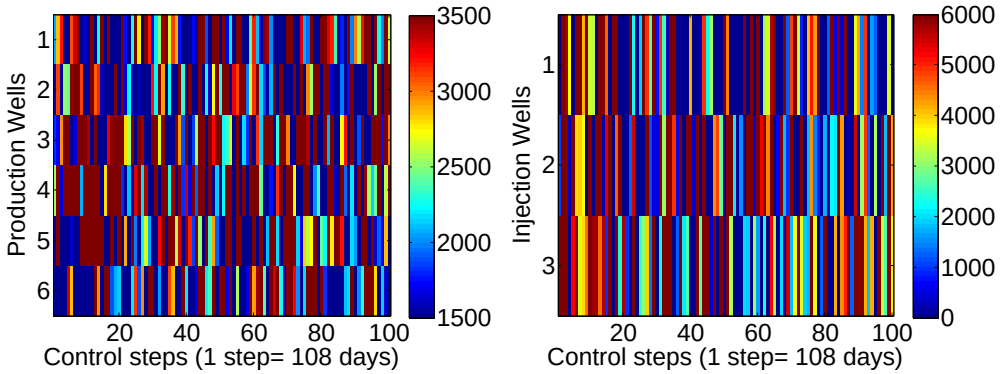
For the SCMA production optimization step using, initial guess 5 resulted in the highest NPV (same initial guess that yielded the highest NPV for EnOpt), which is equal to $\$2.627 \times 10^9$. In the case of the CMA production optimization step, initial guess 4 yielded the best NPV with a value of $\$2.473 \times 10^9$. The final well controls for these solutions are shown in Figs. 6.24 and 6.25 for CMA and SCMA, respectively. The well locations and trajectories for case 4 (the best case for CMA-CMA sequential) are given in Fig. 6.23.

Comparing the curves of the NPV averages in Fig. 6.22, it can be observed that the performance of EnOpt algorithm is superior to the CMA algorithm for production optimization, however, the SCMA algorithm in which the initial covariance matrix imposes a temporal correlation on the well controls outperformed EnOpt. Initializing the covariance



(a) Trajectories (b) Perforations

Figure 6.23: The well trajectories and perforations corresponding to the solution with the highest NPV value for the sequential optimization approach (optimization run 4) obtained by CMA algorithm.

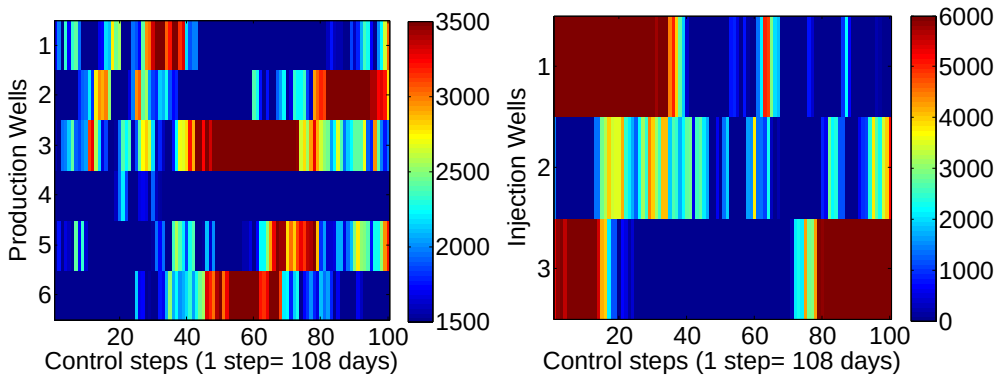


(a) Production wells (b) Injection wells

Figure 6.24: The final well controls corresponding to the solution with the highest NPV value for the CMA-CMA sequential algorithm (optimization run 4).

matrix in SCMA algorithm resulted in relatively smooth well controls; see Fig. 6.25. However, the final well controls from the CMA algorithm with the initial covariance matrix set equal to the identity matrix are very rough; see Fig. 6.24. As one expects, the well controls obtained by EnOpt are very smooth.

In addition to the sequential optimization scheme presented above, the iterative sequential scheme was also tested. In the iterative sequential scheme the algorithm alternates



(a) Production wells

(b) Injection wells

Figure 6.25: The final well controls corresponding to the solution with the highest NPV value for the CMA-SCMA sequential algorithm (optimization run 5).

between the well placement and production optimization steps as proposed by Jafarpour and Li in [27]. This would be an iterative scheme in which at each iteration, the well placement optimization step is performed with the fixed specified well controls, followed by a well control optimization step that uses the well locations obtained from the previous well placement step. Here, only two iterations are performed and this scheme is referred to as the iterative sequential optimization framework. For the sake of simplicity, only the EnOpt algorithm is used for the production optimization step. Two iterations of the sequential optimization approach are considered. The well placement optimization step is run at each iteration for 3540 simulation runs (236 CMA-ES well placement algorithm iterations). In each iteration of the iterative sequential scheme, the well control optimization is performed for 31 EnOpt iterations which is equivalent to about 682 simulation runs. The plot of NPV versus the number of simulation runs for the iterative sequential optimization run with different initial well locations is shown in Figs. 6.26.

The final mean value of NPV for the two iterative sequential runs are 2.491×10^9. From Fig. 6.26, it can be seen that the improvement of the NPV in the second iteration of the iterative sequential optimization scheme is marginal and both well placement and the production optimization steps in the second iteration of the sequential scheme resulted in only marginal improvements in the values of optimized NPV. A summary of the results from the sequential optimization runs and the proposed algorithm for simultaneously optimizing

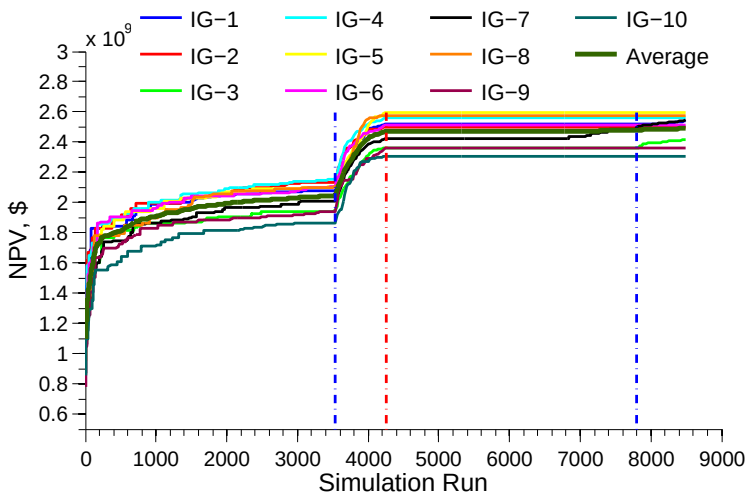


Figure 6.26: Optimal NPV vs. simulation runs for the iterative sequential optimization scheme with two well placement steps alternating with two optimal well control steps.

Table 6.2: The summary of the optimized NPV values with the simultaneous and sequential joint optimization algorithms using the PUNQ model.

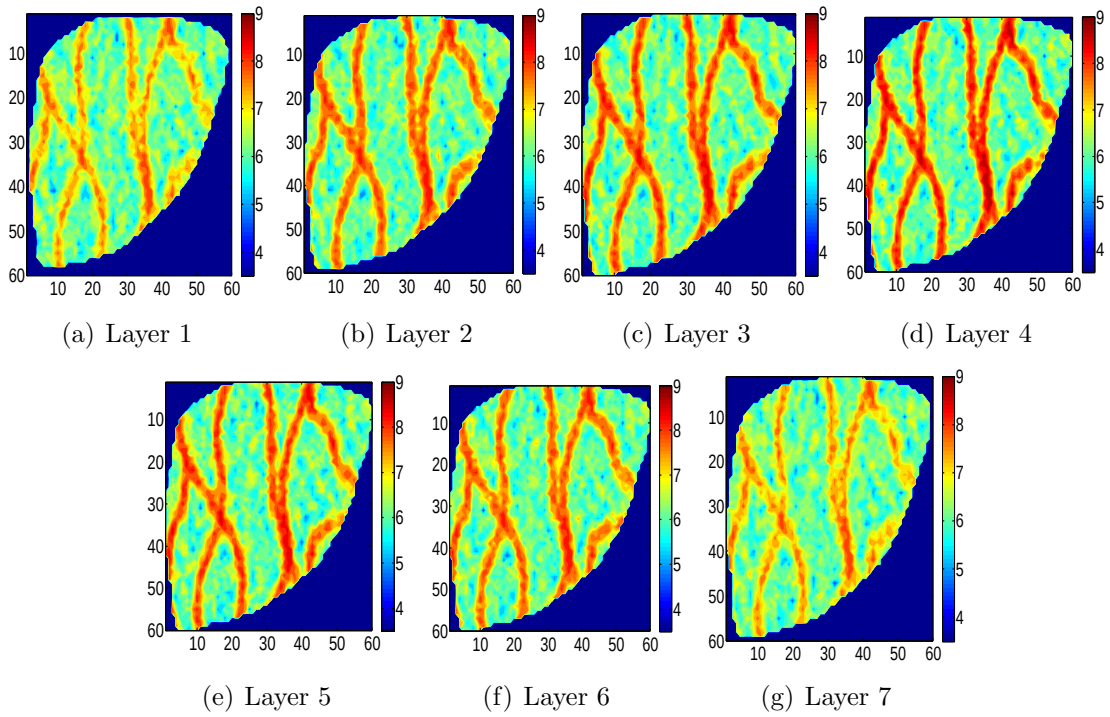
NPV, \$ $\times 10^9$	Simulation Runs	Mean	Min	Max	SD
Proposed Algorithm (Case 2b)	4012	2.657	2.389	2.867	0.142
Proposed Algorithm (Case 3b)	4012	2.724	2.443	2.870	0.145
CMA-EnOpt Sequential	4288	2.481	2.345	2.601	0.095
CMA-CMA Sequential	4284	2.359	2.189	2.473	0.101
CMA-SCMA Sequential	4284	2.492	2.330	2.627	0.101
Iterative CMA-EnOpt Sequential	8444	2.491	2.308	2.593	0.097

both well trajectories and controls is given in Table 6.2.

6.2 Egg reservoir model

In this section, the results for the application of the proposed algorithm to solve the joint optimization of well trajectories and controls for the egg reservoir model are presented. The egg reservoir model is described in [29]. The egg model is a two phase three dimensional reservoir model that has a $60 \times 60 \times 7$ grid. Each gridblock occupies a volume of $26.24 \times 26.24 \times 13.12$ cubic feet. The horizontal log-permeability distribution of reservoir simulation layers are shown in Fig. 6.27.

The egg model does not contain an aquifer or a gas cap. Its primary production is almost negligible and the mechanism used to aid production is water flooding [29]. The initial reservoir pressure is 5,801.5 psi at the datum depth of 4,000 ft (the top layer of the model).



(a) Layer 1 (b) Layer 2 (c) Layer 3 (d) Layer 4

(e) Layer 5 (f) Layer 6 (g) Layer 7

Figure 6.27: Horizontal log-permeability distribution of reservoir simulation layers of the egg model.

The reservoir life is considered to be 3,600 days. The original case has 8 injector wells and 4 production wells. The original positions of the well center points are shown in Fig. 6.28. The injectors are controlled by their injection rates and inject at a rate of 500 STB/day for the entire reservoir life. The producers are controlled by their bottomhole pressures and are all set to a BHP of 5,729 psi for the whole reservoir life in the original case (normally the NPV of this case is used as a reference case against which production optimization solutions can be compared to). The maximum length of a well used in the followings tests is 82 ft. The values of the economic parameters r_{o}, r_{w} and r_{winj} are set to 70, 10 and 10 \$/STB, respectively, and $b = 0.05$.

To perform joint optimization and well placement optimization the original positions of the wells are changed. The change is motivated by the fact that the wells are originally placed in a pattern that is already close to optimum and running any well placement algorithm with those locations could potentially trap the algorithm in a local optimum. In order to get a better "start" for the algorithm to explore the solution space more freely

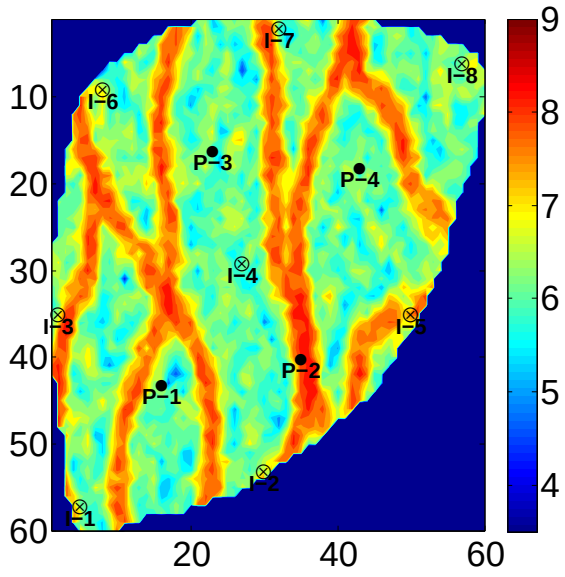
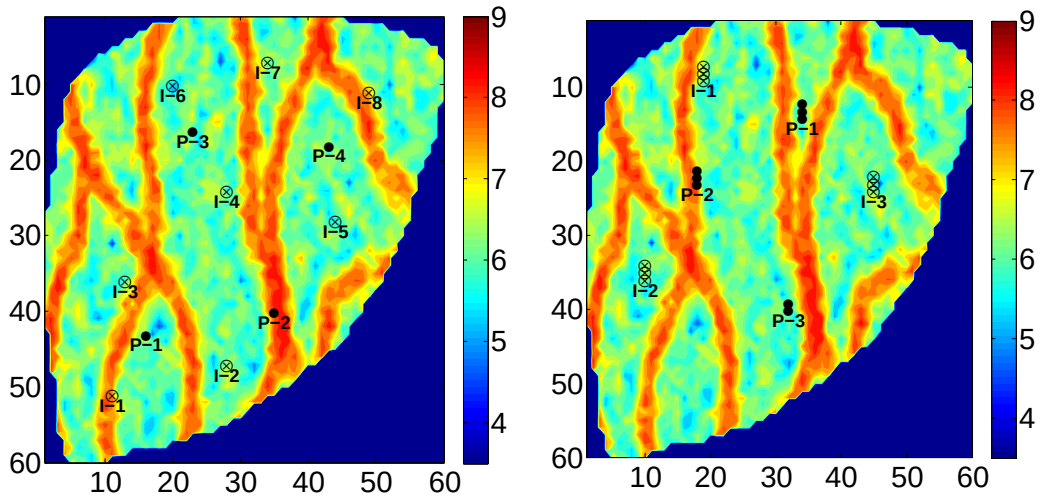


Figure 6.28: The initial locations of the well center points for the original egg model.

the injection wells are slightly moved into the reservoir. The new initial locations are used as an initial guess for many of the tests performed in this section and they are shown in Fig. 6.29(a). The wells in Fig. 6.28 are all vertical and penetrate all the layers. In addition to the initial guess presented in Fig. 6.29(a), referred to as case 1, other tests are performed using a different set of injectors and producers. The other initial guess is referred to as case 2 and has 3 injectors and 3 producers. The perforations and trajectories corresponding to case 2 are shown in Fig. 6.29(b). It is important to add that the wells shown in Fig. 6.29(b) are horizontal, hence, the multiple dots (or x's in the case of injectors) associated with each well which show the gridblocks that are penetrated by the well. The layer that the wells in Fig. 6.29(b) perforate is layer 4 (middle layer). It should be mentioned that in the well placement algorithms presented in this subsection the center points of the wells are constrained by vertical planes to a convex region within which there are only active gridblocks. This measure is taken so that the wells generated in the sampling step of the CMA-ES algorithm always lay inside the active region of the reservoir. In order to honor these bounds imposed on the well center points the method described in the subsection 5.6 is used.

