POLITECNICO
MILANO 1863

SCUOLA DI INGEGNERIA INDUSTRIALE
E DELL'INFORMAZIONE

Integrating machine learning and derivative-free optimizers for oil production optimization by water-flooding

Tesi di Laurea Magistrale in
Energy Engineering - Ingegneria Energetica

Author: **Guido Di Federico**

Student ID: 947540
Advisor: Prof. Enrico Zio
Co-advisors: Dr. Ahmed Shokry, Dr. Giorgio Fighera and Dr. Emanuele Vignati
Academic Year: 2020-21

Abstract

The first stage of primary production from oil fields is driven by the natural pressure of reservoir fluids, which inevitably drops after a period of time. Then, a secondary production phase is initiated, in which an external fluid is pumped underground to improve oil recovery. The most common method for this secondary production is waterflooding, which consists in injecting water from wells to displace hydrocarbons. In waterflooding, the optimization of the water injection schedule is essential to maximize the economic production of the field, while reducing wastes and emissions.

Waterflooding optimization can be performed by field engineers and operators, supported by surveillance methods, and by using numerical optimization with physics-based reservoir models. Surveillance methods allow operators to drive their decisions through the analysis of production data and decline curves. This way is heavily dependent on the operators' experience and, in most cases, leads to suboptimal policies, especially over long-term operation. On the other hand, simulation-based optimization is difficult to apply in practice because of the huge computational cost required to repeatedly run the complex reservoir models and also the time and effort required to build and tune the models itself. This becomes even more challenging for mature fields, which have undergone many years of production of a high number of wells. Actually, to reduce the computational burden, many studies suggest the use surrogate-based optimization (SBO) methods, in which the physics-based reservoir model is replaced by a data-driven model of fast execution. Many studies employ shallow artificial neural networks (ANNs) as simple static function evaluators, approximating the mapping between the decision variables and optimization criteria, but lacking of information about reservoir behavior and not providing engineering interpretations of the obtained optimal decisions.

This thesis presents an innovative SBO framework for waterflooding management in mature oilfields, which integrates machine learning (ML) models, such as long short-term memory (LSTMs) and physics informed neural networks (PINNs), with optimization techniques, like ensemble-based and genetic algorithms. The framework builds a ML model that approximates a complex physics-based reservoir model to accurately predict the future state of the reservoir in terms of the wells' water and oil production profiles, as a

function of the water injection profiles. Then, the ML-based reservoir model is integrated within an optimization algorithm to search for the optimal injection profiles, considering the net present value (NPV) as objective function while satisfying different operational and economic constraints.

The effectiveness of the proposed framework is validated through its application to two case studies. The first considers a 2D reservoir model with homogeneous geological properties, including 4 production and 5 injection wells. PINNs are used as forward models to integrate physical constraints during training. The second is the well-known case of the Olympus field, which is a 3D reservoir model with complex geological properties, including 7 injection and 11 production wells. In this case, LSTMs are used as forward models, since time needs to be integrated explicitly into the forecast. The optimal injection schedules provided by the proposed method are validated by comparing them to those recommended by standard reservoir practices and state-of-the-art software. The results show a good prediction accuracy, significant reduction in the required computational time and, even more, provide high-level operational recommendations and guidelines.

Keywords: oil production, waterflooding, reservoir simulation, optimization, machine learning, PINN, LSTM, genetic algorithms

Abstract in lingua italiana

La prima fase di produzione dai giacimenti di petrolio è sostenuta dalla pressione dei fluidi di giacimento, che, dopo un periodo di tempo, inevitabilmente diminuisce. È necessaria quindi una fase secondaria di produzione, che prevede l'iniezione di un fluido esterno, iniettato nel sottosuolo al fine di incrementare il recupero di idrocarburi. Il metodo più comune è il *waterflooding*, che consiste nell'iniettare acqua da alcuni pozzi per facilitare l'estrazione del petrolio. L'ottimizzazione del programma di iniezione è essenziale per massimizzare il rendimento economico del giacimento e ridurre gli sprechi, nonché limitare le emissioni.

L'ottimizzazione del processo di *waterflooding* può avvenire o mediante metodi tradizionali di sorveglianza del giacimento, cioè tramite azione diretta degli ingegneri e operatori, oppure con l'ausilio di tecniche di ottimizzazione numeriche, basate sulla simulazione di modelli fisici del giacimento. I metodi di sorveglianza, in cui gli operatori prendono decisioni sulla base dei dati di produzione, dipendono fortemente dalla loro esperienza e possono condurre a una gestione subottimale, specialmente sul lungo termine. D'altro canto, l'ottimizzazione basata su un simulatore è complessa da realizzare nella pratica, sia a causa dell'alto costo computazionale che deriva dalla complessità dei modelli di giacimento, sia a causa del tempo e delle risorse necessarie per costruire e calibrare i modelli stessi. Ciò è particolarmente rilevante per i giacimenti maturi, con molti anni di storia produttiva e numerosi pozzi. Per limitare i tempi computazionali, diversi studi suggeriscono l'uso di modelli surrogati per l'ottimizzazione (SBO), che sostituiscono il modello fisicamente basato con un modello basato sui dati di produzione (*data-driven*), con tempi computazionali ridotti. La maggioranza di tali studi sfrutta reti neurali artificiali (ANN) tradizionali come semplici approssimazioni della relazione tra le variabili di controllo e i criteri di ottimizzazione, senza tuttavia fornire informazioni sul comportamento del giacimento o interpretazioni ingegneristiche delle soluzioni ottimali ottenute.

Questa tesi presenta una metodologia innovativa SBO per la gestione del processo di *waterflooding* per giacimenti maturi, che integra modelli di intelligenza artificiale, come reti neurali *long short-term memory* (LSTM) e *physics informed* (PINN), con tecniche di ottimizzazione, tra cui algoritmi genetici ed *ensemble-based*. La metodologia propone

di costruire un modello di *machine learning* (ML) che approssimi il complesso modello di giacimento e che predica con precisione le condizioni future del giacimento stesso in termini di produzione di acqua e olio, in funzione delle iniezioni di acqua. In seguito, il modello ML del giacimento è integrato con un algoritmo di ottimizzazione per ottenere i profili di iniezione ottimali, con il valore attuale netto (NPV) come funzione obiettivo, rispettando alcuni vincoli operazionali ed economici.

L'efficacia della metodologia proposta è validata attraverso l'applicazione a due casi di studio. Il primo caso è un giacimento sintetico 2D, con proprietà geologiche omogenee, quattro pozzi produttori e cinque iniettori. Reti neurali PINN sono impiegate come modello, al fine di integrare vincoli fisici durante la fase di training. Il secondo caso è il noto Olympus, un giacimento sintetico 3D con proprietà geologiche complesse, sette iniettori e undici produttori. In questo caso, reti neurali LSTM sono impiegate come modello, poiché il tempo deve essere integrato esplicitamente nella predizione della rete. Le strategie di iniezione ottimali ottenute sono poi validate confrontandole con altri scenari di produzione tipici dei giacimenti e un software commerciale per ottimizzazione del *waterflooding*. I risultati mostrano una efficace previsione da parte del modello, una significativa riduzione dei tempi computazionali e strategie di ottimizzazione coerenti con i casi di studio.

Parole chiave: produzione, waterflooding, simulazione di giacimento, ottimizzazione, machine learning, PINN, LSTM, algoritmi genetici

Contents

1 | Introduction

New discoveries of oil and gas reservoir are declining [1], while most of the existing fields have already reached their mature stage of production, in which the natural pressure of the reservoir has already dropped and become insufficient for natural depletion of the reservoir. In such situation, an external fluid, e.g., water, is injected to add energy into the reservoir-fluid system and maintain the reservoir pressure, acting as a driving force to keep oil flowing [2]. This process is known as waterflooding. Then, optimizing the waterflooding process is essential to increase oil recovery from reservoirs, decrease operational costs and reduce environmental hazards, while continuing to satisfy the increasing global demand for energy [3].

Traditionally, waterflooding optimization is performed through surveillance or model-based optimization methods [3]. With surveillance methods, decisions are taken reactively by field engineers and operators, based on the observed conditions around each well, and on the analysis of production data and decline curves [4]. This approach is complicated and depends heavily on the experience of field engineers and operators, with the results that it often leads to suboptimal solutions. On the other hand, model-based optimization methods rely on the use of a physics-based model that accurately describes the reservoir's behavior [5]. Although these methods are potentially accurate and have a desired lifecycle perspective, their application is practically challenging due to the complexity and computational expense of the reservoir models. To construct an accurate 3D geological model, extensive effort is required from a variety of professionals, including engineers, geologists and geophysicists, typically for several months. Geological models, then, undergo the process of history matching, in which the model's parameters are tuned onto historical data and measurements. History matching is also a long, complex, and time-consuming process: the higher the complexity and size of the field, the more time-consuming and computationally heavy history matching becomes; the same is true for simulation runs. For these reasons, well-tuned models are not always available or apt to the purpose.

Furthermore, for large fields with dozens of wells, the dimensionality of the optimization problem quickly rises if well patterns, bottom-hole pressures (BHPs) and injection rates

are all taken as control variables [3].

To overcome these modelling challenges, surrogate-based optimization (SBO) methods have been proposed, where a surrogate model, capable of predicting the reservoir's behavior in a fast but still accurate way, replaces the complex physics-based model in the optimization problem, so as to accelerate the convergence of its solution [6]. At this point, it is worth mentioning that a well-tuned geological model is a powerful tool that can be used to forecast the reservoir's behavior in a variety of engineering problems, such as waterflooding, enhanced oil recovery, well placement, geomechanics evaluations; on the contrary, reservoir model surrogates are typically constructed ad hoc for the specific problem of interest.

Two classes of surrogate models are considered in literature, including shortcut physics-based models and machine learning (ML) techniques. Shortcut physics-based models are obtained by applying specific techniques (e.g., hierarchical modeling [7–9] and model order reduction [10–12]) to simplify the high-fidelity physics-based model, to obtain a simpler and computationally cheaper model of the reservoir. However, the applications of these simplifying techniques require some knowledge on the geology and physics of the reservoir, as well as access to the explicit equations of the high-fidelity physics-based model [13]. Besides, their accuracy can be compromised when applied to large-scale reservoir models. On the other hand, ML models [6, 12, 14] can be built either using simulation data generated by the full-physics model or real data collected from the field by means of modern well-sensor technologies, in the case where a model for the reservoir not available [3, 15]. ML models do not require any knowledge of the geology/physics of the reservoir and have the ability to handle complex, large-scale fields [13, 16]. Among different types of ML models, artificial neural networks (ANNs) are widely used in reservoir design and operation optimization problems due to their potential capabilities to approximate the challenging behavior of such systems, in terms of high dimensionality, nonlinearity, noise, and non-smoothness. Nevertheless, they have been extensively applied to tackle history matching problems [16, 17], whereas their application in oil production optimization problems is still limited and, certainly, not extensive at industrial level [15].

Specifically, in waterflooding optimization problems, ANNs have been highly recommended as surrogate models [18], although they were applied for problems with relatively few input control variables. Queipo et al. (2002)[19] used ANNs as a surrogate model in a steam-flooding optimization problem involving a 2D synthetic field with one injection well and two production wells. An ANN model is used to approximate an objective expressed as a weighted sum of the cumulative produced oil and cumulative injected steam, as a function of design and operational variables that include vertical spacing of the injection

well, injected steam enthalpy, injection pressure and subcooling. The ANN model is coupled with the Efficient Global Optimization algorithm (EGO) to search for the optimal solution. Zangl et al. (2006)[20] developed an ANN-based optimization workflow and applied it to gas injection optimization in a field composed of 7 injection and 12 production wells. The ANN is developed to approximate the mapping between the injected gas pressures and temperatures, and the objective function is expressed in terms of the total oil production. Then, a genetic algorithm (GA) is used to identify optimal values of the decision variables. The works reviewed above have employed ANNs as simple static function evaluators that approximate the mapping between the decision variables and a given optimization criterion, formulated in terms of an objective function, neglecting dynamic information about the reservoir's behavior and, thus, preventing to impose physical constraints on the behavior of certain variables (e.g. water production rates).

Advanced approaches have employed ANNs as dynamic models to predict the time-evolution of the reservoir state variables (e.g. water and oil production rates) over long time horizons. This is especially useful when constraints are to be imposed to the optimization problem. Even though the risk of error propagation on the objective function is higher, these dynamic representations allow a much deeper and realistic insight into the reservoir's behavior. Golzari et al. (2015)[21] applied ANNs to the production optimization of a 3D synthetic field including 2 injection and 4 production wells. An ANN-based dynamic model is trained, by adaptive sampling, to approximate the oil and water production rates as a function of BHPs. Then, considering the net present value (NPV) as objective function, a GA is used to identify the optimal BHP schedule. Salam et al. (2015)[22] used ANNs coupled to a GA for design and production optimization of a realistic 3D reservoir model composed of 10 wells. ANNs are harnessed to approximate mappings between the decision variables (well locations, production and injection rates, and scheduling of conversion of production wells into injection wells) and two objective functions, including the cumulative oil production and the NPV. Teixeira and Secchi (2019)[23] employed ANNs for waterflooding optimization of a 3D mature field composed by 14 production and 11 injection wells. Different designs of ANN-based dynamic models have been developed to forecast the future values of the oil production rates as a function of current and previous values of the water injection rates, oil production rates or BHPs. Then, the developed ANN is coupled to a gradient-based algorithm to search the optimal solution, considering the total oil production as objective function. Bruyelle et Guérillot (2019)[24] proposed a method for multi-objective optimization of waterflooding based on ANNs and evolutionary algorithms, and applied it to the Brugge field benchmark. The ANN model is used to approximate two objectives, including the NPV and its risk (NPV

variance) over multiple geological realizations. A covariance matrix adaptation evolutionary algorithm (CMA-ES) is used to identify the Pareto front of the BHP schedules, taking into account the previously mentioned objectives. Chai et al. (2020)[25] tested different combinations of different surrogate model types (e.g. ANN and extreme gradient boosting) with optimization algorithms (e.g. particle swarm optimization (PSO), GA and genetic swarm optimization (GSO)) for waterflooding optimization, and applied them to a 2D model of a huge mature field. The surrogate models take as inputs the injection and production well BHPs and provide as output the oil and water production rates.

Most of the these works have focused on feedforward ANNs as surrogate models, which have two main limitations:

- risk of under-fitting dynamics: hydrocarbon reservoir possess very complex dynamic behaviors, which evolve over time, i.e. the response to the same input varies over time;
- risk of predicting non-physically supported behaviors.

With respect to the first limitation, recurrent neural networks (RNNs) [13], such as long short-term memory (LSTM) networks [26], are specifically designed to predict dynamic behaviors more efficiently than traditional ANNs in such problems, thanks to their powerful capabilities to capture and learn temporal and spatial patterns among multivariate time series. Yong and Durlofsky (2021)[13] used LSTMs as a surrogate model for production optimization of 2D and 3D synthetic reservoir cases, including 2 injection and 5 production wells. The LSTM network is used to predict future time profiles of the oil and water production rates at each well as a function of injection rates and BHPs. A PSO algorithm is used to obtain the optimal BHPs, considering the NPV as the objective function and using a filter-based procedure for nonlinear constraints treatment. A similar approach was adopted by Deng and Pan (2021)[27], who applied echo state networks (ESNs) as surrogate models for waterflooding optimization. ESNs receive the water injection rates and the BHPs as input, and provide the water and oil production rates as output, prior to the use of a fractional flow model. A mesh adaptive direct search (MADS) algorithm is employed to search for the optimal injection rates considering the non-discounted NPV as the objective function. Their method has been applied to two test case studies that include a 2D reservoir model with one injector and 4 producers, and a 3D reservoir model with 8 injectors and 4 producers.

With respect to the second limitation, physics informed neural networks (PINNs) represent a potential solution due to their ability to integrate physical laws and constraints into the loss function of an ANN during the training phase [28]. Recently, few studies [29, 30]

investigated the use of PINNs for history matching problems of reservoirs. Quite recently, Yewgat et al. (2020)[31] and Maniglio et al. (2021)[32] used PINNs in combination with a capacitance resistance model (CRM), which represents the physical constraints of the network, to forecast production rates from injection rates and BHPs.

To the best of the author's knowledge, the use of LSTMs has not yet been considered for the injection schedule optimization of mature or brownfields, and also the use of PINNs for waterflooding optimization.

This work presents a novel SBO framework for optimizing water injection schedules in mature oil fields, relying on PINNs and LSTMs. The proposed framework consists of three stages. In the first stage, the procedure used for data generation is designed, which imposes tight, practical and realistic constraints on the reservoir model simulations to generate input-output signals, i.e. water injection rates and corresponding water and oil production rates, as if they were collected from the real field. In this way, the scope of the framework is enlarged to handle both situations, whereby an accurate reservoir model is available or only real field data are obtainable. The second stage includes the efficient development (i.e. training, validation and testing) of a ML model (a PINN or a LSTM), which is able to accurately predict the future oil and water production rates at each well, as a function of the water injection rates. In the third stage, the developed ML model is coupled to an optimization algorithm to identify the optimal water injection profiles to be applied to the field over the future time period, conserving the NPV as objective function and different operational and economic constraints. Three different optimization algorithms, including ensemble-based, GA and gradient-based, are used in this thesis, to evaluate the flexibility and robustness of the framework. The obtained optimal solutions are, then, compared to standard reservoir practices and software in terms of objective function, computational time and strategies. Finally, optimization results are deeply analyzed to interpret the identified strategies through a variety of tools, ranging from streamline maps to pressure and saturation distributions. Optimization results cannot be blindly translated into action: instead, they must be adapted to economic and operational constraints of the real field that, for the sake of simplicity, cannot be all implemented in the optimization problem in the first place. Therefore, the optimized waterflooding strategy can be viewed as the best unconstrained recommendation from the optimization algorithm, whereas to what extent this strategy can be implemented in the real field is ultimately left to production engineers to decide.

With respect to the approaches developed in literature, the novelty of the proposed framework lies in:

- using PINNs and LSTMs in the context of injection schedule optimization of mature fields;
- presenting a comparison with a state-of-the-art commercial software to understand if time and effort required for a 3D reservoir model are justified or if a data-driven approach is more convenient;
- the deep interpretation and verifications of the obtained optimal solutions;
- the use of EnOpt (by Chen et al. (2009)[33]) in SBO of oil field production.

The effectiveness of the proposed framework is validated through its application to two case studies. The first case considers a 2D reservoir model with homogeneous geological properties, including 4 production and 5 injection wells. The second case is the well-known Olympus field, which is a 3D reservoir model with complex geological properties, including 7 injection and 11 production wells. The results show a good prediction accuracy, significant reduction in the required computational time and even more, provide high-level operational recommendation and guidelines.

The remaining part of the thesis is organized as follows: Chapter 2 presents the waterflooding optimization problem statement, Chapter 3 shows the proposed methodology, Chapter 4 describes its application to the two case studies, and Chapter 5 concludes the work.

2 | Problem Statement

This thesis addresses the model-based optimization of waterflooding of a generic oil field. The generic field is made up of N_{in} injection wells, with injection rates $q_{in,i}$ and N_p production wells with water and oil production rates $q_{w,j}$ and $q_{o,j}$, located at specific positions, as in Figure 2.1.

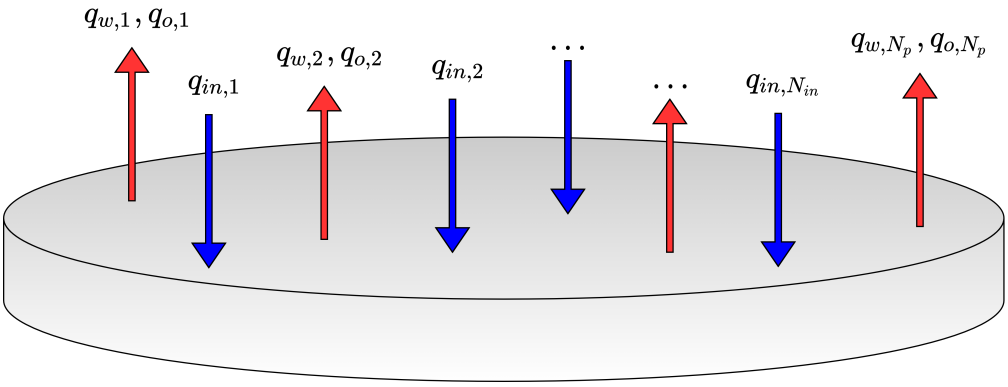


Figure 2.1: Schematic field and well designation.

For simplicity, gas is neglected in this approach, which deals with an oil-water system. In field management operations, injection rates can be set by operators, while production rates are a function of pressure, saturations and geological properties of the reservoir, which in turn vary over time based on injection and production rates, as wells as on the initial properties of the reservoir.

The generic problem of waterflooding optimization aims at finding the control variables $\boldsymbol{u}$ which result in the optimal value of the objective function J, while satisfying predefined constraints $\boldsymbol{c}$:

$$\begin{cases} \min\limits_{\boldsymbol{u} \in U} J(\boldsymbol{x}, \boldsymbol{u}) \\ \boldsymbol{c}(\boldsymbol{x}, \boldsymbol{u}) \leqslant \boldsymbol{0} \\ \boldsymbol{g}(\boldsymbol{x}, \boldsymbol{u}) = \boldsymbol{0} \end{cases} \tag{2.1}$$

where:

- $\boldsymbol{x}$ is the vector of dynamic state variables of the model (pressure, saturation, etc.);
- $\boldsymbol{u}$ is the vector of well control variables, of dimension n;
- $U = \{\boldsymbol{u} \in \mathbb{R}^n; \boldsymbol{u}_{lb} \leqslant \boldsymbol{u} \leqslant \boldsymbol{u}_{ub}\}$ defines the allowable values for $\boldsymbol{u}$;
- $\boldsymbol{c}$ is the set of linear and nonlinear constraints on all control variables;
- $\boldsymbol{g}$ is the reservoir model (set of reservoir simulation equations) to be solved to evaluate J and $\boldsymbol{c}$.

The problem also requires the definition of a timespan, including both the previous period over which the field was operated and the future period over which the field is required to be optimized. In this way, the optimization timespan is divided into N_t discrete time steps, Δt, such that $t_k = k\Delta t$ for $i = 1, ..., N_t$, as in Figure 2.2:

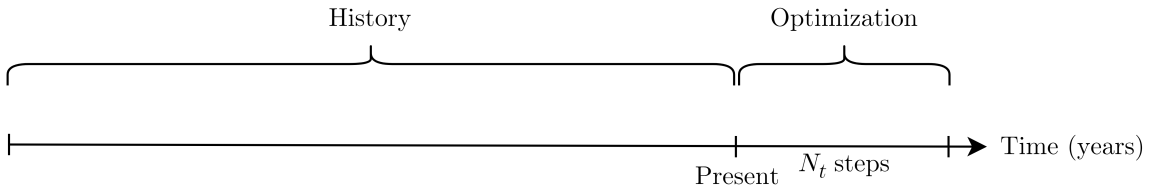


Figure 2.2: Waterflooding optimization timeline.

Since injection rates are chosen as control variables, the dimension of the optimization problem (i.e. number of control variables) is given by the product of the number of injection wells by the number of times steps to optimize, $n = N_{in}N_t$. Hence, the control variable vector can be represented as a concatenated vector in time

$$\boldsymbol{u} = [q_{in,1}^{\Delta t}, ..., q_{in,N_{in}}^{\Delta t}, ..., q_{in,1}^{N_t\Delta t}, ..., q_{in,N_{in}}^{N_t\Delta t}]^T \tag{2.2}$$

where $q_{in,i}$ is the water injection rate of injection well i.

The most common objective function for waterflooding optimization problems is the Net Present Value (NPV) [34]. It gives an economic evaluation of the field's performance in terms of costs and revenues over time, while accounting for the time value of resources (hydrocarbons produced today are worth more than if produced in the future). Since the NPV is to be maximized, $J(\boldsymbol{u}) = -NPV(\boldsymbol{u})$. Mathematically, the NPV can be defined as:

$$NPV(\boldsymbol{u}) = \sum_{k=1}^{N_t} \left[\sum_{j=1}^{N_p} (r_o q_{o,j}^{k\Delta t} - r_w q_{w,j}^{k\Delta t}) - \sum_{i=1}^{N_{in}} r_{in} q_{in,i}^{k\Delta t} \right] \frac{\Delta t}{(1+d)^{\frac{k\Delta t}{365}}} \tag{2.3}$$

where:

- $q_{o,j}$ is the oil production rate of production well j;
- $q_{w,j}$ is the water production rate of production well j;
- r_o is the price of produced oil per unit volume;
- r_w is the cost of produced water per unit volume;
- r_{in} is the cost of injected water per unit volume;
- d is the yearly discount factor (such that if time steps are expressed in days the ratio $\frac{k\Delta t}{365}$ is dimensionless).

The objective function is in general a function of $\boldsymbol{u}$ and $\boldsymbol{x}$, while the outputs of interest $q_{o,j}$ and $q_{w,j}$ are omitted since the relationship between them and $\boldsymbol{u}$ the lies within the forward model, either the reservoir simulator or any other surrogate, based on $\boldsymbol{x}$. In addition, in this problem, J can be considered solely a function of $\boldsymbol{u}$ since only the vector of controls variables is subject to optimization.

Hence, with this definition of the NPV, oil produced is considered a revenue, while water, whether injected or produced, is considered a cost. In this way, pumping costs for injected water, and separation or disposal costs for produced water are accounted for. Thus, the optimization process will steer towards injection configurations that allow to extract the most resources (oil) for the minimum cost (injected and produced water). Nonetheless, the objective function can be selected *ad hoc* for the specific field, depending on its features and optimization requirements. For example, a field might incur extra costs only in case the produced water exceeds the injected water, or if oil production falls behind a set limit. Besides, the objective function can be extended to other fluids, such as gas or steam, which can represent costs or revenues for the field.

Additionally, the problem is subject to the set of constraints $\boldsymbol{c}$, as in (2.1), which model the technical and operational limits of the field. Just like the objective function, constraints to the optimization problem are defined *ad hoc* for the specific field, and in general can take a wide variety of mathematical formulations. For the purposes of this thesis, the following are implemented:

- bound constraints: representing the upper and lower limits of injection rate for each

well at any time step:

$$q_{in,i}^{min} \leqslant q_{in,i}^{k\Delta t} \leqslant q_{in,i}^{max} \quad \text{for } i = 1, ..., N_{in} \text{ and } k = 1, ..., N_t \tag{2.4}$$

– linear constraints: representing the maximum water injection rate at a field level at any time step:

$$\sum_{i=1}^{N_{in}} q_{in,i}^{k\Delta t} \leqslant q_{in,F}^{max} \quad \text{for } k = 1, ..., N_t \tag{2.5}$$

– injection variation over time: representing the unwanted abrupt (exceeding a limit Δq_{in}^{lim}) changes over time of injection rates, which would be impractical to put into action:

$$|q_{in,i}^{k\Delta t} - q_{in,i}^{(k+1)\Delta t}| \leqslant \Delta q_{in}^{lim} \quad \text{for } i = 1, ..., N_{in} \text{ and } k = 1, ..., N_t - 1 \tag{2.6}$$

From a physical perspective, the process of waterflooding is a complex, nonlinear problem. To model it mathematically, the following equations are required:

– equation of state for all phases;

– multiphase flow equations;

– mass conservation equations;

– deliverability equations at wells.

The effectiveness of the waterflooding process can be measured by means of sweep efficiency and the local displacement efficiency [2]. The sweep efficiency is the fraction of the volume of the reservoir contacted by the injected water: it depends on a variety of factors, such as injection pattern, permeability, flow rate, fluid properties. The local displacement efficiency is the fraction of oil that has been recovered from a zone swept by the injected water. Analytical solutions for the problem of waterflooding exist only for simplified cases, such as the Buckley-Leverett equation [35], for a 1D, immiscible and incompressible oil-water system. More details can be found in Appendix A. In real applications though, waterflooding is usually solved through traditional finite difference reservoir simulation. The reservoir is discretized as a 3D grid of blocks (or cells), with given properties (porosity, permeabiltiy, etc.) and the above equations are applied to each grid block: the resulting system of non-linear PDEs is then solved numerically to obtain pressure, velocity and saturation distributions. In this way, a geological model paired to a simulator allows to analyze the relation between control variables and state variables. Due to the complex,

non-linear nature of the 3D reservoir model, the simulator-based optimization process can become computationally unfeasible for large scale applications. Therefore, a data-driven methodology, where neural networks serve as forward model instead of the simulator, is proposed. More information on oilfield production life and flow in oil reservoirs can be found in Appendix A, while a more generalized version of the waterflooding optimization problem can be found in the Appendix B.

In waterflooding optimization problems, it is useful to quickly refer to the vector of control variables for a specific injection well. In this thesis, $\boldsymbol{u}_i$ refers to the vector of control variables of injection well i at all time steps:

$$\boldsymbol{u}_i = [q_{in,i}^{\Delta t}, ..., q_{in,i}^{N_t \Delta t}]^T \tag{2.7}$$

Conversely, there is often the need to refer to all the wells' injection rates at a specific time step k. For this purpose, another index (k) can be used:

$$\boldsymbol{u}_{(k)} = [q_{in,1}^{k\Delta t}, ..., q_{in,N_{in}}^{k\Delta t}]^T \tag{2.8}$$

3 | Methodology

The proposed methodology is based on three main stages, as reported schematically in Figure 3.1.

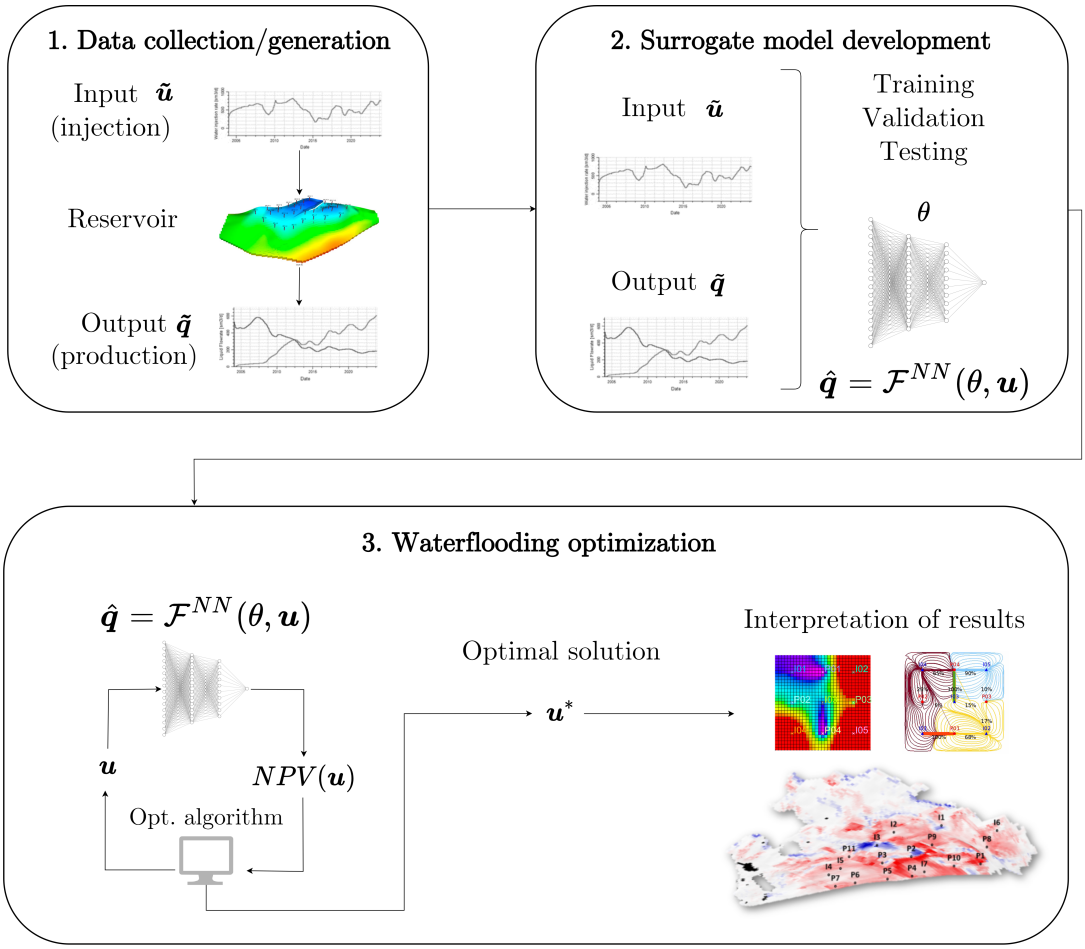


Figure 3.1: Schematic diagram of the proposed methodology.

In the first stage, a procedure for data generation is presented, which imposes tight, practical, and realistic constraints on the reservoir model simulations to generate input-output signals, i.e., water injection rates and corresponding water and oil production rates, as if they were collected from a real field. The second stage includes the efficient development

(i.e., training, validation and testing) of a ML model, which is able to accurately predict future production at each well as a function of the water injection rates. In the third stage the developed ML model is coupled to an optimization algorithm to identify the optimal water injection profiles to be applied to the field over the future time period, adopting the NPV as objective function and different operational and economic constraints. Finally, an in-depth interpretation of the results is conducted with the aid of streamline maps, pressure and saturation distributions. Each step is discussed in detail in this section.

This thesis was conceived as a fast waterflooding optimization tool for cases in which only field data, but no model, is available. However, the scope of the framework can be enlarged to handle cases where an accurate reservoir model is, indeed, available: historical production data can be simulated for a wide range of input values by running several simulations. The computational time required to run such simulations needs to be evaluated carefully, though: if excessive, it would defeat the purpose of a time-saving surrogate model in the first place. Due to difficulty of obtaining real field data, this methodology is applied to synthetic fields, which are considered as the real field: synthetic data are generated in a specific way, as reported in the next section, such that they display realistic characteristics and constraints. On the other hand, if real data is available, the methodology is still applicable and is even more straightforward, since it only requires data collection.

It should be noted that this methodology does not aim at fully replacing the reservoir simulator as forward model. Instead, the idea is to adopt data-driven optimization for cases in which ML surrogates are able to replicate the behavior of the reservoir accurately, avoiding the need for simulation, serving as a first-guess for more complex simulator-based optimization, or as a quick tool for real-time applications.

3.1. Data collection/generation

Due to the intrinsic dynamic nature of the problem, the forward model is requested to reproduce the reservoir's behavior over time. The historical data on which the model is trained always refers to an earlier behavior of the reservoir compared to the one the model is required to predict. In this thesis, the developed workflow is applied to synthetic reservoirs. Therefore, the process of data acquisition, filtering and assimilation is replaced by synthetic data generation. The full-physics reservoir simulator is run along the historical timeline to generate data and the matched input-output couples $(\tilde{\boldsymbol{u}}, \tilde{\boldsymbol{q}}) = ([\tilde{\boldsymbol{u}}_1^T, ..., \tilde{\boldsymbol{u}}_{N_{data}}^T]^T, [\tilde{\boldsymbol{q}}_1^T, ..., \tilde{\boldsymbol{q}}_{N_{data}}^T]^T)$, which are collected and employed as historical production data, as in Figure 3.1.

3.1.1. Synthetic data generation

Synthetically generated data should, to a certain extent, resemble real historical data to ensure the proposed methodology can be applied to real-life fields. To obtain realistic synthetic data, the following aspects are considered:

- each well has a single injection or production profile over time;
- profiles do not cover the whole range of allowable values;
- profiles are not randomly generated, but time-correlated.

The first aspect represents the fact that the available data is intrinsically limited to the historical production strategy of the field itself. The second aspect originates from the fact that historical data may not cover the complete range of acceptable values for controls and parameters, but only the ones that were adopted during the field's actual production history. The third aspect ensures that historical data changes smoothly over time. Oil displacement through waterflooding is a slow process: its timescale is in the order of months to years [2]. Hence, variations in production strategies cannot be applied continuously, but with a frequency on the order of months.

In addition, production data from fields always suffers from noise and uncertainty in measurements, often recorded with variable frequency. For the purposes of this work, however, these issues were not addressed: historical data consists of matched input-output couples over time with a constant frequency of measurement.

To achieve realistic synthetic data, injection profile generation is based on the formulation provided by Chilès (2012)[36]. The idea is to generate smoother injection profiles with respect to a random initialization, by imposing a correlation. Correlation between values means that the closer two points are in the variable space, the closer their values are. This can be applied to time, space, or a combination of time and space. Time-correlated injection profiles express the need to change injection rates for the same well at consequent points in time gradually. Space-correlated injection profiles represent the need to avoid flooding neighbor wells disproportionately at the same point in time. For simplicity, only time correlation is considered in this work. Mathematically, correlation is expressed in terms of a specific function [37]. Since the quantity to correlate in time is the injection profile of a single well i, called $\boldsymbol{u}_i$, the following parameters are specified:

- number of elements of the vector N_t (i.e. time steps for each injection well);
- number of dimensions along which to correlate n_{corr} (i.e. time only);
- correlation length or range a;

- type of correlation (also known as variogram model γ, it expresses the mathematical expression of the correlation: a cubic variogram is used in this work);
- upper and lower bounds for each injection well $q_{in,i}^{min}, q_{in,i}^{max}$;
- average values $\bar{u}_i$ and standard deviation σ_i for each injection well.

Time-correlated profiles $\boldsymbol{u}_{corr,i}$ are obtained starting from a Gaussian distribution profile with zero mean and unitary variance, $\boldsymbol{u}_i$, by centering it around its average $\bar{\boldsymbol{u}}_i$, as in:

$$\boldsymbol{u}_{corr,i} = \boldsymbol{L}_i \boldsymbol{u}_i + \bar{\boldsymbol{u}}_i \quad \text{for } i = 1, ..., N_{in} \tag{3.1}$$

where $\boldsymbol{L}_i$ is the lower triangular matrix extracted by Cholesky decomposition from the covariance matrix $\boldsymbol{C}_i$, such that $\boldsymbol{C}_i = \boldsymbol{L}_i(\boldsymbol{L}_i^T)'$. The covariance matrix is constructed as a function of the above parameters $\boldsymbol{C}_i = \boldsymbol{C}_i(N_t, n_{corr}, a, \gamma, q_{in,i}^{min}, q_{in,i}^{max}, \sigma_i)$ (more details can be found in [36] and in Appendix G). The average value $\bar{\boldsymbol{u}}_i$ is in general time dependent (i.e. it is a vector), but is taken as constant in this case. Finally, excessively low and high values are replaced by their respective minimum and maximum bounds. A pictorial example is reported in Figure 3.2.