The joint optimization algorithm and the CMA-EnOpt sequential optimization algorithm are tested using cases 1 and 2. The joint optimization problem, which uses the well



(a) Initial guess for case 1

(b) Initial guess for case 2

Figure 6.29: The initial locations of the well center points for cases 1 and 2 of the egg model which are used to test the joint optimization and well placement optimization algorithms.

control parametrization described in the section 3, is solved using the CMA-ES algorithm. The CMA-EnOpt sequential algorithm uses CMA-ES for the well placement step and then EnOpt for the production optimization step as mentioned in the previous section.

In the joint optimization runs performed here the reservoir life cycle is divided into 100 equally sized control steps of 36 days each (vector of control variables has a dimension of 100). The temporal correlation described in the well control parametrization section is imposed between the control steps of the same well. For all the joint optimization runs performed in the egg model, the correlation length used is 1800 days, i.e. $N_s = 50$. Only the four largest singular values of the SVD of the covariance matrix shown in Eq. 3.12 are retained in all the cases, therefore, in all the joint optimization problems a vector of 10 variables is associated with each well. The joint problem has a dimension of $12 \times 10 = 120$ for case 1 and a dimension of $6 \times 10 = 60$ for case 2. The maximum number of simulation runs for the joint optimization algorithm is set to 2,502 for case 1 and 2,224 for case 2. The maximum number of simulations was chosen so that the maximum number of CMA-ES iterations is the same for both cases (139 iterations), see Eq. 5.2.

Similar to the tests performed in the PUNQ reservoir model two different sets of controls are used in the well placement step of the sequential algorithm. The purpose of

choosing two different sets is to observe the effect that the fixed controls have on the final solution of sequential algorithm. The two different scenarios tested are the one where the well controls are kept fixed at their mean values and the one where the injectors inject at their maximum rate while the producers are kept open at their lowest BHP. The maximum number of simulation runs for the well placement step is set to 2,224 (139 CMA-ES iterations) for case 1 and 1,946 (139 CMA-ES iterations) for case 2.

In the tests performed in this section, two different sets of bounds are implemented on the well control variables. The first set of bounds, referred to as b1, has an upper bound for the injection rate of 500 STB/day and a maximum injection pressure of 5,874 psi. The lower and upper bounds on the producers given in b1 are 5,729 psi and 5,801.5 psi respectively. The aforementioned bounds are chosen to make the problems that use case 1 similar to the standard egg model case in terms of controls (described at the beginning of the section). The second set of bounds allows a greater variation of the controls than that offered by the first set of bounds. In the second set of bounds, referred to as b2, the maximum injection rate is set to 800 STB/day while the maximum allowable injection pressure is set to 6,990 psi. The upper and lower bounds on the BHP of the producers for the b2 set are 3,988.5 psi and 5,801.5 psi respectively. Both sets of bounds are used in tests involving case 1 while only the second set of bounds is used for the tests that use the setup given in case 2.

For all the tests performed, three different initial seeds are passed to the pseudo-random generator which is used to generate samples from $U[0, 1]$, the uniform distribution on [0,1]. The reason for taking this measure is that usually when different initial seeds are passed to the pseudo-random generator used in a stochastic algorithm different results are obtained. Therefore, to obtain reliable results more than one initial seed is used for each test and the results are averaged. From this point forward the three different initial seeds used will be referred to as seed 1, seed 2 and seed 3.

The average NPV curves obtained for the joint optimization and sequential optimization runs involving case 1 and the set of bounds b1 are shown in Fig. 6.30. The black dashed line in Fig. 6.30 marks the point at which the switch between the well placement step and

production optimization step occurs for the sequential algorithms. On average, the joint optimization algorithm performs slightly better than the sequential CMA-EnOpt algorithm.

From Fig. 6.30 it can be seen that the EnOpt step of the CMA-EnOpt sequential algorithm

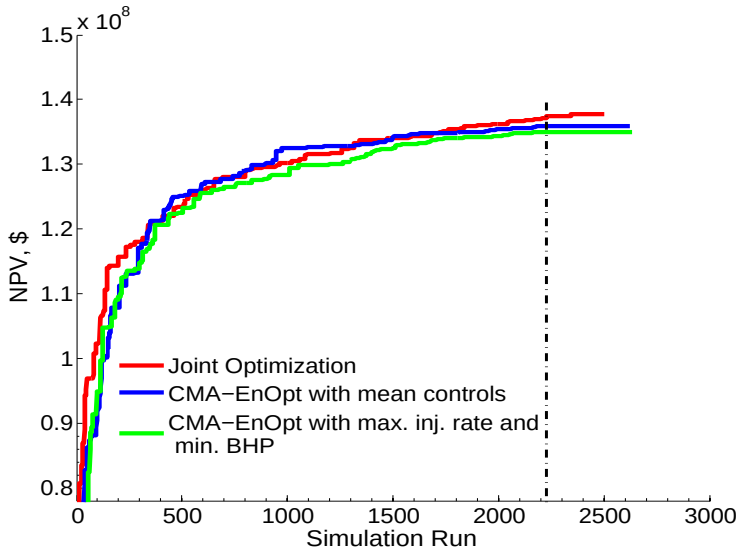
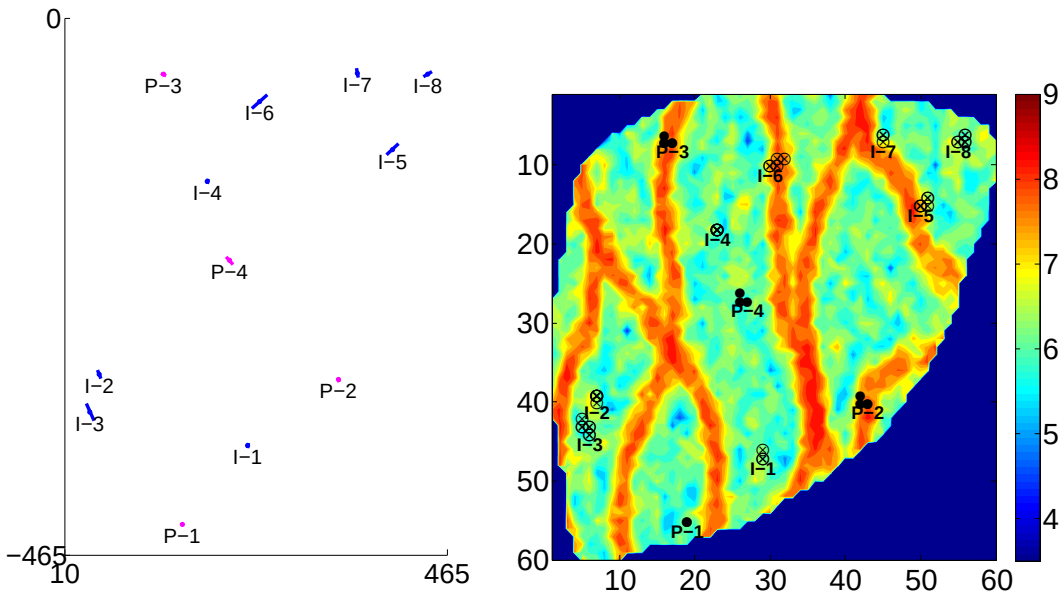


Figure 6.30: Average NPV vs. simulation runs for the joint optimization runs and the CMA-EnOpt sequential runs performed on case 1 with the set of bounds b1.

did not improve the solution previously found by the well placement step of the algorithm. The final average NPV obtained for the joint optimization problem was $\$1.382 \times 10^8$ while the maximum final NPV obtained was $\$1.387 \times 10^8$ which corresponds to the run that used seed 3. The CMA-EnOpt sequential algorithm that used the mean values for the controls (in well placement) obtained a final average NPV of $\$1.360 \times 10^8$ and a maximum NPV of $\$1.370 \times 10^8$ that corresponds to the run with initial seed 2. The average final NPV obtained by applying the CMA-EnOpt sequential algorithm with the maximum injection rate for injectors and minimum BHP for producers was $\$1.349 \times 10^8$ while the maximum NPV obtained was $\$1.371 \times 10^8$ which corresponds to the initial seed 3. The results are summarized in Table 6.3, which is given at the end of this chapter. In Table 6.3 "CMA-EnOpt mean" refers to the sequential optimization runs that set the controls of the wells to their mean values in the well placement step while "CMA-EnOpt bounds" refers to the sequential optimization runs that set the controls of the injectors to their upper bound and the controls of the producers to their lower bound.

The well trajectories and perforations corresponding to the best joint optimization run that was performed on case 1 with b1 bounds are displayed in Fig. 6.31 while the controls corresponding to these wells are displayed in Fig. 6.32. The liquid production rates and the water cuts obtained by applying the aforementioned controls are shown in Fig. 6.33. Looking



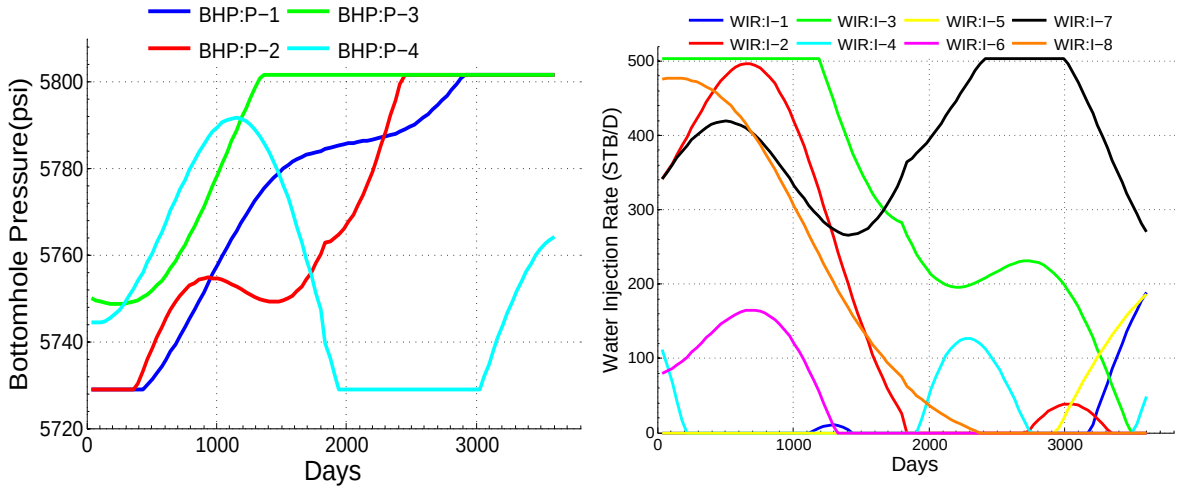
(a) Trajectories

(b) Perforations

Figure 6.31: The well trajectories and perforations corresponding to the solution with the highest NPV value for the joint optimization approach obtained by the CMA algorithm on case 1 with b1 bounds (seed 3).

at injection rates shown in Fig. 6.32(b) it can be seen that the field development strategy consists in flooding the reservoir from its northeast and southwest corners. In the northeast corner of the reservoir, only I-7 and I-8 are injecting at high rates while I-5 injects at very low rates (closed for most of the reservoir life). On the other hand, in the southwest corner I-3 and I-2 are injecting at high rates for the first half of the reservoir life. The remaining wells (I-1, I-4 and I-6) are basically closed for most of the reservoir life because none of them are located in one of the previously mentioned corners of the reservoir. The strategy seeks to produce as much oil as possible early in the reservoir life by, injecting at high rates and producing at low BHP, when the discount factor in the NPV equation is smaller. It then increases the BHP of the producers as the water saturation around them starts to become

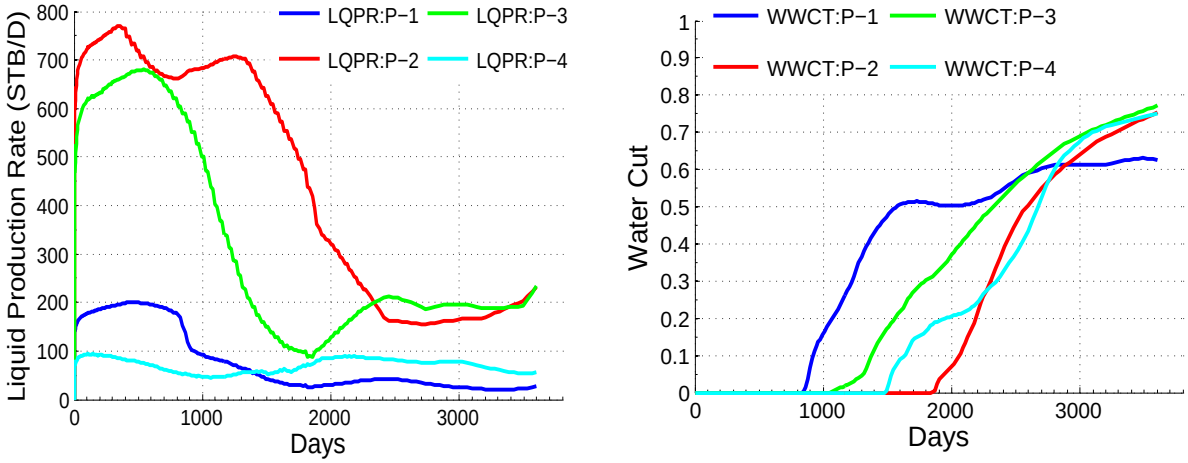
high. It must be mentioned that setting the BHP of the producers to the original reservoir pressure does not close the wells for this case because the reservoir pressure increases to a higher value than the original one (5,801.5 psi). The slight increase in reservoir pressure



(a) Production wells

(b) Injection wells

Figure 6.32: The final well controls corresponding to the solution with the highest NPV value for the joint optimization approach obtained by the CMA algorithm on case 1 with b1 bounds (seed 3) obtained by the CMA algorithm.



(a) Liquid production rate for producers

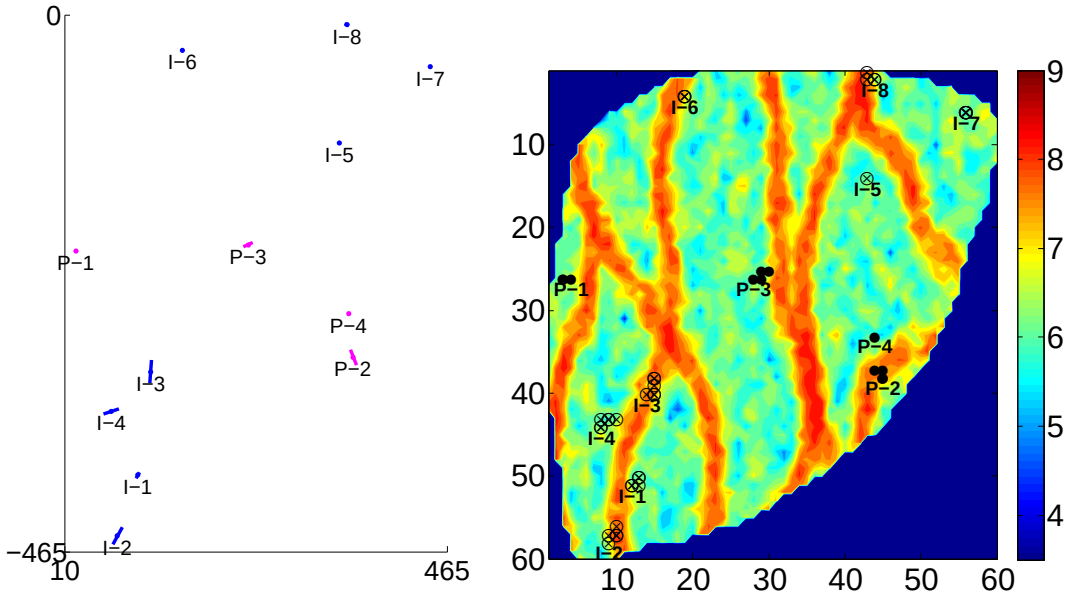
(b) Water cut for producers

Figure 6.33: The liquid production rates and water cuts obtained by applying the well controls shown in Fig. 6.32 to the wells shown in Fig. 6.31.

occurs because more liquid is being injected than is being produced. The injected water from wells I-3 and I-2 reaches P-3 early due to the fact that it is located in a channel that

is close to the two injectors, which also allows it to produce more oil at the beginning of the reservoir life. The injected water takes longer to reach P-4 and P-2 because these two wells are in the middle of the reservoir. It must be mentioned that due to the economic parameters selected in these problems, the watercut above which there is an economic loss in production is 0.875.