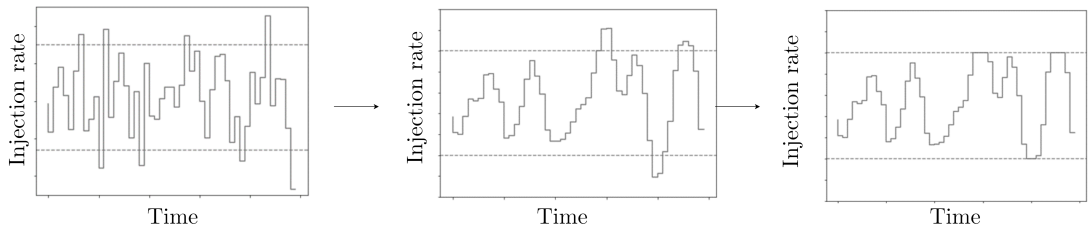


Figure 3.2: Example of time-uncorrelated and time-correlated profiles.

3.2. Surrogate model development

Once historical data is generated, it is used to train neural network surrogates of the simulator. The whole set of training data is split into training, validation, and testing, following the commonly used 70-20-10% rule in a chronological order. Historical data, collected over the "past" region of the timeline, is used for training and validation of the surrogate model, while the testing phase is performed on the same time period of the "future" to optimize. The present-day moment is the boundary between the two (Figure 3.3). Although specific features of the neural networks vary between the case study applications, all networks have injection rates as inputs, and water cuts or production rates as outputs. The total timespan is divided into a number of time steps, assumed

as the smallest possible period over which input variables (i.e. injection rates) are kept constant.

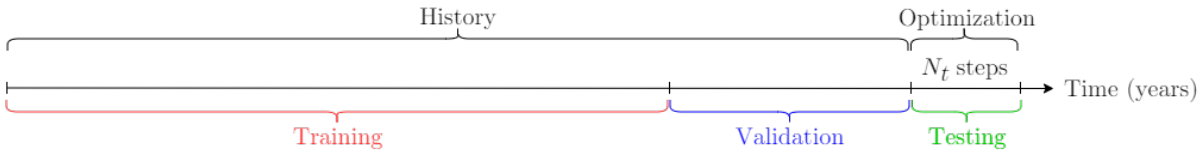


Figure 3.3: Generic timeline for the problem.

The network's approximation of the generic output from the reservoir, $\boldsymbol{q}$, is denoted as $\hat{\boldsymbol{q}} = \mathcal{F}^{NN}(\theta, \boldsymbol{u}) \approx \boldsymbol{q}$: it is a function of inputs $\boldsymbol{u}$ and its parameters (weights and biases) θ, which are obtained in the training phase, minimizing the mean-squared error (MSE) on the data:

$$MSE_{data} = \frac{1}{N_{data}} \sum_{i=1}^{N_{data}} [\mathcal{F}^{NN}(\theta, \tilde{\boldsymbol{u}}_i) - \tilde{\boldsymbol{q}}_i]^2 \tag{3.2}$$

Another measure of the accuracy of the forecast is the normalized root-mean-squared error (NRMSE), where the MSE is normalized with the range $\bar{q}$ or the output values in the dataset:

$$NRMSE_{data} = \sqrt{\frac{MSE_{data}}{\bar{q}}} \tag{3.3}$$

3.2.1. Physics informed neural networks

PINNs integrate the information embedded in the set of partial differential equations (PDEs) that model the physical system into the loss function during training. The loss function contains not only the residual between observed and forecast data, but also additional regularization terms which limit the space of admissible solutions. PINNs honor those physical relationships in their forecast, resulting in more accurate learning and generalization power. Any parametrized non-linear PDE can be modeled in the general form:

$$f = \mathcal{N}[\boldsymbol{q}; \lambda] = 0 \tag{3.4}$$

where $\boldsymbol{q} = \boldsymbol{q}(t, \boldsymbol{u})$ is the solution of the PDE and $\mathcal{N}$ is a generic non-linear operator parametrized by λ, which represents the model's parameters. Within $\mathcal{N}$ are encapsulated specific forms of typical PDE problems with the respective derivatives. PINNs can be employed both in the forward problem (i.e. to solve PDEs) and in the inverse problem (i.e. to find their parameters). Since geological properties of the reservoir are as-

sumed to be unknown, this application is a case of inverse problem. The inverse problem entails finding the value of the function $\boldsymbol{q}(t, \boldsymbol{u})$ and λ model parameters, given a set of training data $(\tilde{\boldsymbol{u}}, t, f(\tilde{\boldsymbol{u}}, t)) = ([\tilde{\boldsymbol{u}}_1^T, t_1, ..., \tilde{\boldsymbol{u}}_{N_{data}}^T, t_{Ndata}]^T, [f(\tilde{\boldsymbol{u}}_1^T, t_1), ..., f(\tilde{\boldsymbol{u}}_{N_{data}}^T, t_{Ndata})]^T)$, such that (3.4) is honored. The additional loss function is:

$$MSE_f = \frac{1}{N_{data}} \sum_{i=1}^{N_{data}} [f(t_i, \tilde{\boldsymbol{u}}_i)]^2 \tag{3.5}$$

such that the total loss function is defined by a linear combination of the above loss functions:

$$MSE_{tot} = MSE_{data} + \lambda_f MSE_f \tag{3.6}$$

where λ_f value is a Lagrange sensitivity value. Additional loss functions can be integrated into (3.6) to ensure specific constraints on PDEs are honored. A schematic representation of the PINN concept is reported in Figure 3.4, while more details on PINNs are found in Appendix F.

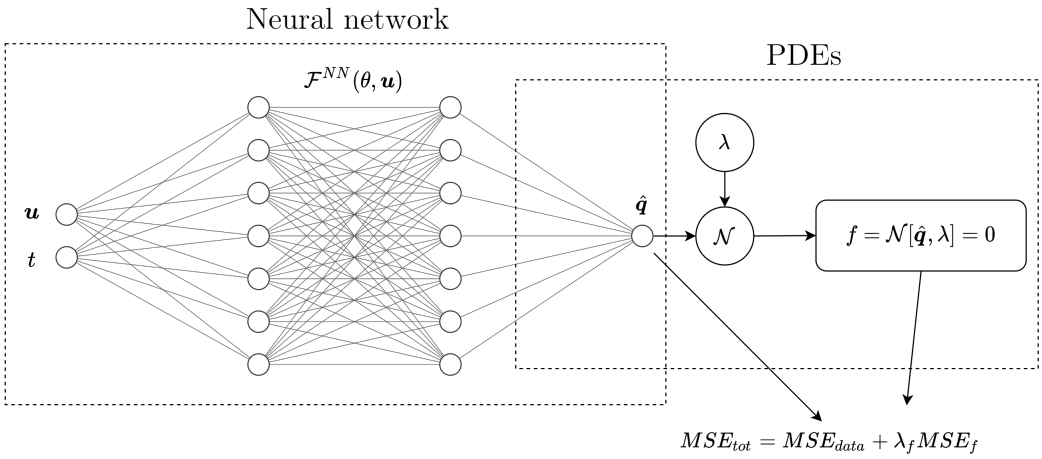


Figure 3.4: Generic PINN concept.

CRM-based PINNs

In this work, the prior knowledge of the physical laws of the system is represented by capacitance resistance models (CRMs) (see Appendix C), in which the reservoir is schematized as a circuit with time constants τ_j and well connectivities f_{ij} as parameters. Liquid production rate q_j is a function of injection rates $q_{in,i}$, bottom-hole pressures $p_{wf,j}$ and the wells' productivity index PI_j. In particular, the CRMP model [38] is applied. Maniglio et al. (2021)[32], developed and trained CRM-based PINNs for oil reservoirs. The goal is to apply the optimization workflow to such surrogate types. Thus, the additional loss

components are:

– the residual on the CRMP equation:

$$f = \tau_j \frac{dq_j(t)}{dt} + q_j(t) - \sum_{i=1}^{N_{in}} f_{ij} q_{in,i}(t) + \tau_j PI_j \frac{dp_{wf,j}(t)}{dt} = 0 \tag{3.7}$$

– the residual on the time constant constraint:

$$g_1 = \sum_{j=1}^{N_p} ReLU(-\tau_j) \tag{3.8}$$

– the residual on the connectivities constraints:

$$g_2 = \sum_{i=1}^{N_{in}} \sum_{j=1}^{N_p} ReLU(-f_{ij}) \tag{3.9}$$

$$g_3 = \sum_{i=1}^{N_{in}} \sum_{j=1}^{N_p} ReLU(f_{ij} - 1) \tag{3.10}$$

$$g_4 = \sum_{i=1}^{N_{in}} ReLU\left[\sum_{j=1}^{N_p} (f_{ij} - 1)\right] \tag{3.11}$$

where $ReLU(\cdot) = max(0, \cdot)$ is the Rectified Linear Unit function. The total loss function (3.6) becomes a weighted average of the single losses:

$$\begin{aligned} \frac{1}{N_{data}} \sum_{i=1}^{N_{data}} \big\{ [\mathcal{F}^{NN}(\theta, \tilde{\boldsymbol{u}}_i) - \tilde{\boldsymbol{y}}_i]^2 + \lambda_f f(\mathcal{F}^{NN}(\theta, \tilde{\boldsymbol{u}}_i))^2 + \\ \lambda_1 (g_{1,i})^2 + \lambda_2 (g_{2,i})^2 + \lambda_3 (g_{3,i})^2 + \lambda_4 (g_{4,i})^2 \big\} \end{aligned} \tag{3.12}$$

Weights can be seen as Lagrange multipliers of a constrained minimization problem. Training of the network on historical data will result in optimal θ, f_{ij}, PI_j, and τ_j values. In particular, Maniglio et al. (2021)[32] trained two types of networks, which are used in this work. They both receive injection rates and times at all injectors as inputs: one is a traditional ANN and forecasts water cut, as shown in Figure 3.5, while the other is a PINN (ANN+CRMP) and forecasts liquid production rates at all producers, as shown in Figure 3.6. From liquid production rate and water cut, oil and water production rates can be easily calculated.

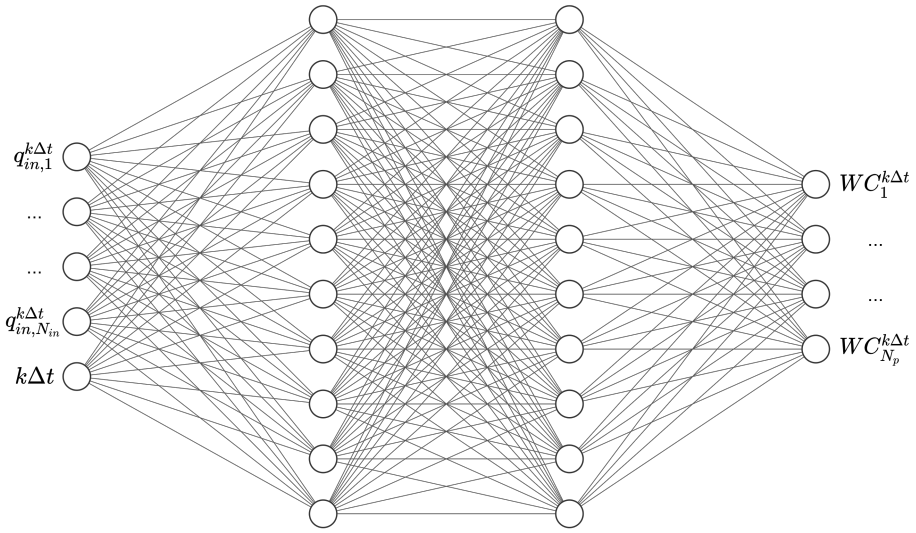


Figure 3.5: Generic water cut ANN.

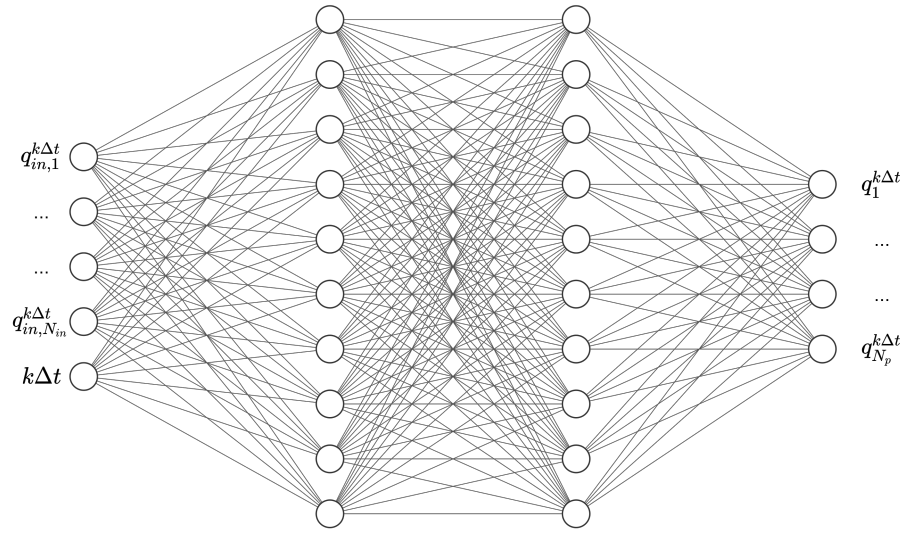


Figure 3.6: Generic liquid production rate PINN.

This choice comes primarily from their integration with CRMs, which are designed to forecast liquid production rates in the first place. Moreover, in this way the water cut forecast is not tied to a specific fractional flow model.

3.2.2. Recurrent neural networks

Recurrent neural networks (RNNs) are specifically designed to predict dynamic behavior problems, in which data is a time sequence. In fact, the network's prediction at a specific time step is based not only on its input at the same time step, but also on the previous

state(s) of the network, which are stored temporarily. A single recurrent neuron can be thought of as a cell. The state of a cell at the generic time step k is denoted as $\boldsymbol{h}_{(k)}$ and is a function of both inputs at the same time step $\boldsymbol{u}_{(k)}$ and the state at the previous time step $\boldsymbol{h}_{(k-1)}$.

Long short-term memory cells

Among RNNs, long short-term memory (LSTM) cells, have been proven successful at tackling the short-term memory issue of traditional RNNs, showing better overall performance, faster training and the ability to detect long-term patterns in data. Differently from traditional RNN cells, the state vector is split into two parts: $\boldsymbol{h}_{(k)}$, representing the short-term state, and $\boldsymbol{c}_{(k)}$, representing the long-term state. An example is provided in Figure 3.7.

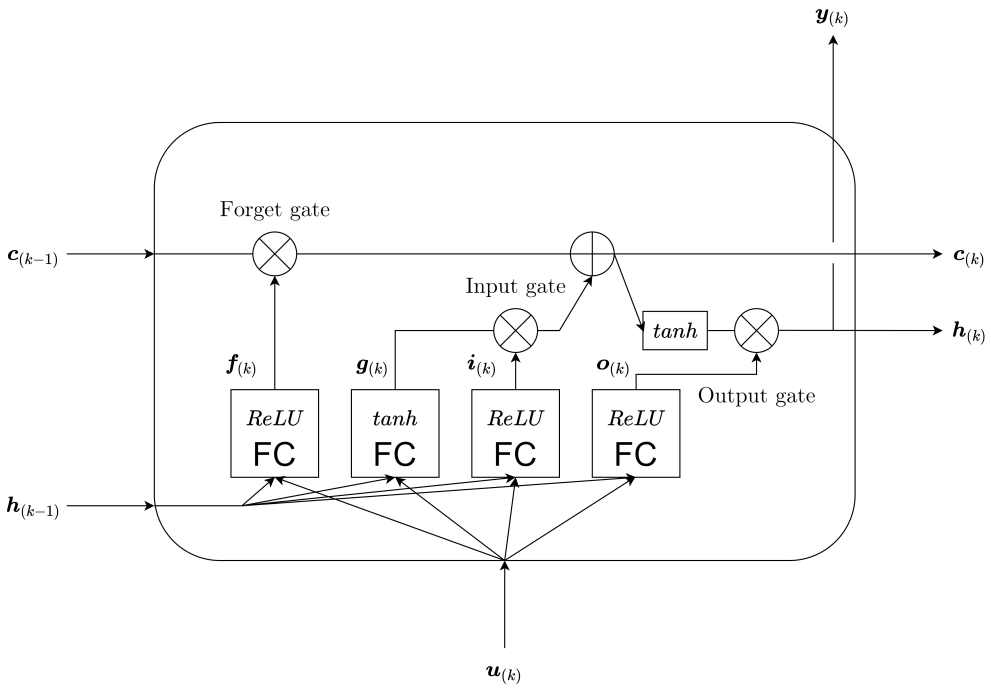


Figure 3.7: Scheme of a LSTM cell.

The input vector $\boldsymbol{u}_{(k)}$ is fed to the cell along with the previous short-term state vector $\boldsymbol{h}_{(k-1)}$. In particular, they both pass through four different fully connected layers, with specific purposes, as described in Géron (2019)[39]:

– the main output layer, as in a traditional RNN cell. Its output $\boldsymbol{g}_{(k)}$, that would be normally used to calculate $\boldsymbol{h}_{(k)}$ and $\boldsymbol{y}_{(k)}$, passes instead through another operation that stores its most important parts in the long-term state and drops the rest (see

below):

$$\boldsymbol{g}_{(k)} = tanh(\boldsymbol{W}_{ug}^T \boldsymbol{u}_{(k)} + \boldsymbol{W}_{hg}^T \boldsymbol{h}_{(k-1)} + \boldsymbol{b}_g) \tag{3.13}$$

– the forget gate. It controls which parts of the long-term state are erased:

$$\boldsymbol{f}_{(k)} = \sigma(\boldsymbol{W}_{uf}^T \boldsymbol{u}_{(k)} + \boldsymbol{W}_{hf}^T \boldsymbol{h}_{(k-1)} + \boldsymbol{b}_f) \tag{3.14}$$

– the input gate. It controls which parts of $\boldsymbol{g}_{(k)}$ are added to the long-term state:

$$\boldsymbol{i}_{(k)} = \sigma(\boldsymbol{W}_{ui}^T \boldsymbol{u}_{(k)} + \boldsymbol{W}_{hi}^T \boldsymbol{h}_{(k-1)} + \boldsymbol{b}_i) \tag{3.15}$$

– the output gate. It controls which parts of the long-term state are read and output at the present time step, both to $\boldsymbol{h}_{(k)}$ and to $\boldsymbol{y}_{(k)}$:

$$\boldsymbol{o}_{(k)} = \sigma(\boldsymbol{W}_{uo}^T \boldsymbol{u}_{(k)} + \boldsymbol{W}_{ho}^T \boldsymbol{h}_{(k-1)} + \boldsymbol{b}_o) \tag{3.16}$$

where:

– $\boldsymbol{W}_{ui}, \boldsymbol{W}_{uf}, \boldsymbol{W}_{uo}, \boldsymbol{W}_{ug}$ are the weight matrices for each of the four layers for their connection to the input vector $\boldsymbol{u}_{(k)}$;

– $\boldsymbol{W}_{hi}, \boldsymbol{W}_{hf}, \boldsymbol{W}_{ho}, \boldsymbol{W}_{hg}$ are the weight matrices for each of the four layers for their connection to the previous short-term state $\boldsymbol{h}_{(k-1)}$;

– $\boldsymbol{b}_i, \boldsymbol{b}_f, \boldsymbol{b}_g, \boldsymbol{b}_o$ are the bias terms for each of the four layers.

Cells (neurons) are, as usual, packed into layers, which make up the overall network. A final dense layer processes the cell states and produces the output of interest for the network. Depending on the purposes, an LSTM neural network can predict more than one step ahead in the future. Moreover, the past time steps used to forecast can range from the single previous time step, to several or all time steps in the past.

The long-term state is a combination of the forget and input gate outputs through an element-wise multiplication $\otimes$. The outputs of the three gate controllers layers range from 0 to 1, since they use logistic activation functions:

$$\boldsymbol{c}_{(k)} = \boldsymbol{f}_{(k)} \otimes \boldsymbol{c}_{(k-1)} + \boldsymbol{i}_{(k)} \otimes \boldsymbol{g}_{(k)} \tag{3.17}$$

$$\boldsymbol{h}_{(k)} = \boldsymbol{o}_{(k)} \otimes tanh(\boldsymbol{c}_{(k)}) \tag{3.18}$$

such that the overall output of the cell is $\boldsymbol{y}_{(k)} = \boldsymbol{h}_{(k)}$. In this case, a single time step is requested to be forecast, using a variable number of previous time steps. Hence, the cell output is transformed in the network's output in the last dense layer as:

$$\hat{\boldsymbol{q}}_{(k)} = \sigma(\boldsymbol{W}_{out}^{T}\boldsymbol{y}_{(k)} + \boldsymbol{b}_{out}) \tag{3.19}$$

where $\boldsymbol{W}_{out}$ and $\boldsymbol{b}_{out}$ are proper weights and bias. Specifically, the LSTMs applied in this work receive injection schedules (not single rates) at wells as inputs and forecast oil or water production rate of a well as output, as in Figure 3.8.

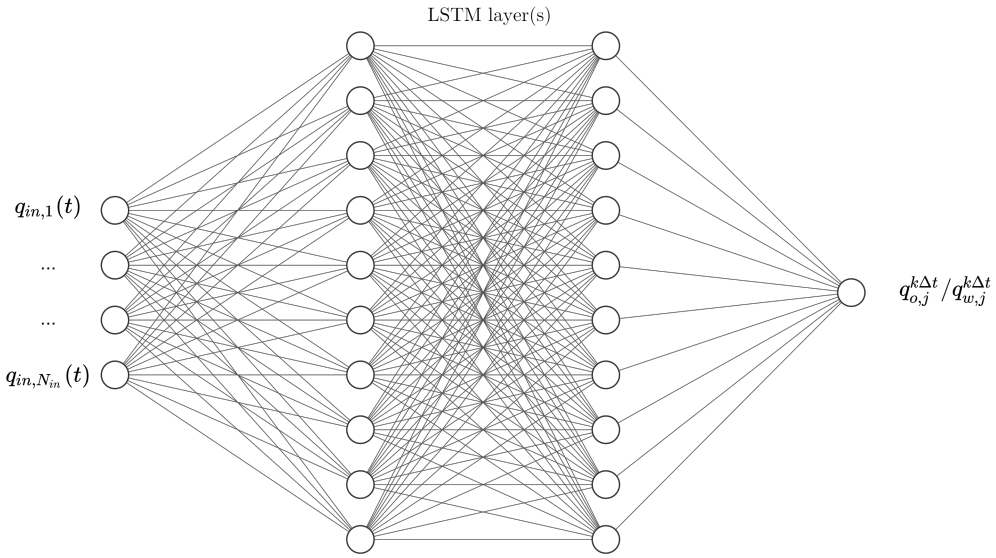


Figure 3.8: Generic oil or water LSTM.

3.3. Waterflooding optimization

Once trained, either a single network or a system of networks can forecast production rates from injection rates, providing all the necessary components to compute the objective function $J(\boldsymbol{u})$ and bypassing the reservoir simulator. The optimization algorithm is then coupled with the neural networks as forward models.

3.3.1. Optimization algorithms

Three optimization algorithms with different operating principles are used. An implementation of an algorithm from the gradient-based, gradient-free, and ensemble-based algorithm families is chosen. The reason behind this choice is to give robustness to the

methodology and compare the different performances on the proposed workflow. The chosen algorithms are the trust-region (gradient-based), genetic (gradient-free) and EnOpt [33] (ensemble-based):

Trust-region algorithm It is a gradient-based optimization algorithm based on quadratic function approximation in a trust region around the evaluation point. The trust region is expanded or contracted depending on the accuracy of the approximation. SciPy's optimization library provides an implementation of the trust-region sequential quadratic programming (SQP) method for equality constraints, by Lalee et al. (1998)[40] and an implementation of the nonlinear interior point trust-region optimizer for inequality constraints, by Nocedal et al. (1999)[41]. The most important parameter is the type of gradient (jacobian) calculation.

Genetic algorithm (GA) It is a population-based, gradient-free algorithm that operates through the basic operations of selection, mutation and crossover. An initial population evolves to reach the optimal solution through the evaluation of a fitness function. The applied implementation can be found in the bibliography [42]. The most important parameters are population size, number of generations, mutation probability, parents portion, crossover probability and crossover type.

EnOpt algorithm It is an ensemble-based algorithm which has gained popularity in the oil production optimization field. An initial ensemble of solutions evolves through the computation of a surrogate gradient and a subsequent line-search method. Developed by Chen et al. (2009)[33], it is applied using an in-company developed implementation. The most important parameter is the step value to be used in the line-search method.

A more detailed explanation of how they work is provided in Appendix D. The choice of a stopping criterion for the optimization algorithms stems from the need to preserve the reduced runtime benefit of the data-driven methodology and on the objective function value (more information is provided in Chapter 4).

3.3.2. Generation and use of initial candidates

For a fair comparison of the selected optimization algorithms, a pool of initial candidates $[\boldsymbol{u}_1^T, ..., \boldsymbol{u}_N^T]^T$ is generated, and then used as initial population for the genetic algorithm, as initial ensemble for the EnOpt algorithm and as different initial guesses for the trust-region algorithm. These profiles are generated in the same way as for the synthetic historical data above, ensuring they also honor the problem's constraints. As represented

in Figure 3.9, for the genetic algorithm the best individual in the final population is selected as solution; for the EnOpt algorithm the best member of the ensemble is selected as solution; for the trust-region algorithm the best solution starting from the different initial guesses is selected as solution.

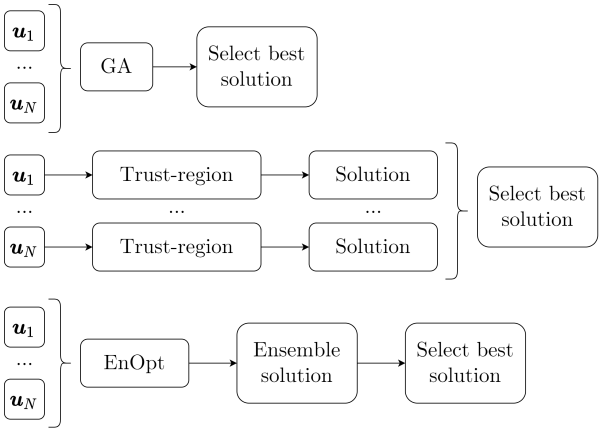


Figure 3.9: Scheme of the use of initial candidates for the three algorithms.

3.3.3. Comparison with existing techniques

Two common production strategies and a state-of-the-art waterflooding optimization software are adopted as a baseline for comparison with the optimization results: the do-nothing strategy, the voidage replacement (VR) strategy and FloodOpt®. Since they require an evaluation of production rates in the future, these scenarios still require the tuned 3D model to be simulated (which in the case of synthetic reservoirs corresponds to the ideal, perfectly known reality). Nevertheless, they make for an interesting comparison benchmark.

Do-nothing In general, if a field is producing at an acceptable rate, with sufficient economic return, the production strategy often consists in simply maintaining production as it is ("do-nothing" or "no-action"). A do-nothing strategy is often used as a baseline to evaluate the performance of waterflooding optimization methodologies for synthetic reservoir cases.

Voidage Replacement In general voidage replacement is the process through which hydrocarbons and water are replaced in the reservoir by fluid injection. The voidage replacement ratio (VRR) is defined as [43]:

$$VRR = \frac{\text{Total injected reservoir volume}}{\text{Total produced reservoir volume}} \tag{3.20}$$

which, in the case of a simplified waterflooding oil-water system reduces to:

$$VRR = \frac{\sum_{i=1}^{N_{in}} q_{in,i} B_w}{\sum_{j=1}^{N_p} (q_{w,j} B_w + q_{o,j} B_o)} \tag{3.21}$$

where flow rates in standard conditions are transformed into reservoir condition through formation volume factors of water B_w and oil B_o (see Appendix A for more details). In voidage replacement (VR), injection rates are managed to reinject a fraction of a unit volume of water for every unit volume of liquid produced. Although there are cases in which specific VRR values are more effective for recovery [43], VRR is often kept around 1. If VRR is unitary, it allows to keep the average field pressure reasonably constant (within constraints), avoiding excessive pressure depletion of the reservoir. The share of the total injection assigned to each well is usually based on the productivity of the production wells they support. In particular, commercial reservoir simulators assign rates based on the concept of well potentials, which represent the production or injection rate that a well would achieve in the absence of any rate constraints, at the current grid block conditions. The acting constraint for a well's potential is thus either its rate or BHP limit. More details can be found in the specific literature [44].

FloodOpt® FloodOpt®, developed by *StreamSim*®[45], is a state-of-the-art commercial software able to perform waterflooding optimization for an oilfield with a given water injection availability target. Optimization is based on sweep efficiency and injected water reallocation, requiring a low number of simulator runs, equal to the number of time step to optimize. FloodOpt® rewards the injector-producer connections with high efficiency and penalizes the ones with low efficiency by a reallocating the available water towards less swept areas of the reservoir. Still, FloodOpt® requires a tuned geological model to run. FloodOpt® is compared to the best performing algorithm on the reservoir of interest. FloodOpt® does not allow to specify an explicit objective function. Thus, for a fair comparison, the best-performing algorithm is adapted in two ways:

– objective function with oil produced only (non-discounted), so (2.3) becomes:

$$J(\boldsymbol{u}) = -\sum_{k=1}^{N_t} \sum_{j=1}^{N_p} q_{o,j}^{k\Delta t} \tag{3.22}$$

with a negative sign since it is minimized;

– target injection at a field level (equality constraint), so (2.5) becomes:

$$\sum_{i=1}^{N_{in}} q_{in,i}^{k\Delta t} = q_{in,F}^{max} \quad \text{for } k = 1, ..., N_t \tag{3.23}$$

while (2.4) is still valid. This additional case, which is referred as “target” case, is performed for comparison purposes only: the increased flexibility of the proposed methodology of specifying the objective function and any type of constraint is, in real life, a benefit to take advantage of. Thus, both comparisons, with the original algorithm and with the modified "target" version, are of interest. A more detailed explanation of how the software works and the rationale behind its heuristic optimization algorithm can be found in Appendix E.

3.3.4. Interpretation of results

After optimization, results are interpreted critically with the aid of commercial software, through a variety of tools: streamline maps, pressure and saturation distributions, single-well contributions to the objective function. Of course, this stage requires the use of the simulator, running the SBO strategy on it: in real-life scenarios, it represents the effect of the optimized strategy on the field as it would occur in the future. For this reason, the final interpretation is only meant for academic purposes and to validate the proposed methodology, since for the chosen synthetic cases the perfect model of the reservoir is available.

4 | Application

In this chapter the proposed methodology is tested on two synthetic reservoir case studies of increasing complexity, which will be referred to as "Streak" and "Olympus" case, or field. In both cases, the reservoir is described, along with its surrogate counterpart. The conditions of the field at the end of the historical period are also outlined.

4.1. Streak case

4.1.1. Geological model

The Streak case is a 2D synthetic reservoir, consisting of 31×31 individual grid blocks of size 80ft. All 961 cells are active. It was conceived as a simple testing tool for reservoir engineering purposes [38, 46, 47]. There are 5 injection wells and 4 production wells placed in an alternating pattern. Geological properties are uniform, except for two high permeability streaks, I01-P01 and I03-P04. The geological model is represented in Figure 4.1, with constant porosity at 0.18 and net-to-gross (or NTG, it is the fraction of gross rock volume occupied by producible oil, i.e. with sufficient porosity, oil saturation, permeability to allow significant production [48]) at 1. Oil and water formation volume factors are taken as constant with pressure, with B_o=1.1066 and B_w=1.0132.

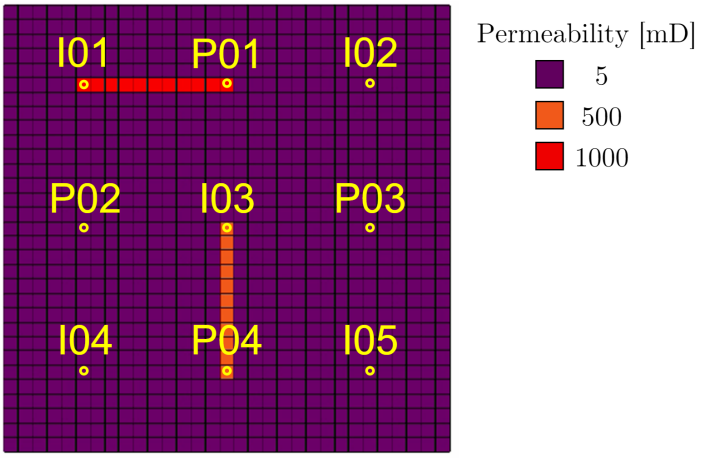


Figure 4.1: Streak: schematic representation of the geological model.

4.1.2. Surrogate model development

The total period considered for reservoir operation, as in Figure 2.2, is 11 years, where the first 8 years were used as historical period, while the remaining 3 are considered as future period for optimization. The historical and the future period are discretized considering time steps of 30 days, but injection rates are not allowed to change with a higher frequency than 90 days. This results in an optimization period of 10 time steps, so considering 5 injection wells, there is a total of 50 control variables.

For this case study the networks developed by Maniglio et al. (2021)[32] will be considered as surrogate models. In particular, the two neural networks described in Chapter 3 are schematized for this field in Figure 4.2 and 4.3. Further reference on the networks' parameters, training process and accuracy can be found in the specific article by Maniglio et al. (2021)[32]. They both receive injection rates and times at all injectors as inputs: one is a traditional ANN and forecasts water cut, while the other is a PINN (ANN+CRMP) and forecasts liquid production rates at all producers.

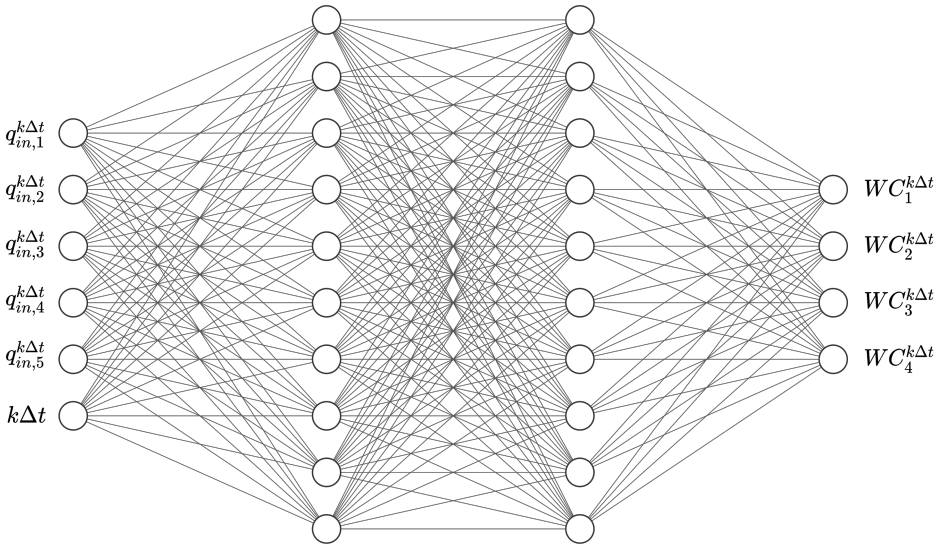


Figure 4.2: Streak: water cut ANN.

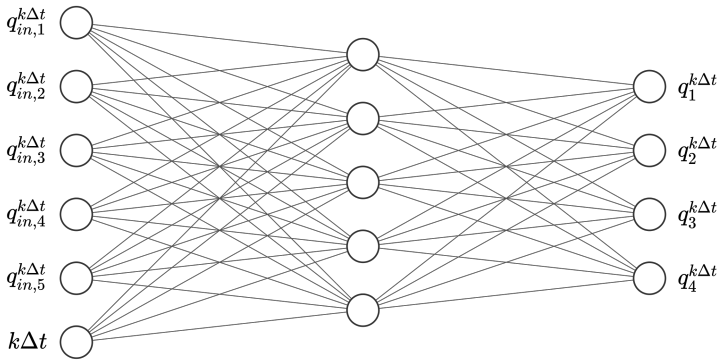


Figure 4.3: Streak: liquid production rate PINN.

4.1.3. Surrogate-based optimization

For this optimization problem, specific costs and oil price are set at r_{in}=1 \$/stb, r_w=1 \$/stb, r_o=70 \$/stb, while the discount factor is d=0.10 (see (2.3)). Bound limits for single injectors and the field are set as in Table 4.1. These values come from minimum and maximum values during the training phase.

Inj. well	$\boldsymbol{q_{in,i}^{min}}$ **[stb/d]**	$\boldsymbol{q_{in,i}^{max}}$ **[stb/d]**
I1	1250	3000
I2	500	1750
I3	200	1750
I4	500	1500
I5	500	2000
Field	-	7600

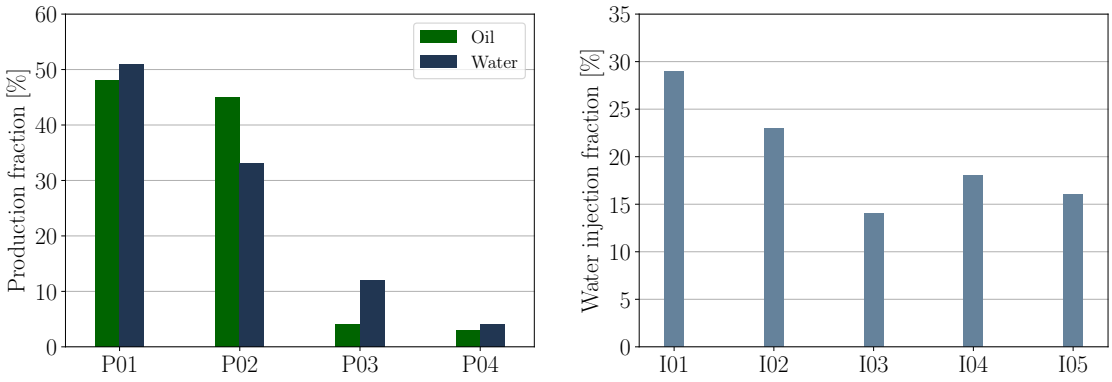
Table 4.1: Streak: bounds for control variables.

For all algorithms, 100 initial candidate solutions are used. For the GA, the mutation probability is set equal to 0.05, parents portion to 0.1, crossover probability to 0.5, and two-point crossover is used. In addition, a 1% elitist ratio is used. The algorithm stops if the variation of the objective function does not improve for more than 10% of the generations, or if the number of generations reaches the maximum value of 1000. For EnOpt, the step value used in the line-search is optimized at every iteration, with a bound reduction of 1%. The algorithm stops if the variation of the objective function does not improve for more than 10 iterations, or if the ensemble of candidates collapses

when all members are the same solution. For the trust-region algorithm, two-point finite difference is used in jacobian calculation. The algorithm stops if the trust-region radius is below 10^{-4} or if the number of iterations reaches the maximum value of 250. Details on the meaning of all the parameters can be found in Appendix D.

Initial situation of the reservoir

The field scenario at the end of the historical time span can provide useful information about the reservoir of interest even before optimization starts. Cumulative bar charts (Figure 4.4) show how, before the optimization period starts, some wells are contributing more to production than others. In particular, P01 and P04 are the major producers, both in terms of oil and water. Conversely, P02 and P03 give only a small fraction of the production. Water cuts on the other hand, at 98% for P4, 97% for P1 and P2, and a 93% for P3, denote how the water breakthrough is already very high (e.g. P01, P04).



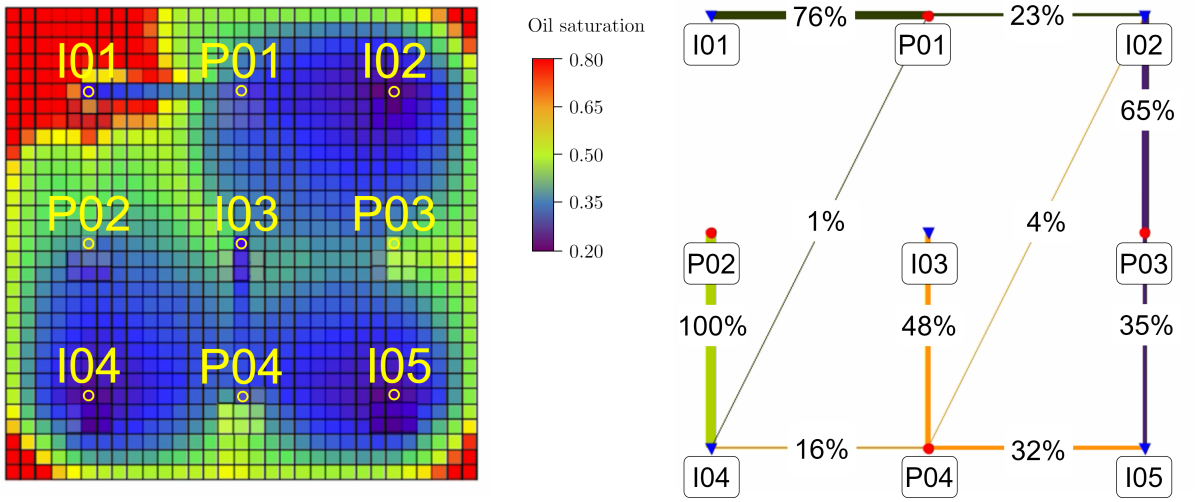
(a) Oil and water production

(b) Water injection

Figure 4.4: Streak: cumulative bar charts at the end of the historical period.

The initial oil saturation map (Figure 4.5a) shows how most of the reservoir has already been flooded (hence the high WCs), while there is a significant remaining volume of oil around I01. Flux maps (Fmaps) are snapshots of the reservoir at a specific moment in time. They display streamlines connecting injectors (sources) and producers (sinks), collapsed in a vector: the thicker the vector the stronger the connection. As such, they allow for a clear visualization of well connections. More details are provided in Appendix E. Figure 4.5b (represented with producers as parents) shows how the high permeability streaks along the I01-P01 and I03-P04 connection act as sinks for any injected water from I1 and I3. On the other hand: I02 supports mostly P01, and, to a lesser extent, P03 and

P04; I4 supports mostly P04, and, to a lesser extent, P02 and P01, I05 supports P04 and, to some degree, P03. The strongest connections are therefore I01-P01, I03-P4, I05-P04 and I04-P04.



(a) Oil saturation distribution (b) Fmap

Figure 4.5: Streak: reservoir condition at the end of the historical period.

Finally, the most efficient well is I01, as reported on the injector efficiency plot in Figure 4.6 (more details about injector efficiency plots can be found in Appendix E).

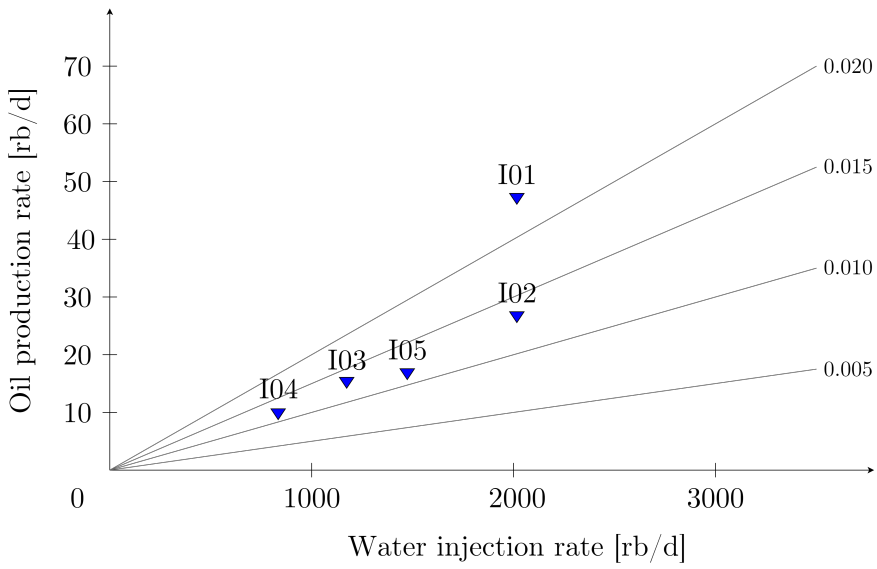


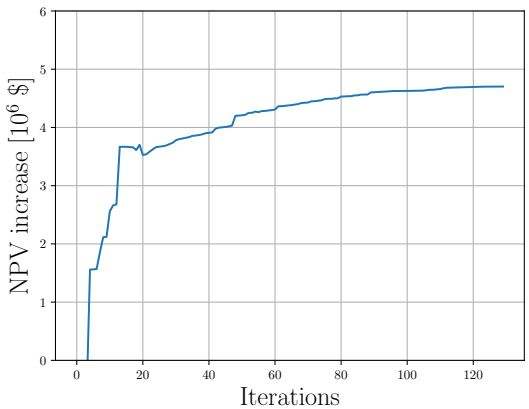
Figure 4.6: Streak: injector efficiency plot at the end of the historical period.

Optimization results

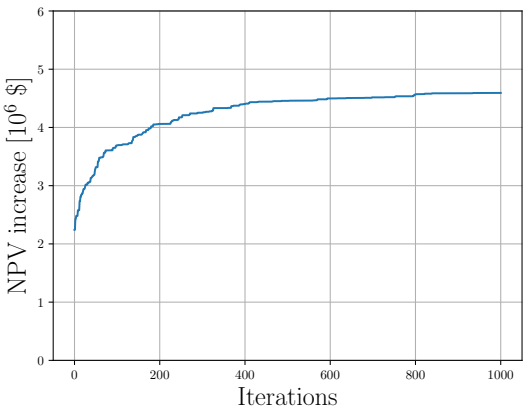
Table 4.2 shows the enhancement achieved by each optimization algorithm with respect to the do-nothing case and the required computational time: the "Simulator" column represents the NPV increase calculated by the full-physics simulator using as input as the optimal injection profiles obtained from the optimization based on the surrogate models (i.e. reality in this case). EnOpt reaches the highest increase and proves to be the fastest, while the GA the slowest, as reported in Figure 4.6 and Table 4.2, where The trust-region computational time is reported for a single candidate, but the different candidates are run in parallel, as in Figure 3.9. In general, the PINN-based forward model of the Streak reservoir achieves a significant reduction in the elapsed time for a single forward evaluation of the objective function, of about 10 times compared to Eclipse® commercial simulator (which for very simple cases as the Streak field, has quite low computational times anyway).

Case	**ANN NPV incr. [10^6 \$]**	**Simul. NPV incr. [10^6 \$]**	**Time [min]**
Trust-region	4.70	4.59	50
GA	4.59	4.56	250
EnOpt	4.77	4.65	45
VR	-	-1.14	-

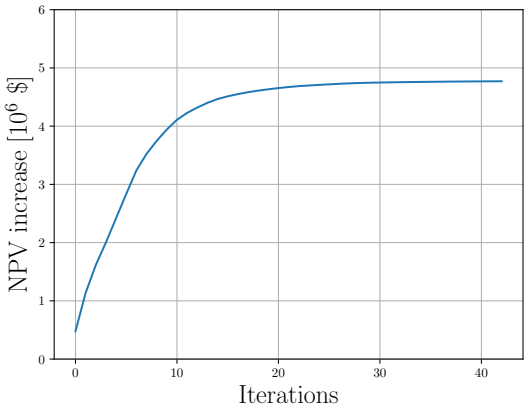
Table 4.2: Streak: computational time and NPV enhancement with respect to the do-nothing case, using ANNs and the simulator, achieved by the obtained optimal solutions.



(a) Trust-region



(b) GA



(c) EnOpt

Figure 4.6: Streak: evolution of the NPV enhancement with respect to the do-nothing case for the three algorithms.

Table 4.3 shows cumulative production of oil and water and injected water. Given the initial underground conditions of the reservoir and its production history, the water cut at all producers is already at very high levels (>90%) at the start of the optimization process. As a consequence, the relative weight of water injection and production costs on the objective function is considerable. For this reason, the VR strategy in this case gives a much lower NPV value compared to the do-nothing case because the total injection rate to balance the total production rate (keeping VRR unitary) results in high costs with limited return.

	Cumulatives [10^6 stb]		
Case	**Water inj.**	**Water prod.**	**Oil prod.**
Trust-region	4.02	3.98	0.102
GA	4.49	4.43	0.115
EnOpt	4.41	4.35	0.114
VR	7.57	7.49	0.108
Do-nothing	7.06	6.97	0.112

Table 4.3: Streak: cumulative injection and production for each simulated strategy.

In particular, most of the oil comes from P01, whose production increases significantly compared to the do-nothing case, as shown in Figure 4.7. In addition, since most of the

production is moved back on the timeline, the NPV is improved not only through the oil production term, but also through the discount factor, which makes oil produced earlier more valuable.

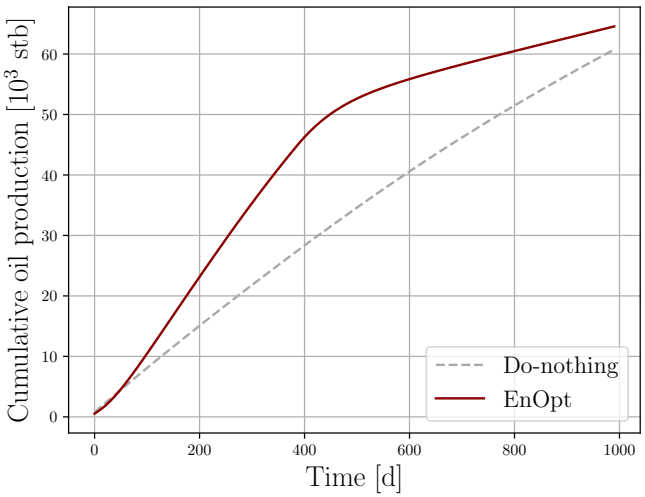


Figure 4.7: Streak: cumulative oil production at P01: EnOpt vs. do-nothing comparison.

On the other hand, the optimization algorithms tend to balance between high oil production (higher profit) and low water injection and production (lower cost), given the high water cut. In particular, while the GA and EnOpt achieve a higher cumulative oil production than the do-nothing, with a significant decrease in the injected water, the trust-region algorithm reaches its NPV increase by reducing injected and produced water even more, penalizing oil production. In other words, the trust-region improves the objective function by slightly reducing oil production, injecting and producing the least water. In real-life scenarios, a lower-producing oil schedule could hardly be implemented: this is, of course, a consequence of both the chosen parameters for the objective function, and the very high water cut at the beginning of the optimization.

As for the optimal control variables, reported in Figure 4.7, injection rates clearly show a common pattern among the different optimization algorithms. In particular, results show high injection rates from I02, I04 and I05 in the first half of the optimization time span, with a decrease in the second half. On the other hand, I01 and I03 are kept at low injection rates all throughout the period by the three algorithms.

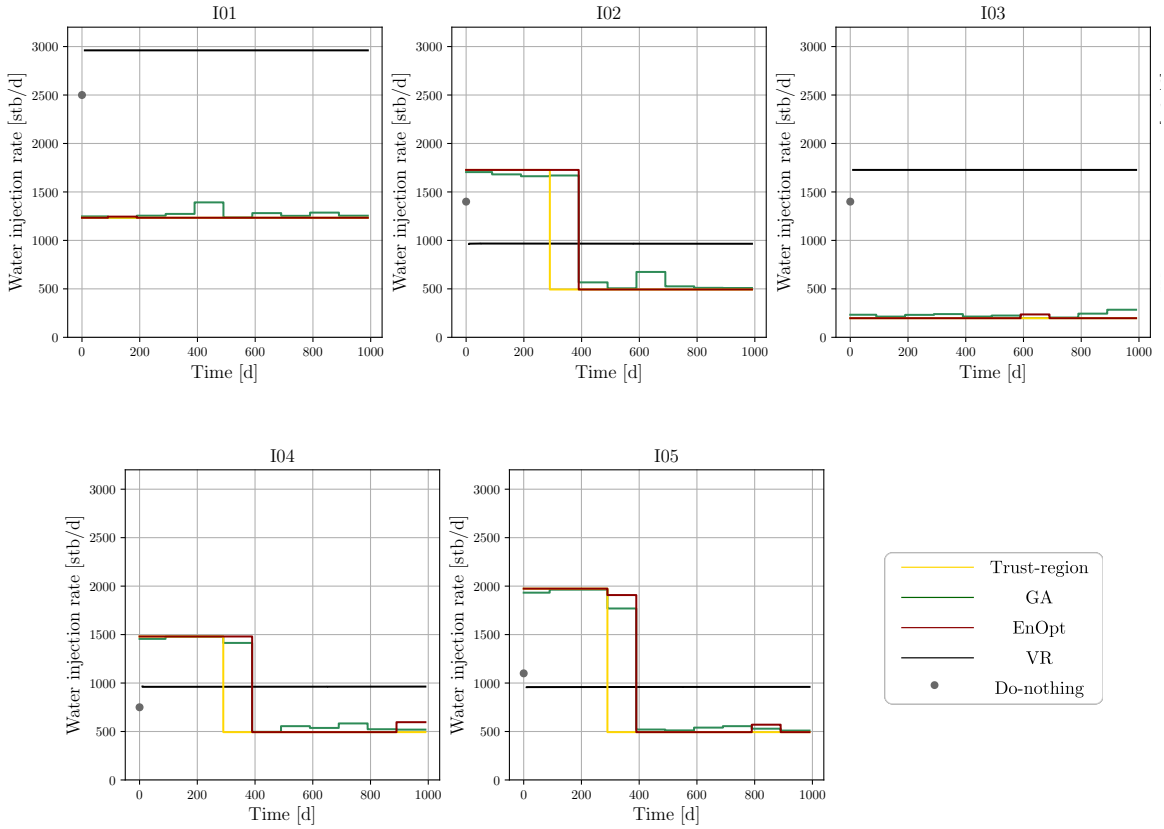


Figure 4.7: Streak: optimal injection rates.

This can be justified by the fact that, as water is injected from I01 and I03, the high permeability streaks act as sinks, draining most of this water towards the producers they are so strongly connected with (P01 and P04). This causes very high water production at the corresponding producers P01 and P04, thus increasing costs. This effect is more pronounced for the 1000mD streak than for the 500mD streak, as expected. This highlights the limitations of injector efficiency plots when deciding where to inject.

The higher injection rates of I02, I04, I05 are maintained by the optimized strategy until the water cut reaches excessive values for economic production, when the price of producing oil becomes higher than the profit. At this point, at about 400 days after the beginning of optimization, even those injection rates are reduced. In real life, reducing injections rate so significantly would likely mean the well needs to be shut completely.

Comparison with FloodOpt

Since EnOpt is the best performing algorithm, it is adapted and compared with FloodOpt's strategy, as explained in Chapter 3. EnOpt reaches a higher cumulative oil produc-

tion with the same cumulative water production and water injection availability target, showing a more efficient allocation of the injected water, as shown in Table 4.4.

	Cumulatives [10^6 stb]		
Case	**Water inj.**	**Water prod.**	**Oil prod.**
EnOpt (target)	7.60	7.44	0.166
FloodOpt	7.60	7.45	0.143

Table 4.4: Streak: cumulative injection and production for each simulated strategy (FloodOpt comparison).

In particular, FloodOpt tends to reward more efficient wells (see 4.6), such as I01, and, to a lesser extent, I02, while penalizing the less efficient ones, I03, I04, with reduced injection rates compared to EnOpt, as reported in Figure 4.7.

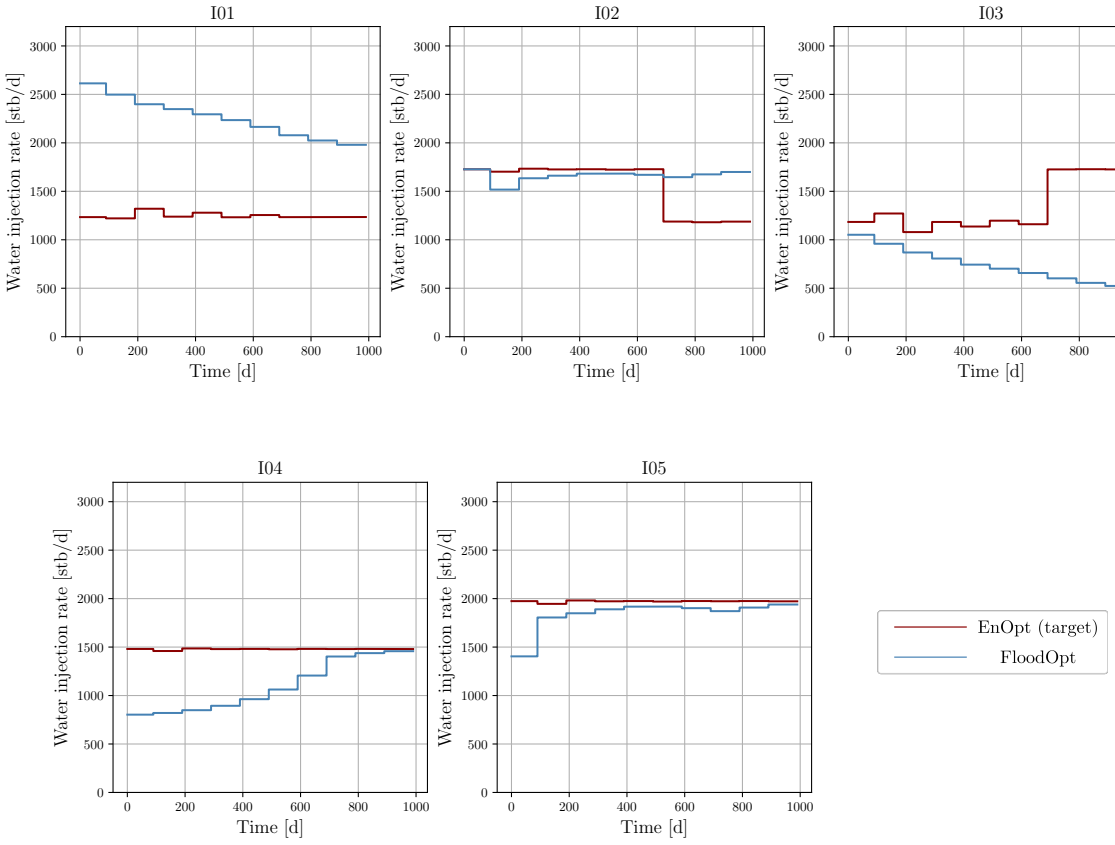


Figure 4.7: Streak: optimal injection rates (FloodOpt comparison).

As shown in the average reservoir pressure maps for EnOpt (target), in Figure 4.8, a significant decrease in injection rates around I01 depressurizes the surrounding area, allowing to drain oil from regions that were previously excluded due to the high permeability channels. These are the along the I02-P01 and I03-P01 connections. Such regions are now at a higher pressure than before.

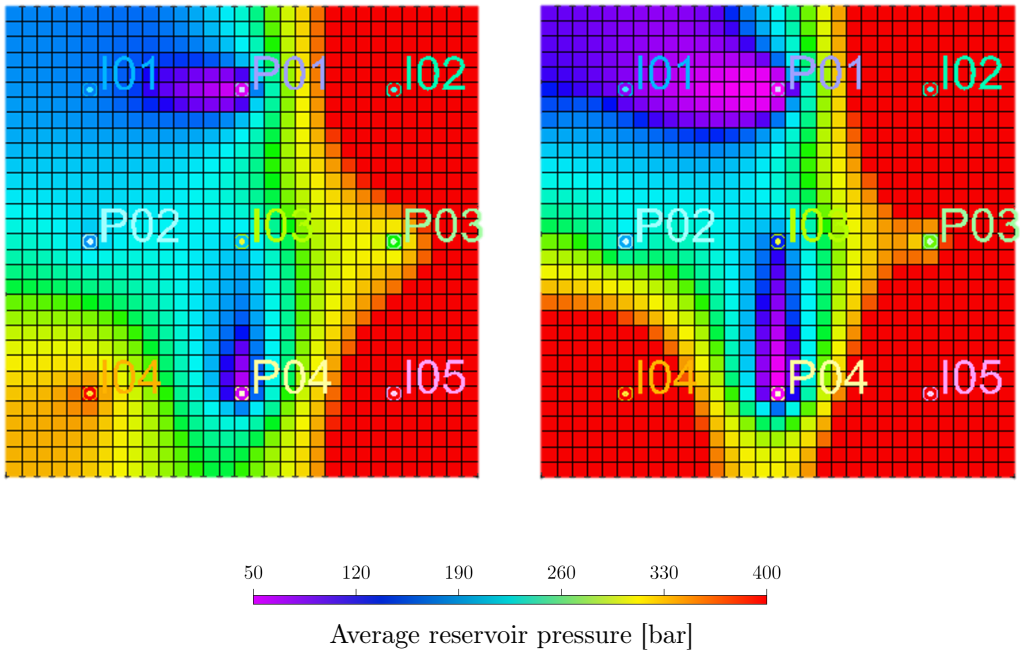


Figure 4.8: Streak: reservoir average pressure maps at the end of the historical (left) and optimization (right) period (EnOpt (target)).

This translates into a change in the oil saturation distribution (reported as a difference with the do-nothing case to highlight the effect, in Figure 4.9), particularly in the region around I01 and I03. Conversely, injecting more in other areas of the reservoir, namely from I02, I04, I05, has the potential to improve production. Since there is a limit on total field water injection, the optimized strategy moves the injection at I04 and I05, which, being further away from most of the leftover oil, act as disposal wells: water can be injected without causing a huge increase in water production rate, since they are far from the high permeability streaks, pressurizing the rest of the region at the same time.

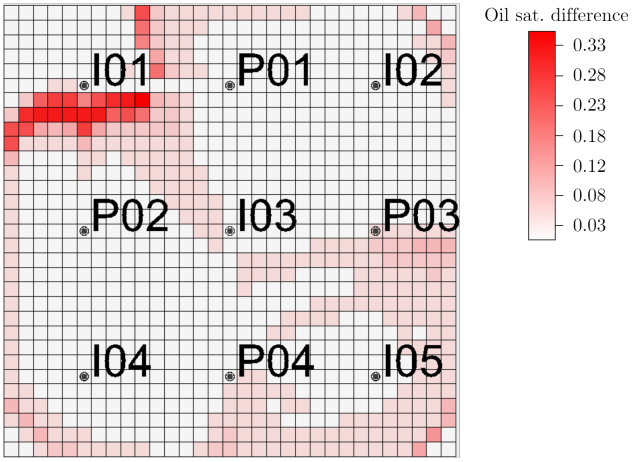
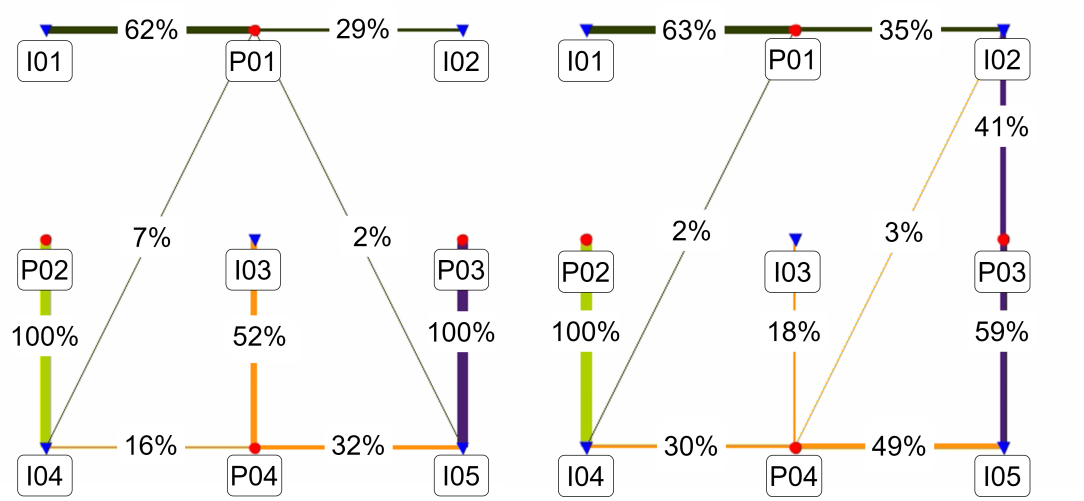


Figure 4.9: Streak: oil saturation difference with do-nothing (EnOpt (target)).

Fmaps for the do-nothing case maintain the same structure as it was at the beginning of the optimization timespan (Figure 4.5b), since the injection rates are not modified, neither in relative nor in absolute terms. Due to the already high WCs at producers, the margin for improvement when the optimization period starts is limited: still, the optimization procedure manages to enhance the injection strategy. By looking at FMpas from a producer perspective (Figure 4.10), it is clear how FloodOpt's strategy tends to reward efficient wells (see 4.19), as seen in the optimal injections' analysis. Apart from the unavoidable sinks along the high permeability streaks, the heuristic algorithm reinforced the I04-P04, I02-P01 and I05-P03. On the other hand, it slightly weakened the I02-P03, I02-P04, I05-P03 and I04-P02 connections.



(a) EnOpt (target) (b) FloodOpt

Figure 4.10: Streak: Fmaps for each simulated strategy.

The better management, as seen above, mainly comes from concentrating I04's water onto the productive P04, letting I05 then support P03, such that I02 can push on P01 to displace oil previously swept areas. The EnOpt target algorithm has a similar layout, but with a more balanced distribution of I02's water towards its neighbors P1 and P03. For completeness, Table 4.5 shows the NPV increased for the FloodOpt comparison, even though the objective function is the cumulative oil production.

Case	ANN NPV incr. [10^6 \$]	Simul. NPV [10^6 \$]	Time [min]
EnOpt (target)	2.28	2.46	30
FloodOpt	-	0.93	10

Table 4.5: Streak: computational time and NPV enhancement with respect to the do-nothing case, using ANNs and the simulator, achieved by the obtained optimal solutions (FloodOpt comparison).

4.2. Olympus case

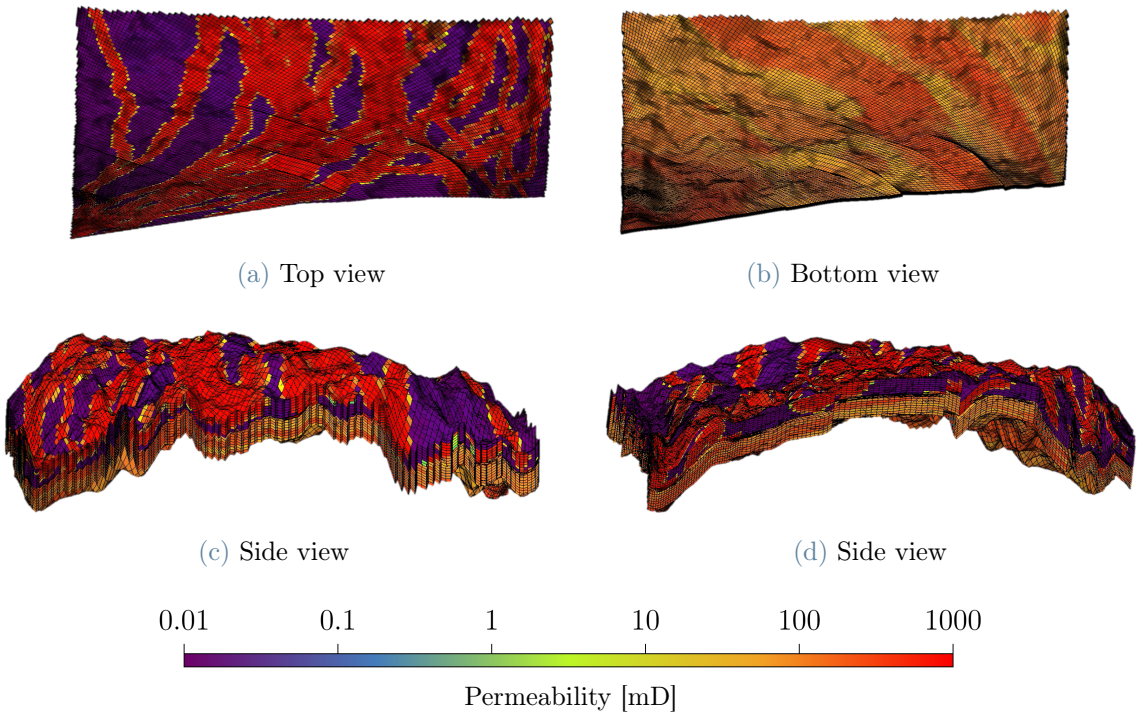
4.2.1. Geological model

The Olympus case is a benchmark synthetic reservoir model developed for the purposes of a benchmark study on field development optimization held in 2017, organized in the context of the ISAPP (Integrated Systems Approach to Petroleum Production) project [49]. It is inspired by an oilfield located in the North Sea.

The field consists of 118×181×16 individual grid blocks of size 50m in the horizontal directions and 3m in the vertical direction. Out of the whole 341,728 grid cells, only 192,750 are active. The field is bounded on one side by a boundary fault. In addition, six minor faults (i.e. a formation break across which there observable displacement [50]) are present. The reservoir consists of two zones, separated by an impermeable shale layer, where most of the inactive cells are located. The top reservoir zone contains high permeability channels embedded in a low permeability matrix, while the bottom reservoir has overall lower permeability. The model is made up of four different facies types (i.e. overall characteristics of a rock unit that reflect its origins and differentiate it from others [50]), as reported in Table 4.6. The model is shown in Figure 4.10 and more in detail with well locations in Figure 4.11. Water formation volume factor is taken as constant with pressure at B_w=1.0132, while oil formation volume factor is variable with pressure, with B_o in the range 1.1069-1.0916.

Facies	Zone	Por. ranges	Perm. ranges [mD]	NTG
Channel sand	Top	0.2-0.35	400-1000	0.8-1
Shale	Top, barrier	0.03	1	0
Coarse sand	Bottom	0.2-3	150-400	0.7-0.9
Sand	Bottom	0.1-0.2	75-150	0.75-0.95
Fine sand	Bottom	0.05-0.1	10-50	0.9-1

Table 4.6: Olympus: geological properties.



(a) Top view (b) Bottom view

(c) Side view (d) Side view

Figure 4.10: Olympus: schematic representation of the geological model.

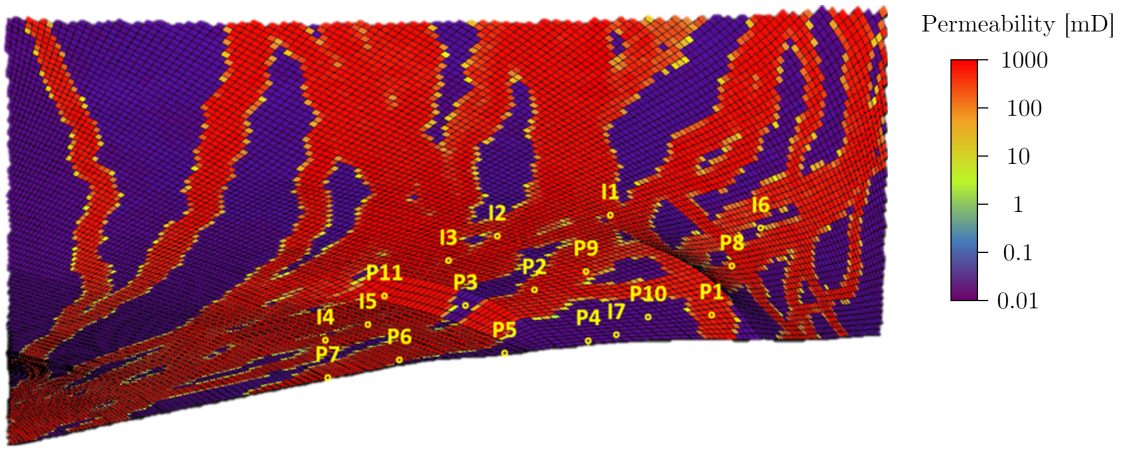
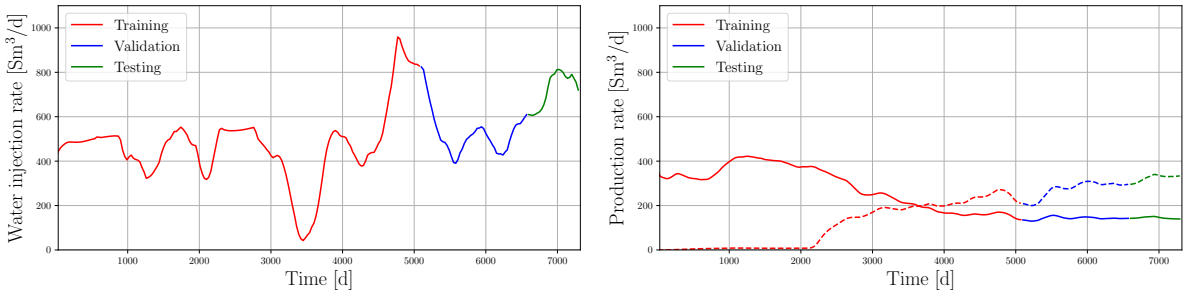


Figure 4.11: Olympus: top view highlighting well positions.

In the Olympus field there are 7 injection wells and 11 production wells.

4.2.2. Surrogate model development

The data generation step described in Chapter 3 is applied to the Olympus field, to produce 7 injection historical schedules and 22 production historical schedules (11 for oil and 11 for water) throughout the whole time span. An example is reported in Figure 4.12, while the rest can be found in Appendix H.



(a) Water injection

(b) Oil (solid) and water (dashed) production

Figure 4.12: Olympus: example of input and output (b) profiles from the generated dataset.

The number of wells and the more complex geology require the direct integration of time dependency into the surrogate model, differently from the simpler Streak case. The total period considered for reservoir operation, as in Figure 2.2, is 20 years, where the first 18 years are used as historical period, while the remaining 2 are considered as future period for optimization (see Figure 4.12). The historical and the future period are discretized

considering time steps of 30 days, but injection rates are not allowed to change with a higher frequency than 90 days. This results in an optimization period of 8 time steps, so considering 7 injection wells, there is a total of 56 control variables.

Two LSTM networks are trained for each production well, one for water and one for oil, for a total of 22 networks. They all receive injection rates at all injectors as inputs and forecast oil and water production rates at each producer, as shown in Figure 4.13. The datasets consist of 7 injection profiles and 22 production profiles, resulting in 170, 50 and 24 time steps for training, validation and testing, respectively.

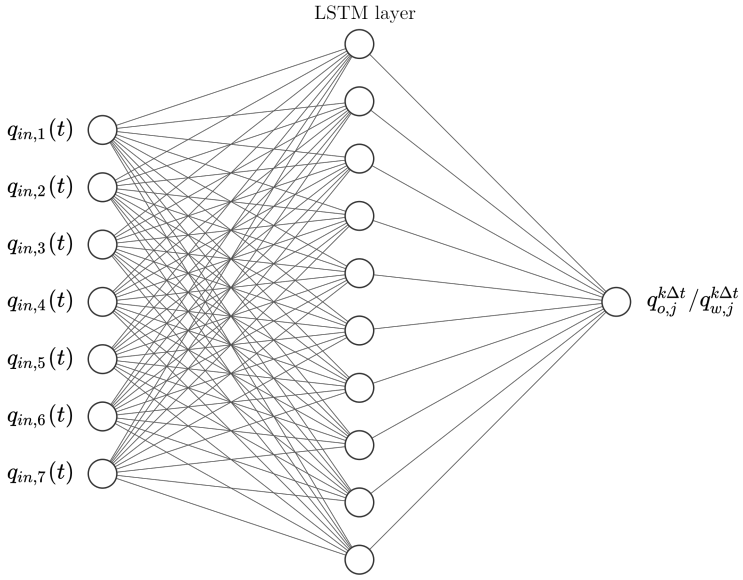


Figure 4.13: Olympus: generic LSTM network surrogate.

The Adam optimizer is used as training algorithm: more details can be found in Géron (2019)[39]. The hyperparameters for the networks are chosen based on a sensitivity analysis carried out using the values reported in Table 4.7 (where "past time steps" refers to the number of previous time steps used for forecasting by the LSTM), resulting in the NRMSEs reported in Table 4.8. It should be noted that the normalization is computed with the same range for training, validation and testing.

Hyperparameter	Values
Hidden layers	1, 2
Neurons per layer	10, 20
Past time steps	3, 5
Learning rate	10^{-4}, 10^{-3}, 10^{-2}
L1 regularization	0, 10^{-2}, 10^{-1}
L2 regularization	10^{-3}, 10^{-2}, 10^{-1}

Table 4.7: Olympus: LSTM hyperparameters tested.

	Water			Oil		
Prod. well	**Training**	**Validation**	**Testing**	**Training**	**Validation**	**Testing**
P1	0.74	1.70	5.54	1.72	1.64	5.28
P2	1.35	1.76	1.82	1.90	1.13	1.91
P3	1.05	3.77	8.26	0.65	2.64	2.00
P4	0.84	4.46	7.87	1.07	1.73	1.28
P5	0.51	2.57	9.32	1.21	0.94	2.75
P6	0.12	0.95	4.76	1.43	0.54	0.44
P7	0.28	1.40	1.13	1.34	1.34	2.50
P8	1.06	2.14	2.92	1.35	0.55	1.98
P9	0.80	5.44	3.47	1.49	2.35	1.89
P10	0.38	2.41	2.12	0.61	0.38	0.57
P11	1.45	0.52	2.96	0.37	1.17	0.91
Avg.	0.78	2.47	4.56	1.19	1.31	1.96

Table 4.8: Olympus: LSTMs NRMSE [%] on water and oil forecast.

A straightforward way to quantify the LSTM model uncertainty is bagging. In this work, an ensemble of models is created by developing the ANNs several times considering the same data, but changing the splitting ratios of the training and validation subsets, maintaining the testing subsets constant. Since the LSTM is developed with time series, the splitting should respect the time sequence in the training, validation and testing. The original 90% of data for training and validation is now randomly split between 65-80% for training and 25-10% for validation, for a total of 10 trials. Using an ensemble of models in the forward evaluation of the objective function can, on one hand, improve

robustness. On the other hand, it could significantly increase the computational time of the optimization, which would also defeat the original purpose of the surrogate, i.e. reducing the computational load of the full-physics simulation. Hence, for computational time reasons, this test is applied to a single well, for oil production only. The chosen well for this test is P11, which the most important well in terms of oil and water production throughout the historical time period (see Figure 4.15). The results are reported in Figures 4.9 and 4.14. Since training and validation datasets are of variable length, in Figure 4.14 they are plotted with the same color, while testing has a fixed dataset, hence a different color.