In Fig. 6.34 the well trajectories and perforations corresponding to the best solution of the CMA-EnOpt sequential runs that used mean controls for the case 1 with b1 bounds are shown. In this solution, the injectors are placed in the northeast and southwest corners of the reservoir in a similar fashion to the one seen in the joint optimization solution. The clustering of wells occurs due to the lack of a constraint on the distance between wells. As previously mentioned, the production optimization step was not able to improve the solution obtained by the well placement step in any of the two sequential optimization cases, therefore, the solution shown corresponds to the well placement step. The controls of the



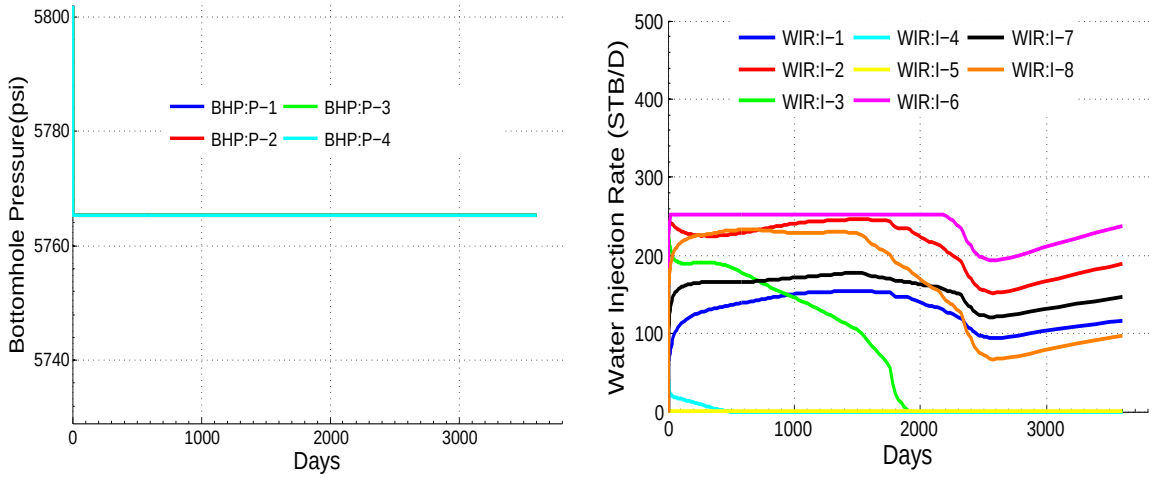
(a) Trajectories (b) Perforations

Figure 6.34: The well trajectories and perforations corresponding to the solution with the highest NPV value for the CMA-EnOpt sequential algorithm approach when used on case 1 with b1 bounds (seed 2) and the well controls are fixed at their mean values in the well placement step.

producers are the same ones used in the well placement step and are displayed in Fig. 6.35(a).

The specified controls for the injectors corresponding to this solution are set to their mean value throughout the reservoir life, but when trying to implement these controls in the reservoir simulation, the BHP of the injectors are taken to their maximum allowable BHP and the actual injection rates in the simulation are lower than the intended rates; see the actual injection rates of the injectors in Fig. 6.35(b). Therefore the injection rates used in the well placement algorithm are not being honored. This could be considered a way that the well placement algorithm has to decrease the over injection of water into the reservoir by picking the adequate well placement parameters for the injectors (such as their lengths and coordinates in the reservoir) so that these are forced to inject at their maximum allowable BHP without being able to inject at the originally specified well control rate. It could be argued that the upper bound of the BHP of the injectors should be increased in order to avoid this from happening but it is very likely that the well placement optimization algorithm will continue to favor well parameters that decrease the injection of water for this case. The liquid rates and water cuts obtained after applying the controls in Fig. 6.35 are shown in Fig.6.36. The drop in the injection rates after around 2,300 days have passed is due to an increase in the reservoir pressure (everywhere in the reservoir except very close to the producers the pressure increases by around 28 psi compared to the original reservoir pressure) which occurs due to the fact that more liquid is being injected than is being produced. The drops in the liquid production rates of the producers coincide with the water breakthroughs in the wells (see Fig. 6.36(a)), therefore, it could be assumed that due to the increase of water saturation in the vicinity of the wells the pressure drawdown (difference in pressure between the bottomhole pressure and the reservoir pressure) is not enough to maintain the same total liquid rates as before (water is heavier than oil). The final water cuts obtained with this solution are higher than those obtained in the joint optimization case, however, the water breakthrough occurs later in the sequential optimization case than in the joint optimization case for most of the wells.

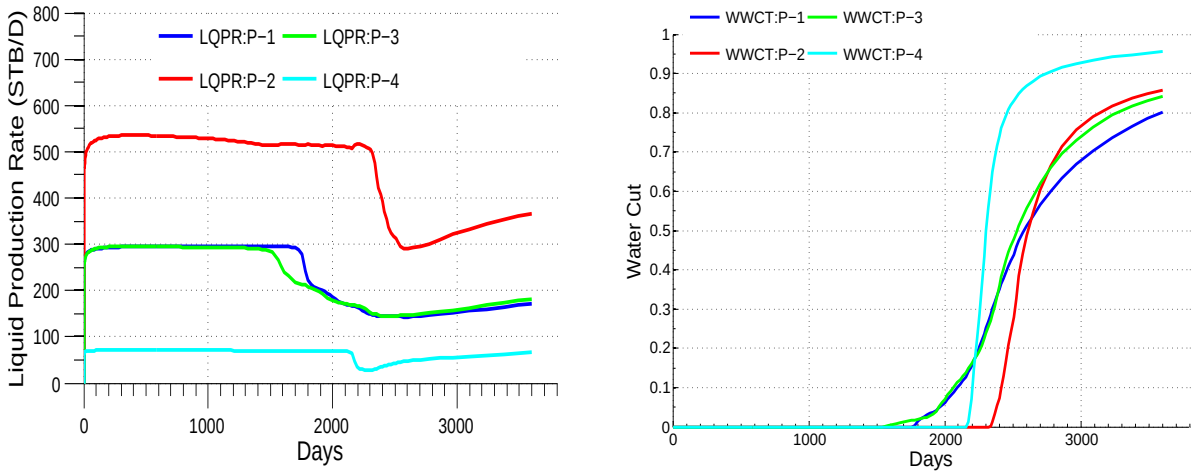
The well trajectories and perforations corresponding to the best run obtained when the CMA-EnOpt sequential algorithm was applied to case 1 with b1 bounds, where the



(a) Production wells

(b) Injection wells

Figure 6.35: The well controls corresponding to the solution with the highest NPV value for the CMA-EnOpt sequential algorithm approach when used on case 1 with b1 bounds (seed 2) and the well controls are fixed at their mean values in the well placement step.

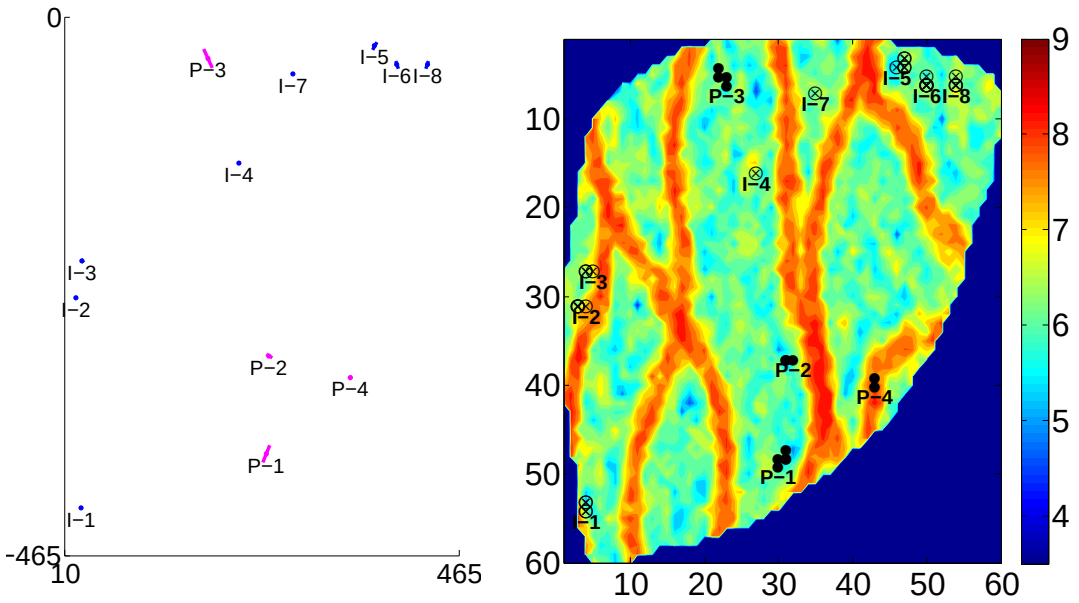


(a) Liquid production rate for producers

(b) Water cut for producers

Figure 6.36: The liquid production rates and water cuts obtained by applying the well controls shown in Fig. 6.35 to the wells shown in Fig. 6.34.

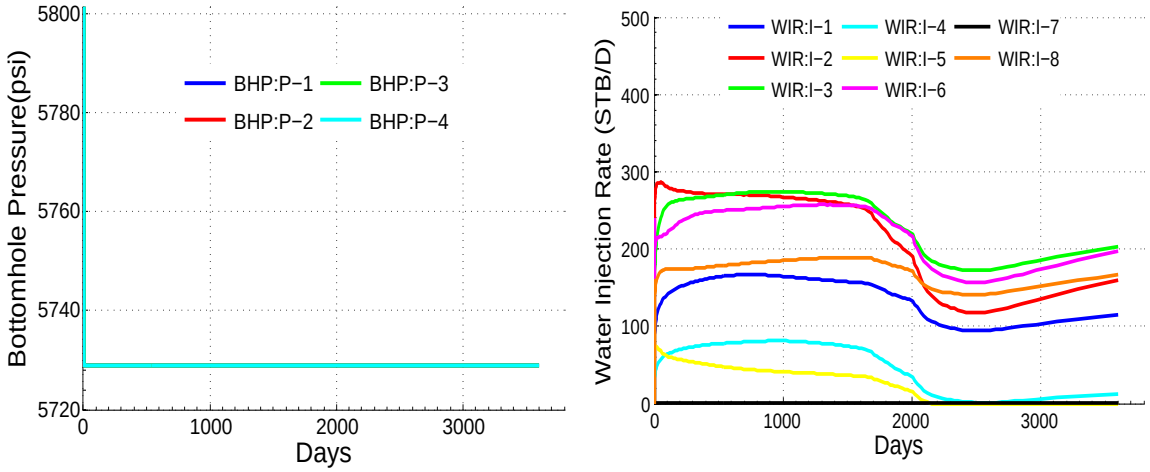
controls of the injectors were initially set to their upper bound while the controls of the producers were initially set to their lower bound (in well placement step), are shown in Fig. 6.37. The BHP controls corresponding to the producers shown in Fig. 6.37 are given by Fig. 6.38(a) while the actual water injection rates of the injectors (which inject at their maximum allowable BHP for this solution) obtained when the original injection controls



(a) Trajectories (b) Perforations

Figure 6.37: The well trajectories and perforations corresponding to the solution with the highest NPV value for the CMA-EnOpt sequential algorithm approach when used on case 1 with b1 bounds (seed 3) and the well controls are fixed at the upper bound for the injectors and at the lower bound for the producers.

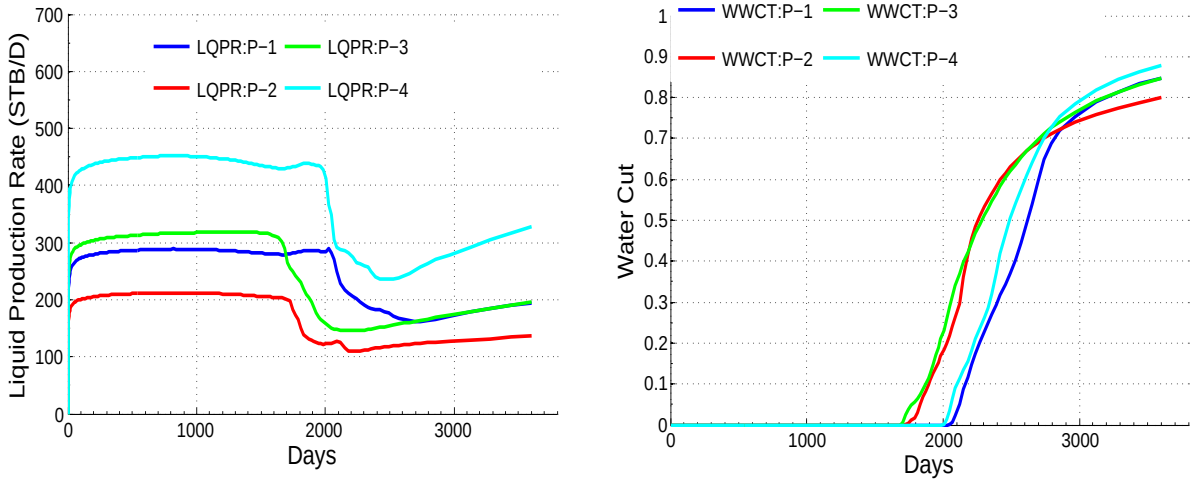
are applied to the simulator are shown in Fig. 6.38(b). The liquid production rates and the water cuts obtained after applying the controls shown in Fig. 6.37 are displayed in Fig. 6.39. The solution is similar to the one obtained by the sequential algorithm when mean values for the controls are used in the well placement step. The injectors in the solution presented in Fig. 6.37 are pushed to the southwest and northeast corners of the reservoir while the producers are placed close to the middle. The injectors are not able to inject at their maximum specified injection rate for the given locations and trajectories because the maximum allowable BHP is too low. As mentioned previously, placing the injectors in low permeability zones or choosing very small lengths is the mechanism that the well placement algorithm has to decrease the total injected water when the maximum allowable BHP for the injectors is not sufficiently high. The drop in the liquid production rates most likely occurs due to the increase of water saturation around the wells and the fixed BHP of the producers. In this solution, the final water cuts obtained are higher than the ones obtained in the joint optimization algorithm.



(a) Production wells

(b) Injection wells

Figure 6.38: The well controls corresponding to the solution with the highest NPV value for the CMA-EnOpt sequential algorithm approach when used on case 1 with b1 bounds (seed 3) and the well controls are fixed at the upper bound for the injectors and at the lower bound for the producers.



(a) Liquid production rate for producers

(b) Water cut for producers

Figure 6.39: The liquid production rates and water cuts obtained by applying the well controls shown in Fig. 6.38 to the wells shown in Fig. 6.37.