	NRMSE [%]		
Trial	**Training**	**Validation**	**Testing**
#1	1.81	2.87	2.75
#2	1.52	2.12	4.59
#3	0.91	1.55	1.43
#4	0.98	0.39	0.95
#5	1.02	5.08	0.40
#6	1.24	0.64	0.74
#7	0.89	4.93	1.11
#8	1.27	0.76	1.09
#9	1.81	1.52	4.89
#10	1.62	0.62	4.63
Avg.	1.31	2.05	2.26

Table 4.9: Olympus: LSTMs NRMSE [%] for bagging on P11.

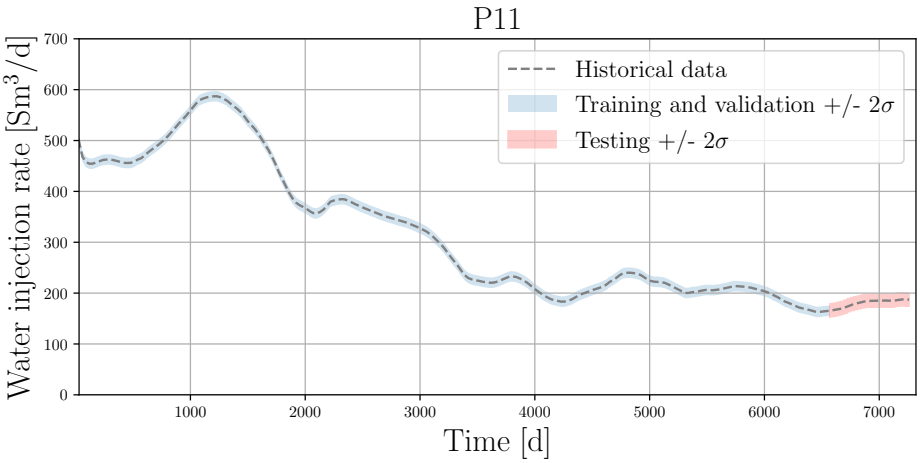


Figure 4.14: Olympus: plot for P11's oil production dataset with standard deviation among different models resulting from bagging on training and validation.

Results show the prediction is accurate even when the dataset is randomly split in different ways between training and validation for the tested well, P11.

4.2.3. Surrogate-based optimization

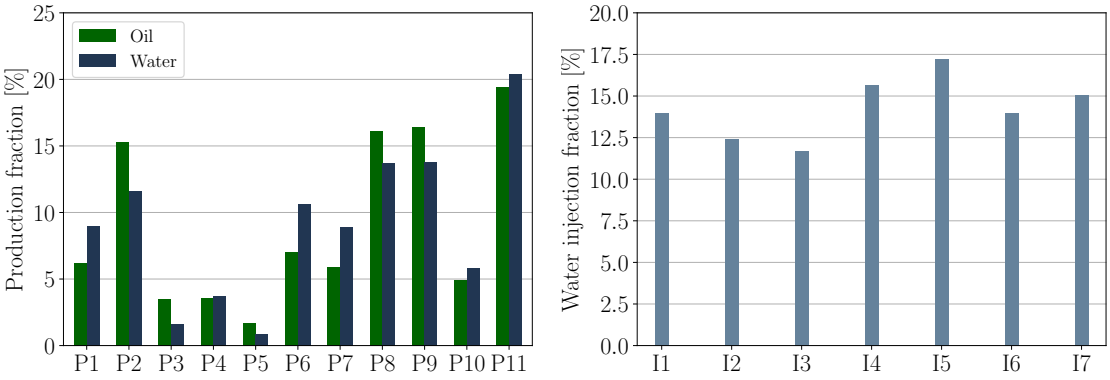
For this optimization problem, specific costs and oil price are set at r_{in}=1 \$/stb, r_w=1 \$/stb, r_o=70 \$/stb, while the discount factor is d=0.10 (see (2.3)). Bound limits for single injectors are set at 0 and 1000 Sm^3/d, while at field level the maximum value allowed is 5000 Sm^3/d. These values come from minimum and maximum values during the training phase. The injection rate variation limit for the constraint in (2.6), modeled as a penalty function on the NPV, is set at 200 Sm^3.

For all algorithms, 50 initial candidate solutions are used. For the GA, the mutation probability is set equal to 0.05, parents portion to 0.1, crossover probability to 0.5, and two-point crossover is used. In addition, a 1% elitist ratio is used. The algorithm stops if the variation of the objective function does not improve for more than 10% of the generations, or if the number of generations reaches the maximum value of 300. For EnOpt, the step value used in the line-search is optimized at every iteration, with a bound reduction of 1%. The algorithm stops if the variation of the objective function does not improve for more than 10 iterations, or if the ensemble of candidates collapses when all members are the same solution. For the trust-region algorithm, two-point finite difference is used in jacobian calculation. The algorithm stops if the trust-region radius is below 10^{-4} or if the number of iterations reaches the maximum value of 200. Details on the meaning of all the parameters can be found in Appendix D. A sensitivity analysis

on the parameters of the GA and EnOpt algorithm is carried out and is presented at the end of the case study.

Initial situation of the reservoir

The field scenario at the end of the historical time span can provide useful information about the reservoir of interest even before optimization starts. Cumulative bar charts (Figure 4.15) show how, before the optimization period starts, some wells are contributing more to production than others. In particular, P2, P8, P9 and P11 are the major producers, both in terms of oil and water. Conversely, P3, P4, P5 give only a small fraction of the production.



(a) Oil and water production

(b) Water injection

Figure 4.15: Olympus: cumulative bar charts before optimization

Water cuts (Figure 4.16) on the other hand show how for some wells the water breakthrough is already significant (e.g. P4, P5), while for others it is still reasonably low for a mature field (e.g. P1, P3, P6).

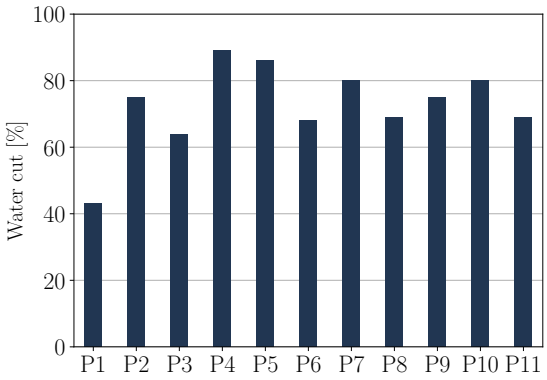


Figure 4.16: Olympus: water cuts at the end of the historical period.

To visualize saturations on a 2D representation of the 3D reservoir, the vertically averaged oil saturation at location (x, y) and time t, $S_o^{x,y}(t)$, is:

$$S_o^{x,y}(t) = \frac{\sum_{z=1}^{N_z^{x,y}} S_o^{x,y,z}(t) PORV^{x,y,z}}{\sum_{z=1}^{N_z^{x,y}} PORV^{x,y,z}} \tag{4.1}$$

.

where $N_z^{x,y}$ is the number of cells in the model over the vertical direction at location (x, y), $S_o^{x,y,z}(t)$ is the oil saturation at location (x, y, z) at time t and $PORV^{x,y,z}$ is the pore volume of grid cell (x, y, z). The initial vertically-averaged oil saturation map (Figure 4.17) shows areas where there is high potential for recovery, such as between I1 and P9, or close to P2, P3, P5. On the other hand, regions such as the I4-P7 connection or the region around I2 and I3 have already been significantly swept.

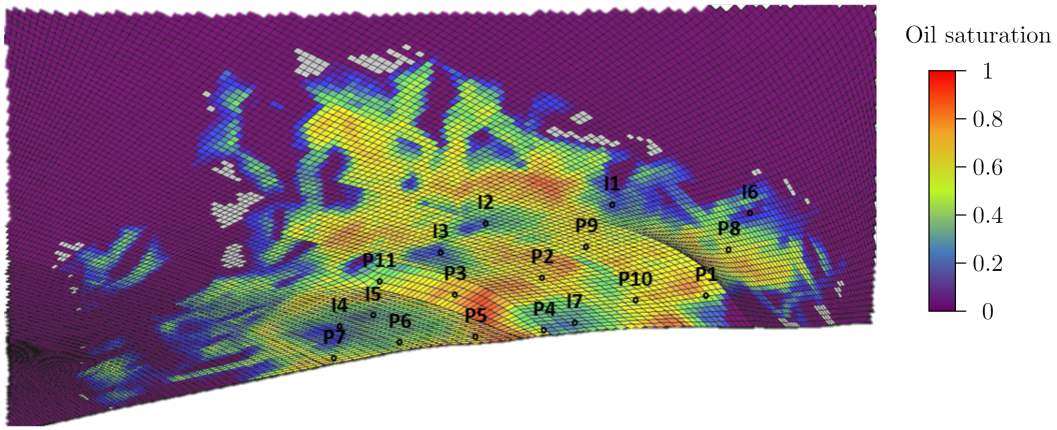


Figure 4.17: Olympus: vertically-averaged oil saturation distribution at the end of the historical period.

Figure 4.18 shows the Fmaps (represented with producers as parents) and main connections before optimization starts. These are, for the most important wells, I1-P9, I2-P9, I3-P2, I7-P2, I5-P11 and I6-P8. The presence of faults, not only the heterogeneous permeability and porosity field, also affects the injector-producer connections.

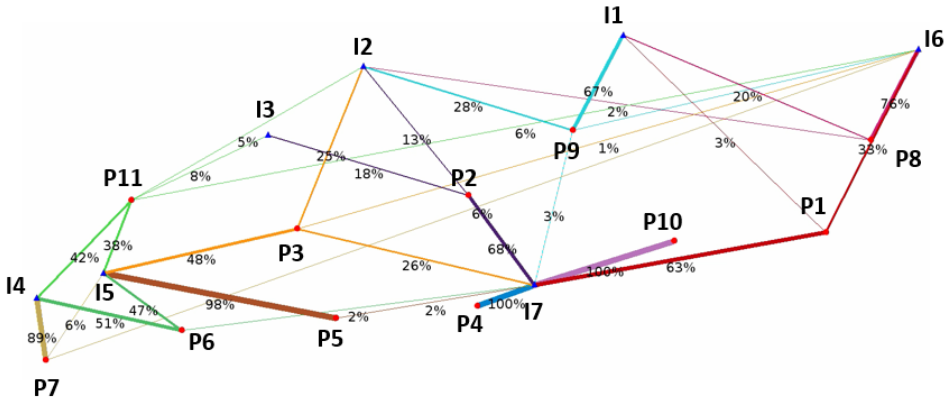


Figure 4.18: Olympus: Fmap at the end of the historical period.

Finally, the most efficient wells are I3, I1, I7 and I6, as reported on the injector efficiency plot in Figure 4.19 (more details about injector efficiency plots can be found in Appendix E).

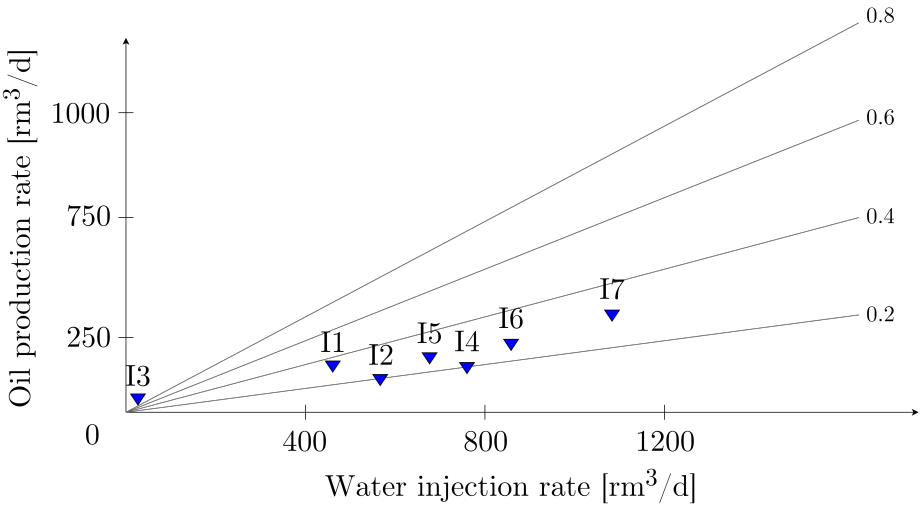
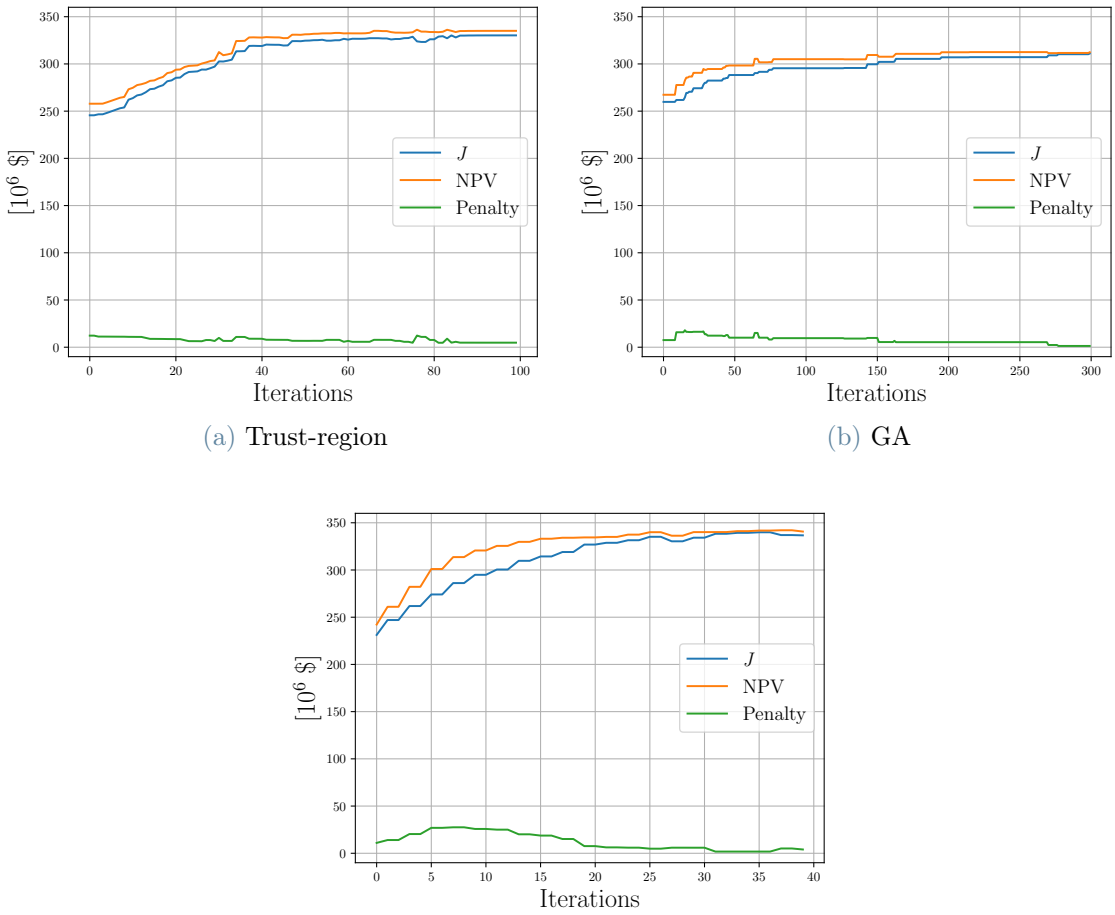


Figure 4.19: Olympus: injector efficiency plot at the end of the historical period.

Optimization results

The more heterogeneous geology and the higher number of wells make the Olympus field a more realistic application compared to the Streak case. This, however, also means that the interpretation of the results is a more complicated task, where the effect of injection schedule on the production is more complex to infer. For this reason, only the most important wells and connections are analyzed. The interpretation task becomes even more complex in real-life brownfields, with dozens of wells and highly heterogeneous geological properties.

Table 4.10 shows the enhancement achieved by each optimization algorithm and the required computational time: the "Simulator" column represents the NPV calculated by the full-physics simulator using as input as the optimal injection profiles obtained from the optimization based on the surrogate models (i.e. reality in this case). EnOpt reaches the highest increase and proves to be the fastest, while the GA the slowest, as reported in Figure 4.19 and Table 4.10. The trust-region computational time is reported for a single candidate, but the different candidates are run in parallel, as in Figure 3.9. In general, the LSTM-based forward model of the Streak reservoir achieves a significant reduction in the elapsed time for a single forward evaluation of the objective function, of about 100 compared to Eclipse® commercial simulator. In this case, the VR strategy yields a better result than the do-nothing case, but lower compared to the algorithms.



(a) Trust-region (b) GA

(c) EnOpt

Figure 4.19: Olympus: evolution of the NPV for the three algorithms.

Case	ANN NPV [10^6 $]	Simul. NPV [10^6 $]	Time [min]
Trust-region	334	288	110
GA	312	286	250
EnOpt	337	289	50
VR	-	238	-
Do-nothing	-	228	-

Table 4.10: Olympus: computational time and NPV, using ANNs and the simulator, achieved by the obtained optimal solutions.

Table 4.11 shows cumulative production of oil and water and injected water. Given the initial underground conditions of the reservoir and its production history, the water

cut at all producers is at a lower level compared to the Streak case, at the start of the optimization process. As a consequence, the relative weight of water injection and production costs on the objective function is also lower. For this reason, all algorithms adopted a strategy where an increase in injected water (within constraints), results in a sufficient increase in oil production to counterbalance the costs. In addition, all algorithms exploited the water availability target almost fully, having injected water at a relatively low cost compared to higher water cut fields, such as the Streak case. In particular, all algorithms achieved a higher cumulative oil production than the base case.

	Cumulatives [10^6 Sm^3]		
Case	**Water inj.**	**Water prod.**	**Oil prod.**
Trust-region	3.60	2.57	0.802
GA	3.56	2.53	0.798
EnOpt	3.55	2.55	0.802
VR	2.70	1.99	0.657
Do-nothing	2.59	1.91	0.630

Table 4.11: Olympus: cumulative injection and production for each simulated strategy.

As for the optimal control variables, reported in Figure 4.18, the injection rates show a common pattern among the different optimization algorithms, which are not, though, always in agreement. Specifically, the trust-region and the EnOpt algorithm show more similar optimal profiles, since they are both based on a gradient or a gradient surrogate. However, the GA, due to its inherently stochastic nature, reaches a different, more oscillating solution, especially for I2, and I5, which have lower rates all throughout the optimization period, and I4, which has higher rates instead. The EnOpt algorithm tends to approach boundary limits at the top and bottom, due to the collapse of its ensemble population when members get too similar to each other. Overall, all optimal profiles have higher values for I1, I3, I6 and I7, lower values for I2 and I4. All algorithms agree on I5, I6, I7; more visible differences are found for I1, I2, I3; I4 is the most variable of the profiles among the solutions.

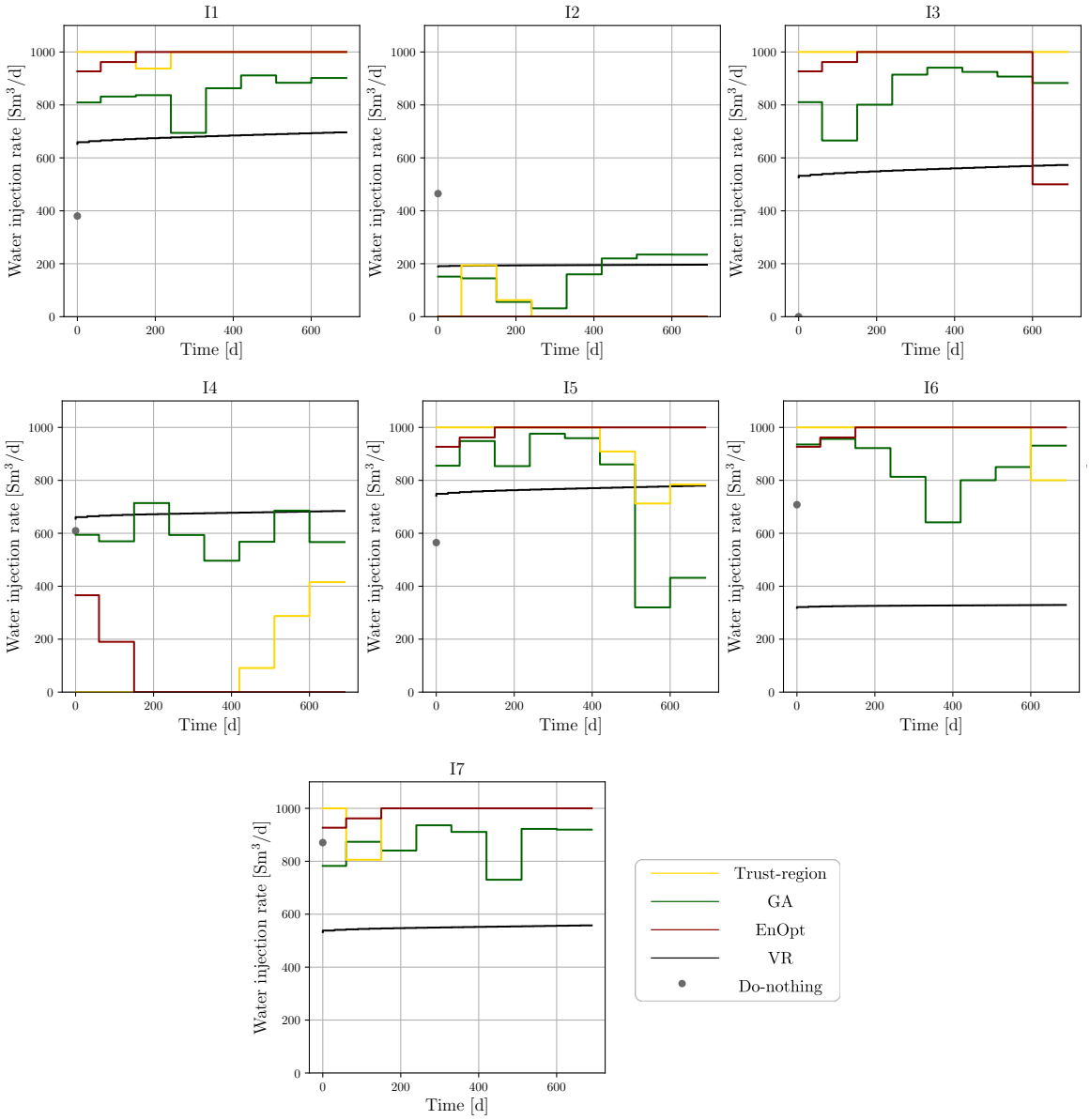


Figure 4.18: Olympus: optimal injection rates.

With respect to the do-nothing I1 injects more, being connected mostly to P9, which is the biggest producer, I2 injects less, favoring the movement of I1's water toward P9, I3 injects more, being connected to P2 and P11, I4 injects less, being connected to P6 (it is more convenient to penalize this connection to inject more at I5, which is connected to P7, a fairly productive well), I6 injects more in order to support P8 more strongly, and I7 stays at high rates as before.

Oil volume moved plots help understand where oil moved from or into a specific cell between two moments in time, such as before and after optimization. Hence, oil volume

moved is a quick way to show how much and where oil has moved over time. The volume of oil moved $V_{mob}(t)$ at time t is defined as:

$$V_{mob}(t) = \sum_{k=1}^{N_z^{x,y}} (S_o^{x,y,z}(t) - S_{or}^{x,y,z}) PORV^{x,y,z} NTG^{x,y,z} \tag{4.2}$$

and the difference in oil volume moved from before to after optimization is:

$$\Delta V_{mob} = V_{mob}(t_0) - V_{mob}(t_{N_t}) \tag{4.3}$$

where $NTG^{x,y,z}$ is the net-to-gross of cell (x, y, z) and $S_{or}^{x,y,z}$ is the residual oil to water saturation of cell (x, y, z).

For the case of EnOpt, the map is reported in Figure 4.19. Red areas show where oil moved from the cell, blue areas show where oil moved into the cell. Although the specific pathways cannot be fully reconstructed, from mobile oil maps additional and valuable information can be inferred. Compared to the initial situation, a significant movement of oil takes place in the region between I1 and I2. I2, being practically cut off from injection, depressurizes its surrounding area, draining oil from other regions of the reservoir. A similar process occurs around I4, which has also been cut off. A considerable volume of oil moves in the region between I1 and P9, which is the second most important producer. A similar process occurs between I6 and P8, as well as between I3 and P11.

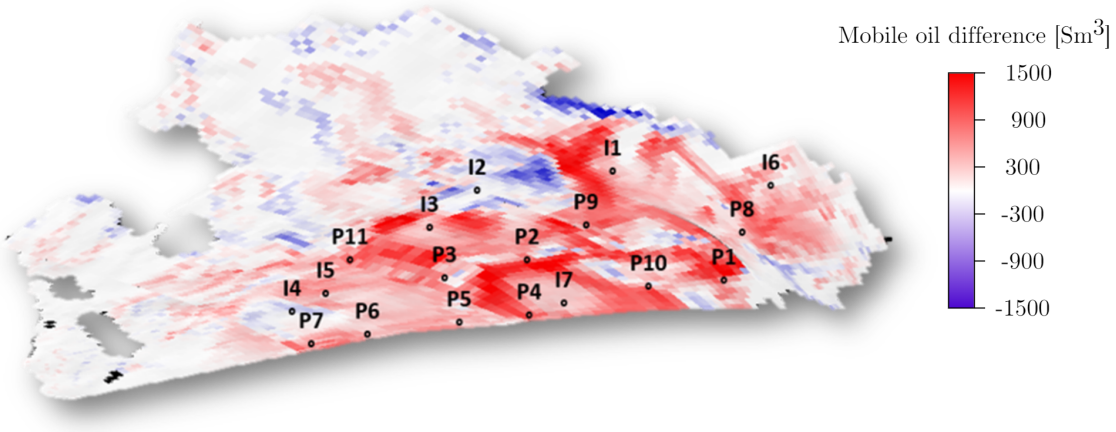


Figure 4.19: Olympus: mobile oil map (EnOpt).

By mapping the difference between the EnOpt and do-nothing mobile oil maps, another useful representation is obtained, reported in Figure 4.20. The green areas show regions where, at the end of the optimization process, there is less oil than for the do-nothing case

(which is hopefully produced, or moved elsewhere), red areas where there is more (which is left underground). The sum of red and green areas is negative, confirming how EnOpt reaches a higher cumulative oil production. In particular, the moved oil that is missing at the end between I1 and P9 is likely partially produced towards P9 (where the red region can be seen) and partially left off (shutting down I2, and depressurizing the region, some oil is probably drained). The compromise between the two directions results in a higher overall NPV and oil production than the do-nothing strategy. Finally, around P11 and between P2 and P9 much more oil is produced in the optimized scenario.

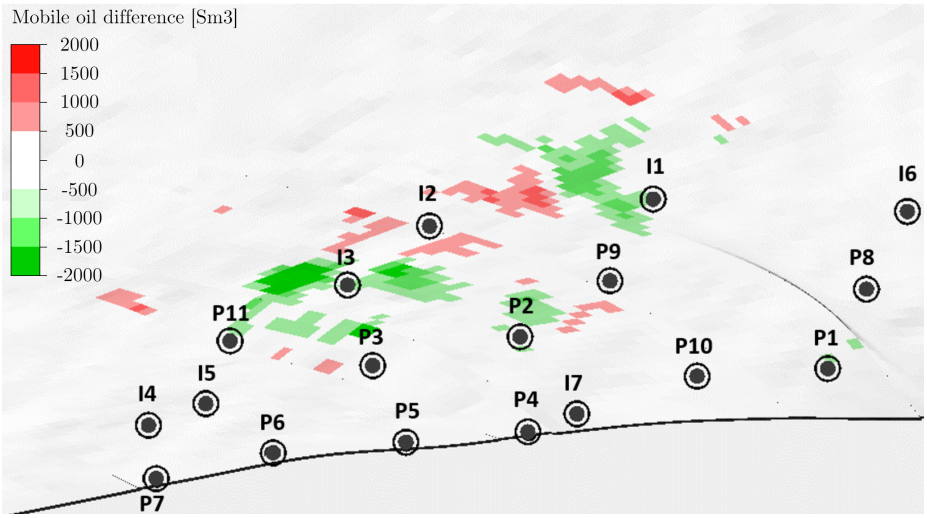
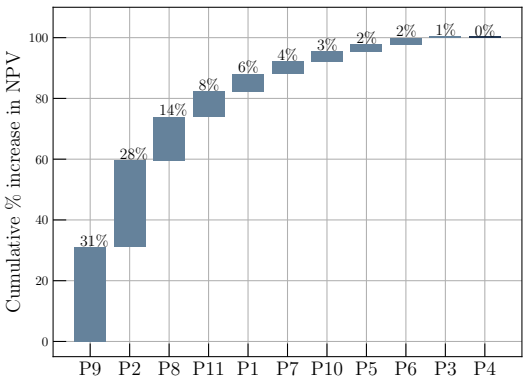
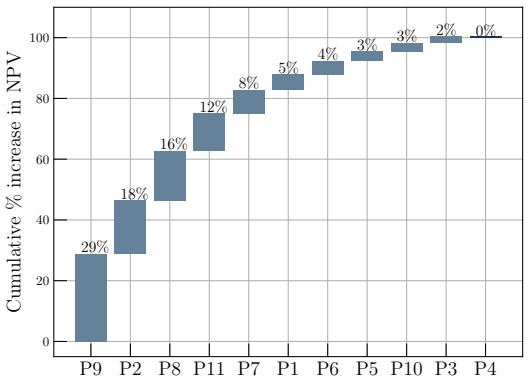


Figure 4.20: Olympus: mobile oil difference map, with respect to the do-nothing case (EnOpt).

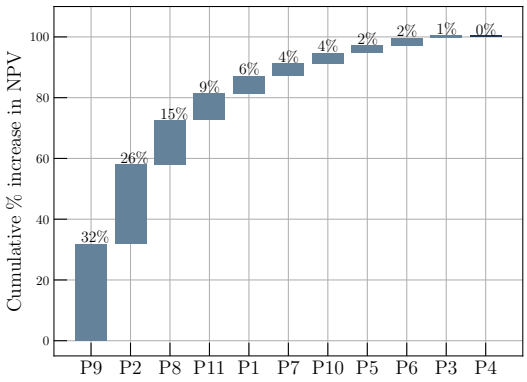
A useful approach when the number of wells makes it hard to interpret results at a first glance is to identify the most important wells in terms of their contribution to the objective function. A fast way to visualize this is a cumulative plot (Figure 4.20), where the relative weight of each well in increasing the objective function, compared to its do-nothing counterpart, is reported. This allows to understand which wells are the major contributors to the objective function improvement can bring little improvement to the objective function.



(a) Trust-region



(b) GA



(c) EnOpt

Figure 4.20: Olympus: cumulative well contribution to the NPV increase, compared to do-nothing case, for the three algorithms.

As it was during the historical period, the wells that contributing the most to the NPV increase are P9, P2, P8, P11, for all three algorithms. Fmaps before optimization (4.18) showed the injectors that are more strongly connected to such producers. They are P11-I4 and I5, P9-I1 and I2, P8-I6 and I1, P2-I7, I3 and I2.

For the trust-region and EnOpt, these producers contribute to about 80% of the NPV increase compared to the do-nothing case. On the Fmaps, it is clear how these wells are more strongly connected to the injectors with the highest rates. The GA's solution also requires P7 to reach the same 80% threshold: these wells is, as seen on the Fmaps, strongly connected to I4, which is exactly the injector with the most different optimal profile compared to the trust-region and EnOpt, where it has a lower rate. Conversely, in the GA's solution P2 has a lower relative weight to the NPV: P2 is mostly connected

to I7 and I3, which, have, indeed, lower rates compared to the trust-region and EnOpt. Finally, for the GA I2's rates are slightly higher: this reflects onto P3, rather than on P9 or P2, which are the other wells connected to I2. P3 has a marginal relative weight on the oil production, hence the lower NPV for the GA.

It is worth noticing how P4 has a practically negligible weight: this means it is producing approximately at the same profit of its do-nothing counterpart. The reason, as on the Fmaps, lies in its strong connection to I7, which is already injecting at high rates to support P2 and P1, but is actually making no additional profit on P4. In addition, I4 is well-connected to P6, which is the well with the highest WC, up to 90%, hence its lower economic value. To be truthful, also P4 is in similar conditions, but I7, its connected neighbor, already injects to support P2.

Maps for the do-nothing case maintain the same structure as it was at the beginning of the optimization timespan (Figure 4.18), since the injection rates are not modified, neither in relative nor in absolute terms. By comparing the Fmaps before and after optimization with the best performing algorithm EnOpt (since the layout before practically corresponds to the final layout of the do-nothing case), as in Figure 4.21, it can also be said that I1 is more strongly connected to P9 and P8, the second and third most productive wells respectively. On the other hand I2 is cut off in its connection with P9 to favor I1. I3 is more strongly connected to P2, P3 and P11, penalizing its connection with I7, which is already supported by P1. I4 is cut off in its connection with P6 to favor I5. In turn, I5 is more strongly connected to P6. I6 maintains the same connections, slightly more with far away wells P3, P7, P11. Finally, I7 reduces its support to its surrounding wells to support P4.

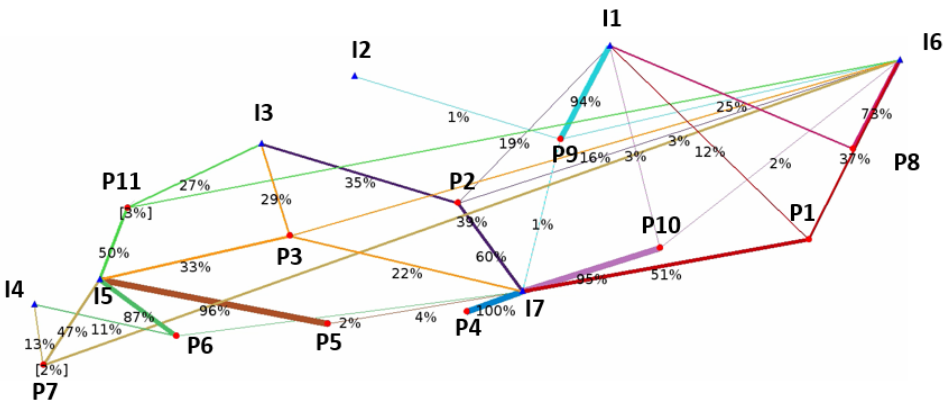


Figure 4.21: Olympus: Fmap (EnOpt).

The VR strategy, compared to the optimization results, adopts more distributed injection

rates, keeping the VRR ratio close to the unitary value (Figure 4.18). With respect to the do-nothing case, injection rates are maintained closer to their historical values, with the slight exceptions of stronger I1-P8 connection (penalizing I6), stronger I5-P3 connection (penalizing I2), stronger I6-P8 connection (penalizing I1), stronger I1-P9 and I4-P9 connections, stronger I2-P9 connection (penalizing P1), as shown in Figure 4.22.

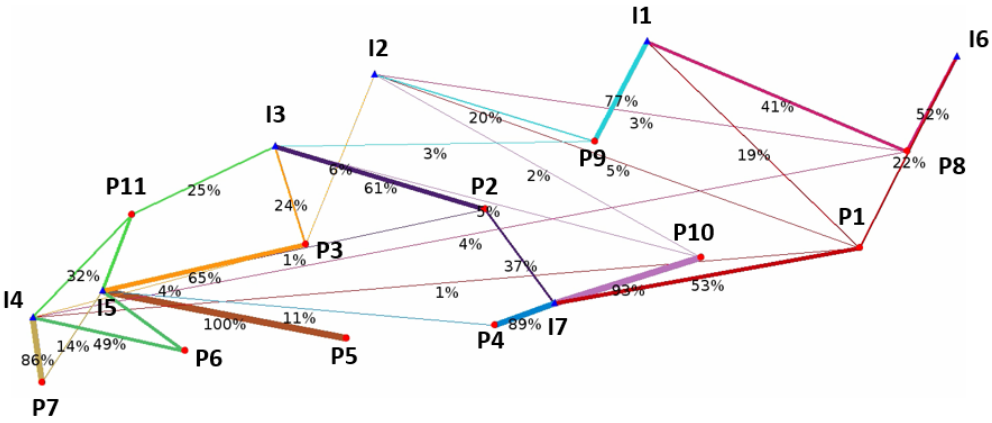


Figure 4.22: Olympus: Fmap (VR).

Mobile oil maps (Figure 4.23) show how the VR strategy tends to maintain a smoother layout, without extreme conditions (less strongly red or blue zones).

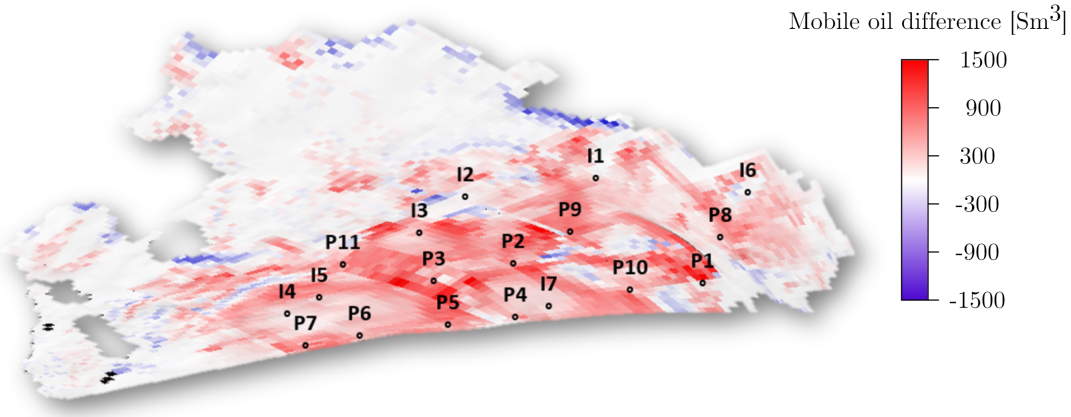


Figure 4.23: Olympus: mobile oil map (VR).

Comparison with FloodOpt

Since EnOpt is the best performing algorithm, it is adapted and compared with FloodOpt's strategy, as explained in Chapter 3. EnOpt reaches a similar cumulative oil produc-

tion with the same cumulative water production and water injection availability target, showing an efficient allocation of the injected water. For completeness, Table 4.12 shows the NPV increased for the FloodOpt comparison, even though the objective function is the cumulative oil production.

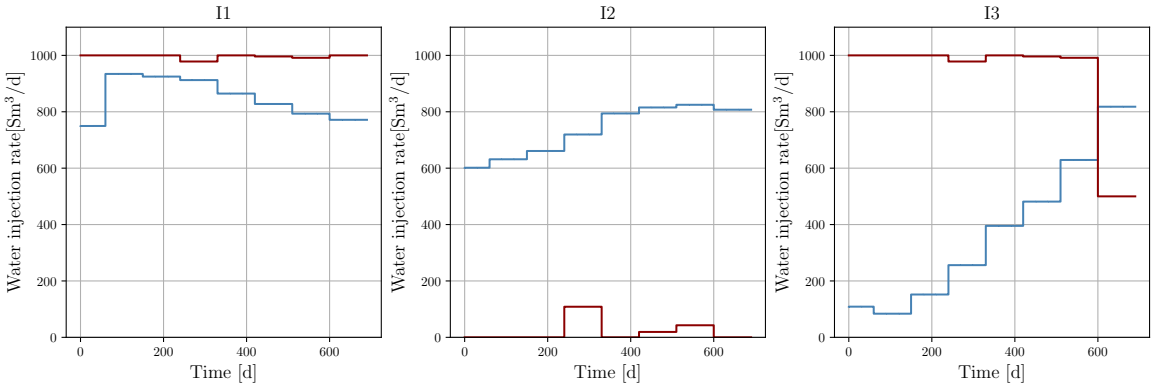
Case	ANN NPV [10^6 \$]	Simul. NPV [10^6 \$]	Time [min]
EnOpt (target)	342	288	30
FloodOpt	-	287	15

Table 4.12: Olympus: computational time and NPV, using ANNs and the simulator, achieved by the obtained optimal solutions (FloodOpt comparison).

	Cumulatives [10^6 Sm^3]		
Case	**Water inj.**	**Water prod.**	**Oil prod.**
EnOpt (target)	3.59	2.55	0.807
FloodOpt	3.59	2.52	0.799

Table 4.13: Olympus: cumulative injection and production for each simulated strategy (FloodOpt comparison).

In particular, FloodOpt tends to reward more efficient wells, such as I1, I6, I7, while penalizing the less efficient ones, I2, I4 and I5, with reduced injection rates compared to EnOpt. It is interesting to investigate why I3, despite its high efficiency, is kept at low to medium rates. Compared to EnOpt, the most significant differences are on I2, I3, I4, I5.



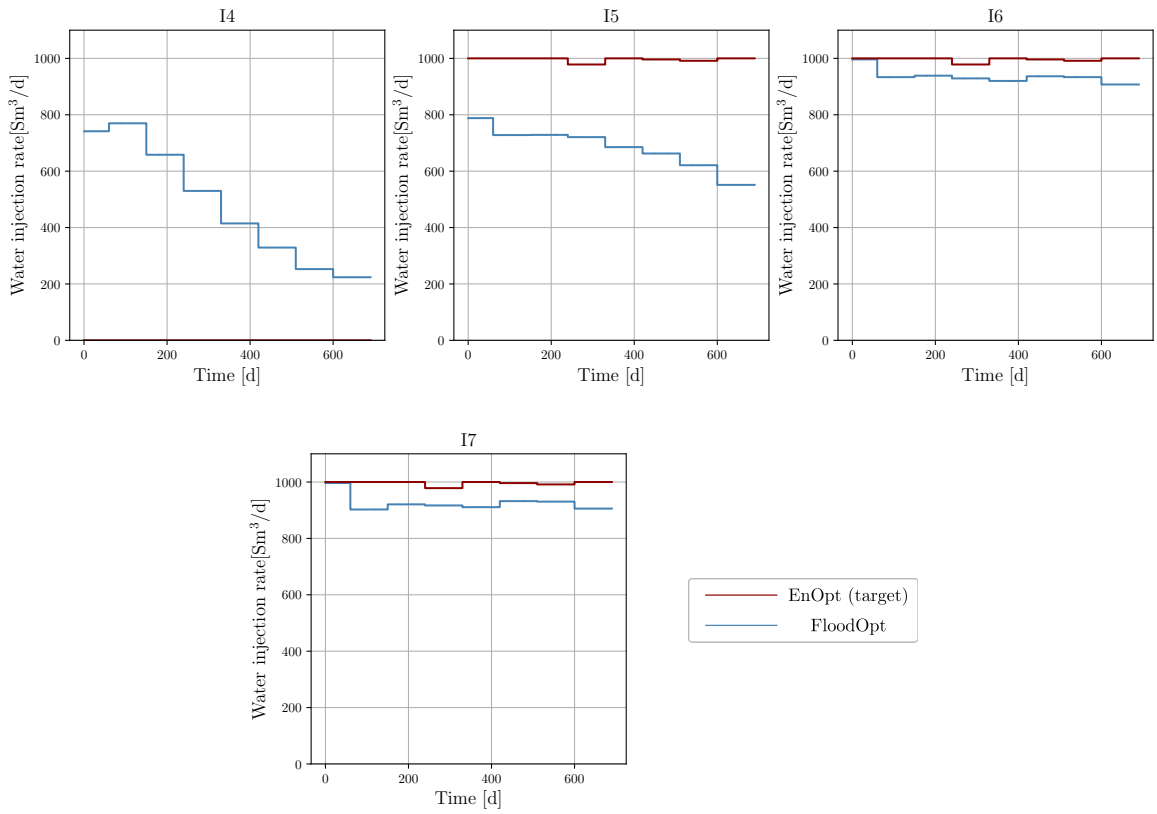
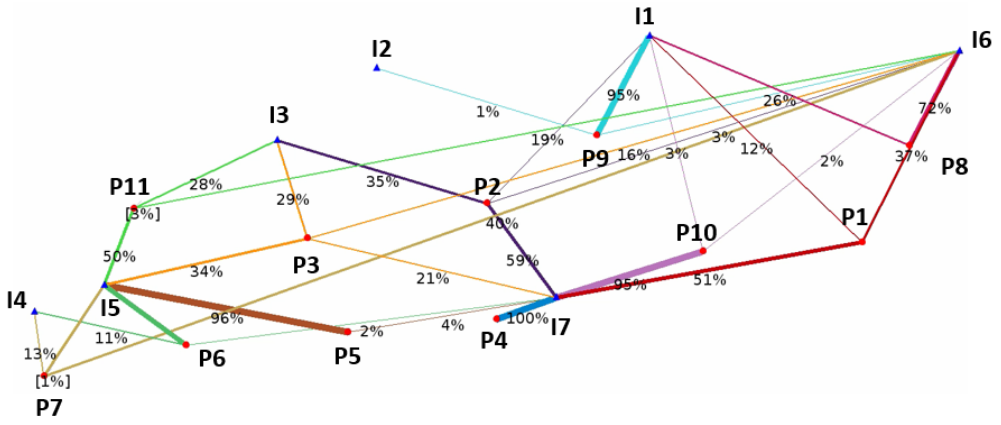
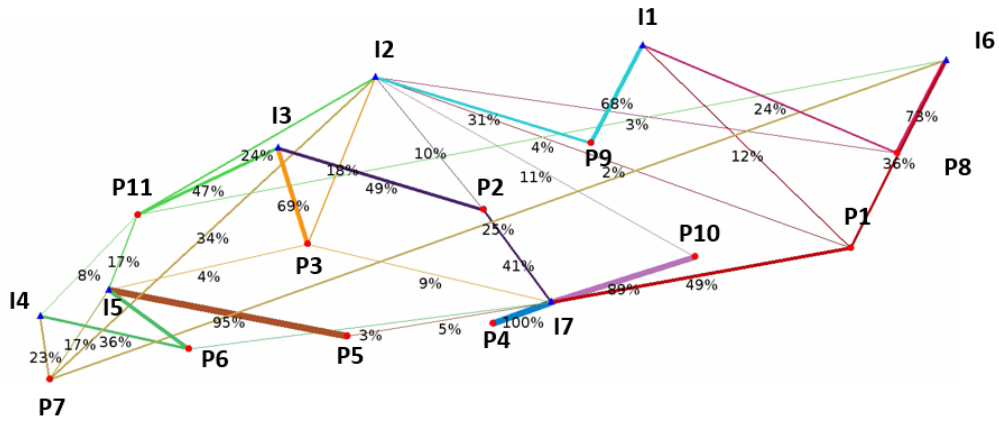


Figure 4.23: Olympus: optimal injection rates (FloodOpt comparison).

As it can also be seen on Fmaps (Figure 4.23b compared to Figure 4.24a), this results in a stronger I2-P9 connection (penalizing P1), stronger I3-P3 and P2 connection (penalizing P11), stronger I5-P6 and P2 connection (penalizing P11), stronger I4-P6 and P7 connection (without penalizing other wells since I4 is practically closed in EnOpt).



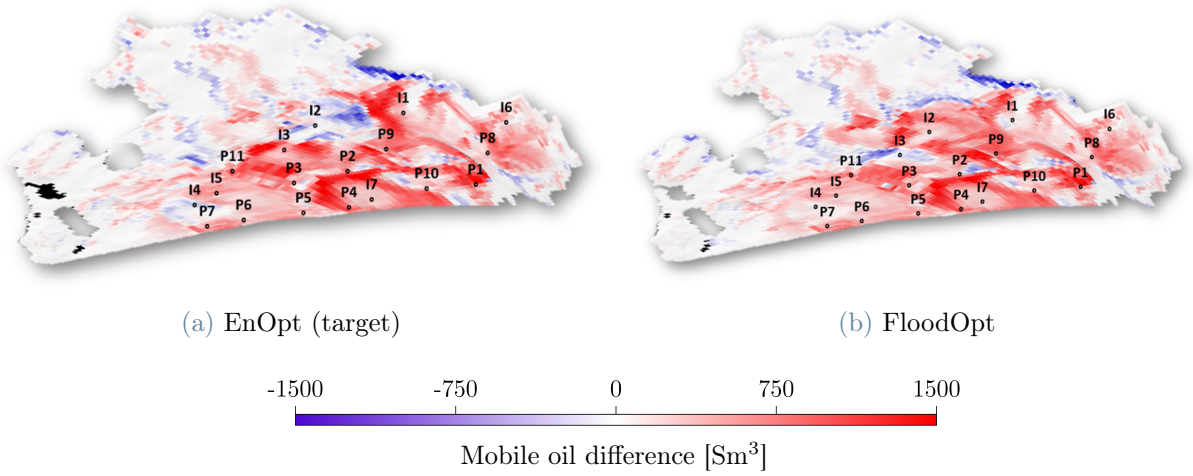
(a) EnOpt (target)



(b) FloodOpt

Figure 4.23: Olympus: Fmaps for each simulated strategy (FloodOpt comparison).

On mobile oil maps (Figure 4.24b compared to Figure 4.24a), this reflects onto how by injecting less from I1, more oil is left in the region around P9 and how by injecting less from I3, more oil concentrates between I3 and P2 and P11. On other hand, by injecting more from I2 (FloodOpt), the region between I2 and I1 is drained, while by injecting more from I4, the oil around it is not left off as in EnOpt, but it is produced. The remaining regions display similar behaviors.



(a) EnOpt (target)

(b) FloodOpt

Figure 4.24: Olympus: mobile oil maps for each simulated strategy (FloodOpt comparison).

Sensitivity analysis on optimization parameters

A sensitivity analysis is carried out on the main parameters of the GA and EnOpt algorithm. This study is presented here, after the optimization results, due to computational

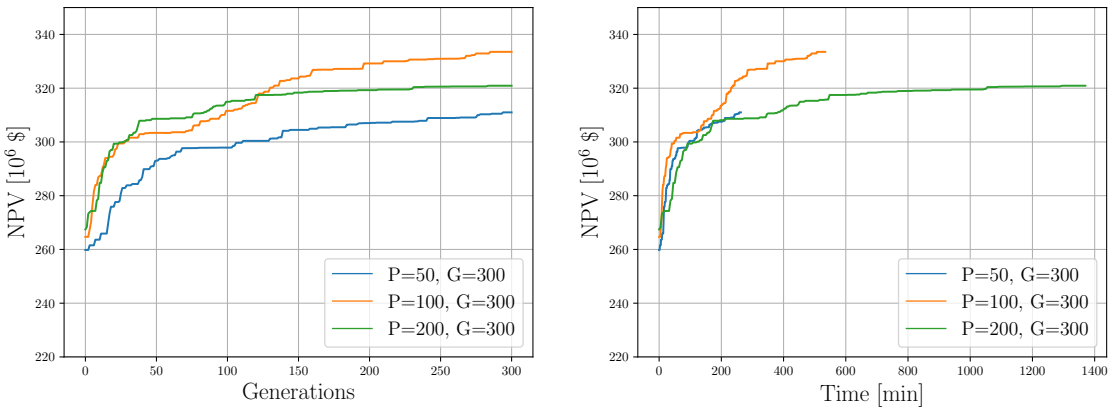
time reasons, explained below, which can prevent the full implementation of the sensitivity analysis results in the optimization workflow.

Genetic algorithm For the GA, different combinations of the parameters described in Appendix D are tested to investigate their impact on the objective function and computational time. Tested and optimal values are reported in Table 4.14.

Parameter	Values	Best for NPV	Best for time
Mutation prob.	0.05, 0.1, 0.2	Low	High
Parents portion	0.1, 0.3, 0.5	Low	Mid-high
Crossover prob.	0.1, 0.5, 0.7	Low-mid	Low/high
Crossover type	One-point, two-point, uniform	Two-point/uniform	One/two-point

Table 4.14: Olympus: GA sensitivity analysis parameters.

In addition, the effect of population number (P) and maximum number of generations (G) is investigated, keeping the same optimal parameters found previously and applied in the two case studies. The results are showed in Figure 4.24, with objective function plots reported over iterations (generations) and over time.



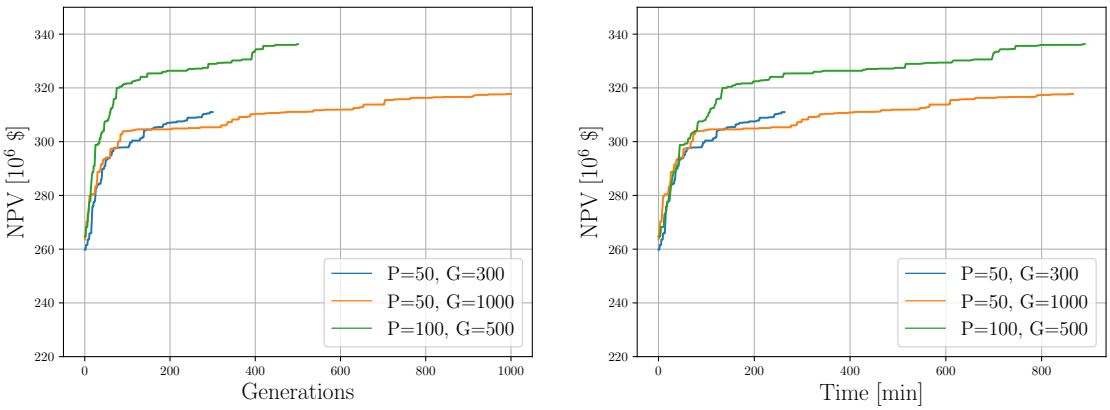


Figure 4.24: Olympus: GA sensitivity analysis.

The two top plots show the effect of the P, at constant G. P is suboptimal if too low (not enough variability in the population) or too high (excessive variability in the population) in terms of objective function. As for computational time, a higher P generally leads to an increase in the number of objective function evaluations if individuals are not evaluated in parallel. The two bottom plots show the effect of G. While a higher G can generally lead to increased computational time, a compromise between P and G seems to give the best overall result in terms of objective function, with the same computational time as a lower-P and higher-G configuration.

The GA parameters leading to the higher objective function, though, result in excessively high computational times for this application. For this reason, to find a compromise between optimization and computational time, and to respect the fair comparison among algorithms described in Figure 3.9, the chosen population and maximum number of generations for the GA in the previous optimization process of the two case studies are G=300 and P=50. Anyhow, as example, optimal profiles for the best configuration, with P=100 and G=500, are reported in Figure 4.24.

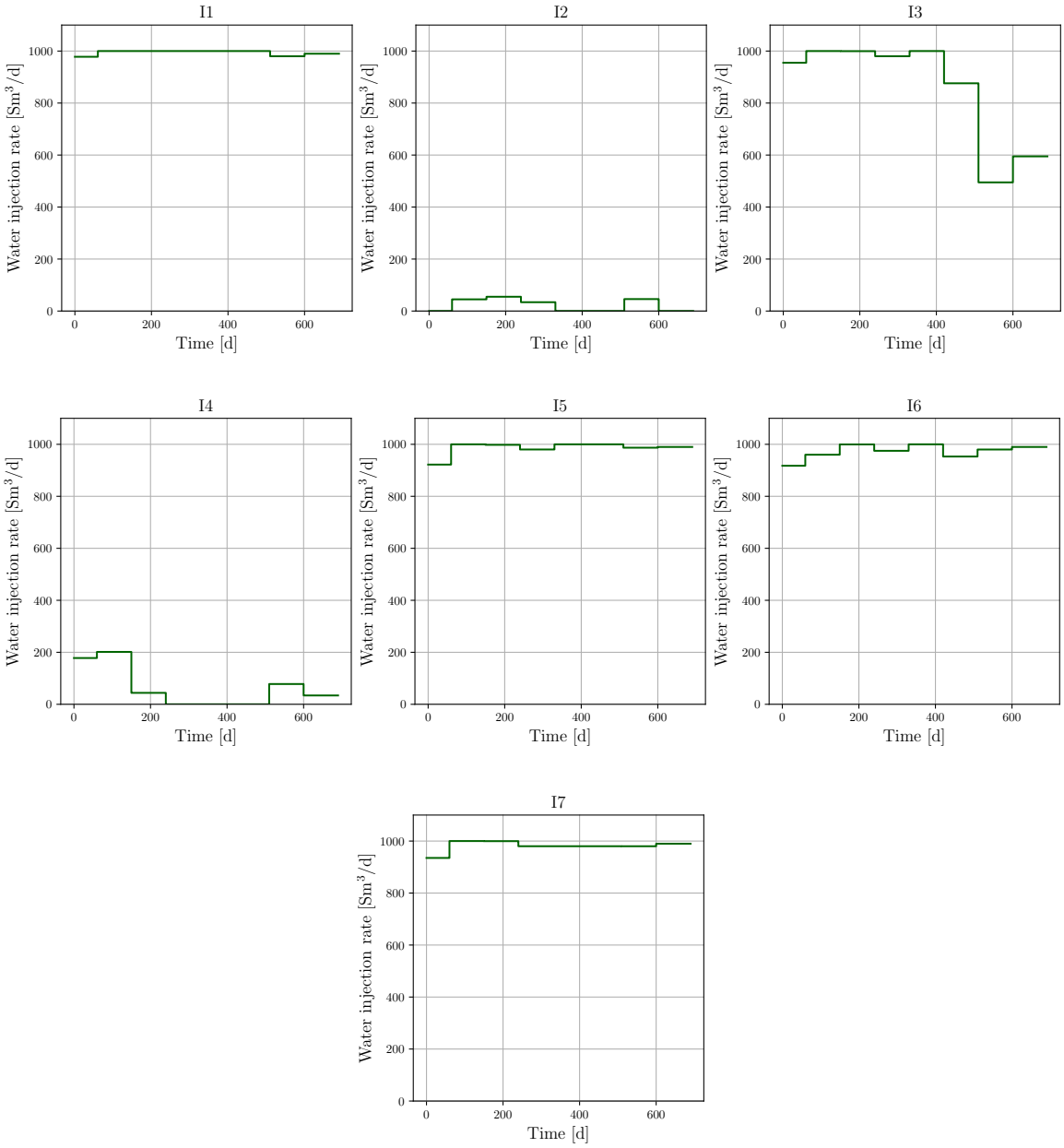
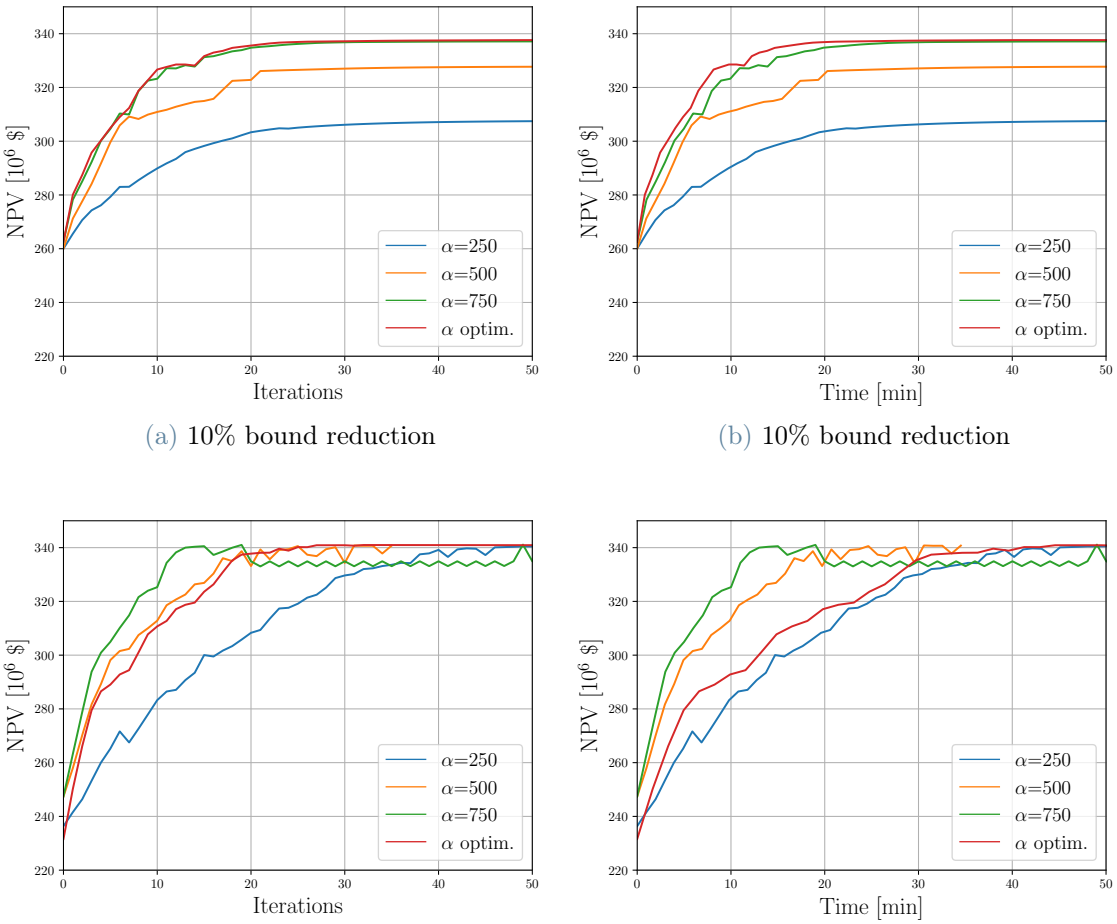


Figure 4.24: Olympus: optimal injection rates (GA).

Results improves but are still slightly lower than with EnOpt. This result can be visualized better when compared to the base case (see Figure 4.18).

EnOpt The main parameters of the EnOpt algorithm are the length α of the linesearch in the desired direction at each iteration and the bound percentage reduction at each iteration. The α value can be kept constant or can be optimized at each iteration, following the best objective function value. A sensitivity analysis is performed on these

two parameters, as reported in 4.24 with objective function plots reported over iterations (generations) and over time.



(a) 10% bound reduction (b) 10% bound reduction

(c) 1% bound reduction (d) 1% bound reduction

Figure 4.24: Olympus: EnOpt sensitivity analysis.

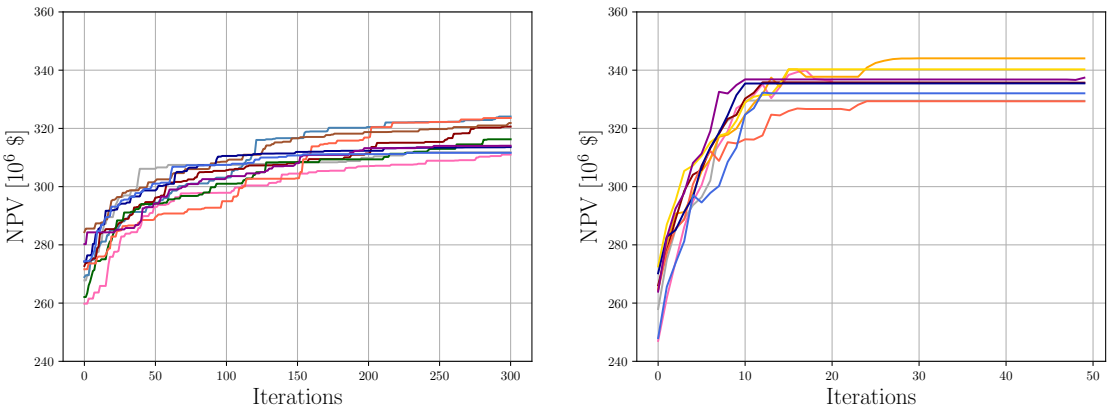
Results show a 1% reduction in the bounds at each iteration leads to better overall objective function values than a 10% reduction. In addition, optimizing the α value at each iteration generally leads to higher objective function values, but usually requires slightly more computational time. The compromise is then between the increased evaluations required to optimize α versus the benefit in terms of objective function.

Uncertainty analysis on initial candidates

To quantify the uncertainty of the obtained optimal solution with respect to the initial candidates, a brief study is carried out on the non-deterministic algorithms, GA and

EnOpt. It should be mentioned that this analysis is performed without integrating the results in the previous optimization workflow, which would require an entirely new optimization procedure that should exploit the prediction uncertainty directly in its search. These results are not validated through the full-physics reservoir simulator as for the rest of the application.

The optimization process is repeated 10 times by changing the seed (generated with the same procedure described in Chapter 3), acting as initial population for the GA or ensemble for EnOpt, with the same size as before. Figure 4.25 shows the objective function for the best of the candidates from each of the 10 trials, whose use was explained in Figure 3.9.



(a) GA (b) EnOpt

Figure 4.25: Olympus: evolution of the NPV for different initial candidates.

Figure 4.24 shows the average optimal injection profiles obtained by the algorithms, along with two standard deviations around each profile, specific for the injection rate of the single well at the specific time step.

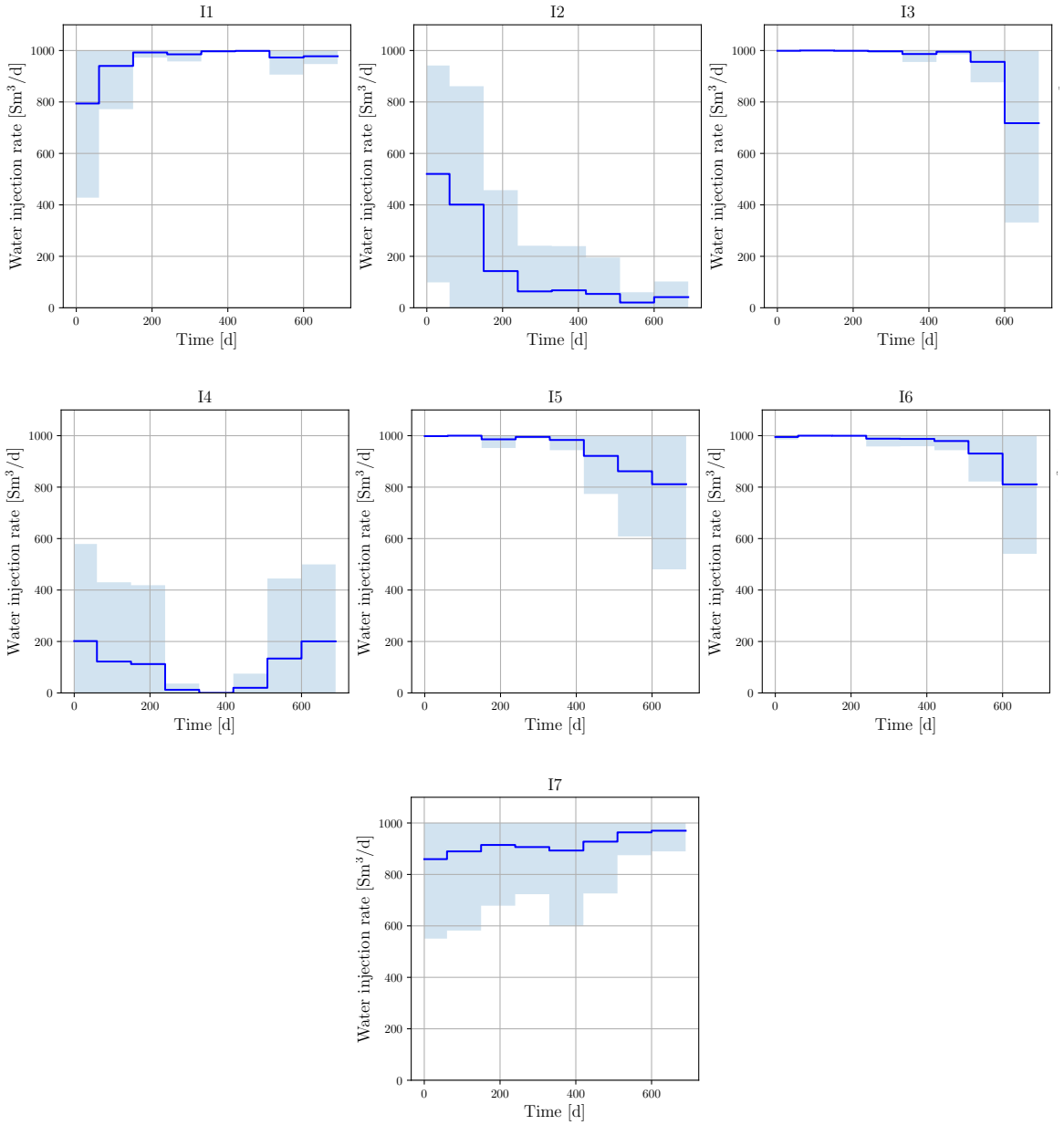


Figure 4.24: Olympus: optimal injection rates for different initial candidates, with average injection rates +/- 2σ, within bounds (EnOpt).

The EnOpt algorithm gives a lower uncertainty on the optimized injection profiles than the GA (not shown for simplicity). Although there is high uncertainty in optimization in some wells (e.g. I2, I4, I5 the injectors connected to the least important producers), for others all trials converge to similar profile (I1, I3, I6, the injectors connected to the main producers). For reference, see Figures 4.21 and 4.20. Furthermore, in some profiles, the uncertainty of optimization is significantly lower in certain points in time.

This analysis suggests that certain wells (I2, I4, I7) have little impact on the objective function, showing higher uncertainty in the optimal profiles when changing the seed. Thus, a further analysis is performed to investigate the impact on the optimization process when these wells are excluded from control variables and simply given a constant injection profile throughout the optimization period. Specifically, five values for the excluded injectors are chosen, for each injector, as follows: the average value of the average profile shown in 4.24, the average value of the average profile +/- $2\sigma_i$, and the two midpoints between the first and second value and between the first and third value, as shown in Table 4.15.

Results on the objective function and the controls of non-excluded wells (Table 4.15) show that optimization requires a compromise between injecting less from I2 and I4 and injecting more from I7, in agreement with previous optimization results. The best trial from the optimization uncertainty test still reaches better values than the best trial of this "excluded wells" test, using EnOpt. Even in this case, the GA does not reach better results than EnOpt. However, this trial reaches a very close value to the base-case optimization (Table 4.10), but with slightly more than half of the control variables (32 out of 56), reducing computational time.

Trial	$q_{in,2}$ **[Sm^3/d]**	$q_{in,4}$ **[Sm^3/d]**	$q_{in,7}$ **[Sm^3/d]**	**ANN NPV [10^6 $]**
#1	0	0	488	325
#2	75	57	694	330
#3	150	113	900	332
#4	341	316	950	318
#5	530	519	1000	303

Table 4.15: Olympus: injection rates for wells excluded from control variables.

5 | Conclusions

The continuous rise in global energy demand and the simultaneous decline in new substantial discoveries of oil fields require a careful management of the existing oil fields. In such context, waterflooding optimization consists in identifying the optimal injection scheme for a given field, so as to maximize its economic production and reduce waste. While traditional surveillance methods can lead to suboptimal production strategies, simulation-based optimization is often computationally heavy for practical applications, in terms of the time and resources required to construct and tune full-physics models and to run simulations. On the contrary, in surrogate-based optimization (SBO), the detailed physics-based simulation code is replaced by a fast-running data-driven model, requiring only production data and significantly reducing computational times. On the other hand, while full-physics models are multi-purpose, surrogate models are typically developed for the specific problem.

The research work in the present thesis proposes a framework for the efficient integration of ANN-based surrogate models for the solution of the problem of waterflooding optimization of brownfields. Compared to other types of SBO methods, which make use of physics-based proxy models (e.g. reduced-order models), ANNs do not require geological information. ANNs, in the form of physics informed (PINN) or long short-term memory (LSTM) neural networks, are trained on historical data, assuming only the actual historical production of the field is available as matched input-output pairs. These ANNs are used to approximate the oil and water production rates, based on the water injection rates and are, then, coupled to an optimization algorithm, using the net present value (NPV) as objective function. Simplifying assumptions include the timeline, where all wells share the same start date, the high quality of the datasets (neglecting noise, uncertainty or sparsity) and the constant dynamics of wells, which are not modified by workover operations.

The methodological framework proposed is validated by its application to two synthetic fields. The first case is a 2D synthetic field with homogeneous geological properties, 5 injectors and 4 producers, named Streak field. The surrogate models used in this first problem are based on PINN networks. The second case is a complex, 3D synthetic field

with highly heterogeneous geological properties, 7 injectors and 11 producers, named Olympus field. The surrogate models in this case are based on LSTM networks. Both applications provide positive results, which are deeply interpreted and analyzed in the thesis, in order to add a level of extent of validation to the numerical validation of optimized controls run on the simulator. In particular, it is found that:

- surrogate models are able to approximate the behavior of the reservoirs with good accuracy relative to a full-physics simulation model, thanks to the physical constraints imposed by PINNs and the integration of dynamics through time sequences in the LSTMs. In addition, they require only a small fraction of the computational time of the simulation runs for a single forward evaluation of the objective function, with a ratio of 1:10 and 1:100 for the Streak and Olympus fields, respectively. A study on model uncertainty shows that the prediction is accurate even when the dataset is randomly partitioned between training and validation, with the analysis performed on a single well only for time reasons;
- compared to the "do-nothing" and "pressure maintenance" scenarios, the optimization process leads to improved values of the NPV objective, even when the optimal injection schedules obtained by the surrogate models are run through the full-physics simulator. With respect to a simulation-based software for waterflooding optimization, the workflow is able to reach the same, or higher, cumulative oil production, with comparable computational times. The analysis on the robustness of the optimal solutions highlights high uncertainty;
- the optimization process is able to provide high-level operational guidelines for the field of interest and additional insights into the reservoir's behavior, in terms of injector-producer connections and the impact of wells on the NPV. Nevertheless, the obtained optimal solutions cannot be blindly translated into action, and still require the supervision of experts and operators to be implemented;
- the developed workflow provides high flexibility in that any objective function and constraint on the control variables can be specified, differently from the optimization software used as benchmark.

The demonstrated benefits of the proposed workflow confirm that it can be applied as a fast, practical tool for the problem of waterflooding optimization in realistic brownfields, without the support of geological information. The surrogate models can be updated continuously with new production data from the field to ensure they are up-to-date with the reservoir's conditions, making them available for real-time applications.

Future developments can include:

- integrating PINNs and LSTMs, to benefit from the peculiarities of both network types;
- using additional input data during training and/or optimization (e.g. BHPs), in order to increase the predictivity of the models as well as the flexibility of the workflow;
- enhancing the parallelization of the workflow by computing the objective function for independent candidate solutions, or the output from independent neural networks, in parallel rather than in series. This could further reduce computational times, expanding the workflow to the use of more complex networks or computationally heavier optimization algorithms;
- investigating further the model uncertainty quantification, through full bagging or Gaussian processes.

A | Oil and gas production

A.1. Oil field production life

Oil and gas production refers to all the processes which allow fluids to flow from reservoir rock to production wells as a result of a pressure difference between the reservoir and the surface. Production life is generally divided into three phases: primary, secondary, and tertiary production (or recovery) [2].

Primary production It occurs during the first years of life. The field produces in natural depletion, exploiting the natural energy available in the reservoir (i.e. expansion capacity of the formation fluids and the contraction of the rock) and the supporting action of an underlying aquifer or overlying gas cap. The main driving force for production is referred to as "drive": in this sense, fields can produce under expansion drive, dissolved gas drive, gas cap drive, water drive. However, once the reservoir's pressure has dropped sufficiently (usually within 1-10 years), production gradually decreases, leaving most of the resources underground (recoveries are usually 5-20% of the initial oil in place) [48]. This can make production inefficient or the investment not worthwhile. In addition to this, if pressure drops below the bubble point, the hydrocarbon mixture separates into two phases: liquid and vapor. The latter is richer in light components, has lower density, viscosity and economic value. Fields are referred to as "mature" when they have reached a stage of plateau or decline in production rate. Therefore, other supplementary processes are designed to add energy into the reservoir-fluid system and to maintain pressure in the reservoir (ideally above the bubble point), known as secondary production [4].

Secondary production It occurs when an external fluid is injected into the reservoir through injection wells. This helps maintain pressure and acts as a driving force to keep oil flowing. In addition, the injected fluid can displace oil from the pore space of the rock, leading to potentially higher recoveries (up to 30-60% of the initial oil in place) [48]. The most common injected fluids are natural gas and water. The use of one or the other depends on a variety of factors, such as the type of reservoir and the availability of the

fluid. Secondary production can extend the field's production life up to 20 years [4].

Tertiary production It occurs when secondary processes have exhausted their potential. It is the last and most expensive phase of production life, aimed at enhancing the recovery factor up to 50-70% by injecting special fluids such as CO2, surfactants, polymers, steam [48]. These processes are based on a chemical or thermal mechanism that improves oil displacement or flow.

Note: in reservoir engineering, decisions often must be taken in presence of very large uncertainties about subsurface properties. To address the issue, a set of different subsurface models can be used, known as ensemble of geological realizations. "Nominal" is the word used if only one geological model is considered, or else "robust" in case multiple geological models are employed.

A.2. Waterflooding

Water injection is also called waterflooding, and is performed through a series of injection wells, strategically drilled in the field. Water can be pumped directly into the surrounding aquifer (water-bearing zone below oil), in the oil-bearing area, or peripherally. It maintains pressure and has a sweep effect on the oil-saturated rock, improving recovery [51]. Oil wells produce a complex, multiphase mixture at the well-head. The main components are usually oil, water, and gas [51]. The water fraction of the liquid production rate of a well is known as water cut (WC). When injected water is first produced at production wells, water breakthrough has occurred. From this point onwards the water cut usually starts to rise, as the reservoir is being gradually flooded and, hopefully, oil is being swept. In mature fields, water cut can reach values up to 90-95% while still maintaining production economically feasible [4].