In Fig. 6.40, the average NPV values obtained for the joint and sequential optimization runs performed on case 1 with the set of bounds b2 are displayed. The final average NPV for the joint optimization runs was $\$1.505 \times 10^8$ and the maximum NPV obtained was $\$1.538 \times 10^8$ which corresponds to the run that used seed 1. The final average NPV for the

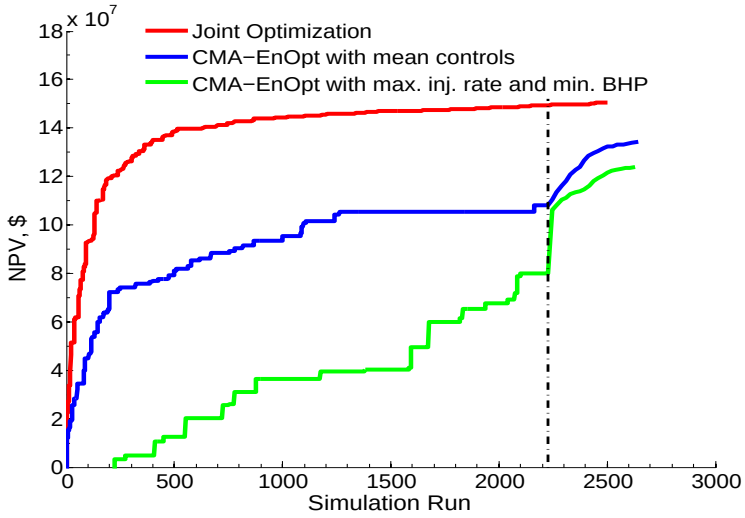
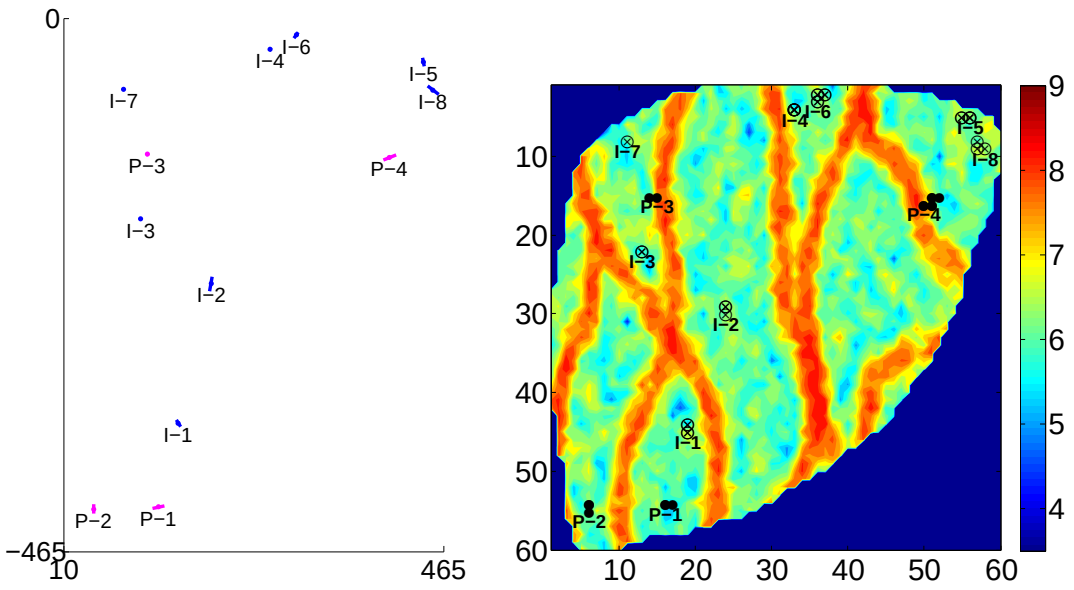


Figure 6.40: Average NPV vs. simulation runs for the joint optimization runs and the CMA-EnOpt sequential runs performed on case 1 with the set of bounds b2.

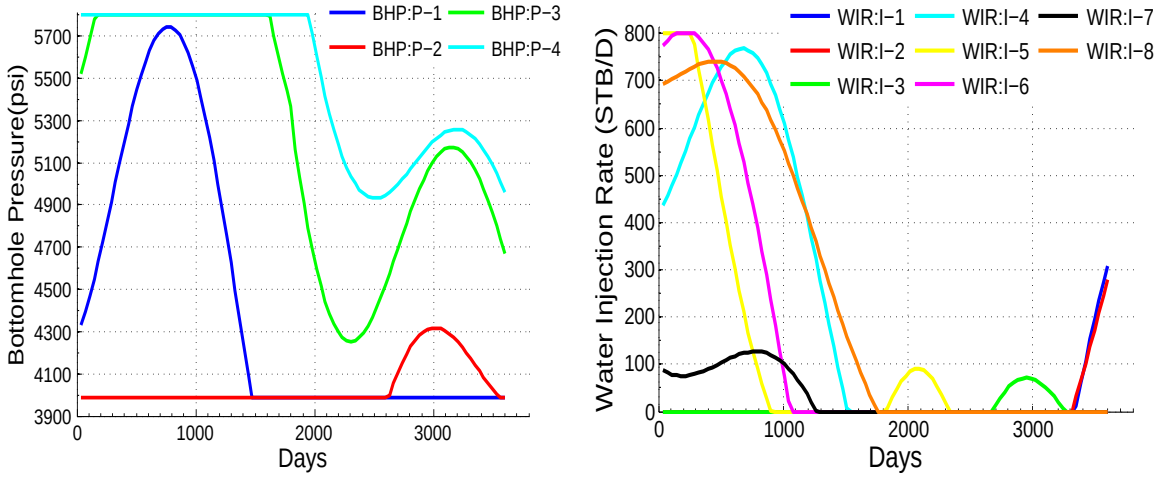
sequential optimization runs that used mean controls in the well placement step was $\$1.342 \times 10^8$ while the maximum NPV obtained among the different runs was $\$1.439 \times 10^8$ which corresponds to the run that used seed 2. The final average NPV obtained for the CMA-EnOpt sequential runs that used the upper bound value for the controls of the injectors and the lower bound value for controls of the producers in the well placement step was $\$1.237 \times 10^8$ and the best NPV obtained was $\$1.386 \times 10^8$ which corresponds to the run that used seed 1. The results are summarized in Table 6.3 at the end of this section.

The well locations and perforations corresponding to the best solution obtained by the joint optimization algorithm applied to case 1 with b2 bounds are shown in Fig. 6.41. The well controls corresponding to the solution are displayed in Fig. 6.42. The liquid production rates and water cuts obtained after applying the aforementioned controls to the wells shown in Fig. 6.41 are shown in Fig. 6.43. By looking at the location of the wells and their corresponding controls, it can be deduced that the strategy is to inject at high rates from the northeast side of the reservoir (I-3, I-2 and I-1 are closed for most of the reservoir life) to sweep the oil towards the southwest corner of the reservoir (P-3 and P-4 are closed for the whole reservoir life). The well I-7 aids the sweeping process by injecting at relatively low rates from the northwest side of the reservoir so that as much oil as possible is swept towards



(a) Trajectories (b) Perforations

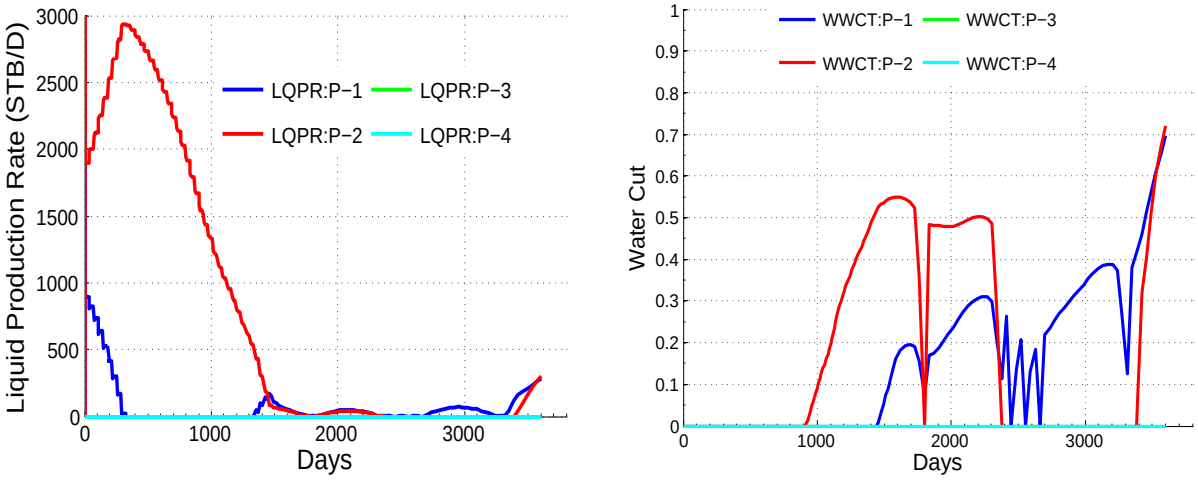
Figure 6.41: The well trajectories and perforations corresponding to the solution with the highest NPV value for the joint optimization approach obtained by the CMA algorithm on case 1 with b2 bounds (seed 1).



(a) Production wells (b) Injection wells

Figure 6.42: The final well controls corresponding to the solution with the highest NPV value for the joint optimization approach obtained by the CMA algorithm on case 1 with b2 bounds (seed 1).

the southwest corner. From the liquid production rates in Fig. 6.43(a) it is clear that most of the production occurs in P-2. In this solution P-2 is kept fully open for most of the reservoir life so that the water being injected moves towards the southwest corner and in the last



(a) Liquid production rate for producers

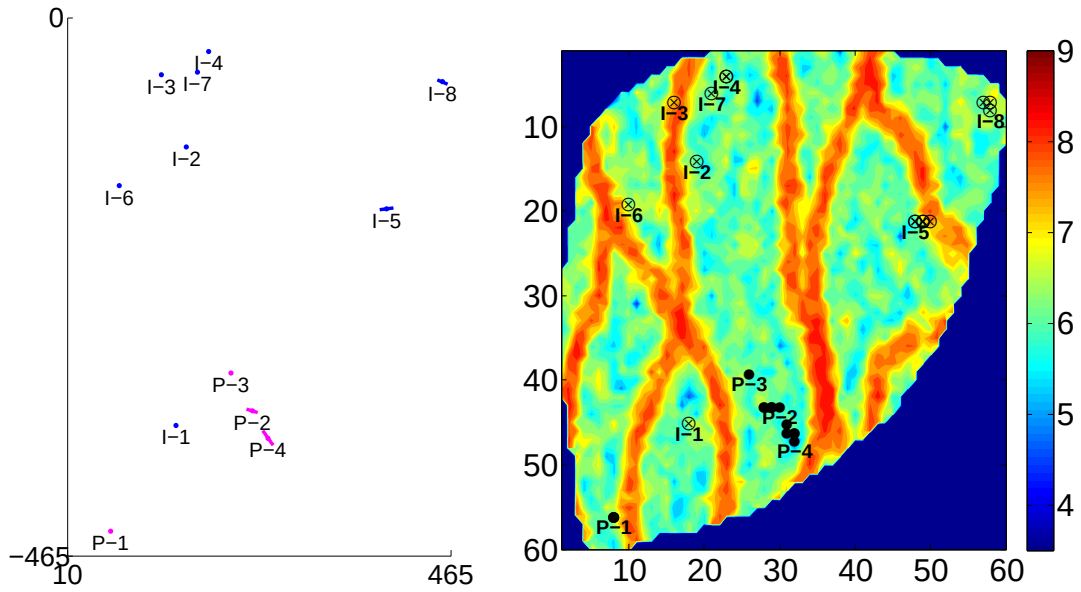
(b) Water cut for producers

Figure 6.43: The liquid production rates and water cuts obtained by applying the well controls shown in Fig. 6.42 to the wells shown in Fig. 6.41.

quarter of the reservoir life when the water cut becomes high it is closed, so that the water sweeps part of the remaining oil towards P-1.

The trajectories and perforations of the wells corresponding to the best solution obtained by using the CMA-EnOpt sequential algorithm on case 1 with bounds b2 and mean values for the controls (for the well placement step) are shown in Fig. 6.44. The controls of the wells, found by the production optimization part of the algorithm, are displayed in Fig. 6.45 while the liquid production rates and the water cuts obtained by using those controls are shown in Fig. 6.46. The relatively low injection rates of wells I-1, I-2, I-3, I-6 and I-7 shown in Fig. 6.45 are a consequence of the well trajectory parameters obtained after the well placement step as all these wells inject at their maximum allowable BHP in this solution. By analyzing the controls of the wells and their locations it can be assumed that the strategy is to flood the reservoir from the north region and move the oil towards the south where the producers are located. Although well P-3 is open throughout the reservoir life, its liquid production rate is negligible when compared to the production of wells P-2 or P-1. Well P-1 is open for the whole reservoir life while P-2 is closed after the first quarter of the reservoir life and later opened for the last 400 days. The reason for keeping P-1 open for longer than P-2 could be that the former is farther away from the injectors than the latter

and the injected water would take longer to reach P-1. Once the water cut becomes high in P-1 the BHP of the well is increased and production rate declines.

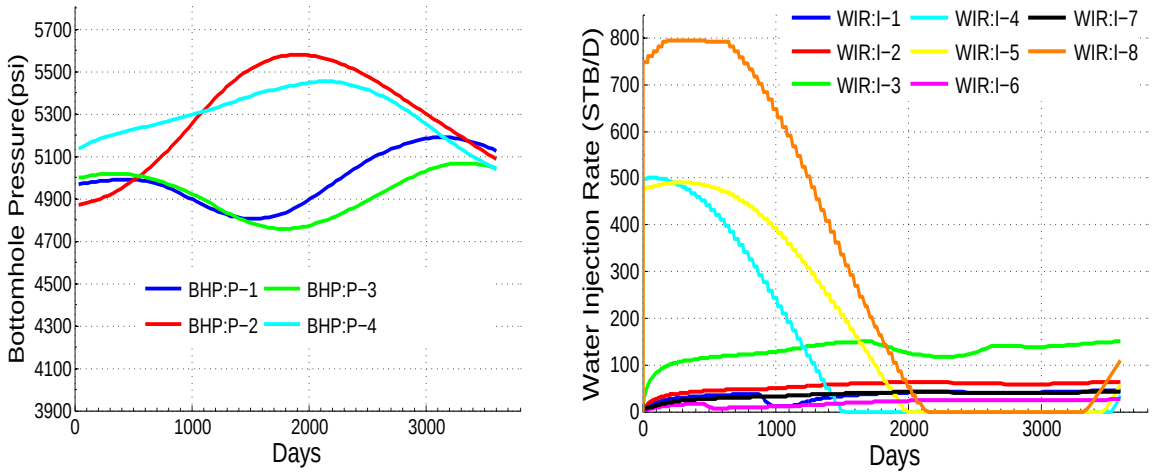


(a) Trajectories (b) Perforations

Figure 6.44: The well trajectories and perforations corresponding to the solution with the highest NPV value for the CMA-EnOpt sequential algorithm approach when used on case 1 with b2 bounds (seed 2) and the well controls are fixed at their mean values in the well placement step.

The perforations and trajectories of the wells corresponding to the best run of the CMA-EnOpt sequential algorithm performed on case 1 with b2 bounds while the injection rates were kept at their maximum rate and the BHP of the producers at their lowest pressure in the well placement step are shown in Fig. 6.47. The controls corresponding to the wells in Fig. 6.47 are shown in Fig. 6.48 while the liquid production rates and water cuts obtained by the application of these controls are displayed in Fig. 6.49. In the solution, wells I-1, I-4, I-6, I-7 and I-8 are injecting at their maximum allowable BHP for the whole reservoir life and due to the parameters chosen (length, location) in the well placement step of the optimization algorithm are not able to inject at higher rates. Therefore, any fluctuation in the injection rates of these injectors is only due to changes in the pressure in the vicinities of the wells. Wells P-1 and P-3 are open for a relatively brief period of the entire reservoir life (first 36 days in the case of P-1 and first 3 days in the case of P-3) and are later closed for

the rest of the reservoir life. The contribution of P-1 and P-3 towards the obtained NPV is almost negligible in comparison to the contributions of P-2 and P-4. The general strategy of this solution consists of injecting high volumes of water from the west side of the reservoir to move as much oil as possible towards P-4 and P-2 which are located on the east edge of the reservoir respectively. Wells I-7 and I-8 are used to displace any oil that remains in the east side of the reservoir. The closing of well P-4 after 400 days of the reservoir life has passed means that the oil is placed from the northeast corner and the west side of the reservoir towards P-2 for a long period of time until the water cut of the well becomes relatively high (see Fig. 6.49) at which time well P-2 is closed. After closing P-2, well P-4 is re-opened and allowed to produced until the end of the reservoir life.