Nevertheless, simply injecting more water does not directly lead to higher recovery. In fact, from a physical perspective, the process of waterflooding is a complex, nonlinear problem. To model it mathematically, the following equations are required:

- phase behavior of fluids;
- multiphase flow equations;
- mass conservation equations;
- deliverability equations at wells.

Analytical solutions to the waterflooding problem exist only for simplified cases, such as

the Buckley-Leverett equation, for a 1D, immiscible and incompressible oil-water system (see below).

Phase behavior The phase behavior of reservoir fluids is used to describe their PVT properties, both at surface and reservoir conditions. Phase behavior is typically described either using a "black-oil" approach, based on interpolation of PVT properties as a function of pressure, or a "compositional" approach, based on thermodynamically consistent model such as an equation of state (EOS) [44]. Fluid properties have major impact of production mechanisms. Phase behavior of hydrocarbons can be visualized on a P-T diagram, as in Figure A.1, which is not known *a priori* but is based on experimental measurements from the reservoir [48].

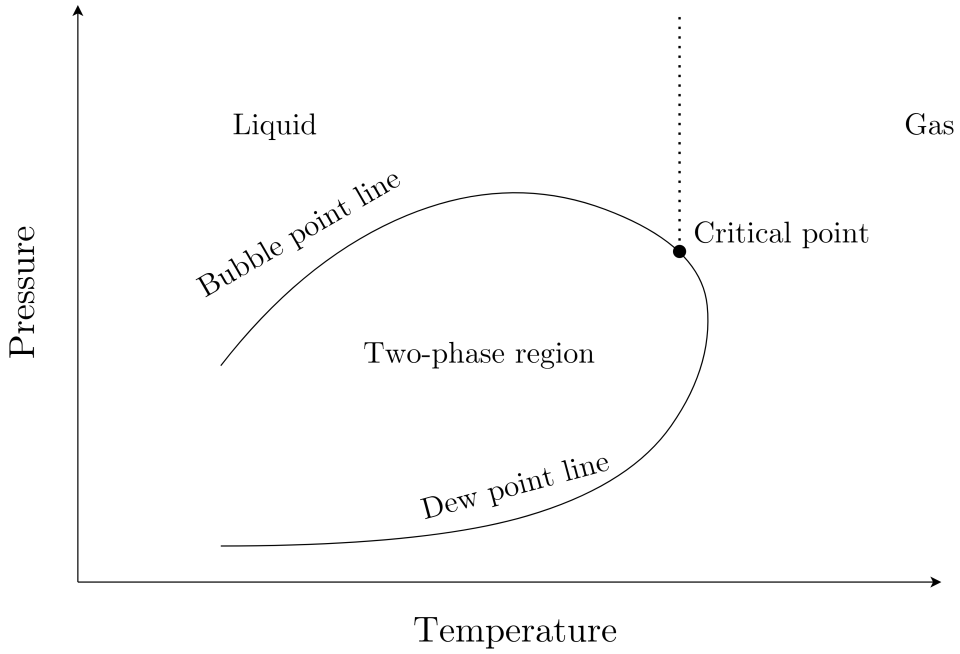


Figure A.1: Hydrocarbon phase diagram.

The critical point marks the transition from the bubble point to the dew point line, which in turn describe how the hydrocarbon separates into two phases as pressure and temperature are varied. Surface conditions have much lower pressure (and, to a lesser extent, temperature) than reservoir conditions. As a consequence, formation volume factors are used to describe the ratio between reservoir volumes (subscript r) and surface volumes (subscript s). This is also the reason why flow rates are sometimes expressed in rm^3 or rb.

Oil formation volume factor is defined as the ratio between the oil reservoir volume V_{or}

and surface volume V_{os}:

$$B_o = \frac{V_{or}}{V_{os}} \tag{A.1}$$

Due to light components which separate from the liquid phase as pressure decreases, B_o is greater than 1 $\mathrm{rm}^3/\mathrm{Sm}^3$, with values in the range 1-2 $\mathrm{rm}^3/\mathrm{Sm}^3$ from heavy oils to volatile oils. It reaches its maximum at the bubble point pressure.

Water formation volume factor is defined as the ratio between the water reservoir volume V_{wr} and surface volume V_{ws}:

$$B_w = \frac{V_{wr}}{V_{ws}} \tag{A.2}$$

Due to dissolved gases at reservoir conditions, such as carbon dioxide, it is typically close to 1 $\mathrm{rm}^3/\mathrm{Sm}^3$ or slightly larger.

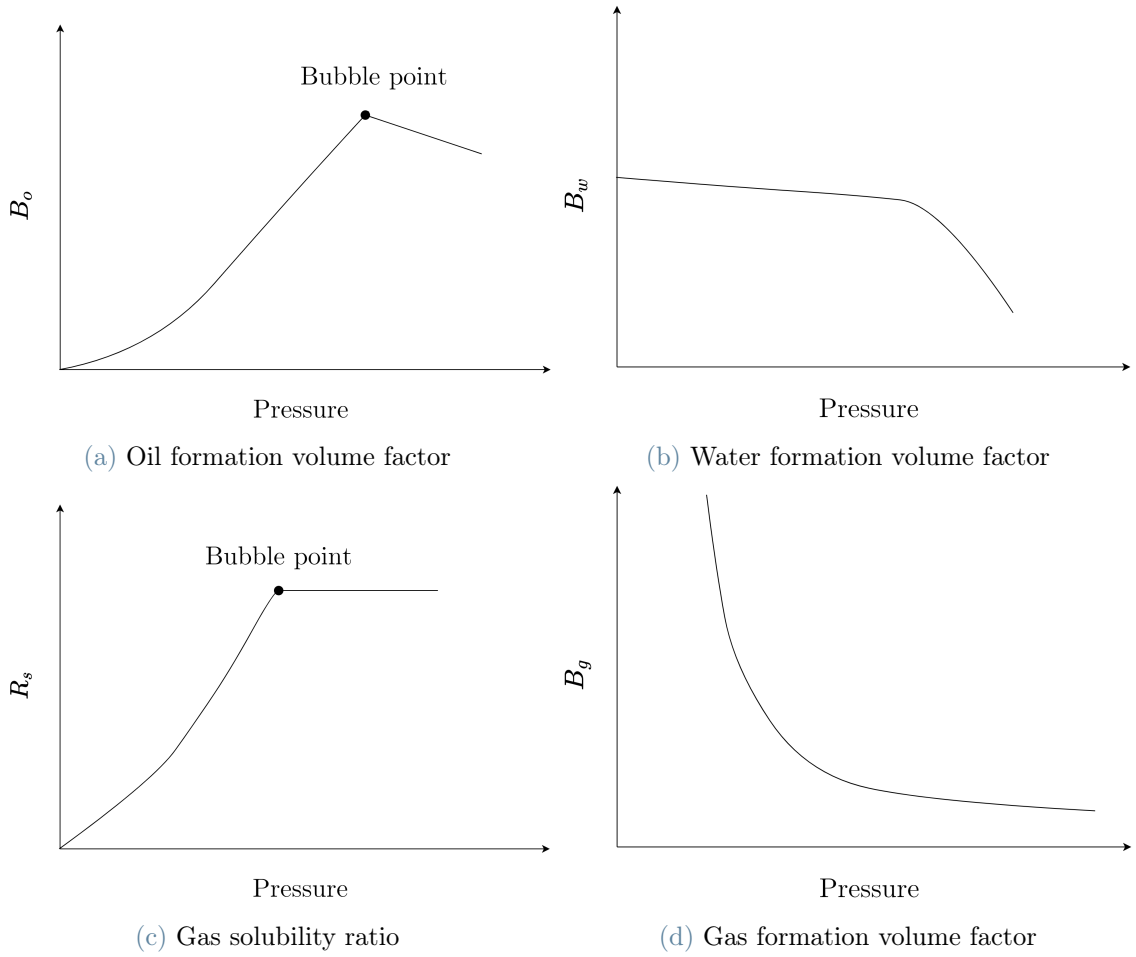
Gas formation volume factor is defined as the ratio between free gas reservoir volume V_{gr} and the surface volume of that free gas (excluding solution gas) V_{fg}:

$$B_g = \frac{V_{gr}}{V_{fg}} \tag{A.3}$$

It is typically in the range 10^{-2}-$10^{-3}\mathrm{rm}^3/\mathrm{Sm}^3$ and decreases as pressure drops since gas expands. Another important concept is the solution gas-oil ratio, which is defined as the volume of gas V_{sg}, measured at standard conditions, that will dissolve in a unit surface volume of oil in the reservoir V_{os} [48]:

$$R_s = \frac{V_{sg}}{V_{os}} \tag{A.4}$$

It reaches its maximum at the bubble point pressure. Schematic plots of formation volume factors and of R_s are reported in Figure A.1.



(a) Oil formation volume factor

(b) Water formation volume factor

(c) Gas solubility ratio

(d) Gas formation volume factor

Figure A.1: Qualitative trends of formation volume factors and gas solubility ratio with pressure.

Mass conservation equations For one-dimensional flow in porous media, the Darcy velocity q (also called Darcy flux) is related to the average linear velocity v (or seepage velocity) by the porosity ϕ. For a discharge flow rate Q over a cross-sectional area A:

$$v = \frac{Q}{A_e} = \frac{Q}{A\phi} = \frac{q}{\phi} \tag{A.5}$$

where A_e is the effective area, to reflect the fact that fluids can only pass through the connected pore space portion of A.

The material balance equation for incompressible multiphase flow is given by:

$$\nabla \cdot \boldsymbol{q}_t = \nabla \cdot \sum_{j=1}^{N_{ph}} \boldsymbol{q}_j = 0 \tag{A.6}$$

and for every phase j it is given by:

$$\phi \frac{\partial S_j}{\partial t} + \nabla \cdot \boldsymbol{q}_j = 0 \tag{A.7}$$

where:

- N_{ph} is the number of phases;
- $\boldsymbol{q}_t$ is the total Darcy velocity;
- $\boldsymbol{q}_j$ is the Darcy velocity of phase j;
- S_j is the saturation of phase j.

The right side of the equation is different from zero in case of sink or source terms (in proximity of wells).

Multiphase flow equations The generalized form of Darcy's law for incompressible multiphase flow in porous media, due to Muskat and Meres (1936)[52], is:

$$\boldsymbol{q}_j = -\frac{\boldsymbol{K} k_{rj}}{\mu_j}(\nabla P_j - \rho_j \boldsymbol{g}) \tag{A.8}$$

where:

- $\boldsymbol{g}$ is the gravitational acceleration constant
- $\boldsymbol{K}$ is the permeability matrix
- k_{rj} is the relative permeability of phase j
- μ_j is the viscosity of phase j
- ρ_j is the density of phase j

A key role in waterflooding is played by permeability $\boldsymbol{K}$ and relative permeability k_{rj} of phase j. $\boldsymbol{K}$ is a function of the structure of the porous medium, and it is dimensionally a squared length, measured in millidarcy (mD) or m^2. It can span several orders of magnitude in the same geological structure, and is in general a tensor. Relative permeability k_{rj}, on the other hand, accounts for the fact that the flow of any phase is influenced by the presence of the other phases. It is a highly nonlinear function of saturation, but it also depends on displacement path, wettability of the rock, and flow rate [48]. It is dimensionless and equals 1 in case of single-phase flow. An example of permeability curves

is reported in Figure A.2, where they are plotted against water saturation, between the initial value S_{wi} and residual value $1 - S_{or}$.

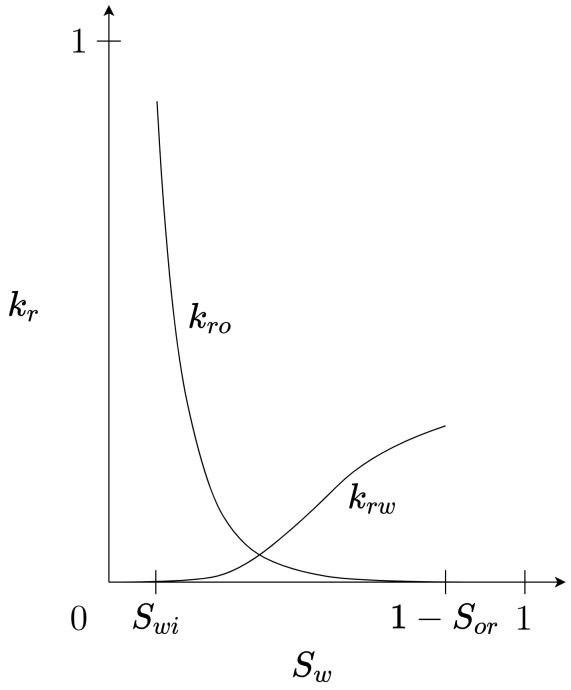


Figure A.2: Example of relative permeability curves against water saturation.

In practice, water does not displace oil uniformly along a single flow direction, following a piston-like motion. Instead, water follows preferential paths in the rock matrix, depending on its absolute permeability distribution and relative permeabilities. This phenomenon, known as "fingering", is the reason why simply injecting more water from a well may not lead to increased recovery: on the contrary, it may result in leaving oil behind and producing only water. The effectiveness of the waterflooding process can be measured by means of sweep efficiency and the local displacement efficiency [2]. The sweep efficiency is the fraction of the volume of the reservoir contacted by the injected water: it depends on a variety of factors, such as injection pattern, permeability, flow rate, fluid properties. The local displacement efficiency is the fraction of oil that has been recovered from a zone swept by the injected water.

Deliverability equation at wells The production rate in ideal, undersaturated (single-phase, homogeneous liquid) oil wells is linearly proportional to the drawdown, i.e. the difference between reservoir pressure $\bar{p}$ and bottomhole well flowing pressure p_{wf}. The

deliverability equation describes this relationship through the productivity index PI:

$$PI = \frac{Q_o}{(\bar{p} - p_{wf})} \tag{A.9}$$

Assuming steady state conditions, pure radial flow, and constant, homogeneous properties, Darcy's law can be expressed by means of the productivity index, in SI units, as:

$$PI = \frac{2\pi K k_r H}{\mu_o B_o \left[ln\left(\frac{r_e}{r_{wb}}\right) + S_k \right]} \tag{A.10}$$

where:

- H is the net pay zone thickness;
- r_e is the drainage radius at the constant pressure boundary;
- r_{wb} is the wellbore radius;
- S_k is the skin factor, which accounts for a pressure drop Δp_{skin} due to near wellbore phenomena.

A schematic representation of the drawdown (pressure difference) cone is reported in Figure A.3, where the effect of the skin factor is highlighted.

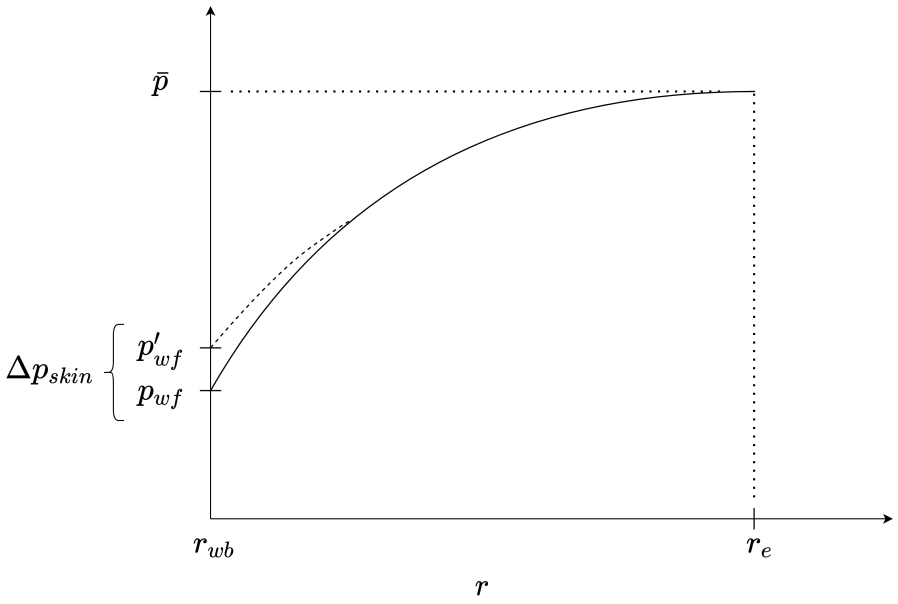


Figure A.3: Schematic pressure cone around an ideal, undersaturated production well.

(A.10) is also known as inflow performance relationship (IPR). Conditions between well-

bore and wellhead, represented in Figure A.4, can be expressed through pressure losses in the tubing, as:

$$p_{wf} = p_{wh} + p_h + \Delta p_{tbg} = p_{wh} + p_h + f(Q_o, p_{wf}, p_{wh}) \tag{A.11}$$

where p_h is the static pressure head and Δp_{tbg} is the pressure drop in the tubing, expressed as a function of flow rate and pressure.

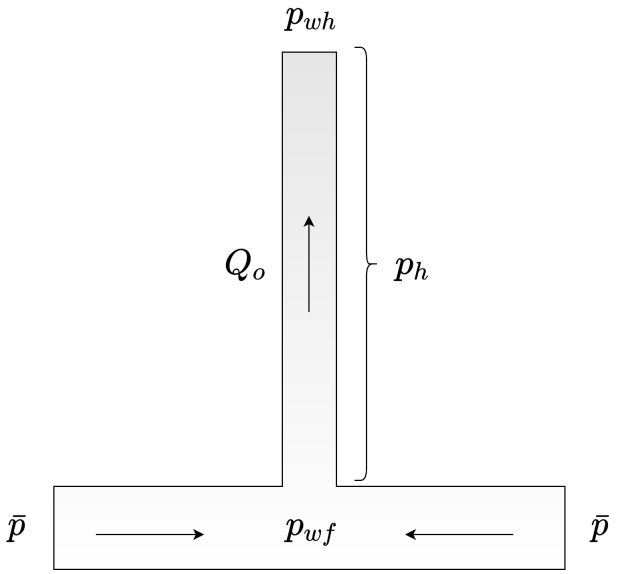


Figure A.4: Production well scheme.

(A.11) is also known as vertical flow performance (VFP) curve. The IPR curve is paired with VFP curve to obtain the working point of the reservoir-well system, as shown in Figure A.5.

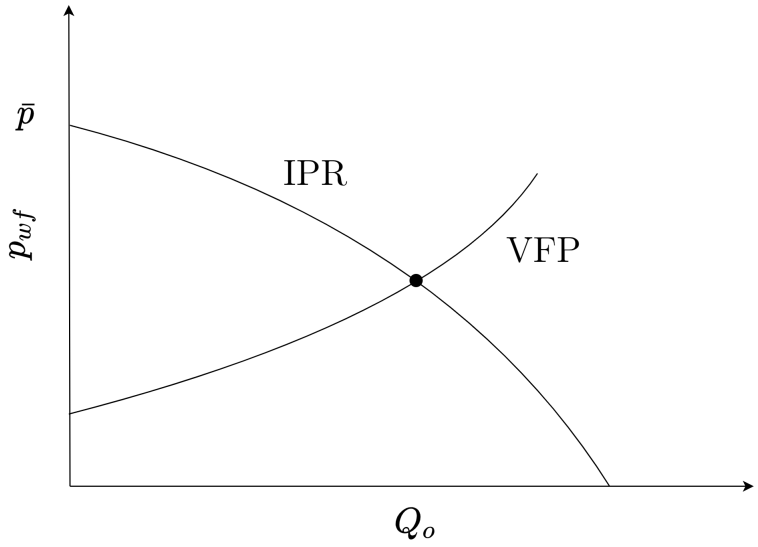


Figure A.5: Graphical method to find a well's working point.

These considerations can be extended to non-saturatd flow and non-ideal conditions: further information can be found in the specific literature [4, 51].

A.2.1. Buckley-Leverett solution

First proposed by Buckley and Leverett (1942)[35], this equation is used for a simplified, 1D water-oil system in porous media. Assuming incompressible, immiscible flow, the Darcy velocities in (A.8) can be expressed as:

$$q_w = -\frac{Kk_{rw}}{\mu_w}(\frac{\partial P_w}{\partial x} - \rho_w g sin\theta) \tag{A.12}$$

$$q_o = -\frac{Kk_{ro}}{\mu_o}(\frac{\partial P_o}{\partial x} - \rho_o g sin\theta) \tag{A.13}$$

where θ represents the slant angle of flow with respect to the horizontal direction. Notice the use of the symbols q_o and q_w, which in this instance refer to a Darcy flux, while in the rest of the thesis refer to flow rates. The phase conservation equations (A.7) become:

$$\phi\frac{\partial S_w}{\partial t} + \frac{\partial q_w}{\partial x} = 0 \tag{A.14}$$

$$\phi\frac{\partial S_o}{\partial t} + \frac{\partial q_o}{\partial x} = 0 \tag{A.15}$$

such that, since $S_w + S_o = 1$, the total Darcy velocity can be expressed as the sum

$q_t = q_w + q_o$:

$$q_t = -\frac{Kk_{rw}}{\mu_w}(\frac{\partial P_w}{\partial x} - \rho_w g sin\theta) - \frac{Kk_{ro}}{\mu_o}(\frac{\partial P_w}{\partial x} + \frac{\partial P_c}{\partial x} - \rho_o g sin\theta) \tag{A.16}$$

where $P_c = P_o - P_w$ is the capillary pressure, assuming oil is the non-wetting phase and water is the wetting phase. By defining the mobility λ_j of phase j and the total mobility λ_t as:

$$\lambda_j = \frac{k_{rj}}{\mu_j} \tag{A.17}$$

$$\lambda_t = \sum_{j=1}^{N_{ph}} \lambda_j \tag{A.18}$$

(A.16) becomes:

$$q_t = -K\lambda_t \frac{\partial P_w}{\partial x} + Kg sin\theta(\rho_w \lambda_w + \rho_o \lambda_o) - K\lambda_o \frac{\partial P_c}{\partial x} \tag{A.19}$$

and (A.12), by substituting $\frac{\partial P_c}{\partial x}$ from (A.19), becomes:

$$q_w = \frac{\lambda_w}{\lambda_t} q_t + K\frac{\lambda_o \lambda_w}{\lambda_t}(\rho_w - \rho_o) g sin\theta + K\frac{\lambda_o \lambda_w}{\lambda_t}\frac{\partial P_c}{\partial x} \tag{A.20}$$

In this way, (A.15) can be expressed in terms of water fractional flow f_w as:

$$\phi \frac{\partial S_w}{\partial x} + q_t \frac{\partial f_w}{\partial x} = 0 \tag{A.21}$$

where f_w is:

$$f_w = \frac{\lambda_w}{\lambda_t}\left[1 + K\frac{\lambda_o}{q_t}\left(\frac{\partial P_c}{\partial x} + (\rho_w - \rho_o) g sin\theta\right)\right] \tag{A.22}$$

Neglecting capillary effects, which are generally small compared to advection and buoyancy effects, (A.22) becomes:

$$f_w = \frac{\lambda_w}{\lambda_t}\left[1 + K\frac{\lambda_o}{q_t}(\rho_w - \rho_o) g sin\theta\right] = \frac{1 + N_G k_{ro} sin\theta}{1 + \frac{k_{ro}\mu_w}{k_{rw}\mu_o}} \tag{A.23}$$

where $N_G = \frac{K\Delta\rho g}{\mu_o q_t}$ is called gravity number. In this way, (A.21) can be rewritten as:

$$\frac{\partial S_w}{\partial t} + v\frac{\partial f_w}{\partial x} = 0 \tag{A.24}$$

where $v = \frac{q_t}{\phi}$. Applying the chain rule:

$$\frac{\partial S_w}{\partial t} + v\frac{\partial f_w}{\partial S_w}\frac{\partial S_w}{\partial x} = 0 \tag{A.25}$$

If dimensionless space x_D and time t_D are defined as:

$$x_D = \frac{x}{L} \tag{A.26}$$

$$t_D = \int_0^t \frac{v}{L}dt = \int_0^t \frac{q_t}{\phi L}dt = \int_0^t \frac{Q}{\phi AL}dt = \frac{1}{V_p}\int_0^t Qdt \tag{A.27}$$

then water saturation partial derivatives become, in dimensionless variables:

$$\frac{\partial S_w}{\partial t_D} = \frac{dS_w}{dv_D}\frac{dv_D}{dt}\bigg|_x = \frac{v_D}{t_D}\frac{dS_w}{dv_D} \tag{A.28}$$

$$\frac{\partial S_w}{\partial x_D} = \frac{dS_w}{dv_D}\frac{dv_D}{dx}\bigg|_t = \frac{1}{t_D}\frac{dS_w}{dv_D} \tag{A.29}$$

to obtain the final equation:

$$\frac{dS_w}{\partial v_D}\left(v_D - \frac{\partial f_w}{\partial S_w}\right) = 0 \tag{A.30}$$

The trivial solution is constant saturation with v_D, while the non-trivial solution is expressed by the characteristic velocity of the saturation wave:

$$v_D = \frac{\partial f_w}{\partial S_w} \tag{A.31}$$

or, in dimensional form:

$$v = v_D\frac{Q}{\phi A} \tag{A.32}$$

As such, water saturation is a function of space and time $S_w = S_w(x,t) = S_w(x - vt)$. The non-convex nature of (A.23) gives rise to the peculiar Buckley-Leverett profile, which consists of a constant state at S_{wi}, a shock wave (i.e. a discontinuity in saturation left L

and right R side of the shock), moving with velocity v_{sh} (or, in dimensionless form, v_{shD}):

$$v_{sh} = \frac{q_t}{\phi}\frac{(f_w^L - f_w^R)}{(S_w^L - S_w^R)} = \frac{q_t}{\phi}\frac{\Delta f_w}{\Delta S_w} \tag{A.33}$$

$$v_{shD} = \frac{\Delta f_w}{\Delta S_w} \tag{A.34}$$

and a rarefaction wave, as reported schematically in Figure A.6. The construction of the Buckley-Leverett solution can be obtained analytically or graphically, with the Welge construction, starting from the $f_w - S_w$ plot (details can be found in Dake (1983)[51] and Buckley and Leverett (1942)[35]).

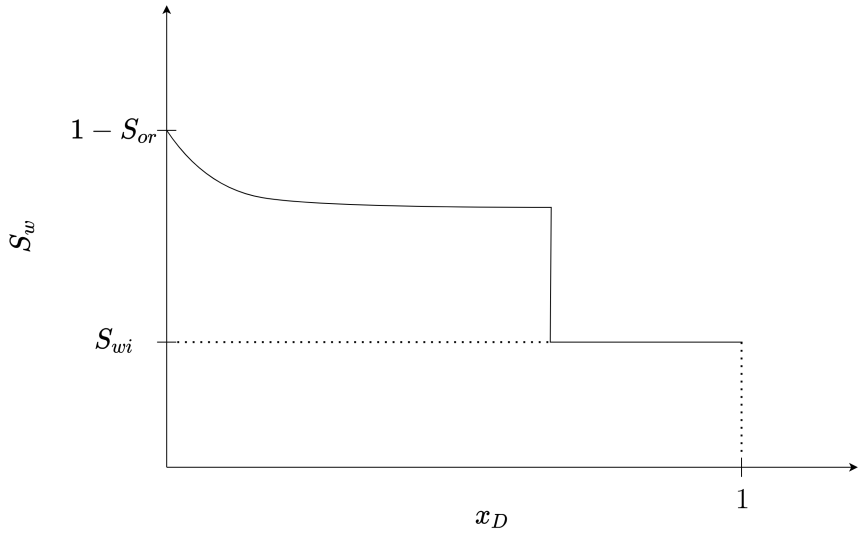


Figure A.6: Example of Buckley-Leverett solution profile.

The final recovery efficiency R_f, the ratio between the produced oil N_P and the total oil in place N, can be thought of as the product between the sweep efficiency E_s and the local displacement efficiency E_d [48]:

$$R_f = \frac{N_P}{N} = E_s E_d = \frac{N_{PD}^{real}}{N_{PD}^{BL}}\frac{N_{PD}^{BL}}{(1 - S_{wi})} \tag{A.35}$$

where $N_{PD} = \frac{N_P B_o}{V_p \phi NTG}$ is the dimensionless form of N_P and superscripts denote the ideal Buckley-Leverett conditions and real conditions.

A.2.2. Other aspects

Water composition, well placement and injection/production schedules are critical to efficient waterflooding [4].

Water composition It must be compatible with the reservoir's aquifer water and rock. Water can come from aquifers, watercourses, or the sea, in case of offshore fields. Water pretreatments include filtering, deaeration, inhibitor addition, in order to prevent any precipitate formation, clogging, corrosion [4].

Well placement The number and location of wells can be determined through simulations using mathematical models, based on reservoir's geometry, type of rock, costs, and position of existing wells. The layout of injection and production wells on the field is called well pattern. Common patterns are the direct and staggered line drive, the 4-5-6-7 spot [4].

Injection/Production schedules They specify how much and when to inject or produce from each well over time. This is done in the form of injection and bottom-hole pressure profiles of wells. Oil displacement through waterflooding is a slow process: its timescale is in the order of months to years [48]. Hence, also for operational reasons, variations in production strategies cannot be applied continuously, but are modified only every so often.

A.3. Waterflooding management

Most of the existing oil and gas fields are already at mature stage of production. Simultaneously, the number of substantial new discoveries is declining [1]. Maximizing the efficiency of existing oilfields, while lowering development and operating costs, is critical to meet the increasing global demand for energy. If waterflooding operations are instead managed poorly, there is a high risk of leaving precious resources behind, ultimately resulting in monetary losses. In practice, the risk is to reach the minimum oil production rate for economic production or the maximum water production rate allowed by facilities. Hence, a poor water injection management can lead to well flooding and subsequent shutdown.

A strategy that might be used to address these issues is optimal control theory applied to production. Production optimization refers to the identification of optimal strategies for hydrocarbon production. In other words, it is the process through which reservoir

performance is adjusted and managed in terms of oil recovery or economic return (net present value, NPV).

Benefits Developing an improved operating plan for waterflooding optimization has the potential to increase oil recovery from the reservoir of interest [3]. Practically, this entails producing more oil for the same investment, or reducing costs and waste for the same production. Specifically, the advent of smart well technology has allowed to achieve substantially higher oil recovery by intelligently managing field operations [5].

Challenges Given that the relationship between reservoir dynamics and control parameters is, in general, non-linear, finding the optimal set of controls is a complex task. The number of constraints and parameters for optimal field control can rise quickly if well patterns, rates and BHPs over time are all used as control variables with dozens of wells in the field. As a consequence, reducing the order of the optimization problem (dimension space) is one of the main challenges of waterflooding optimization. This entails identifying the controls that have the most influence on the objective function to lower the computational complexity of the problem.

B | Waterflooding optimization problem

In this section the general form of the waterflooding optimization problem is described. Waterflooding optimization can be both single and multi-objective. In the former, the objective function is almost always some form of Net Present Value (NPV); in the latter, another objective, such as cumulative oil production or the inverse of NPV variance across multiple geological realizations, is also optimized by means of a Pareto front, along with the expected value of the objective function over the realizations. This thesis focuses on single objective optimization. Anyhow, the choice of the objective function ultimately depends on the specific reservoir and its surrounding conditions. The time evolution of the reservoir state can be represented as a time and spatial discretization of the underlying partial differential equations over N_t time steps [53]. As such, the reservoir simulator can be described as in Onwunalu and Durlofsky (2010)[53]:

$$\boldsymbol{g}_{(k+1)}(\boldsymbol{x}_{(k+1)}, \boldsymbol{x}_{(k)}, \boldsymbol{u}_{(k+1)}) = \mathbf{0} \quad \text{for } k = 0, ..., N_t - 1 \tag{B.1}$$

where:

- $\boldsymbol{g}_{(k)} : \mathbb{R}^d \rightarrow \mathbb{R}^t$ is a vector-valued nonlinear function representing the set of simulator equations at the generic time step k;
- $\boldsymbol{x}_{(k)} \in X_{(k)} \subset \mathbb{R}^t$ is the vector of dynamic state variables of the model (pressure, saturation, component concentration etc.) at all grid blocks at the generic time step k;
- $\boldsymbol{u}_{(k)} \in U_{(k)} \subset \mathbb{R}^p$ is the vector of continuous well control variables, of dimension n (e.g. injection rates, BHPs, etc.).

$X_{(k)}$ and $U_{(k)}$ define the allowable values for each control variable at time step k. The reservoir simulator equations are solved starting from initial conditions at each time step.

Simulator equations can be written as concatenated vectors in time:

$$\boldsymbol{g} = [\boldsymbol{g}_{(1)}^T, ..., \boldsymbol{g}_{(N_t)}^T]^T \quad \boldsymbol{g} : \mathbb{R}^s \to \mathbb{R}^s \tag{B.2}$$

where $s = N_t t$. Applying the same formulation to state and control vectors:

$$\boldsymbol{x} = [\boldsymbol{x}_{(1)}^T, ..., \boldsymbol{x}_{(N_t)}^T]^T \quad \boldsymbol{x} \in X \subset \mathbb{R}^s \tag{B.3}$$

$$\boldsymbol{u} = [\boldsymbol{u}_{(1)}^T, ..., \boldsymbol{u}_{(N_t)}^T]^T \quad \boldsymbol{u} \in U \subset \mathbb{R}^n \tag{B.4}$$

where $n = N_t p$. X and U define the allowable values for each control variable at all time steps. Reservoir simulator equations can be rewritten in a more compact manner, omitting time, as:

$$\boldsymbol{g}(\boldsymbol{x}, \boldsymbol{u}) = \boldsymbol{0} \tag{B.5}$$

B.1. Simplified optimization problem formulation

Waterflooding optimization is subject to a set of operational and economic constraints, making it, in its most generic form, a non-linearly constrained mixed integer optimization problem (MINLP). As such, it can be formulated as finding the optimal control variables which minimize the objective function J, subject to the generic set of linear or non-linear constraints $\boldsymbol{c}$. If the same approach as with $\boldsymbol{g}$ for time dependence is also applied to $\boldsymbol{c}$, as in (B.5), the problem can be formulated as:

$$\begin{cases} \min\limits_{\boldsymbol{u} \in U} J(\boldsymbol{x}, \boldsymbol{u}) \\ \boldsymbol{c}(\boldsymbol{x}, \boldsymbol{u}) \leqslant \boldsymbol{0} \\ \boldsymbol{g}(\boldsymbol{x}, \boldsymbol{u}) = \boldsymbol{0} \end{cases} \tag{B.6}$$

where:

- $\boldsymbol{x}$ is the vector of dynamic state variables of the model (pressure, saturation, etc.) at all grid blocks;
- $\boldsymbol{u}$ is the vector of continuous well control variables, of dimension n (e.g. injection rates, BHPs);
- $U = \{\boldsymbol{u} \in \mathbb{R}^n; \boldsymbol{u}_l \leqslant \boldsymbol{u} \leqslant \boldsymbol{u}_u\}$ defines the allowable values for $\boldsymbol{u}$ at all time steps;
- $\boldsymbol{g}$ is the set of reservoir simulation equations to be solved to evaluate J and $\boldsymbol{c}$ at all

time steps;

– $\boldsymbol{c}$ is the set of linear and nonlinear constraints on all control variables at all time steps.

The dimension of the optimization problem is n. Well controls are represented as piecewise constant functions in time with N_t time intervals. Examples of constraints include:

– maximum BHPs at injectors;

– minimum BHPs at producers;

– bounds on injection and production rates at well or field level;

– maximum water cut at producers;

– minimum profit per barrel of oil.

Depending on whether constraints should simply guide the optimization process, or instead be honored precisely in the solution (which would instead be considered unacceptable), they can be implemented in the optimization algorithm with a penalty function, a filter method, or directly as a "hard" constraint. The implementation of constraints, especially if nonlinear, is also a topic of interest in literature [54].

This work assumes that the most effective way reservoir engineers have to improve field performance is to control injection well rates. Hence, only injection wells are controlled, and each injection well only has its injection rate as control variable, such that the dimension of the problem is $n = N_{in}N_t$, where N_{in} is the number of injection wells.

Note: the above procedure can be extended to discrete and categorical control variables, with their own domains, as for continuous control variables

C | Capacitance resistance models

Capacitance resistance models (CRMs) have been inspired by electrical circuits. Following the electrical analogy, reservoir fluids, pressure and reservoir transmissivity can be viewed as current, voltage and conductance respectively. In this way, the whole reservoir is modeled as a circuit, defined by a number of parameters. At any point in time, if initial conditions and the circuit's parameters are known, production rates at producers can be computed.

CRMs make use of nonlinear regression on historical data to calculate those parameters. In the case of reservoir systems, they are known as interwell connectivities and time constants (or response delays). Connectivities f_{ij} quantify the connection between injector i and producer j: in a way, they are a measure of the reservoir porosity, permeability and fluid properties [14]. Time constants τ quantify the attenuation of the output response of the reservoir (production rates at producers) to an input signal (injection rates at injectors) [14]: as such, they represent a combined measure of the reservoir's compressibility c_t, pore volume V_p and productivity index PI of the production well. Considering only a single injector-producer connection ij, they are defined mathematically as:

$$\tau = \frac{c_t V_p}{PI} \tag{C.1}$$

$$f_{ij} = \frac{q_{ij}}{q_{in,i}} \tag{C.2}$$

for $i = 1, ..., N_{in}$ and $j = 1, ..., N_p$, where f_{ij} is the fraction of the injection rate at injector i contributing to production rate at producer j in steady-state conditions. They are subject to the following constraints, which ensure results are physically meaningful:

$$\tau \geqslant 0 \tag{C.3}$$

$$0 \leqslant f_{ij} \leqslant 1 \tag{C.4}$$

$$\sum_{j=1}^{N_{in}} f_{ij} \leqslant 1 \tag{C.5}$$

C.1. Assumptions and equations

The underlying assumptions for CRM models are described in Holanda et al. (2018)[14] :

- low compressibility of fluids and rock;
- constant total volume with instantaneous pressure equilibrium;
- constant temperature;
- constant productivity index;
- immiscible phases with negligible capillary pressure;
- stepwise variation of injection rates and linear variation in BHPs.

Regression is based on historical data fitted onto the material balance equation:

$$c_t V_p \frac{d\bar{p}(t)}{dt} = f q_{in}(t) - q(t) \tag{C.6}$$

and the deliverability equation:

$$PI = \frac{q(t)}{\bar{p}(t) - p_{wf}(t)} \tag{C.7}$$

where $\bar{p}, p_{wf}, q$ and q_{in} are average reservoir pressure, well bottomhole pressure, liquid production rate and water injection rate, respectively. Combined, they result in the CRM equation:

$$\tau \frac{dq(t)}{dt} + q(t) = f q_{in}(t) - \tau PI \frac{dp_{wf}(t)}{dt} \tag{C.8}$$

For the solution of (C.8), time is discretized as a sequence of intervals of generic length, defined as $t_k = t_0 + \sum_{j=1}^{k} \Delta t_k$ with $k = 1, ..., N_t$ instead of the usual $t_k = t_0 + k\Delta t$ with equally spaced intervals. To further distinguish between phases, a fractional flow model for a phase must be introduced. For the sake of simplicity, only water and oil are considered. The mathematical formulation for the oil fraction (or cut) f_o typically depends on the type and stage of the field. A common oil cut model used for mature fields in literature is some form of two-parameter semi-empirical power law, as suggested in Sayarpour et al. (2009)[47], which correlates cumulative injection rate CWI to oil production rate q_o:

$$f_o(t) = \frac{1}{1 + WOR(t)} = \frac{1}{1 + \alpha CWI(t)^{\beta}} \tag{C.9}$$

Once the oil fractional flow f_o is defined, oil and water rates, q_o and q_w can be computed as:

$$q_o(t) = f_o(t)q(t) \tag{C.10}$$

$$q_w(t) = q(t) - q_o(t) \tag{C.11}$$

where WOR is known as water-oil-ratio. Again, historical data is fitted on the fractional flow model to find α and β values. Depending on the needs, oil rate can be calculated at a well, group of wells or fields level, by fitting data accordingly. No previous knowledge of the reservoir's physical properties is required. Nonetheless, the information about connectivities and time constants can be valuable for other purposes as well, such as identifying the presence of channelized areas or faults [31].

C.2. Analytical solutions

Depending on the control volume to which the above equation is applied, we can distinguish different CRM models, as formulated by Sayarpour at al. (2007)[55]:

– CRMT (as in "tank") with the volume of the entire field F. The CRM equation is:

$$\tau_F \frac{dq(t)_F}{dt} + q_F(t) = q_{in,F}(t) \tag{C.12}$$

and its analytical solution is:

$$q_F(t_{N_t}) = q_F(t_0)e^{-\frac{t_{N_t} - t_0}{\tau_F}} + \sum_{k=1}^{N_t} \left[e^{-\frac{t_{N_t} - t_k}{\tau_F}} \left(1 - e^{-\frac{\Delta t_k}{\tau_F}} \right) q_{in,F}(t_k) \right] \tag{C.13}$$

where $\tau_F = \left(\frac{c_t V_p}{PI}\right)_F$ is a field F parameter. A schematic representation of the CRMT model is reported in Figure C.1;

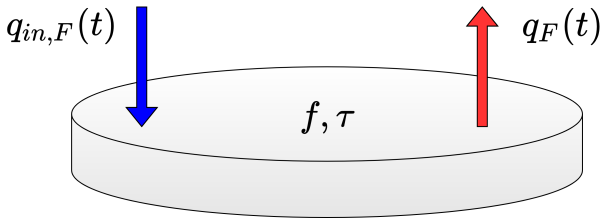


Figure C.1: CRMT model.

– CRMP (as in "producer") with the drainage volume around each producer. The CRM equation is:

$$\tau_j \frac{dq_j(t)}{dt} + q_j(t) = \sum_{i=1}^{N_{in}} f_{ij} q_{in,i}(t) - \tau_j PI_j \frac{dp_{wf,j}(t)}{dt} \tag{C.14}$$

and its analytical solution is:

$$\begin{aligned} q_j(t_{N_t}) &= q_j(t_0) e^{-\frac{t_{N_t}-t_0}{\tau_j}} + \\ &\sum_{k=1}^{N_t} \left\{ e^{-\frac{t_{N_t}-t_k}{\tau_j}} \left(1 - e^{-\frac{\Delta t_k}{\tau_j}}\right) \left[\sum_{i=1}^{N_{in}} q_{in,i}(t_k) - \tau_j PI_j \frac{\Delta p_{wf,j}(t_k)}{\Delta t_k} \right] \right\} \end{aligned} \tag{C.15}$$

where $\tau_j = \left(\frac{c_t V_p}{PI}\right)_j$ is a producer j parameter associated with its effective area. A schematic representation of the CRMP model is reported in Figure C.2;

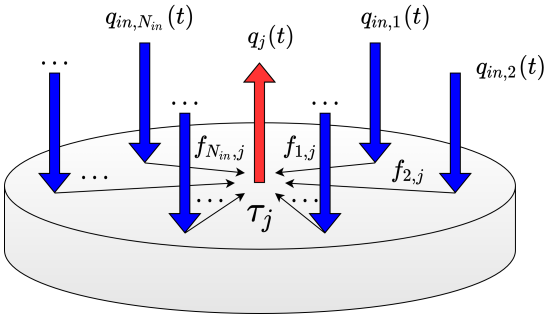


Figure C.2: CRMP model.

– CRMIP (as in "injector/producer") with the drainage volume between each injector-producer pair. The CRM equation is:

$$\tau_{ij} \frac{dq_{ij}(t)}{dt} + q_{ij}(t) = f_{ij} q_{in,i}(t) - \tau_{ij} PI_{ij} \frac{dp_{wf,j}(t)}{dt} \tag{C.16}$$

and its analytical solution is:

$$\begin{aligned} q_j(t_{N_t}) &= \sum_{i=1}^{N_{in}} q_{ij}(t_k) = q_{ij}(t_0) e^{-\frac{t_{N_t}-t_0}{\tau_{ij}}} + \\ &\sum_{i=1}^{N_{in}} \left\{ \sum_{k=1}^{N_t} \left[e^{-\frac{t_{N_t}-t_k}{\tau_{ij}}} \left(1 - e^{-\frac{\Delta t_k}{\tau_{ij}}}\right) \left(f_{ij} q_{in,i}(t_k) - \tau_{ij} PI_{ij} \frac{\Delta p_{wf,j}(t_k)}{\Delta t_k} \right) \right] \right\} \end{aligned} \tag{C.17}$$

where $\tau_{ij} = \left(\frac{c_t V_p}{PI}\right)_{ij}$ is a producer-injector ij parameter associated with the control volume between them. A schematic representation of the CRMIP model is reported in Figure C.3.

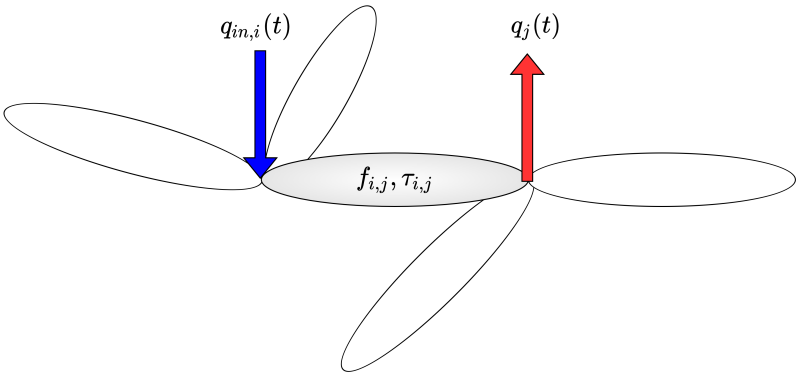


Figure C.3: CRMIP model.

Drainage volume represents the portion of the volume of a reservoir drained by a well. Each analytical solution is the sum of three contributions to oil production: due to primary production or depletion (first term), due to water injection (second term), due to BHP variation (third term).

C.3. Applications in reservoir engineering

Albertoni and Lake (2002)[38] first introduced the use of CRMs with the purpose of identifying injection patterns, while the formal mathematical model using material balance and time constants is due to Yousef et al. (2006)[46, 56]: CRMs were brought from theoretical backgrounds to real field applications. It was not until Sayarpour at al. (2007)[55, 57] that material balance ODEs above were first solved analytically in closed form. In Sayarpour et al. (2009)[47], they were applied for production optimization using heuristic water reallocation techniques. Weber at al. (2009)[58] developed a method to pre-process data and reduce parameter requirements for CRMs, decreasing the dimensionality of the problem. Jahangiri at al.(2014)[59] applied waterflooding optimization for a field in the North Sea, combining a CRMs with a heuristic optimization algorithm based on injector-producer connectivity. Similarly, Temizel et al. (2017)[60] applied waterflooding optimization using CRMs and investigating the most important parameters, such as magnitude of injection, vertical conformance, area conformance of wells.

It is clear how through CRMs, it is possible to forecast production rates from injection rates and BHPs. CRMs' main disadvantage is their limited flexibility: the analytical

solution of the ODE can be obtained only in specific conditions (linear variations of BHP and fixed injection rates between two consecutive time steps) and with a suitable fractional flow model [14]. Finally, CRMs' performance in terms of computational time and accuracy of forecast may deteriorate as field size (thus the number of parameters) increases.

D | Optimization algorithms

In this section a brief description of the optimization algorithms used in this thesis is given, along with a simple process flow diagram.

D.1. Trust-region algorithm

Trust-region methods are a class of algorithms opposed to line-search methods. Instead of finding a direction of improvement and selecting a step length, a trust region is identified (similar to a maximum step length) and then a point of improvement is searched for in the region. A new trust region is centered in the improved point. Similarly to Sequential Quadratic Programming (SQP), the objective function J is approximated by a quadratic function m (or quadratic model) around a point: the model is "trusted" to be an adequate representation of the objective function in the region. The Taylor series expansion of the objective function around the solution at iteration (i), $\boldsymbol{u}^{(i)}$ is:

$$J(\boldsymbol{u}^{(i)} + \boldsymbol{p}) = J^{(i)} + \boldsymbol{g}^{(i)T}\boldsymbol{p} + \frac{1}{2}\boldsymbol{p}^T\nabla^2 J(\boldsymbol{u}^{(i)} + t\boldsymbol{p})\boldsymbol{p} \tag{D.1}$$

with $J^{(i)} = J(\boldsymbol{u}^{(i)})$. The model function is:

$$m^{(i)}(\boldsymbol{p}) = J^{(i)} + \boldsymbol{g}^{(i)T}p + \frac{1}{2}\boldsymbol{p}^T\boldsymbol{B}^{(i)}\boldsymbol{p} \tag{D.2}$$

where $\boldsymbol{g}^{(i)} = \nabla J^{(i)}$, $\boldsymbol{p}$ is the step, t is some scalar in the interval (0,1), and $\boldsymbol{B}^{(i)}$ is an approximation of the Hessian in the second term, in the form of a symmetric matrix (or the actual Hessian in case second-order derivative information is available). The difference between $m^{(i)}(\boldsymbol{p})$ and $(\boldsymbol{u}^{(i)} + \boldsymbol{p})$ is of course $O(\|p\|^2)$ when p is small. For each step, the solution reduces to a minimization problem in the n-dimensional space, formulated as:

$$\min_{\boldsymbol{p}\in\mathbb{R}^n} m^{(i)}(\boldsymbol{p}) \quad \text{subject to } \|\boldsymbol{p}\| \leqslant \Delta^{(i)} \tag{D.3}$$

where $\Delta^{(i)}$ is the trust-region radius and $\|\cdot\|$ is the Euclidean norm. Let us define the ratio:

$$\rho^{(i)} = \frac{J(\boldsymbol{u}^{(i)}) - J(\boldsymbol{u}^{(i)} + \boldsymbol{p}^{(i)})}{m^{(i)}(0) - m^{(i)}(\boldsymbol{p}^{(i)})} \tag{D.4}$$

where the numerator is the actual reduction and the denominator is the predicted reduction, which is always non-negative. Hence, if:

- $\rho^{(i)}<0$, it means $J(\boldsymbol{u}^{(i)}) < J(\boldsymbol{u}^{(i)} + \boldsymbol{p}^{(i)})$, so the step must be rejected, because the step $\boldsymbol{p}^{(i)}$ brought an increase in the objective function (minimization problem);
- $\rho \approx 1$, the model's prediction is accurate, and the trust region is expanded in the next iteration;
- $0 \ll \rho^{(i)} \ll 1$ the trust region remains unaltered;
- $\rho \leqslant 0$ or $\rho \sim 0$ the trust region is shrunk by $\Delta^{(i)}$ in the next iteration.

The trust-region algorithms require thus to solve a series of sub-problems as in (D.3) with quadratic objective function $m^{(i)}(\boldsymbol{p})$ and constraint $\|p\| \leqslant \Delta^{(i)}$. Depending on the method through which (D.3) is solved, different algorithms have been studied, to solve both equality constrained and inequality constrained problems. SciPy's optimization library provides an implementation of the trust-region SQP method for equality constraints, as described by Lalee et al. (1998)[40] and an implementation of the nonlinear interior point trust region optimizer for inequality constraints, as described by Nocedal et al. (1999) [41]. A process flow diagram for the trust-region algorithm is depicted in Figure D.1.

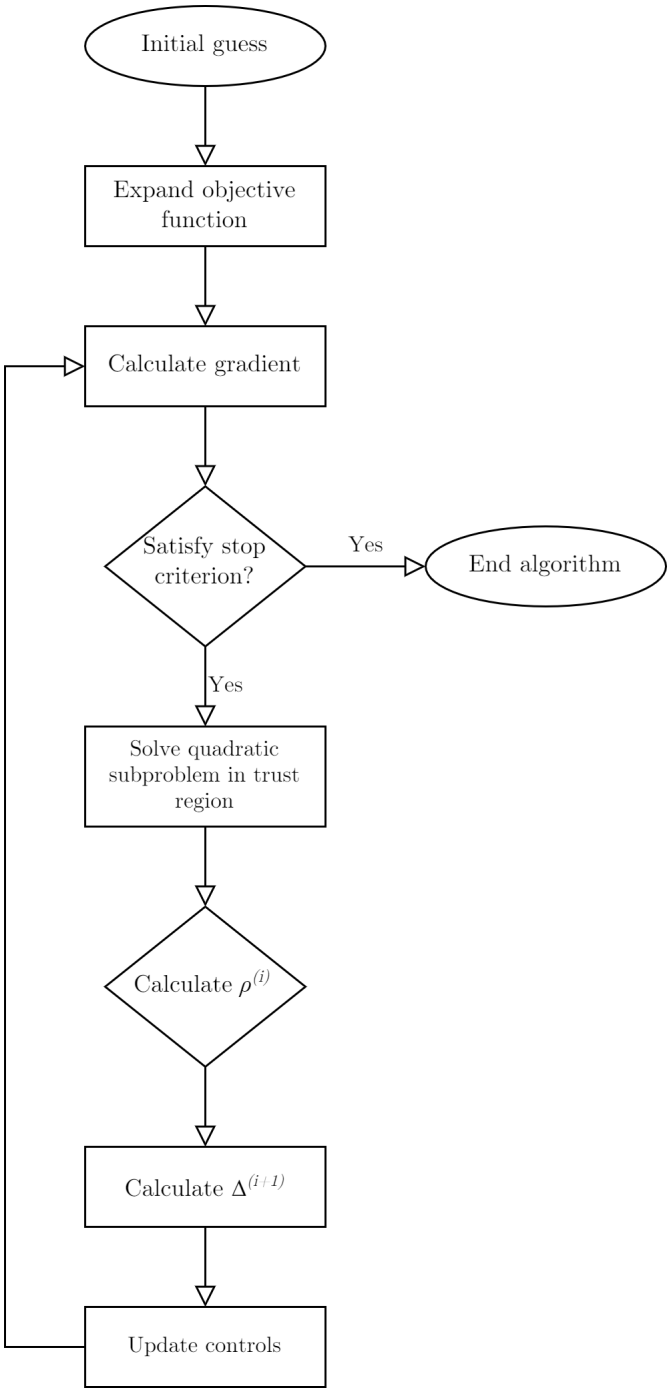


Figure D.1: Trust-region algorithm process flow diagram.

D.2. Genetic algorithm

Holland (1992) introduced genetic algorithms (GAs) [61], inspired by biological evolution in natural environments and the concept of survival of the fittest. In GA, each candidate solution is a chromosome, made up of genes (control variables and their values). Chro-

mosomes evolve through the iterative use of three basic operations: selection, crossover, mutation. The best candidates, according to their objective function value, have greater chance of survival and crossover.

At each iteration, some chromosomes are selected based on the selection operation. They are then combined based on the crossover operation, resulting in an offspring for the next iteration (or generation). Finally, some chromosomes' genes are randomly modified, based on the mutation operation.

The population of P individuals can be described as:

$$\boldsymbol{P} = [\boldsymbol{u}_1^T, ..., \boldsymbol{u}_P^T]^T \tag{D.5}$$

Each individual (or chromosome) is a vector of genes, where each gene is a control variable for the problem, $\boldsymbol{u}_j = [u_{j,1}, ..., u_{j,n}]$, where n is the dimension of the control variable space (i.e. the number of genes in the chromosome). As such, $\boldsymbol{P}$ can be seen as a concatenated vector of vectors.

D.2.1. Selection

The selection operation is based on the fitness of the individuals (i.e. the value of the objective function). Individuals with high fitness are more likely to be selected for reproduction. Both stochastic and deterministic methods can be applied, such as ranking selection (probability of selection is assigned as a 0 to 1 linear function with fitness value), tournament selection (individuals are compared in small groups or couples and the fittest fill the mating pool). A widely used method is the roulette-wheel selection, in which the probability $\mathcal{P}_j$ of an individual $\boldsymbol{u}_j$ of being selected is based on its objective function value:

$$\mathcal{P}_j = \frac{J(\boldsymbol{u}_j)}{\sum_{i=1}^{P} J(\boldsymbol{u}_j)} \tag{D.6}$$

Obviously, in case of a minimization problem, the reverse would be used, such that the lowest J values hold higher chances of being selected. An additional aspect that has been shown to improve convergence for GAs is elitism, in which a specified fraction of the best individuals from one generation, called elitist ratio, is copied directly to the next one.

D.2.2. Crossover

The crossover operation involves only a fraction of the selected individuals (parents portion) and replaces only a specified portion of the previous generation (crossover probability), while the rest are copies of the previous one. Examples of replacement strategies are complete replacement, random replacement or replacement of the worst or oldest individuals. For a two-parent crossover new individuals (offspring), $\boldsymbol{u}^{off,1}$ and $\boldsymbol{u}^{off,1}$, are thus produced by the crossover operation by an information exchange between the two parents, $\boldsymbol{u}^{par,1}$ and $\boldsymbol{u}^{par,1}$, and contain features from both. Commonly used crossover techniques are:

- one-point: a random location r along the genome (i.e. encoded control variables) specifies where the first parent's genome is cut off and where the second parent's starts in one offspring and vice versa for the other:

$$\boldsymbol{u}^{off,1} = [u_1^{par,1}, ..., u_r^{par,1}, u_{r+1}^{par,2}, ..., u_n^{par,2}]^T \tag{D.7}$$
$$\boldsymbol{u}^{off,2} = [u_1^{par,2}, ..., u_r^{par,2}, u_{r+1}^{par,1}, ..., u_n^{par,1}]^T \tag{D.8}$$

- two-point: two random locations r and s along the genome specify where the first parent's genome is cut off, where the second parent's starts, and where the first parent's starts again in one offspring and vice versa for the other:

$$\boldsymbol{u}^{off,1} = [u_1^{par,1}, ..., u_r^{par,1}, u_{r+1}^{par,2}, ..., u_s^{par,2}, u_{s+1}^{par,1}, ..., u_n^{par,1}]^T \tag{D.9}$$
$$\boldsymbol{u}^{off,2} = [u_1^{par,2}, ..., u_r^{par,2}, u_{r+1}^{par,1}, ..., u_s^{par,1}, u_{s+1}^{par,2}, ..., u_n^{par,2}]^T \tag{D.10}$$

- uniform : single elements, rather than sections, of the genome are exchanged between the parents to produce the offspring (i.e. each gene of one offspring is randomly selected between the parents, while for the other offspring the opposite occurs)

D.2.3. Mutation

The mutation operation is used to avoid trapping in local optima, by adding new genetic material in the population. Hence, it can prevent premature convergence. It occurs after crossover and only requires one parent to generate offspring. Mutation probability defines how often genes are mutated. If set to zero, the offspring stays the same after crossover; if instead mutation is present, part of chromosome is changed, with a chance of being mutated equal to the mutation probability. In particular, mutation can be implemented as point mutation (changes occur to single genes, by replacement, deletion or insertion) or

large-scale mutation (changes occur in multiple positions or in a single position involving more than one gene). Mutation operators define the mathematical formulation of the operation. Gaussian mutation is a common method: for every gene i (control variable) that undergoes mutation:

$$u_i^{mut} = u_i + \mathcal{N}(0, \sigma_i) \tag{D.11}$$

where $\mathcal{N}(0, \sigma_i)$ is a random number drawn from a normal distribution with zero mean and standard deviation σ_i, specific to the control variable u_i. A process flow diagram for the genetic algorithm is depicted in Figure D.2.

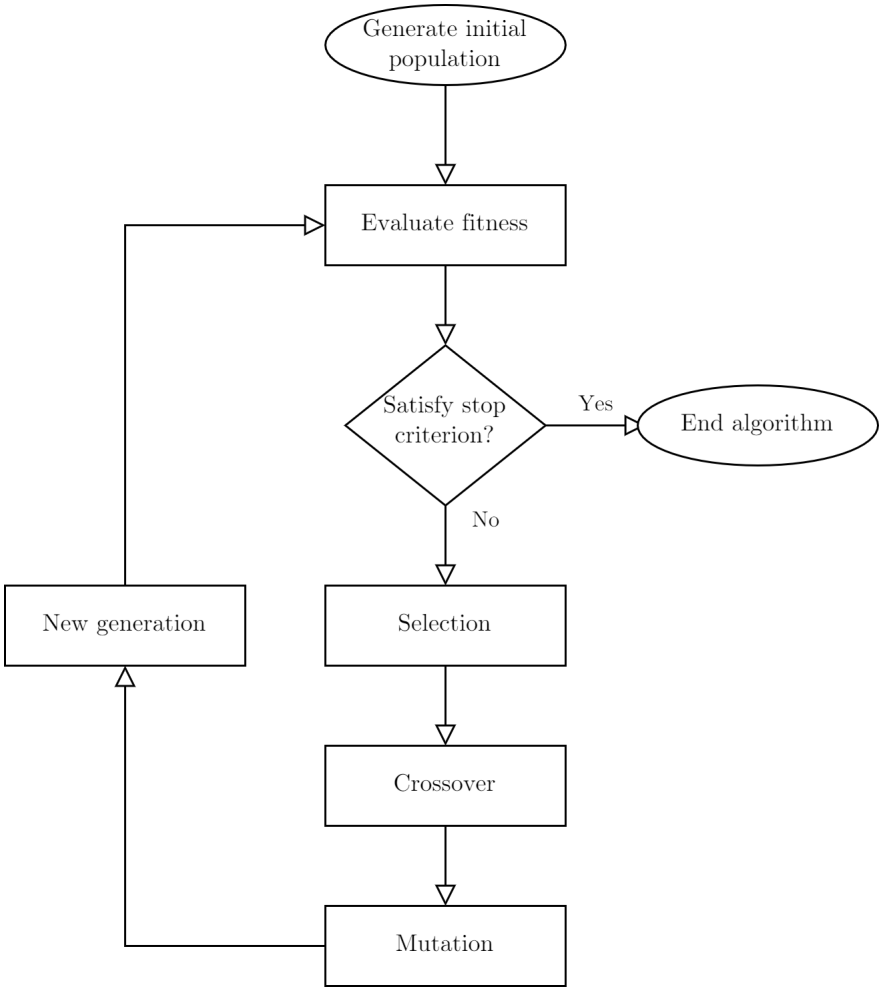


Figure D.2: Genetic algorithm process flow diagram.

D.3. EnOpt algorithm

The ensemble of N_e candidate solutions (control variable vectors) and their respective objective function values can be represented as concatenated control vectors:

$$\boldsymbol{u}_e = [\boldsymbol{u}_1^T, ..., \boldsymbol{u}_{N_e}^T]^T \tag{D.12}$$

$$\boldsymbol{J}_e = [J_1, ..., J_{N_e}]^T = [J(\boldsymbol{u}_1), ..., J(\boldsymbol{u}_{N_e})]^T \tag{D.13}$$

The ensemble average control vector and objective function are defined as:

$$\bar{J} = \frac{1}{N_e}\sum_{j=1}^{N_e}(J_j) \tag{D.14}$$

$$\bar{\boldsymbol{u}} = \frac{1}{N_e}\sum_{j=1}^{N_e}(\boldsymbol{u}_j) \tag{D.15}$$

$$\tag{D.16}$$

where the $\bar{\boldsymbol{u}}$ is strictly a scalar, but the vector representation is more convenient for operations involving other vectors. In other words, it can be seen as a vector with identical scalar values $\bar{u}$ and size N_e. The mean-shifted control vectors and objective function values are defined as:

$$\hat{\boldsymbol{U}} = [\boldsymbol{u}_1^T - \bar{\boldsymbol{u}}^T, ..., \boldsymbol{u}_{N_e}^T - \bar{\boldsymbol{u}}^T]^T \tag{D.17}$$

$$\hat{\boldsymbol{J}} = [J_1 - \bar{J}, ..., J_{N_e} - \bar{J}]^T \tag{D.18}$$

It is convenient for ensemble optimization to compute the gradient of the mean-shifted objective function, defined as $\boldsymbol{g}_u = \nabla_{\boldsymbol{u}}\hat{\boldsymbol{J}}$, with respect to the controls instead of simply the gradient of the objective function. In case the number of ensemble members is higher than the number of controls (i.e. $N_e > n$) the approximate gradient with respect to the controls can be obtained with a classical least square (LS) solution:

$$\boldsymbol{g}_u = (\hat{\boldsymbol{U}}^T\hat{\boldsymbol{U}})^{-1}\hat{\boldsymbol{U}}^T\hat{\boldsymbol{J}} \tag{D.19}$$

or, rearranged differently:

$$\boldsymbol{g}_u = \boldsymbol{C}_{uu}^{-1}\boldsymbol{C}_{uJ} \tag{D.20}$$

where:

$$\boldsymbol{C}_{uu} = \frac{1}{N_e - 1}(\hat{\boldsymbol{U}}^T\hat{\boldsymbol{U}}) \tag{D.21}$$

$$\boldsymbol{C}_{uJ} = \frac{1}{N_e - 1}(\hat{\boldsymbol{U}}^T\hat{\boldsymbol{J}}) \tag{D.22}$$

are the covariance and cross covariance matrices of control variables and control variables to objective function, respectively. However, since typically $N_e < n$ for large-scale ap-

plications (undetermined case), $\hat{\boldsymbol{U}}^T\hat{\boldsymbol{U}}$ becomes non-singular: thus its inverse cannot be computed directly (or the associated system of equations to compute it cannot be solved). Suggested methods in this case are Singular Matrix Decomposition (SVD), through a pseudo-inverse matrix, as used by Fonseca et al. (2013)[62] or, as proposed by Chen et al. (2009)[33]:

$$\boldsymbol{g}'_u = \boldsymbol{C}_{uJ} = \boldsymbol{C}_{uu}\boldsymbol{g}_u \tag{D.23}$$

where the gradient is approximated by the ensemble cross-covariance matrix $\boldsymbol{C}_{uJ}$. In addition, they propose to adopt the covariance matrix $\boldsymbol{C}_{uu}$ as preconditioner, such that:

$$\boldsymbol{g}''_u = \boldsymbol{C}_{uu}\boldsymbol{C}_{uJ} = \boldsymbol{C}_{uu}\boldsymbol{C}_{uu}\boldsymbol{g}_u \tag{D.24}$$

The preconditioner limits the frequency and magnitude of changes in the well controls as described in Chen et al. (2009)[33]. Once the approximate gradient is computed, any gradient-based optimization algorithm can be applied to update the ensemble of control vectors $\boldsymbol{u}_e$ from iteration (i) to iteration $(i+1)$. A common method is a simple line-search combined with the steepest descent method:

$$\boldsymbol{u}_e^{(i+1)} = \boldsymbol{u}_e^{(i)} + \alpha^{(i)}\boldsymbol{g}_u''^{(i)} \tag{D.25}$$

where $\alpha^{(i)}$ is a variable related to the step size during the line-search (some authors also use the formulation with $\frac{1}{\alpha^{(i)}}$). The step length can be assumed constant for each iteration and be reduced by a specified coefficient every time, or it can be optimized for an assigned range of values in the given descent direction for each iteration. The first approach is simpler, but not objective-function dependent, while the second approach is computationally heavier (additional forward runs are needed), but it potentially leads to more accurate results. More complex ensemble-based optimization methods also make use of an approximate Hessian [62]. A process flow diagram for the EnOpt algorithm is depicted in Figure D.3.

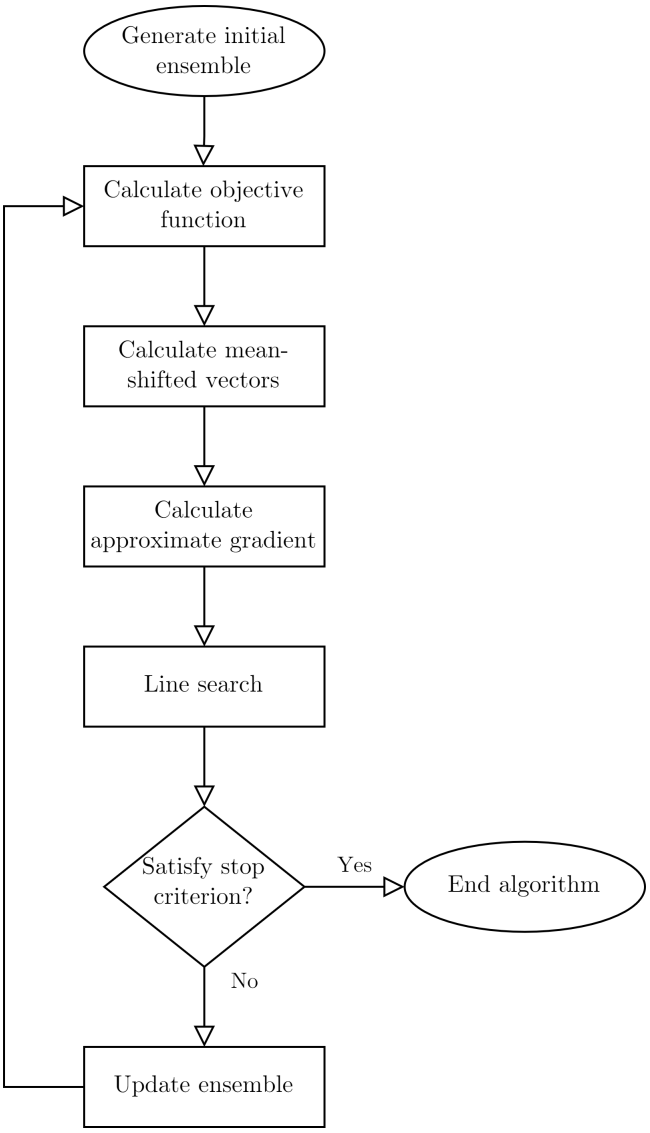


Figure D.3: EnOpt algorithm process flow diagram.

E | Streamline simulation

A streamline is a line tangent to the local velocity field $\boldsymbol{v}$ at a given point in time, that allows to connect source (injector) and sink (producer)[63], such that:

$$\boldsymbol{v} \times d\xi = \mathbf{0} \quad \text{or} \quad \frac{dx}{v_x} = \frac{dy}{v_y} = \frac{dz}{v_z} \tag{E.1}$$

where ξ is the generic streamline coordinate. Streamlines can be used in traditional finite element reservoir simulation for visualization purposes. Alternatively, streamline models can also be used to run simulations. However, streamline simulation is different from traditional reservoir simulation: in the former, transport equations are solved along a flow-based grid, defined by streamlines; in the latter, they are solved from cell to cell as in classic finite element simulation.

The governing equation for pressure, for a multiphase flow without any capillary of diffusion effects, employing the formulation used in Holstein et al. (2007)[64] with the explicit depth below datum D, is:

$$\nabla \cdot \sum_{j=1}^{N_{ph}} \frac{\boldsymbol{K} k_{rj}}{\mu_j} (\nabla P + \rho_j \boldsymbol{g} D) = 0 \tag{E.2}$$

where D is the depth below datum. The explicit material balance equation for each incompressible phase j is then:

$$\phi \frac{\partial S_j}{\partial t} + \boldsymbol{q}_t \cdot \nabla f_j + \nabla \cdot \boldsymbol{G}_j = \mathbf{0} \tag{E.3}$$

with:

$$\boldsymbol{G}_j = \boldsymbol{K} \cdot g f_j \nabla D \sum_{i=1}^{N_{ph}} \lambda_i (\rho_i - \rho_j) \tag{E.4}$$

where $\boldsymbol{G}_j$ is the velocity component of phase j resulting from gravity effects due to phase density differences. While in standard reservoir simulation based on finite difference (E.3)

is discretized and solved on the same underlying grid on which the pressure equation is solved, in streamline simulation it is solved along each streamline using the time of flight (TOF) coordinate transform. The TOF τ is the time required for a neutral tracer to travel a distance s along each streamline:

$$\tau = \int_0^s \frac{\phi(\xi)}{|\boldsymbol{q}_t(\xi)|} d\xi \tag{E.5}$$

Starting from the total velocity field, streamlines are traced from sources (injectors) to sinks (producers): a common method is Pollock's algorithm [65]. (E.5) results in the definition of:

$$|\boldsymbol{q}_t| \frac{\partial}{\partial \xi} \equiv \boldsymbol{q}_t \cdot \nabla = \phi \frac{\partial}{\partial \tau} \tag{E.6}$$

As described in Holstein and Lake (2007)[64], using (E.2) and (E.6), (E.3) becomes:

$$\frac{\partial S_j}{\partial t} + \frac{\partial f_j}{\partial \tau} + \frac{1}{\phi} \nabla \cdot \boldsymbol{G}_j = 0 \tag{E.7}$$

Since the gravity term is not aligned along streamlines, (E.7) can be split into two parts: a convective term and a gravity term (due to phase density differences). The convective term, solved along streamlines, is:

$$\frac{\partial S_j^c}{\partial t} + \frac{\partial f_j}{\partial \tau} = 0 \tag{E.8}$$

while the gravity term, solved along gravity lines is:

$$\frac{\partial S_j^g}{\partial t} + \frac{1}{\phi} \nabla \cdot \boldsymbol{G}_j = 0 \tag{E.9}$$

Both equations can be solved by standard finite difference techniques. However, since fluid distributions and well conditions change over time, streamlines paths change as well. Thus, it is necessary to periodically update the total velocity field, such that the new streamlines can be redrawn to reflect the non-linear nature of the displacement. The following algorithm is used to move the 3D saturation distribution forward in time. For every time step, the algorithm is:

1. Solution of (E.2) from initial well, pressure and saturations conditions for each cell, as in standard finite difference simulation

2. From pressure, total velocity calculation for each cell interface, using Darcy's law
3. Tracing of streamlines from the velocity field using Pollock's algorithm [65], where initial streamlines conditions are obtained by a mapping of the underlying 3D grid onto each streamline
4. Solution of 1D mass conservation equations (E.8) along each streamline
5. Mapping of the new streamline saturations back onto the 3D grid
6. Solution of 1D mass conservation equations (E.9) along vertical gravity lines
7. Mapping of the new gravity lines saturations onto the 3D grid

Figure E.1 shows the procedure schematically.

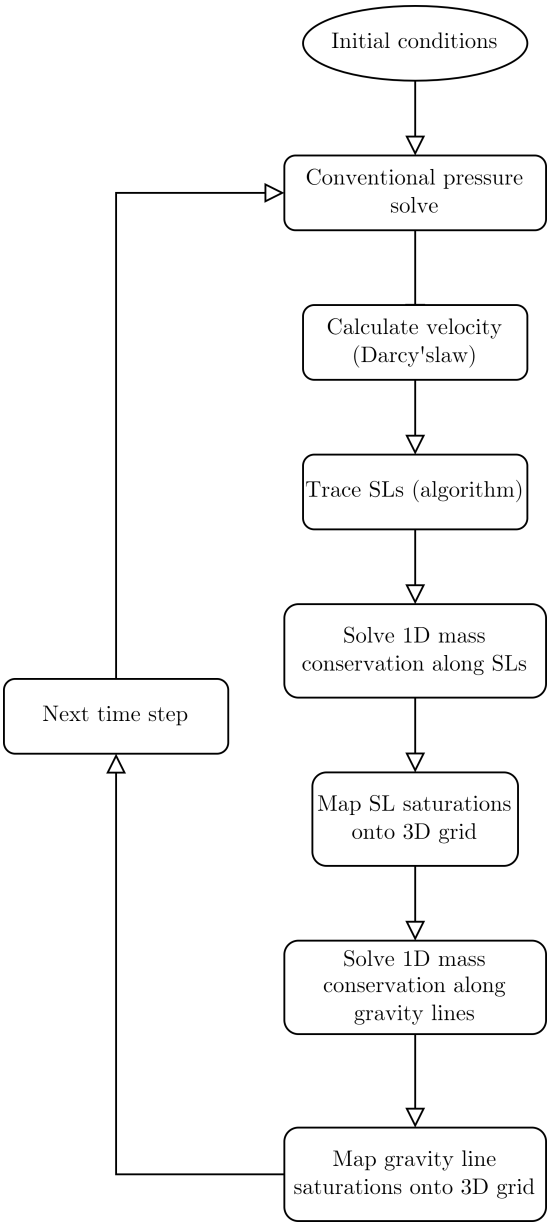


Figure E.1: Streamline simulation process flow diagram.

E.1. Useful definitions

Below are some important definitions for streamline simulation, all valid for $i = 1, ..., N_i$ and $j = 1, ..., N_p$, as defined in Thiele and Batycky (2003)[63]. They are all instantaneous, being ratios of rates. Injection efficiency is defined as:

$$IE_i = \frac{q_{o,ij}}{q_{in,i}} = \frac{\sum_{s=1}^{N_s} q^s_{o,j}}{q_{in,i}} \tag{E.10}$$

Average (or field) injector efficiency is defined as:

$$\overline{IE} = \frac{\sum_{j=1}^{N_p} q_{o,j}}{\sum_{i=1}^{N_{in}} q_{in,i}} \tag{E.11}$$

Well allocation factors ($WAFs$) can be defined with an injection perspective (injector as parent, in this case, the definition is practically the same as inter-well connectivity (C.2)) or producer perspective (producer as parent):

$$WAF_{ij} = \frac{q_{in,ij}}{q_{in,i}} = \frac{\sum_{s=1}^{N_s} q^s_{in,ij}}{q_{in,i}} \quad \text{(injector as parent)} \tag{E.12}$$

$$WAF_{ij} = \frac{q_{o,ij}}{q_{o,j}} = \frac{\sum_{s=1}^{N_s} q^s_{o,ij}}{q_{o,i}} \quad \text{(producer as parent)} \tag{E.13}$$

where:

- N_{in} is the number of injectors;
- N_p is the number of producers;
- N_s is the number of streamlines that make up the bundle connecting injector i to producer j: visually, they represent the oil produced by the various streamline bundles that start in every injector and end in the various producers. In other words, producers whose production is supported by those injectors (also called offset producers);
- $q^s_{o,j}$ is the oil production rate from producer j along streamline s;
- $q_{in,i}$ is the water injection rate from injector i;
- $q^s_{in,ij}$ is the water injection rate flowing through the streamline s;connecting injector i to producer j.