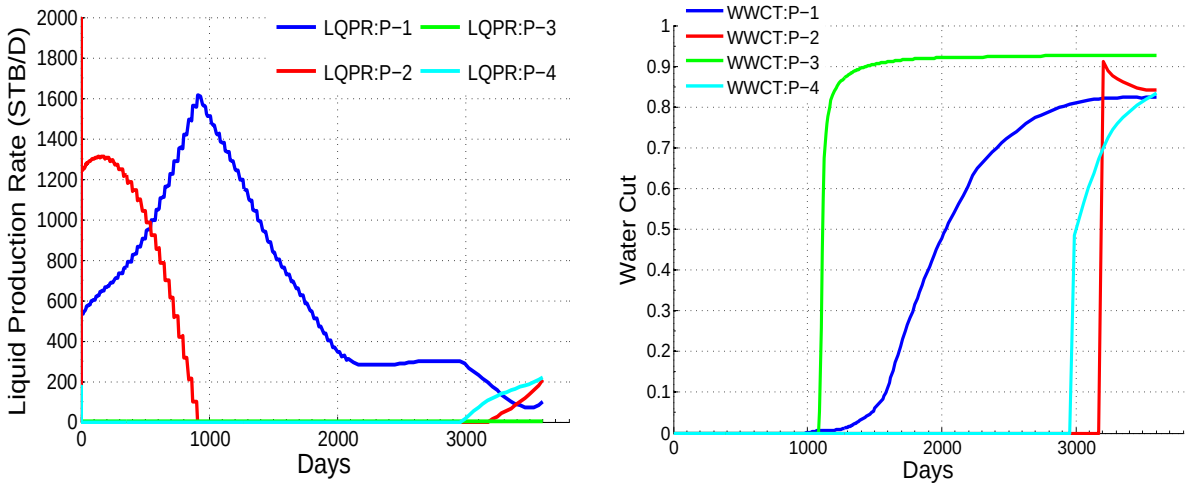


(a) Production wells

(b) Injection wells

Figure 6.45: The well controls corresponding to the solution with the highest NPV value for the CMA-EnOpt sequential algorithm approach when used on case 1 with b2 bounds (seed 2) and the well controls are fixed at their mean values in the well placement step.

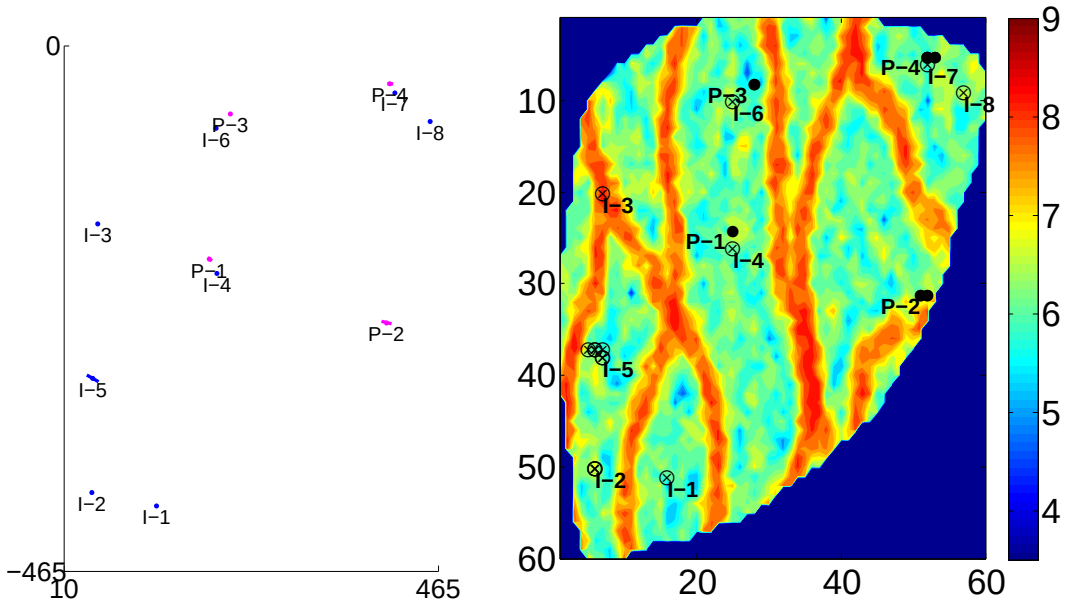
The average NPV values obtained for the joint and sequential optimization runs performed on case 2 with the set of bounds b2 are shown in Fig. 6.50. From the figure, it can be seen that the joint optimization algorithm outperforms the CMA-EnOpt sequential algorithm. The final average NPV for the joint optimization runs was $\$1.479 \times 10^8$ while the maximum NPV obtained was $\$1.494 \times 10^8$ which corresponds to the run that used seed 3.



(a) Liquid production rate for producers

(b) Water cut for producers

Figure 6.46: The liquid production rates and water cuts obtained by applying the well controls shown in Fig. 6.45 to the wells shown in Fig. 6.44.

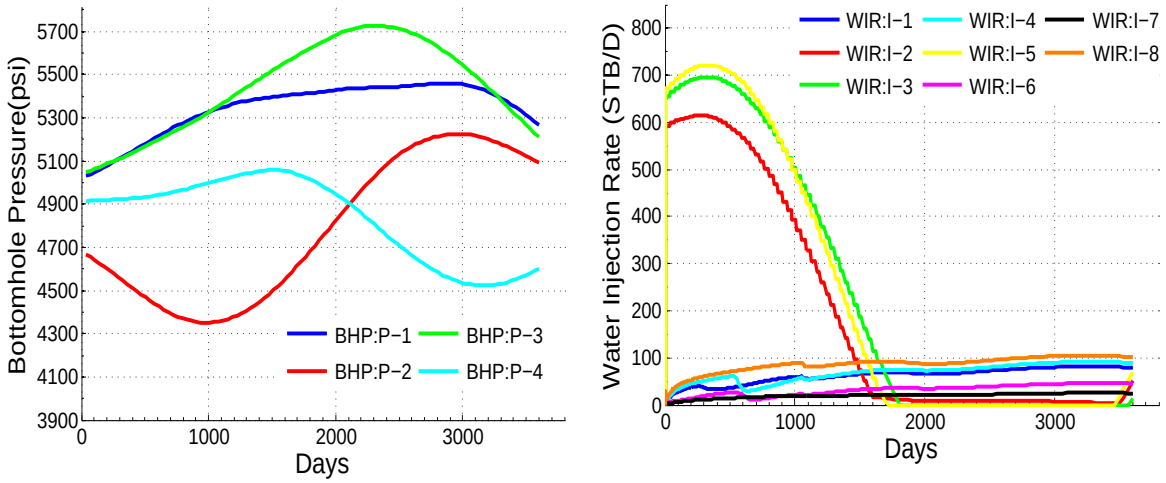


(a) Trajectories

(b) Perforations

Figure 6.47: The well trajectories and perforations corresponding to the solution with the highest NPV value for the CMA-EnOpt sequential algorithm approach when used on case 1 with b2 bounds (seed 1) and the well controls are fixed at the upper bound for the injectors and at the lower bound for the producers.

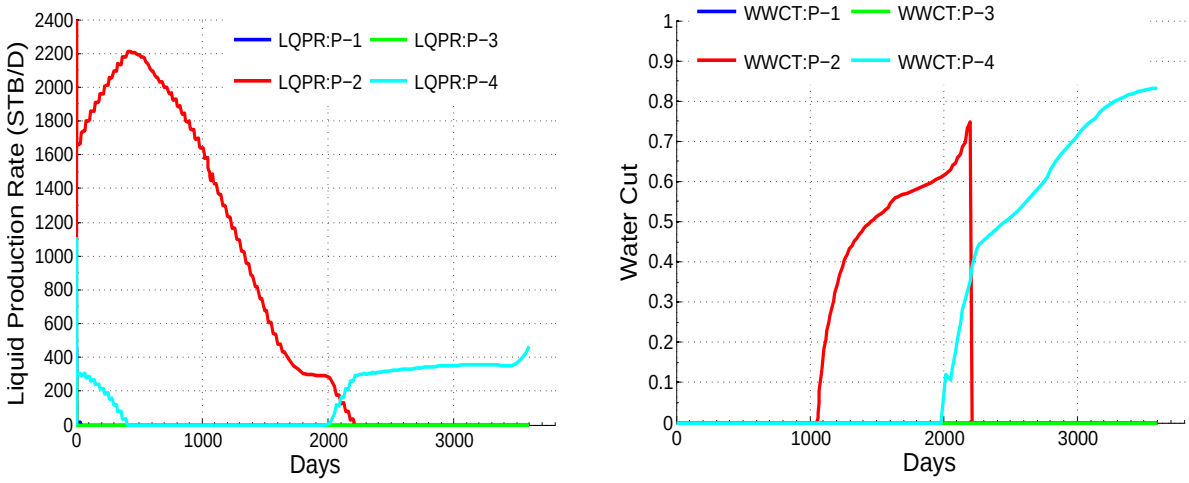
For the sequential optimization runs that used the mean controls in the well placement step the final average NPV was $\$1.434 \times 10^8$ while the maximum NPV obtained was $\$1.456 \times 10^8$. In the case of the sequential optimization runs that used the maximum injection rate for



(a) Production wells

(b) Injection wells

Figure 6.48: The well controls corresponding to the solution with the highest NPV value for the CMA-EnOpt sequential algorithm approach when used on case 1 with b2 bounds (seed 1) and the well controls are fixed at the upper bound for the injectors and at the lower bound for the producers.



(a) Liquid production rate for producers

(b) Water cut for producers

Figure 6.49: The liquid production rates and water cuts obtained by applying the well controls shown in Fig. 6.48 to the wells shown in Fig. 6.47.

injectors and minimum BHP for producers the final average NPV was $\$1.301 \times 10^8$ while the maximum NPV obtained was $\$1.351 \times 10^8$. It should be mentioned that many of the CMA-EnOpt sequential runs actually converged in their well placement step (due to the value of the relative error of the mean of the MVND used in the CMA-ES algorithm) before reaching the limit imposed on the maximum number of simulation runs (1946). Therefore, in order

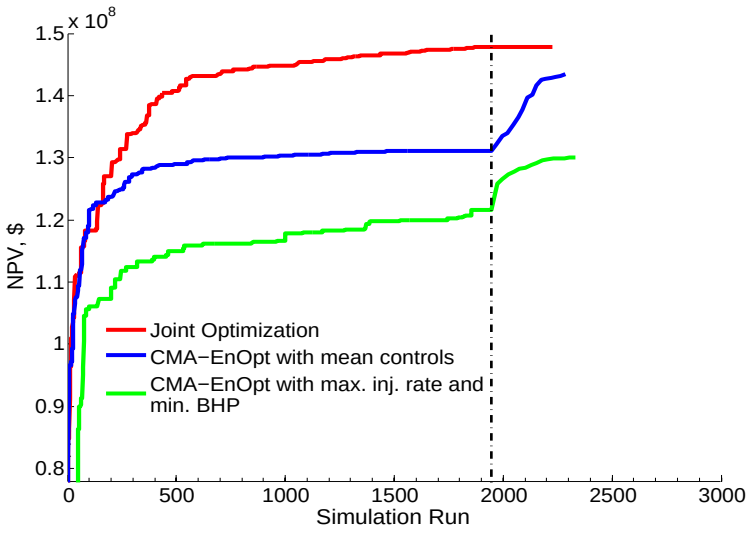
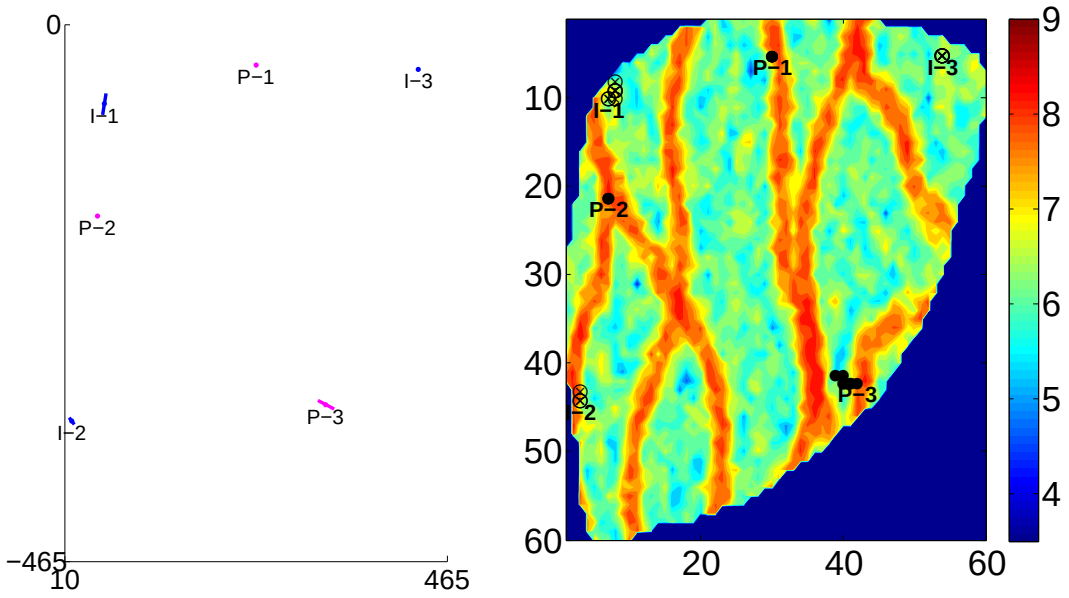


Figure 6.50: Average NPV vs. simulation runs for the joint optimization runs and the CMA-EnOpt sequential runs performed on case 2 with the set of bounds b2.

to generate the average NPV curves shown in Fig. 6.50 the final values of the converged runs were used as many times as required, e.g. if a run converged after 1900 simulation runs then its final value was used in the computation of the remaining 46 average NPV points corresponding to the well placement step. The reader is referred to Table 6.3 for a more accurate comparison of the algorithms where the average number of simulation runs used by each algorithm is shown.

The perforations and trajectories of the wells that correspond to the best case obtained by applying the joint optimization algorithm to case 2 with b2 bounds are shown in Fig. 6.51. The well controls obtained by the algorithm are shown in Fig. 6.52 while the liquid production rates and water cuts obtained by the application of these controls are shown in Fig. 6.53. By comparing the original well locations shown in Fig. 6.29(b) with the ones obtained by the joint optimization algorithm shown in Fig. 6.51, it can be seen that the wells have been pushed away from each other. The injectors are placed in three corners of the reservoir which helps to efficiently sweep the oil in the reservoir. The well P-3 is placed as far as possible from I-1 and is almost equidistant to wells I-2 and I-3 which allows it to produce large volumes of oil early in the reservoir (first third of reservoir life) without experiencing water breakthrough as seen in Fig. 6.53. After the water breakthrough occurs in well P-3

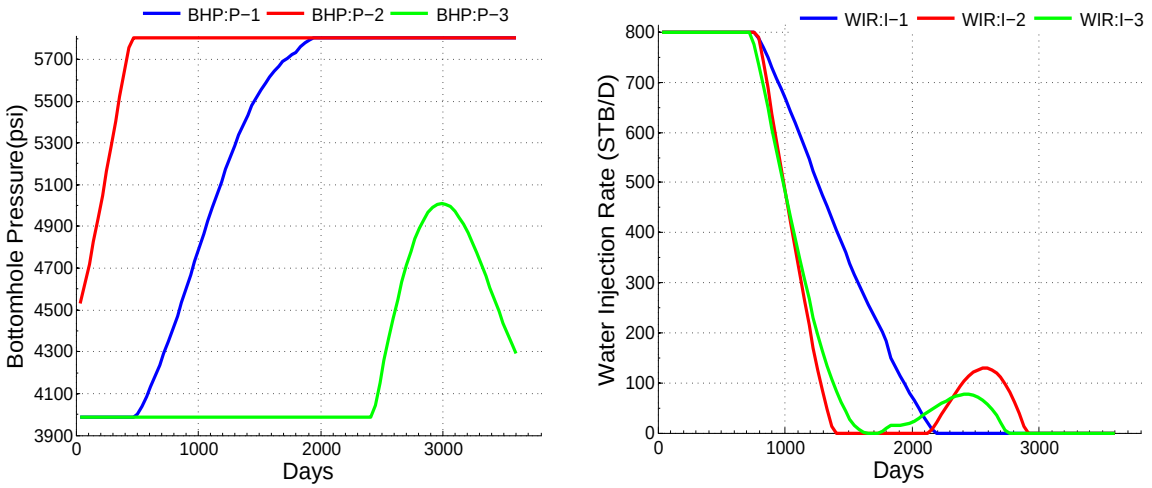


(a) Trajectories

(b) Perforations

Figure 6.51: The well trajectories and perforations corresponding to the solution with the highest NPV value for the joint optimization approach when used on case 2 with b2 bounds (seed 3) obtained by the CMA algorithm.

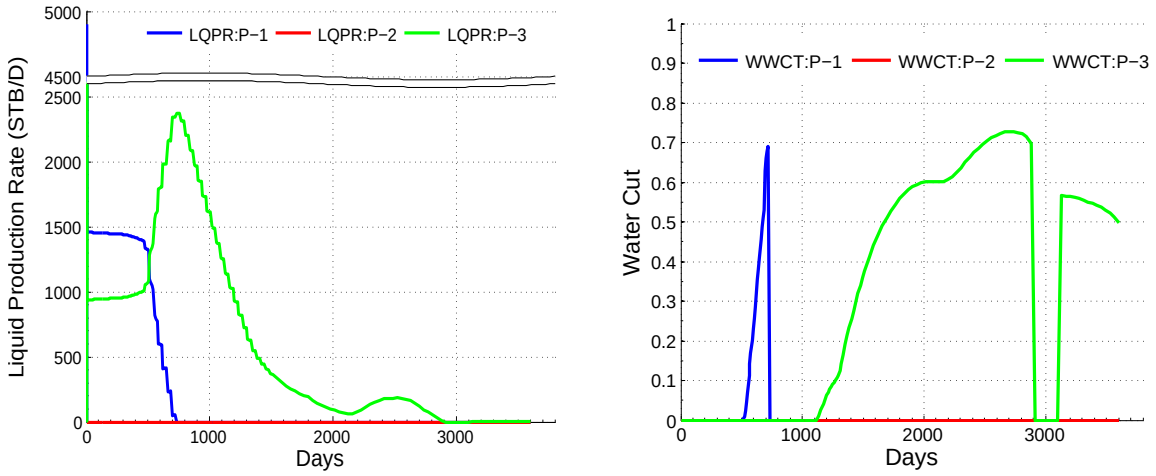
and the water cut becomes high the well is shut for around 200 days and later re-opened with a high BHP for the rest of the reservoir life in order to produce relatively small volumes of oil. In the case of P-1, the well is placed equidistantly to wells I-1 and I-3 and produces high volumes of liquid at the beginning of the reservoir life. However, due to the proximity of P-1 to I-1 and I-3, which inject at their maximum injection rates in the beginning of the reservoir life, the water breakthrough in P-1 occurs early and the water cut increases sharply, therefore, the well is shut after the first quarter of the reservoir life has passed. When the BHP of P-1 is increased the production of P-3 increases but this increase happens for a relatively short period of time (around 300 days) because the injectors are closed around the same time. The slight increase in the liquid production rate of P-3 at around 2,100 days from the beginning of production occurs due to the injection of water by wells I-2 and I-3 which are re-opened for a short period. The well P-2 is closed for the whole reservoir life and has no input on the final solution.



(a) Production wells

(b) Injection wells

Figure 6.52: The final well controls corresponding to the solution with the highest NPV value for the joint optimization approach when used on case 2 with b2 bounds (seed 3) obtained by CMA algorithm.

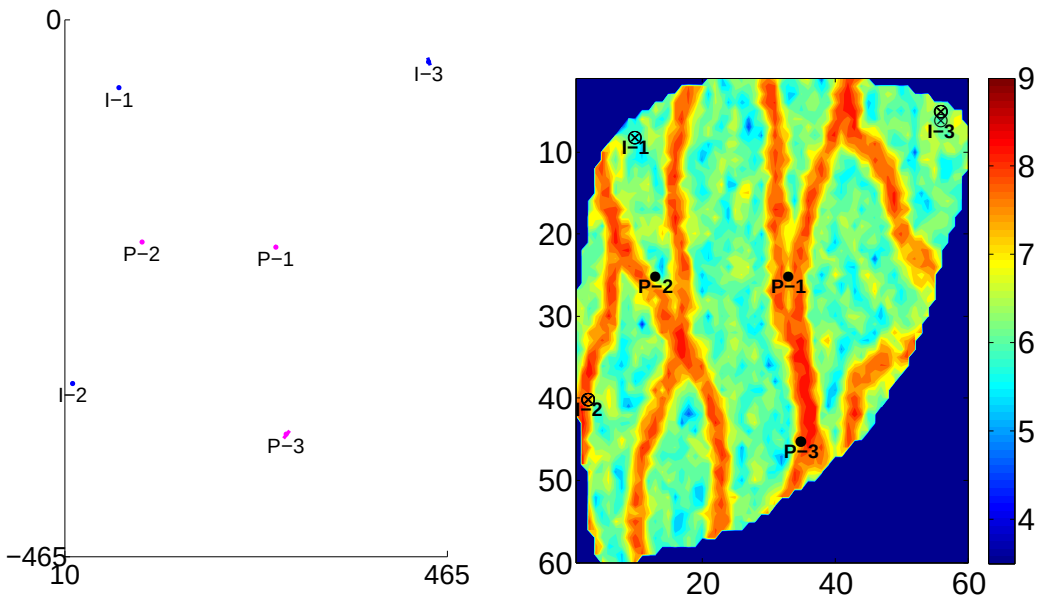


(a) Liquid production rate for producers

(b) Water cut for producers

Figure 6.53: The liquid production rates and water cuts obtained by applying the well controls shown in Fig. 6.52 to the wells shown in Fig. 6.51.

The perforations and trajectories corresponding to the best solution found by applying the CMA-EnOpt sequential algorithm with mean controls (in well placement step) to case 2 with b2 bounds are displayed in Fig. 6.54. The controls corresponding to these locations are shown in Fig. 6.55. The liquid production rates and water cuts obtained after applying the well controls in Fig. 6.55 to the wells in Fig. 6.54 are shown in Fig. 6.56. It is interesting to

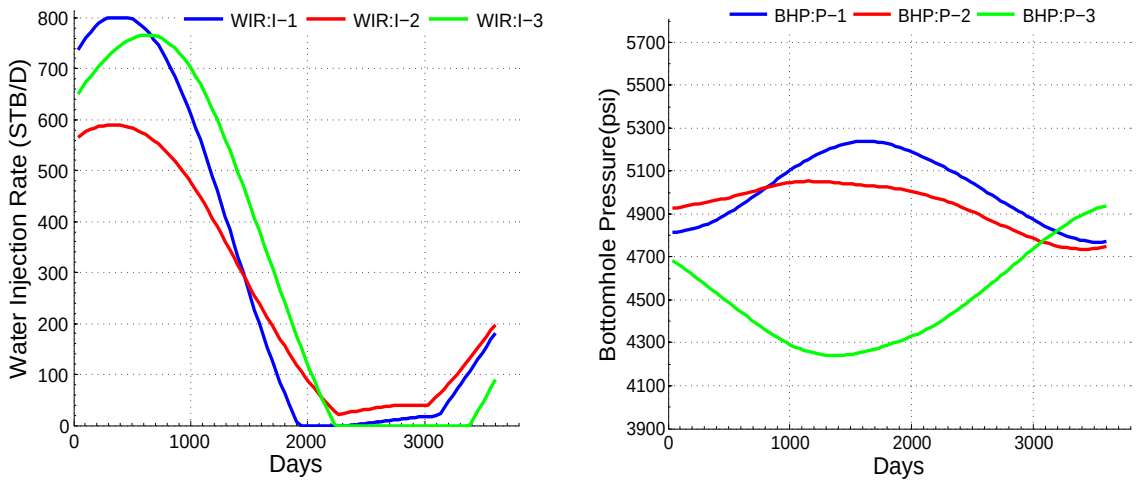


(a) Trajectories (b) Perforations

Figure 6.54: The well trajectories and perforations corresponding to the solution with the highest NPV value for the CMA-EnOpt sequential algorithm approach when used on case 2 with b2 bounds (seed 2) and the well controls are fixed at their mean values in the well placement step.

observe that the locations of the injectors in this solution are somewhat similar to the ones obtained in the joint optimization run in that wells are not close together. The production mainly occurs through well P-3 as the other two producers, P-1 and P-2, are kept at relatively high bottomhole pressures during all the reservoir life and have a shorter length than P-3, see Fig. 6.56(a) and Fig. 6.55(a). In the solution all the injectors inject at high rates at the beginning of the reservoir life and are closed after the first third of the reservoir life has passed. It seems that the strategy seeks to sweep most of the oil towards P-3 which produces at high rates for the first half of the reservoir life and then decrease the production rate of P-3 when its water cut becomes high.

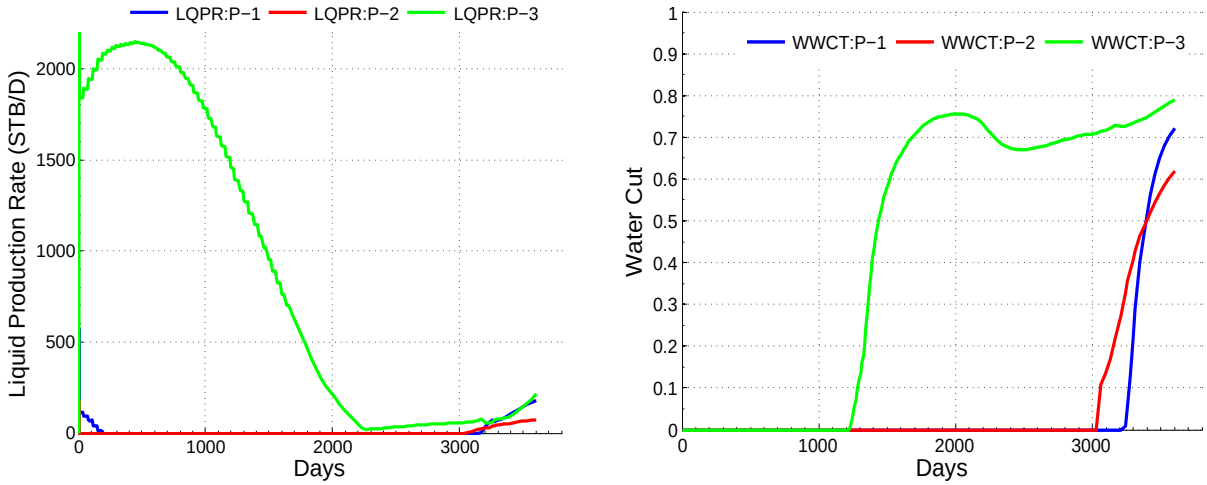
In Fig. 6.57 the perforations and trajectories of the wells corresponding to the best solution obtained by using the CMA-EnOpt sequential algorithm on case 2 with bounds b2 with maximum injection rate for injectors and minimum BHP for producers (during well placement) are shown. The controls corresponding to the solution are shown in Fig. 6.58 while the liquid production rates and the water cuts obtained by applying the controls are



(a) Production wells

(b) Injection wells

Figure 6.55: The well controls corresponding to the solution with the highest NPV value for the CMA-EnOpt sequential algorithm approach when used on case 2 with b2 bounds (seed 2) and the well controls are fixed at their mean values in the well placement step.



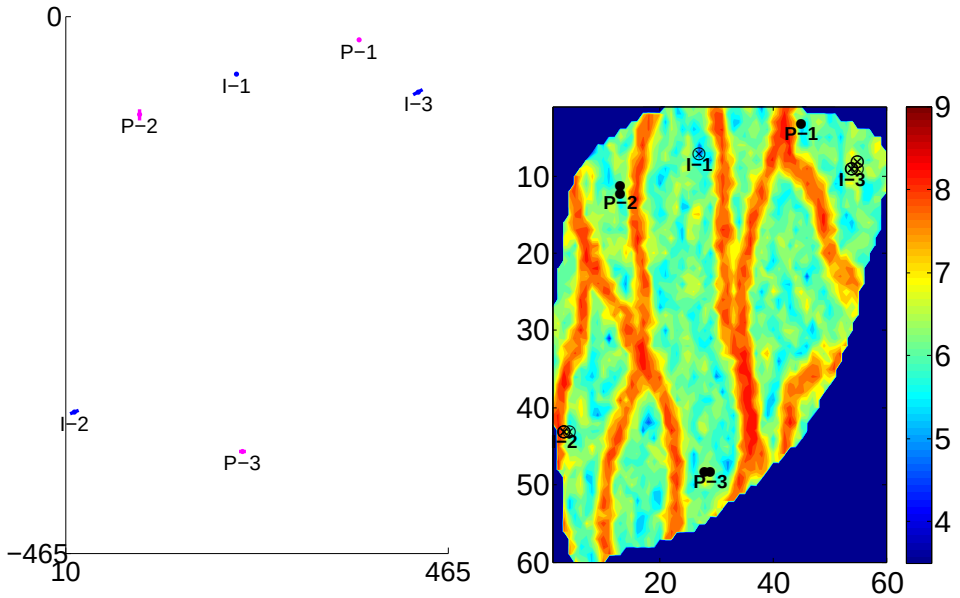
(a) Liquid production rate for producers

(b) Water cut for producers

Figure 6.56: The liquid production rates and water cuts obtained by applying the well controls shown in Fig. 6.55 to the wells shown in Fig. 6.54.

shown in Fig. 6.59. It should be mentioned that the well placement algorithm tried to decrease the production of high volumes of water by reducing the lengths of wells I-1 and P-1 to almost zero feet prior to the production optimization step. The injector I-1 injects a negligible amount of water for the maximum allowable BHP (6990 psi) compared to I-2 and I-3; see Fig. 6.59(b). A similar situation occurs with P-1 which produces a negligible

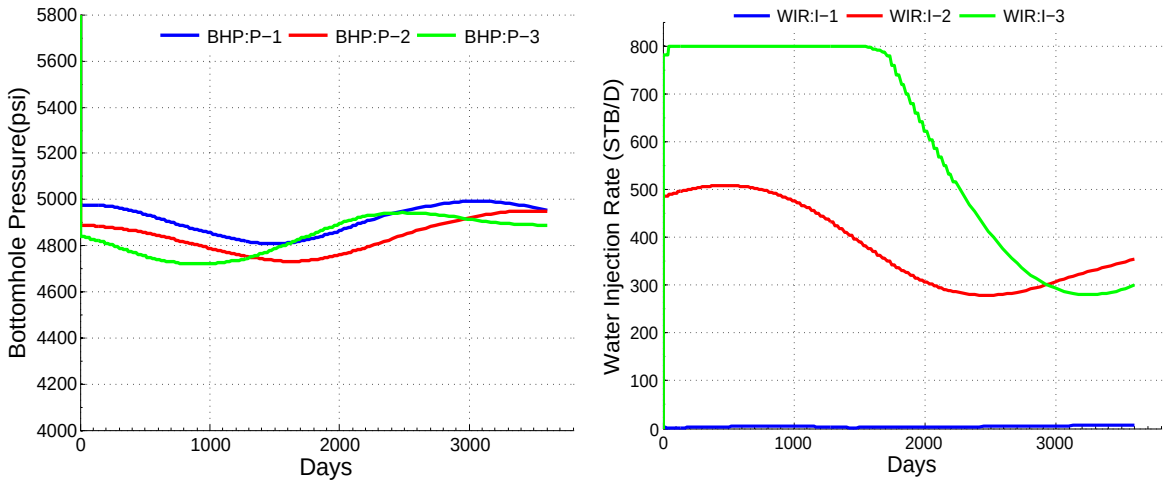
amount of liquid even when its BHP is at its lowest bound for the whole reservoir life. The well placement parameters obtained for I-1 and P-1 reduces the production optimization problem to finding the optimal set of controls for the other four wells. Similar to the previous solutions obtained in the optimization runs performed for the egg model, the strategy is to inject at high rates at the beginning of the reservoir life to later decrease the injection rate as the water breakthroughs in the producers. The BHP of the producers are kept close to the mean value between their upper and lower bounds for the whole reservoir life.



(a) Trajectories (b) Perforations

Figure 6.57: The well trajectories and perforations corresponding to the solution with the highest NPV value for the CMA-EnOpt sequential algorithm approach when used on case 2 with b2 bounds (seed 2) and the well controls are fixed at the upper bound for the injectors and at the lower bound for the producers.

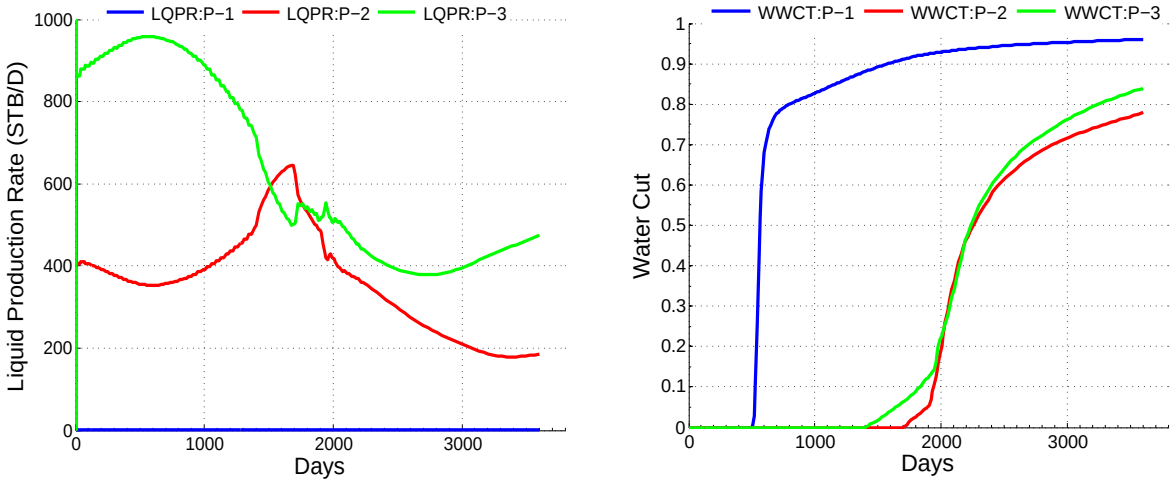
A final case was tested in which life-cycle production optimization was performed on the wells corresponding to Case 1 using the b1 bounds. The EnOpt algorithm was used for this case with an ensemble size of 20 and a maximum number of iterations set to 125. The reservoir life was partitioned into 100 control steps of 36 steps. A temporal correlation was imposed on the controls of each well with a correlation length of 50 control steps (5400 days). The well controls were set at their mean value initially. Three runs with different initial seeds were performed. The final average NPV obtained for this case was 1.048×10^8 while the



(a) Production wells

(b) Injection wells

Figure 6.58: The well controls corresponding to the solution with the highest NPV value for the CMA-EnOpt sequential algorithm approach when used on case 2 with b2 bounds (seed 2) and the well controls are fixed at the upper bound for the injectors and at the lower bound for the producers.



(a) Liquid production rate for producers

(b) Water cut for producers

Figure 6.59: The liquid production rates and water cuts obtained by applying the well controls shown in Fig. 6.58 to the wells shown in Fig. 6.57.

maximum NPV obtained was 1.049×10^8. The results of the runs corresponding to this case are summarized in Table 6.3. The results suggest that performing joint optimization (whether it is a simultaneous or a sequential approach) yields better results than optimizing the controls alone.

Table 6.3: The summary of the optimized NPV values with the simultaneous and sequential joint optimization algorithms for using the egg model.

NPV, \$ $\times 10^8$	Case	Bounds	Ave. Sim. Runs	Mean	Min	Max	SD
Proposed Algorithm	1	b1	2502	1.378	1.368	1.387	0.010
CMA-EnOpt mean	1	b1	2624	1.356	1.354	1.370	0.009
CMA-EnOpt bounds	1	b1	2633	1.349	1.332	1.371	0.020
Proposed Algorithm	1	b2	2502	1.505	1.474	1.538	0.032
CMA-EnOpt mean	1	b2	2639	1.342	1.285	1.439	0.085
CMA-EnOpt bounds	1	b2	2642	1.237	1.115	1.386	0.137
Proposed Algorithm	2	b2	2224	1.479	1.467	1.494	0.014
CMA-EnOpt mean	2	b2	1940	1.434	1.406	1.456	0.025
CMA-EnOpt bounds	2	b2	2283	1.301	1.258	1.351	0.047
EnOpt	1	b1	2864	1.048	1.046	1.049	0.002

CHAPTER 7

DISCUSSIONS AND CONCLUSIONS

In this thesis, simultaneous and sequential algorithms for the joint optimization of well trajectories and controls were investigated. The trajectory of a general directional well is considered as a three dimensional line which is parameterized in terms of a line in a spherical coordinate system using six parameters. In the simultaneous joint optimization algorithm, the vector of controls of a well is parameterized in terms of the singular vectors corresponding to the largest singular values of the temporal covariance matrix which is imposed on the controls of the well. Therefore, the number of optimization variables corresponding to the controls of the well is significantly reduced and the final well controls obtained by this algorithm are very smooth. An implementation of the Covariance Matrix Adaptation algorithm is used to solve the joint optimization problem in both sequential and simultaneous algorithms. The results obtained from these tests are compared to the CMA-EnOpt sequential algorithm where CMA-ES is used to solve a well placement optimization problem, where the well controls are kept fixed, followed by a production optimization step that uses the standard EnOpt algorithm with fixed well locations (the locations obtained in the well placement step). The performance of CMA-ES algorithm is not compared to other stochastic evolutionary algorithms in this work, however based on previous experimentation with other algorithms, the CMA-ES performance is superior to that of particle swarm (PSO) and differential evolution (DE) optimization algorithms for the proposed joint optimization algorithm. This might be due to the intensive randomness of these other algorithms compared to the CMA-ES.

As mentioned before, the proposed algorithm for parameterizing the well controls imposes some degree of smoothness on the controls of a well. Although imposing a tem-

poral smoothness on the well controls is generally a good strategy which could improve the performance of the production optimization algorithms (e.g., EnOpt and G-SPSA), if too much smoothness is imposed, it could result in obtaining sub-optimal solutions for the well controls [30]. Oliveira and Reynolds [41] proposed hierarchial multi-scale algorithms to avoid over smoothing well controls but that methodology is not considered here. Moreover, the strategy presented to parameterize the well controls in terms of only the singular vectors corresponding to the largest singular values of the covariance matrix results in the reduction of the degrees of freedom for the well controls which generally may result in a sub-optimal solution. A disadvantage of only retaining a small portion of the singular values of the temporal covariance matrix is that bang-bang solutions are not possible. Therefore, the proposed algorithm cannot completely replace the production optimization step which is normally performed after determining the optimum well locations. However, this algorithm tries to mitigate the effect of the specified well controls on the well placement step.

The proposed algorithm is tested for optimizing the trajectories and controls of injectors and producers in the PUNQ and egg reservoir models. In the PUNQ reservoir model the objective was to find the optimal trajectories and controls for three injection wells and six production wells while in the egg reservoir model two different cases were tested: one where the trajectories and controls were optimized for eight injectors and four producers and another one where the trajectories and controls were optimized for three injectors and three producers. In order to perform the tests on the egg reservoir model the reservoir was bounded by vertical planes that separated the active gridblocks from the non-active ones. The area enclosed by these vertical planes was transformed into linear inequality constraints that were imposed on the well trajectory parameters. These constraints were handled by sampling candidates only within the convex region defined by the vertical planes during the sampling step of the CMA-ES algorithm.

In this work, it was considered that the reservoir model is known, i.e., the uncertainties in the reservoir description were not considered. The results showed superior performance for the simultaneous joint optimization algorithm compared to the sequential frameworks

in all the cases presented. Moreover, the CMA-ES algorithm was applied to solve the life-cycle production optimization problem for the given number and locations of the wells in the PUNQ reservoir model. The results show that when the initial covariance matrix in the CMA-ES algorithm is obtained from imposing a temporal correlation on the controls of a well, the CMA-ES algorithm for production optimization outperformed the standard EnOpt algorithm in average.

BIBLIOGRAPHY

[1] W. Bangerth, H. Klie, M. F. Wheeler, P. L. Stoffa, and M. K. Sen. On optimization algorithms for the reservoir oil well placement problem. *Computational Geosciences*, 10(3):303–319, 2006.

[2] Mathias C. Bellout, David Echeverra Ciaurri, Louis J. Durlofsky, Bjarne Foss, and Jon Kleppe. Joint optimization of oil well placement and controls. *Computational Geosciences*, 16(4):1061–1079, 2012.

[3] H. G. Beyer. *The Theory of Evolutions Strategies*. Natural Computing Series. Springer, 2001.

[4] H. G. Beyer and K. Deb. On self-adaptive features in real-parameter evolutionary algorithms. *Evolutionary Computation, IEEE Transactions on*, 5(3):250–270, Jun 2001.

[5] Zyed Bouzarkouna, Didier Ding, and Anne Auger. Well placement optimization with the covariance matrix adaptation evolution strategy and meta-models. *Computational Geosciences*, 16:75–92, 2012.

[6] D. R. Brouwer and J. D. Jansen. Dynamic optimization of water flooding with smart wells using optimial control theory. *SPE Journal*, 9(4):391–402, 2004.

[7] Chaohui Chen, Gaoming Li, and Albert C. Reynolds. Closed-loop reservoir management on the Brugge test case. *Computational Geosciences*, 14(4):691–703, 2010.

[8] Yan Chen, Dean S. Oliver, and Dongxiao Zhang. Efficient ensemble-based closed-loop production optimization. *SPE Journal*, 14(4):634–645, 2009.

[9] D. Echeverra Ciaurri, O. J. Isebor, and L. J. Durlofsky. Application of derivative-free methodologies to generally constrained oil production optimization problems. *Procedia Computer Science*, 1(1):1301 – 1310, 2010.

[10] Sy T. Do and Albert C. Reynolds. Theoretical connections between optimization algorithms based on an approximate gradient. *Computational Geosciences*, 17(6):959–973, 2013.

[11] Alexandre A. Emerick, E. Silva, B. Messer, L. Almeida, D. Szwarcman, M. Pacheco, and M. Vellasco. Well placement optimization using a genetic algorithm with nonlinear constraints. In *Proceedings of the SPE Reservoir Simulation Symposium*, 2009. Paper SPE 118808.

[12] M. Evans, N. Hastings, and B. Peacock. *Statistical Distributions*. Wiley, 3rd edition, 2000.

[13] F. J. T. Floris, M. D. Bush, M. Cuypers, F. Roggero, and A. R. Syversveen. Comparison of production forecast uncertainty quantification methods - an integrated study. *Petroleum Geoscience*, 7:S87–S96, March 2001.

[14] R. M. Fonseca, O. Leeuwenburgh, P. M. J. Van den Hof, and J. D. Jansen. Improving the ensemble optimization method through covariance matrix adaptation (CMA-EnOpt). In *Proceedings of the SPE Reservoir Simulation Symposium*, 2013. Paper SPE 163657.

[15] Fahim Forouzanfar, Gaoming Li, and Albert C. Reynolds. A two-stage well placement optimization method based on adjoint gradient, SPE-135304. In *Proceedings of the 2010 SPE Annual Technical Conference and Exhibition, 20-22 September 2010, Florence, Italy*, 2010.

[16] Fahim Forouzanfar and Albert C. Reynolds. Well-placement optimization using a derivative-free method. *Journal of Petroleum Science and Engineering*, 109(0):96 – 116, 2013.

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

[18] Fahim Forouzanfar, Albert C. Reynolds, and Gaoming Li. Optimization of the well locations and completions for vertical and horizontal wells using a derivative-free optimization algorithm. *Journal of Petroleum Science and Engineering*, 86-87:272–288, 2012.

[19] S. Geman and Geman D. Stochastic relaxation, gibbs distributions , and the bayesian restoration of images. *IEEE Transactions On Pattern Analysis and Machine Intelligence*, 6(6):721–741, 1984.

[20] M. Handels, M. J. Zandvliet, D. R. Brouwer, and J. D. Jansen. Adjoint-based well-placement optimization under production constraints. In *Proceedings of the SPE Reservoir Simulation Symposium*, number SPE 105797, 2007.

[21] Nikolaus Hansen. The CMA evolution strategy: A tutorial, 2011.

[22] Nikolaus Hansen, Andre S. P. Niederberger, Lino Guzzella, and Petros Koumoutsakos. Evolutionary optimization of feedback controllers for thermoacoustic instabilities. In J.F. Morrison, D.M. Birch, and P. Lavoie, editors, *IUTAM Symposium on Flow Control and MEMS*, volume 7 of *IUTAM Bookseries*, pages 311–317. Springer Netherlands, 2008.

[23] Nikolaus Hansen and Andreas Ostermeier. Completely derandomized self-adaptation in evolution strategies. *Evol. Comput.*, 9(2):159–195, June 2001.

[24] John H. Holland. *Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology, Control and Artificial Intelligence*. MIT Press, Cambridge, MA, USA, 1992.

[25] Thomas D. Humphries, Ronald D. Haynes, and Lesley A. James. Simultaneous and sequential approaches to joint optimization of well placement and control. *Computational Geosciences*, pages 1–16, 2013.

[26] Obiajulu J. Isebor, David E. Ciaurri, and Louis J. Durlofsky. Generalized field development optimization using derivative-free procedures. In *Proceedings of the SPE Reservoir Simulation Symposium*, number SPE 163631, 2013.

[27] B. Jafarpour and L. Li. Generalized field development optimization: Coupled well-placement and control under geologic uncertainty. In *Proceedings of the ECMOR XIII - 13th European Conference on the Mathematics of Oil Recovery*, 2012.

[28] J. D. Jansen, D. R. Brouwer, G. Naevdal, and C. P. J. W. van Kruijsdijk. Closed-loop reservoir management. *First Break*, 23:43–48, 2005.

[29] J. D. Jansen, R. M. Fonseca, S. Kahrobaei, M. M. Siraj, G. M. Van Essen, and P. M. J. Van den Hof. The egg model a geological ensemble for reservoir simulation. *Geoscience Data Journal*, 1(2):192–195, 2014.

[30] Jan Dirk Jansen, Okko H. Bosgra, and Paul M. J. Van den Hof. Model-based control of multiphase flow in subsurface oil reservoirs. *Journal of Process Control*, 18(9):846 – 855, 2008.

[31] Muller Christian L. Continuous black-box optimization in linearly constrained domains using efficient gibbs sampling. Technical report, ETH Zurich, 2011.

[32] Robert Michael Lewis, Virginia Torczon, and Robert Michael Lewis. Rank ordering and positive bases in pattern search algorithms. Technical report, Institute for Computer, 1996.

[33] Gaoming Li and Albert C. Reynolds. Uncertainty quantification of reservoir performance predictions using a stochastic optimization algorithm. *Computational Geosciences*, 15(3):451–462, 2011.

[34] Lianlin Li and Behnam Jafarpour. A variable-control well placement optimization for improved reservoir development. *Computational Geosciences*, pages 1–19, 2012.

[35] Lianlin Li, Behnam Jafarpour, and M. Reza Mohammad-Khaninezhad. A simultaneous perturbation stochastic approximation algorithm for coupled well placement and control optimization under geologic uncertainty. *Computational Geosciences*, 17(1):167–188, 2013.

[36] I. Loshchilov. CMA-ES with restarts for solving CEC 2013 benchmark problems. In *Evolutionary Computation (CEC), 2013 IEEE Congress on*, pages 369–376, June 2013.

[37] N. S. Mera. Passive gamma tomography reconstruction of layered structures in nuclear waste vaults. *Inverse Problems*, 23:385–403, February 2007.

[38] Geir Nævdal, D. Roald Brower, and Jan-Dirk Jansen. Waterflooding using closed-loop control. *Computational Geosciences*, 10(1):37–60, 2006.

[39] Jorge Nocedal and Stephen J. Wright. *Numerical Optimization*. Springer, New York, 2006.

[40] E. Nwankwor, A. K. Nagar, and D. C. Reid. Hybrid differential evolution and particle swarm optimization for optimal well placement. *Computational Geosciences*, 17(2):249–268, 2013.

[41] Diego Felipe Barbosa Oliveira and Albert C. Reynolds. An adaptive hierarchical algorithm for estimation of optimal well controls. In *Proceedings of the SPE Reservoir Simulation Symposium*, number SPE 163645, 2013.

[42] Dean S. Oliver, Albert C. Reynolds, and Ning Liu. *Inverse Theory for Petroleum Reservoir Characterization and History Matching*. Cambridge University Press, Cambridge, UK, 2008.

[43] Jerome Onwunalu and Louis Durlofsky. Application of a particle swarm optimization algorithm for determining optimum well location and type. *Computational Geosciences*, 14:183–198, 2010.

[44] Umut Ozdogan and Roland N. Horne. Optimization of well placement under time-dependent uncertainty. *SPE Reservoir Evaluation & Engineering*, 9(2):135–145, 2006.

[45] D. W. Peaceman. Interpretation of well block pressures in numerical reservoir simulation. *SPE Journal*, 18(6):183–194, 1978.

[46] D. W. Peaceman. Interpretation of well-block pressures in numerical reservoir simulation with non-square grid blocks and anisotropic permeability. *SPE Journal*, 23(6):531–543, 1983.

[47] G. Rodriguez-Yam, R. A. Davis, and L. L. Scharf. Efficient Gibbs sampling of truncated multivariate nomal with application to constrained linear regression. Technical report, Colorado State University, 2004.

[48] H. H. Rosenbrock. An automatic method for finding the greatest or least value of a function. *The Computer Journal*, 3:175–184, 1960.

[49] P. Sarma, W. H. Chen, L. J. Durlofsky, and K. Aziz. Production optimization with adjoint models under nonlinear control-state path inequality constraints. *SPE Reservoir Evaluation & Engineering*, 11(2):326–339, 2008.

[50] Schlumberger. *Schedule, ECLIPSE reservoir simulation software*. Schlumberger, 2013.

[51] Y. Shi and R.C. Eberhart. A modified particle swarm optimizer. In *proceedings of the IEEE Conference on Evolutionary Computation. AK, Anchorage*, 1998.

[52] James C. Spall. Multivariate stochastic approximation using a simulataneous perturbation gradient approximation. *IEEE Transactions Automat. Control.*, 37(3):332–341, 1992.

[53] G. M. van Essen, M. J. Zandvliet, P. M. J. Van den Hof, O. H. Bosgra, and J. D. Jansen. Robust waterflooding optimization of multiple geological scenarios. *SPE Journal*, 14(1):202–210, 2009.

[54] S. Vlemmix, G. J. P. Joosten, D. R. Brouwer, and J. D. Jansen. Adjoint-based well trajectory optimization in a thin oil rim, SPE-121891. In *Proceedings of the SPE EUROPEC/EAGE Annual Conference an Exhibition, 8-11 June 2009, Amsterdam, The Netherlands*, 2009.

[55] Chunhong Wang, Gaoming Li, and A. C. Reynolds. Optimal well placement for production optimization. In *Proceedings of the SPE Eastern Reginal Meeting*, number SPE 111154, 2007.

[56] Jesse W. Wilson, Philip Schlup, Monte Lunacek, Darrell Whitley, and Randy A. Bartels. Calibration of liquid crystal ultrafast pulse shaper with common-path spectral interferometry and application to coherent control with a covariance matrix adaptation evolutionary strategy. *Review of Scientific Instruments*, 79(3):–, 2008.

[57] Burak Yeten, Louis J. Durlofsky, and Khalid Aziz. Optimization of nonconventional well type, location and trajectory. In *Proceedings of the 2002 SPE Annual Technical Conference and Exhibition, 29 September-2 October 2002, San Antonio, Texas*, 2002.

[58] Maarten Zandvliet, Martijn Handels, Gijs van Essen, Roald Brouwer, and Jan-Dirk Jansen. Adjoint-based well-placement optimization under production constraints. *SPE Journal*, 13(4):392–399, 2008.

[59] Hui Zhao, Chaohui Chen, Sy Do, Diego Oliveira, Gaoming Li, and Albert Reynolds. Maximization of a dynamic quadratic interpolation model for production optimization. *SPE Journal*, 18(4):1012–1025, 2013.

APPENDIX A

UNBIASEDNESS OF THE UPDATE OPERATORS IN THE CMA-ES ALGORITHM

The notion of unbiasedness of the update refers to the requirement in evolution strategies, explained in [4] by Beyer, that operations which generate variations of the parameters should not change the expected value of the parameter from one generation to next, e.g. $\mathrm{E}\left[\boldsymbol{C}^{(g+1)}|\boldsymbol{C}^{(g)}\right] = \boldsymbol{C}^{(g)}$. This stems from the fact that evolutionary algorithms consists of two basic operations: selection and recombination. The selection operator, as the name suggests, selects the parents for the next generation out of a pool of candidates while the variation operator uses these parents to create new candidates. Their roles differ in that the variation operator is intended to search the solution space without using any fitness information while the selection operator exploits the information gained by the fitness information. If there is a bias in the variation process, created by the use of the fitness information, then there is a danger of converging prematurely because the parameters of the distribution will not be able to explore the search space effectively. For example, if the mutation operator that controls the change in the overall variance of the distribution, σ, has a bias towards decreasing its value, the variance of the distribution would have a high probability of decreasing with each step of the algorithm independently of the manner in which parents of the new population are selected [21]. In other words, the updates should not have a bias towards a particular value (or values in the case of covariance matrices and vectors) because this would mean that independently of the parents that the algorithm selects the update already has a built-in tendency to shift the parameter that needs to be updated towards a particular value. The proofs for the unbiasedness of each of the updates are given below.

The proof of unbiasedness of the mean update can be shown by calculating the ex-

pected value of $\boldsymbol{m}^{(g+1)}$ given $\boldsymbol{m}^{(g)}$. Using the fact that $\boldsymbol{v}_{i:\lambda}^{(g+1)} \sim \mathcal{N}\left(\boldsymbol{m}^{(g)}, {\sigma^{(g)}}^2 \boldsymbol{C}^{(g)}\right)$,

$$\begin{aligned} \mathrm{E}\left[\boldsymbol{m}^{(g+1)} | \boldsymbol{m}^{(g)}\right] &= \mathrm{E}\left[\sum_{i=1}^{\mu} w_i \boldsymbol{v}_{i:\lambda}^{(g+1)}\right], \\ &= \sum_{i=1}^{\mu} w_i \mathrm{E}\left[\boldsymbol{v}_{i:\lambda}^{(g+1)}\right], \\ &= \sum_{i=1}^{\mu} w_i \boldsymbol{m}^{(g)}, \\ &= \boldsymbol{m}^{(g)}. \end{aligned} \tag{A.1}$$

Next it is shown that the unbiasedness condition for the rank-μ update is satisfied, i.e., $\mathrm{E}\left[\boldsymbol{C}^{(g+1)} | \boldsymbol{C}^{(g)}\right] = \boldsymbol{C}^{(g)}$. First,

$$\begin{aligned} \mathrm{E}\left[\boldsymbol{C}^{(g+1)} | \boldsymbol{C}^{(g)}\right] &= \mathrm{E}\left[(1 - c_\mu)\boldsymbol{C}^{(g)}\right] + \mathrm{E}\left[c_\mu \frac{\boldsymbol{C}_\mu^{(g+1)}}{{\sigma^{(g)}}^2}\right] \\ &= (1 - c_\mu)\boldsymbol{C}^{(g)} + c_\mu \mathrm{E}\left[\frac{\boldsymbol{C}_\mu^{(g+1)}}{{\sigma^{(g)}}^2}\right]. \end{aligned} \tag{A.2}$$

To complete the proof, it is only necessary to show that $\mathrm{E}\left[\frac{\boldsymbol{C}_\mu^{(g+1)}}{{\sigma^{(g)}}^2}\right] = \boldsymbol{C}^{(g)}$ as the linear combination of this term with the other term in the right hand side of Eq. A.2 then yields the desired result $\mathrm{E}\left[\boldsymbol{C}^{(g+1)} | \boldsymbol{C}^{(g)}\right] = \boldsymbol{C}^{(g)}$. Using $\boldsymbol{z} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I})$ and the fact that $\left(\frac{\boldsymbol{v}_{i:\lambda}^{(g+1)} - \boldsymbol{m}^{(g)}}{\sigma^{(g)}}\right) \sim \mathcal{N}\left(\boldsymbol{0}, \boldsymbol{C}^{(g)}\right)$ (see Eq. 5.1), it follows that

$$
\begin{aligned}
\mathrm{E}\left[\frac{\boldsymbol{C}_{\mu}^{(g+1)}}{\sigma^{(g)^2}}\right] &= \mathrm{E}\left[\sum_{i=1}^{\mu} w_i \left(\frac{\boldsymbol{v}_{i:\lambda}^{(g+1)} - \boldsymbol{m}^{(g)}}{\sigma^{(g)}}\right)\left(\frac{\boldsymbol{v}_{i:\lambda}^{(g+1)} - \boldsymbol{m}^{(g)}}{\sigma^{(g)}}\right)^T\right] \\
&= \mathrm{E}\left[\sum_{i=1}^{\mu} w_i \left(\frac{\boldsymbol{v}_{i:\lambda}^{(g+1)} - \boldsymbol{m}^{(g)}}{\sigma^{(g)}}\right)\left(\frac{\boldsymbol{v}_{i:\lambda}^{(g+1)} - \boldsymbol{m}^{(g)}}{\sigma^{(g)}}\right)^T\right] \\
&= \mathrm{E}\left[\sum_{i=1}^{\mu} w_i \left(\boldsymbol{C}^{(g)^{1/2}}\boldsymbol{z}_{i:\lambda}\right)\left(\boldsymbol{C}^{(g)^{1/2}}\boldsymbol{z}_{i:\lambda}\right)^T\right] \\
&= \mathrm{E}\left[\sum_{i=1}^{\mu} w_i \boldsymbol{C}^{(g)^{1/2}}\boldsymbol{z}_{i:\lambda}\boldsymbol{z}_{i:\lambda}^T\left(\boldsymbol{C}^{(g)^{1/2}}\right)^T\right] \\
&= \mathrm{E}\left[\left(\boldsymbol{C}^{(g)^{1/2}}\right)\left(\sum_{i=1}^{\mu} w_i \boldsymbol{z}_{i:\lambda}\boldsymbol{z}_{i:\lambda}^T\right)\left(\boldsymbol{C}^{(g)^{1/2}}\right)^T\right] \\
&= \left(\boldsymbol{C}^{(g)^{1/2}}\right)\left(\sum_{i=1}^{\mu} w_i \mathrm{E}\left[\boldsymbol{z}_{i:\lambda}\boldsymbol{z}_{i:\lambda}^T\right]\right)\left(\boldsymbol{C}^{(g)^{1/2}}\right)^T \\
&= \left(\boldsymbol{C}^{(g)^{1/2}}\right)\left(\sum_{i=1}^{\mu} w_i \boldsymbol{I}\right)\left(\boldsymbol{C}^{(g)^{1/2}}\right)^T \\
&= \left(\boldsymbol{C}^{(g)^{1/2}}\right)(\boldsymbol{I})\left(\boldsymbol{C}^{(g)^{1/2}}\right)^T \\
&= \boldsymbol{C}^{(g)}.
\end{aligned}
\tag{A.3}
$$

Note that the derivation of A.3 used $\mathrm{E}\left[\boldsymbol{z}_{i:\lambda}\boldsymbol{z}_{i:\lambda}^T\right] = \boldsymbol{I}$ which follows from the fact that the components of $\boldsymbol{z}_{i:\lambda}$ are independent standard random normal deviates.

The unbiasedness of the rank-one update can be proved by following the same steps as the ones shown in Eq. A.3 because the outer product $\boldsymbol{p}_c^{(g+1)}\boldsymbol{p}_c^{(g+1)^T}$ has the same distribution as the term $\frac{\boldsymbol{C}_{\mu}^{(g+1)}}{\sigma^{(g)^2}}$.

In the case of the step size, the unbiasedness condition is only satisfied through $\ln \sigma^{(g+1)}$ as shown below:

$$
\begin{aligned}
\ln \sigma^{(g+1)} &= \ln\left[\sigma^{(g)} \exp\left(\frac{c_\sigma}{d_\sigma}\left(\frac{\|\boldsymbol{p}_\sigma^{(g+1)}\|}{\mathrm{E}[\|\boldsymbol{z}\|]} - 1\right)\right)\right] \\
&= \ln \sigma^{(g)} + \left(\frac{c_\sigma}{d_\sigma}\left(\frac{\|\boldsymbol{p}_\sigma^{(g+1)}\|}{\mathrm{E}[\|\boldsymbol{z}\|]} - 1\right)\right).
\end{aligned}
\tag{A.4}
$$

By taking the conditional expectation of the terms in Eq. A.4 given $\sigma^{(g)}$, it follows that

$$\begin{aligned}
\mathrm{E}[\ln \sigma^{(g+1)} | \sigma^{(g)}] &= \mathrm{E}[\ln \sigma^{(g)}] + \mathrm{E}\left[\left(\frac{c_\sigma}{d_\sigma}\left(\frac{\|\boldsymbol{p}_\sigma^{(g+1)}\|}{\mathrm{E}[\|\boldsymbol{z}\|]} - 1\right)\right)\right] \\
&= \ln \sigma^{(g)} \ + \left(\frac{c_\sigma}{d_\sigma}\left(\frac{\mathrm{E}[\|\boldsymbol{p}_\sigma^{(g+1)}\|] - \mathrm{E}[\|\boldsymbol{z}\|]}{\mathrm{E}[\|\boldsymbol{z}\|]}\right)\right) \\
&= \ln \sigma^{(g)} \ + \left(\frac{c_\sigma}{d_\sigma}\left(\frac{\mathrm{E}[\|\boldsymbol{p}_\sigma^{(g+1)}\|] - \mathrm{E}[\|\boldsymbol{z}\|]}{\mathrm{E}[\|\boldsymbol{z}\|]}\right)\right)
\end{aligned} \tag{A.5}$$

and since both $\boldsymbol{p}_\sigma^{(g+1)}$ and $\boldsymbol{z}$ have the same distribution they have the same expectation and cancel each other

$$\begin{aligned}
&= \ln \sigma^{(g)} \ + \left(\frac{c_\sigma}{d_\sigma}\left(\frac{0}{\mathrm{E}[\|\boldsymbol{z}\|]}\right)\right) \\
&= \ln \sigma^{(g)}.
\end{aligned}$$