$WAFs$ with an injector perspective give a measure of how water injection from an injector distributes to support the different producers. $WAFs$ with a producer perspective give a measure of how oil production from a producer is supported by the different injectors.

Injected water lost to the aquifer surrounding the reservoir is included in the denominator of (E.10), while oil produced due to a supporting aquifer is not included in the numerator

of (E.10) [63].

E.2. Waterflooding optimization through streamlines

Waterflooding optimization through streamline simulation was first studied to expand flood management schemes beyond standard surveillance methods and traditional workflows based on finite difference simulation [63]. In this way, it has emerged in the last decades as a powerful complementary tool to those more traditional optimization methods. One powerful aspect of streamline simulation is the ability to visualize, at any instant in time, how the reservoir is connected and how much fluid is allocated between injector-producer pairs. In a way, it gives a "snapshot" of the reservoir [66]. As such, streamlines allow to identify producer-injector connections and are a widely used tool also in other areas of reservoir engineering, such as production data analysis. As highlighted above, streamline simulation does not simply trace streamlines, but it solves 1D transport equations along each one of them. This is what allows to extract injector-producer connections, that would not be obtainable from streamline tracing alone.

E.2.1. FloodOpt ®software

Although streamline-based simulation can be integrated into a standard, optimization-guided procedure, as in Wen et al. (2014)[67] and Park and Datta-Gupta (2013)[68], a valid alternative can be found in simulator-based commercial software to perform waterflooding optimization. Simulator-based means that they still require a geological model, but their operating principle allows them to run fewer simulator runs compared to a standard simulator coupled with an optimization algorithm. In fact, information on how to tweak control variables is extrapolated from simulation outputs. FloodOpt®, developed by *StreamSim*®[45], is a state-of-the-art software that performs streamline-based waterflooding optimization. FloodOpt® uses the output of a reservoir simulation to draw streamlines, which are then used for optimization, with a given injection water availability target. The software tends to "reward" efficient (i.e. with higher than average efficiency) injectors and "penalize" inefficient ones (i.e. with lower than average efficiency), by reallocating injected water from the latter to the former, within the field target limit.

Optimization algorithm

FloodOpt®'s optimization algorithm [69] can be summarized as follows:

1. Simulation of the reservoir at initial time step and calculation the velocity field

2. Tracing of streamlines from velocity fields and calculation $WAFs$, as explained above
3. Calculation of average and single injector efficiencies with (E.10)
4. Calculation of weights associated with each injector, with:

$$w_i = \min\left(w_{max}, w_{max}\left(\frac{IE_i - \overline{IE}}{IE_{max} - \overline{IE}}\right)^a\right) \quad \text{if } IE_i > \overline{IE} \tag{E.14}$$

$$w_i = \max\left(w_{min}, w_{min}\left(\frac{IE_i - \overline{IE}}{IE_{min} - \overline{IE}}\right)^a\right) \quad \text{if } IE_i < \overline{IE} \tag{E.15}$$

where a, w_{max}, w_{min} are arbitrary parameters. The new unconstrained injection rate target is:

$$q'_{in,i} = q_{in}(1 + w_i) \quad \text{for } i = 1, ..., N_{in} \tag{E.16}$$

5. Rescaling of the unconstrained new injection rates to ensure that they add up to the field injection target $q^{max}_{in,F}$ using the formula:

$$q^{new}_{in,i} = \frac{q'_{in,i}}{\sum_{i=1}^{N_{in}} q'_{in,i}} q^{max}_{in,F} \quad \text{for } i = 1, ..., N_{in} \tag{E.17}$$

6. Update of production rates at offset wells accordingly using $WAFs$

The same procedure is reported schematically in Figure E.2.

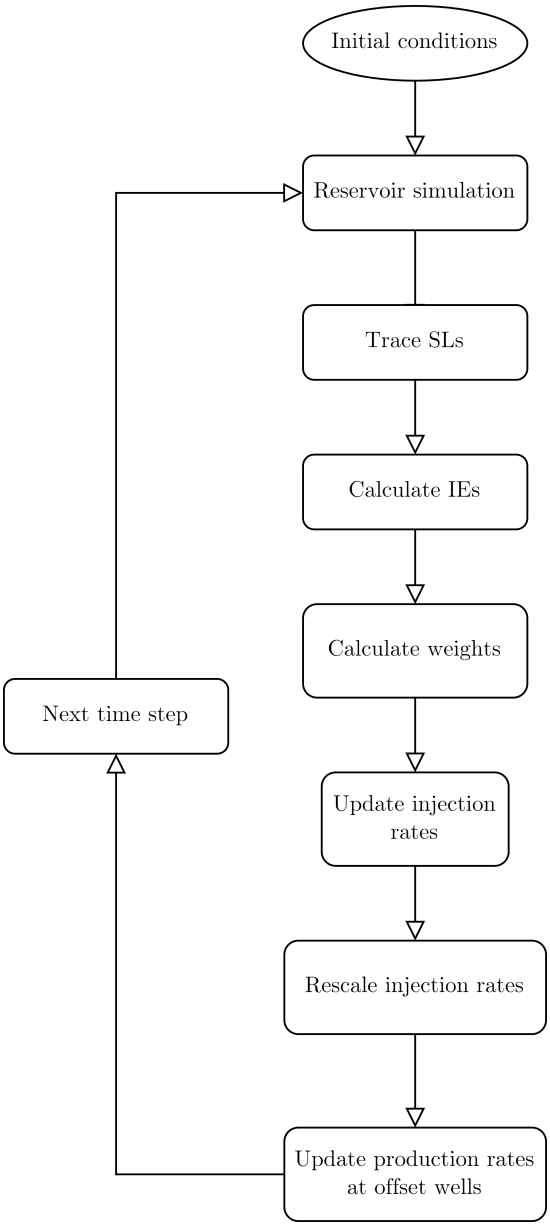


Figure E.2: FloodOpt optimization process flow diagram.

This procedure is repeated for each of the time steps of the optimization problem. Frequency of optimization, i.e. the number of time steps to optimize, cannot exceed certain limits (order of months). Not only would it significantly increase runtimes and might introduce undesirable numerical artifacts in the mapping process between streamlines and the underlying grid [63], but it would also be unrealistic.

Details on the specific weight function to use and its parameters for weights w_i are not discussed in further detail. Although in general they are arbitrary, they should have a smooth behavior near the average efficiency and promote increasing or decreasing weights with

increasing or decreasing efficiencies [63]. More details can be found in Thiele (2005)[69].

The reason why offset production rates of all the producers must be updated as well is that FloodOpt®'s strategy aims at preserving the streamline pattern of the previous time step. Otherwise, streamlined geometry would change significantly at the next time step, yielding a totally different pattern from the one used to calculate the new rates in the first place, leading to an inconsistent workflow [66]. Thus, an increase in injection rate must be followed by an increase in production rate at all its offset producers, such that the total increase in production rate is approximately equal to the increase in injection rate. The reason why streamline patterns would change significantly is that if an increase in "push" from one end is not met by an increase in "pull" from the other end, the in-situ volumes would not flow in the desired direction, hence there would be an excess of injected water that is not counterbalanced at the producing end and that will need to find new flow paths, altering streamline patterns [63]. The exact same reasoning applies to a decrease of injected water, hence a shortage instead of an excess.

E.2.2. Other tools

FloodOpt®'s parent software, 3DSL Studio®, allows to extract useful information from any simulation run by visualizing it on streamline maps, flux pattern maps and injector efficiency plots. Below they are explained briefly.

Streamline maps They are 3D or 2D snapshots of the reservoir at a specific moment in time. They display streamlines connecting injectors (sources) and producers (sinks). Streamline maps give a quick idea of how fluids are moving within the reservoir and how strong sink-source connections are. If paired with $WAFs$, they allow for an even clearer visualization of fluid pattern. In fact, for complex reservoirs, streamline bundles can become harder to interpret: hence, to ease reading, they are often collapsed into straight vectors connecting source and sink, where the thicker the vector, the stronger the connection. In this case the maps are called flux pattern maps or simply flux maps (Fmaps).

Flux maps They can be drawn with a producer perspective (producers as parents) or injector perspective (injectors as parents), depending on the definition of $WAFs$ used:

- injectors as parents: for every injector, percentages shown on Fmaps represent the percentage of injection rate supporting each offset producer, at the considered time step ($WAFs$ are defined as in (E.12));

– producers as parents: for every producer, percentages shown on Fmaps represent the percentage of production rate supported by each connected injector, at the considered time step ($WAFs$ are defined as in (E.13)).

Both types can be used, depending on the purpose, and are computed through the aid of FloodOpt®, at a specific point in time.

Injector efficiency plots They are a quick way to visualize the wells that are responsible for the most efficient injection in terms of unit of oil produced per unit of water injection. Injectors are placed along iso-efficiency lines on the plot, which show how efficient they are in relative terms, and on that specific line at a certain distance from the origin, which shows how much they are injecting and how much they are contributing to production in absolute terms. As for Fmaps, they are computed through the aid of FloodOpt®, at a specific point in time.

Benefits

The main benefit of a software like FloodOpt® is employing an automated workflow to identify injection patterns, areas of efficiency and inefficiency within the field, and outline an optimized flooding plan [66]. The distinguishing feature of streamline simulation of solving transport equations along each streamline allows to quantify how much oil is being produced at one end (producer) as a result of injected water on the other end (injector). Furthermore, the number of full physics and subsequent streamline simulations required is simply the number of time steps chosen for optimization, independently of the number of controls or wells. As a result, this approach is computationally efficient [66]. Streamline based optimization is particularly efficient in the cases of constant field injection capacity, with high water production and voidage replacement conditions [63].

Drawbacks

However, in FloodOpt®'s algorithm there is no formal attempt to optimize production through the minimization of a specified objective function [63]. As stated by the developers, there can be no rigorous mathematical proof that the strategy is moving toward the optimal solution, as the performance improvement is the result of enhanced volumetric displacement efficiency, driven by a reallocation of injected water to less swept areas of the reservoir. In other words, the approach in determining new target injection rates and managing constraints is heuristic. In addition, as any simulation-based software, its algorithm relies on a history-matched geological model. Only in those circumstances would well connections and streamline bundles be reasonably accurate.

F | Artificial neural networks

A generic artificial neural network (ANN) layer (i) can be modeled as:

$$\boldsymbol{x}^{(i+1)} = \sigma(\boldsymbol{W}^{T(i)}\boldsymbol{x}^{(i)} + \boldsymbol{b}^{(i)}) \tag{F.1}$$

with:

$$\boldsymbol{x}^{(i)} = (x_1^{(i)}, ..., x_R^{(i)}) \tag{F.2}$$

$$\boldsymbol{b}^{(i)} = (b_1^{(i)}, ..., b_R^{(i)}) \tag{F.3}$$

$$\boldsymbol{x}^{(i+1)} = (x_1^{(i+1)}, ..., x_S^{(i+1)}) \tag{F.4}$$

$$\boldsymbol{W}^{(i)} = \begin{bmatrix} w_{1,1}^{(i)} & \cdots & w_{1,S}^{(i)} \\ \vdots & \ddots & \vdots \\ w_{R,1}^{(i)} & \cdots & w_{R,S}^{(i)} \end{bmatrix} \tag{F.5}$$

where:

- σ is the activation function;
- $\boldsymbol{W}^{(i)}$ is the matrix of $R \times S$ weights of the layer;
- $\boldsymbol{x}^{(i)}$ is the vector of R neurons of the layer;
- $\boldsymbol{b}^{(i)}$ is the vector of R biases of the layer;
- $\boldsymbol{x}^{(i+1)}$ is the vector of S neurons of the next layer.

In this way, the output from a layer (i) is the input for the following layer $(i+1)$, $\boldsymbol{y}^{(i)} = \boldsymbol{x}^{(i+1)}$. Overall, the neural network output vector can be defined as a function $\mathcal{F}^{NN}$ of all the weights and biases θ of its N_l internal layers and the input vector $\boldsymbol{x}_0$ at layer 0:

$$\boldsymbol{y} = \mathcal{F}^{NN}(\theta, \boldsymbol{x}_0) \tag{F.6}$$

where:

$$\theta = \{\boldsymbol{W}^{(i)}, \boldsymbol{b}^{(i)}\}_{i=0}^{N_l} \tag{F.7}$$

A schematic representation of an ANN layer is reported in Figure F.1.

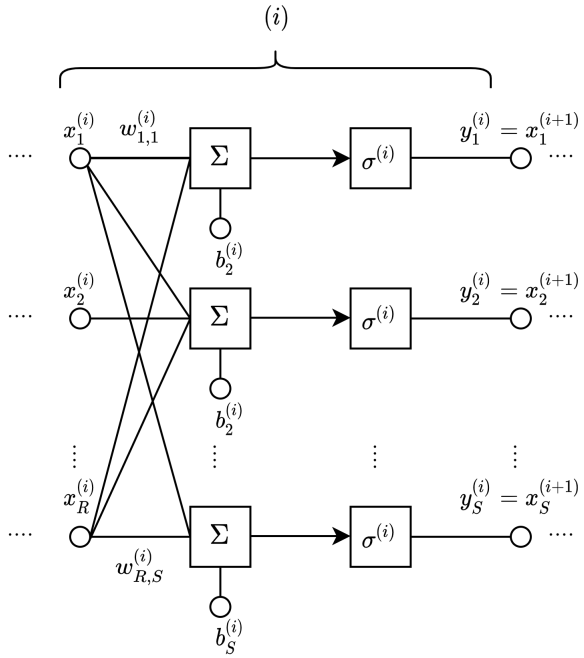


Figure F.1: Generic ANN layer.

Note: for the following formulas, the generic layer index will be dropped for ease of reading.

For a given neural network structure, optimal weights and biases are obtained during the training phase, in which they are adjusted until the loss function L between historical data and neural network forecast is minimized. The loss function is typically the mean squared error (MSE) on the training dataset $(\tilde{\boldsymbol{x}}, \tilde{\boldsymbol{y}})$:

$$L = MSE_{data} = \frac{1}{N_{data}} \sum_{i=1}^{N_{data}} [\mathcal{F}^{NN}(\tilde{x}_i, \theta) - \tilde{y}_i]^2 \tag{F.8}$$

where $(\tilde{\boldsymbol{x}}, \tilde{\boldsymbol{y}}) = ([\tilde{x}_1, ..., \tilde{x}_{N_{data}}]^T, [\tilde{y}_1, ..., \tilde{y}_{N_{data}}]^T)$.

F.1. Physics informed neural networks

Physics informed neural networks (PINNs) integrate the information embedded in the physical laws that model the system into the loss function during the training phase

of the network. Our understanding of fluid flow is usually formulated through partial differential equations (PDEs), constraints on their parameters, and boundary conditions on the unknown function. As such, PINNs' loss function contains not only the residual between observed and forecast data, but also additional regularization terms, in the form of properly weighted penalties, which limit the space of admissible solutions. PINNs honor those physical relationships in their forecast, resulting in more accurate learning and generalization power. This is especially important to avoid ANNs overfitting and non-physically supported results. PINNs can be employed both in the forward problem (i.e. to solve PDEs) and in the inverse problem (i.e. to find their parameters).

F.1.1. Forward and inverse problem

As described in Raissi et al. (2017), any parametrized non-linear PDE can be modeled in the general form:

$$\mathcal{N}[u;\lambda] = 0 \quad x \in \Omega \subset \mathbb{R}^d,\ t \in [0,T] \tag{F.9}$$

where $u = u(t,x)$ is the solution of the PDE and is a function of x and t in the $(d+1)$-dimensional space $\Omega \times [0,T]$. $\mathcal{N}$ is a generic non-linear operator parametrized by λ, which represents the model parameters. Within $\mathcal{N}$ are encapsulated specific forms of typical PDE problems, such as conservation laws, diffusion processes, kinetic equations or advection-diffusion-reaction systems, with the respective derivatives, expressed through subscripts, in the Ω space and time. Since PDEs in physical systems are associated with time and space derivatives, they are often reported explicitly.

The forward problem (also called data-driven solution of the PDE) entails finding the value of the function $u(x,t)$, given a set of training data and known λ model parameters, such that:

$$\mathcal{N}[u] = 0 \quad x \in \Omega \subset \mathbb{R}^d,\ t \in [0,T] \tag{F.10}$$

where (F.9) is no longer function of λ, which is known.

The inverse problem (also called data-driven discovery of the PDE) entails finding the value of the function $u(x,t)$ and λ model parameters, given a set of training data, such that:

$$\mathcal{N}[u,\lambda] = 0 \quad x \in \Omega \subset \mathbb{R}^d,\ t \in [0,T] \tag{F.11}$$

In traditional ANNs, $u(x,t)$ can be approximated with a neural network as seen in (F.6), with the loss function defined as in (F.8). Explicitly, (F.8) becomes:

$$MSE_{data} = \frac{1}{N_{data}} \sum_{i=1}^{N_{data}} [\mathcal{F}^{NN}(\theta, t_i, x_i) - u(t_i, x_i)]^2 \tag{F.12}$$

where t_i, x_i and $u(t_i, x_i)$ is the training data set, as in (F.8).

To integrate into the standard ANN the physical laws in (F.9), let us define f as the left side of (F.10) or (F.11), depending on the problem of interest:

$$f = 0 \tag{F.13}$$

f can be combined with the network approximating $u(x, t)$, resulting in a PINN. In particular, the physical laws in (F.9) can be enforced by introducing a loss on its residual, measured at sparse locations in the domain:

$$MSE_f = \frac{1}{N_{data}} \sum_{i=1}^{N_{data}} [f(t_i, x_i)]^2 \tag{F.14}$$

Another loss can be added to enforce N_u boundary and initial conditions for (F.9):

$$MSE_u = \frac{1}{N_u} \sum_{i=1}^{N_u} [\mathcal{F}^{NN}(\theta, t_i^u, x_i^u) - u(t_i^u, x_i^u)]^2 \tag{F.15}$$

where t_i^u, x_i^u and $u(t_i^u, x_i^u)$ are known boundary and initial conditions for the PDE. The total loss function is defined by a linear combination of the above loss functions:

$$L_{tot} = L_{data} + \lambda_f L_f + \lambda_u L_u = MSE_{data} + \lambda_f MSE_f + \lambda_u MSE_u \tag{F.16}$$

where λ values are proper weights.

F.2. Recurrent neural networks

Recurrent neural networks (RNNs) are specifically designed to predict dynamic behavior problems, in which data is a time sequence. In fact, the network's prediction at a specific time step is based not only on its input at the same time step, but also on the previous state(s) of the network, which are stored temporarily. RNNs differ from feed-forward neural networks, in which activations flow only in one direction from input to output. The word "cell" is often used in the context of neural networks to refer to a part of a network

that preserves some form of state across time steps: in a way, it is a form of memory. A single recurrent neuron can be thought of as a cell. A schematic representation of a RNN cell is reported in Figure F.2, along with the common representation with "unraveled" time sequences.

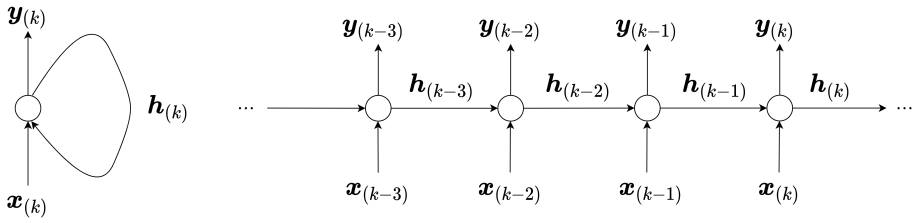


Figure F.2: RNN cell and time unraveling.

The state of a cell at the generic time step k is denoted as $\boldsymbol{h}_{(k)}$ and is a function of both inputs at the same time step $\boldsymbol{x}_{(k)}$ and the state at the previous time step $\boldsymbol{h}_{(k-1)}$. It is clear how in RNNs, the state of the cell is not directly equal to its output, differently from standard feed-forward networks, as in (F.6).

As such, each neuron has a set of weights for inputs and a set of weights for previous states: considering all neurons, these can be packed into two matrices, $\boldsymbol{W}_{hx}$ and $\boldsymbol{W}_{hh}$, respectively. As always, a bias $\boldsymbol{b}_h$ is present. Hence, the state of a single RNN cell can be described by:

$$\boldsymbol{h}_{(k-1)} = \sigma_1(\boldsymbol{W}_{hh}^T \boldsymbol{h}_{(k-1)} + \boldsymbol{W}_{hx}^T \boldsymbol{x}_{(k)} + \boldsymbol{b}_h) \tag{F.17}$$

and its output, using again a proper weight matrix $\boldsymbol{W}_{yh}$ and bias $\boldsymbol{b}_y$, by:

$$\boldsymbol{y}_{(k)} = \sigma_2(\boldsymbol{W}_{yh} \boldsymbol{h}_{(k)} + \boldsymbol{b}_y) \tag{F.18}$$

where σ_1 and σ_2 are the two activation functions. RNNs are prone to the problem of vanishing (or disappearing) gradient and short-term memory problem [39], where, due to the transformation that data goes through in the RNN, some information is lost at each time step, leaving virtually no trace of the first inputs after a while.

Long short-term memory (LSTM) cells, first proposed by Hochreiter and Schmidhuber (1997) [26] and gradually improved over the years by several researchers, have been proven successful at tackling the short-term memory issue of traditional RNNs, showing better overall performance, faster training and the ability to detect long-term patterns in data. Differently from traditional RNN cells, the state vector is split into two parts: $\boldsymbol{h}_{(k)}$, representing the short-term state, and $\boldsymbol{c}_{(k)}$, representing the long-term state. In other

words, they are able to recognize important inputs, store them, preserve them as long as needed (until they are forgotten), and extract them when needed. LSTM networks are used in the Olympus field case study in this thesis.

G | Correlated profiles

The covariance matrix $\boldsymbol{C}$ expresses the degree of statistical dependence for a random variable. It is expressed mathematically as:

$$\boldsymbol{C} = \mathbb{E}[(\boldsymbol{Z} - \bar{\boldsymbol{Z}})(\boldsymbol{Z} - \bar{\boldsymbol{Z}})^T] \tag{G.1}$$

where $\mathbb{E}$ is the expected value operator and $\bar{\boldsymbol{Z}}$ is the vector of average values for the variable. For a time-correlated random variable $\boldsymbol{Z} = \boldsymbol{Z}(t)$, assuming a time discretization $t_k = k\Delta t$ for $k = 0, ..., N_t$, then $\boldsymbol{Z} = \boldsymbol{Z}(t_k) = [Z(t_1), ..., Z(t_{N_t})]$. Hence, $\boldsymbol{C}$ is a square $N_t \times N_t$ matrix and its generic element c_{kl} is:

$$c_{kl} = \mathbb{E}[(Z(t_k) - \bar{Z}(t_k))(Z(t_l) - \bar{Z}(t_l))] \tag{G.2}$$

where for $k = l$, (G.2) reduces to:

$$c_{kk} = \sigma_k^2 \tag{G.3}$$

The variogram (or semi-variogram) γ is defined as the mean quadratic increment of the time-correlated quantity $\boldsymbol{Z} = \boldsymbol{Z}(t_k)$ between two points separated by a distance h. Mathematically:

$$\gamma(h) = \frac{1}{2}\mathbb{E}[(\boldsymbol{Z}(t_k) - \boldsymbol{Z}(t_k + h))^2] \tag{G.4}$$

where:

- σ_k is the standard deviation of quantity $\boldsymbol{Z}$ at time t_k, also known as sill;
- a is the range of the variogram, i.e. the distance at which the variogram reaches 95% of its sill value;
- h is the distance between two points in time, known as lag; for a discretized time system, $h = (k-1)\Delta t$ for $k = 1, ..., N_t$, since the value for zero lag corresponds to σ_k.

Assuming second-order stationarity [36], the covariance matrix is only a function of the

lag h (in time) between two points. As such, (G.2) reduces to:

$$c_{kl} = \sigma_k^2 - \gamma(h) \tag{G.5}$$

where $h = t_k - t_l$. For a cubic variogram with range a, G.4 is expressed as:

$$\gamma(h) = \begin{cases} \sigma_k^2 \left[7 \left(\frac{h}{a}\right)^2 - \frac{35}{4} \left(\frac{h}{a}\right)^3 + \frac{7}{2} \left(\frac{h}{a}\right)^5 - \frac{3}{4} \left(\frac{h}{a}\right)^7 \right] & \text{for } h < a \\ \sigma_k^2 \quad \text{for } h > a \end{cases} \tag{G.6}$$

In this problem, the quantity to correlate in time is the injection profile of a single well i, called $\boldsymbol{u}_i$ (with the same meaning as in (2.2)). The following parameters are specified:

- number of elements of the vector N_t (i.e. time steps for each injection well);
- number of dimensions along which to correlate n_{corr} (i.e. time only);
- correlation length or range a;
- type of correlation (also known as variogram model γ, it expresses the mathematical expression of the correlation: a cubic variogram is used in this work);
- upper and lower bounds for each injection well $q_{in,i}^{min}, q_{in,i}^{max}$;
- average values $\bar{u}_i$ and standard deviation σ_i for each injection well.

Once these parameters are specified, the covariance matrix $\boldsymbol{C}_i$ is constructed through (G.5) and (G.6) (more details can be found in [36]), then its lower triangular $\boldsymbol{L}_i$ is extracted, through Cholesky decomposition, such that:

$$\boldsymbol{C}_i = \boldsymbol{L}_i (\boldsymbol{L}_i^T)' \tag{G.7}$$

Uncorrelated profiles $\boldsymbol{u}_i$ can be generated starting from a Gaussian distribution with zero mean and unitary variance. For every injection well, time-correlated profiles are obtained by multiplying this matrix by the random profile $\boldsymbol{u_i}$, then centering it around the average value $\bar{\boldsymbol{u}}_i$, which is in general time dependent but not in this case:

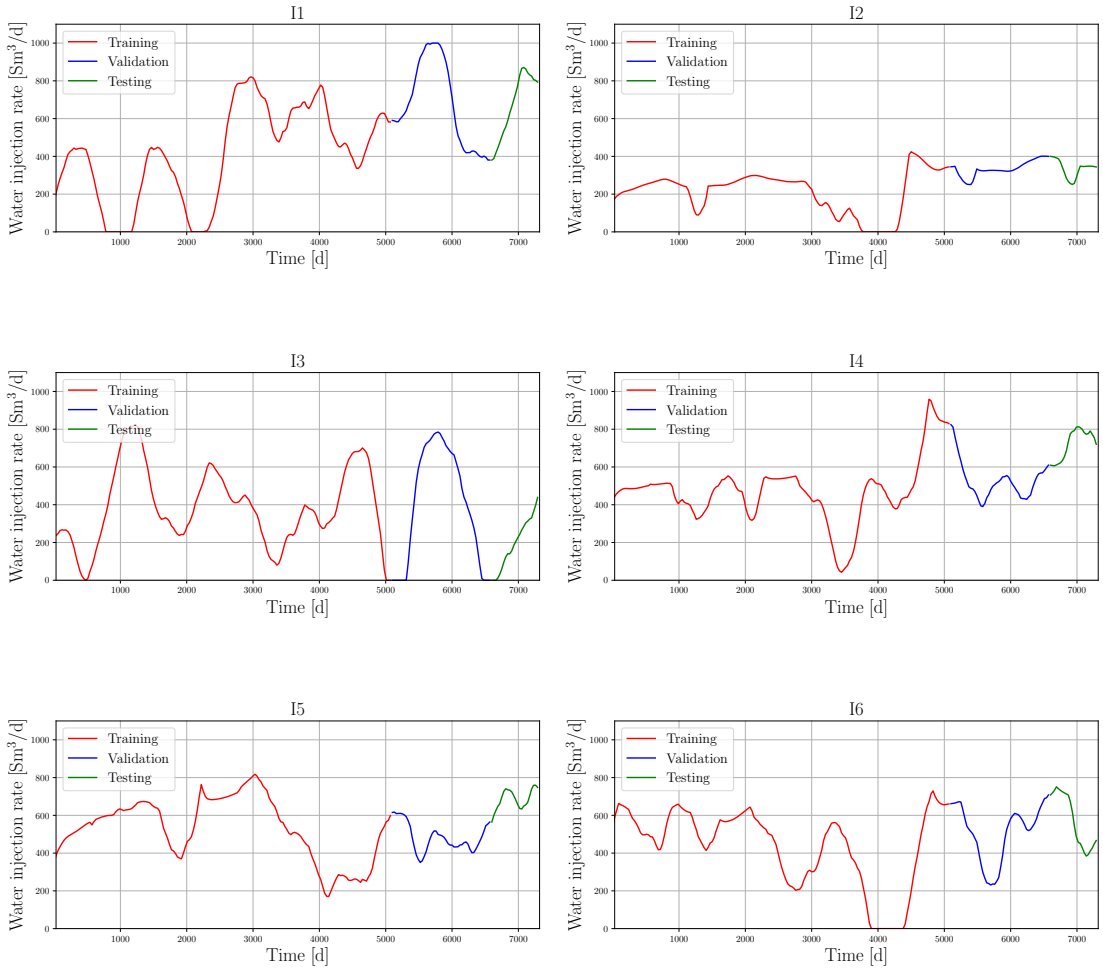
$$\boldsymbol{u}_{corr,i} = \boldsymbol{L}_i \boldsymbol{u}_i + \bar{\boldsymbol{u}}_i \tag{G.8}$$

Finally, excessively low and high values are replaced by their respective minimum and

maximum bounds.

H | Olympus field datasets

Figures H.1 and H.2 show the used input and output datasets for the LSTM surrogate models of the Olympus field.



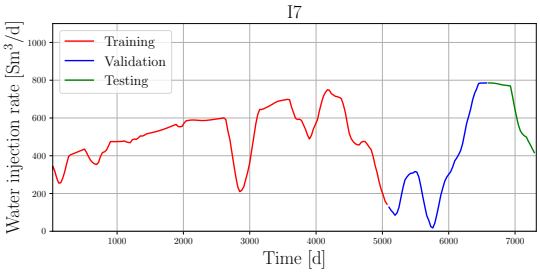
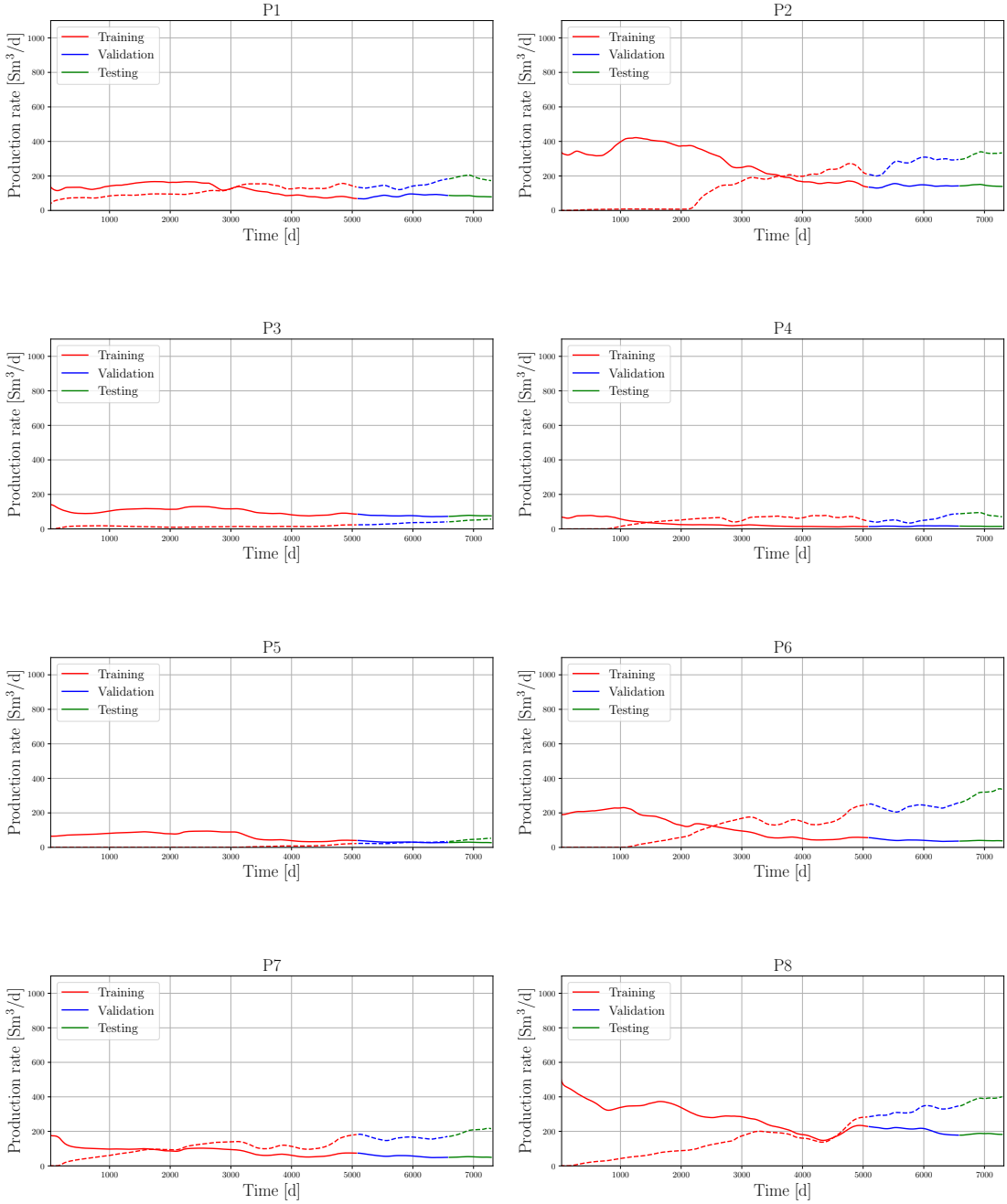


Figure H.1: Olympus: input water injection rate profiles of the generated dataset.



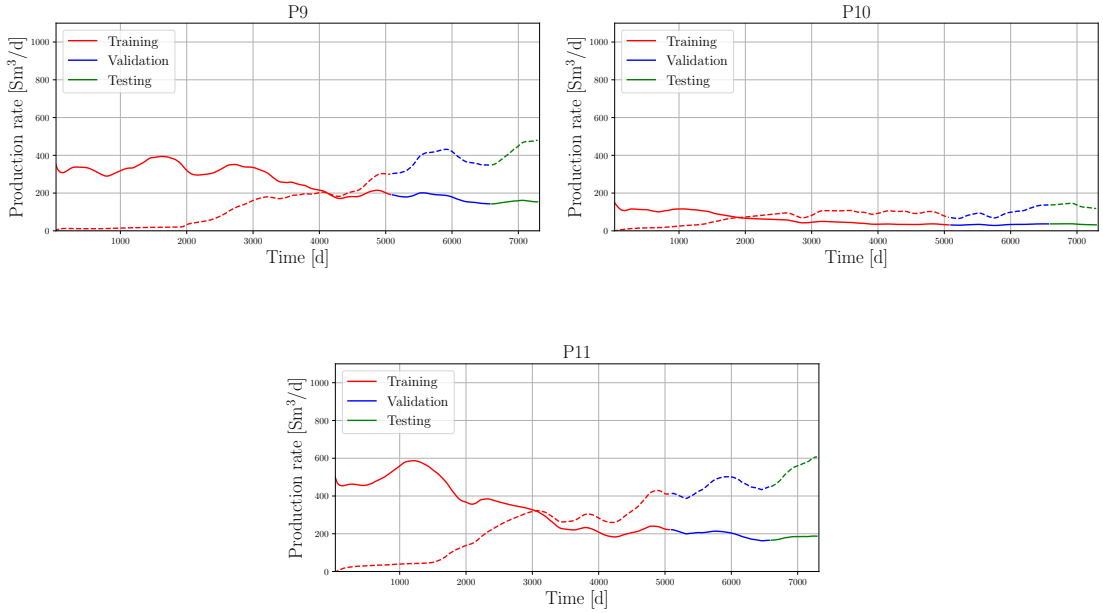


Figure H.2: Olympus: oil (solid) and water (dash) production rate profiles of the generated dataset.

For details on how the dataset is generated, see Appendix G.

Bibliography

[1] BP. World Energy Outlook, 2022. `https://www.bp.com/en/global/corporate/energy-economics/energy-outlook.html`.

[2] M. J. Blunt. *Multiphase Flow in Permeable Media: A Pore-Scale Perspective*. Cambridge University Press, 2017.

[3] D. Brouwer and J.D. Jansen. Dynamic optimization of waterflooding with smart wells using optimal control theory. *SPE Journal*, 9:391–402, 10 2004.

[4] L. Novelli et al. *Hydrocarbons: origin, exploration and production*. Eni Corporate University, 2005.

[5] G. Nævdal et al. Waterflooding using closed-loop control. *Computational Geosciences*, 10:37–60, 01 2006.

[6] Y. Nasir et al. Hybrid derivative-free technique and effective machine learning surrogate for nonlinear constrained well placement and production optimization. *Journal of Petroleum Science and Engineering*, 186:106726, 2020.

[7] E. Aliyev and L. Durlofsky. Multilevel field development optimization under uncertainty using a sequence of upscaled models. *Mathematical Geosciences*, 49, 07 2016.

[8] A. Y. Abukhamsin. *Optimization of well design and location in a real field*. Master's thesis, Stanford University, 2009.

[9] S. Krogstad et al. Reservoir management optimization using calibrated transmissibility upscaling. In *14th European Conference on the Mathematics of Oil Recovery 2014, ECMOR 2014*, volume 2014, pages 1–11. European Association of Geoscientists & Engineers, 2014.

[10] M. Cardoso and L. Durlofsky. Linearized reduced-order models for subsurface flow simulation. *Journal of Computational Physics*, 229(3):681–700, 2010.

[11] S. Trehan and L. Durlofsky. Trajectory piecewise quadratic reduced-order model

for subsurface flow, with application to pde-constrained optimization. *Journal of Computational Physics*, 326:446–473, 2016.

[12] J.D. Jansen and L. Durlofsky. Use of reduced-order models in well control optimization. *Optimization and Engineering*, 18, 03 2017.

[13] Y.D. Kim and L. Durlofsky. A Recurrent Neural Network–Based Proxy Model for Well-Control Optimization with Nonlinear Output Constraints. *SPE Journal*, 26(04):1837–1857, 08 2021.

[14] R. Holanda et al. A state-of-the-art literature review on capacitance resistance models for reservoir characterization and performance forecasting. *Energies*, 11(12), 2018.

[15] D. Koroteev and Z. Tekic. Artificial intelligence in oil and gas upstream: Trends, challenges, and scenarios for the future. *Energy and AI*, 3:100041, 2021.

[16] H. Rahmanifard and T. Plaksina. Application of artificial intelligence techniques in the petroleum industry: a review. *Artificial Intelligence Review*, 52, 12 2019.

[17] B. Negash and D. Atta. Artificial neural network based production forecasting for a hydrocarbon reservoir under water injection. *Petroleum Exploration and Development*, 47(2):383–392, 2020.

[18] J. Bruyelle and D. Guérillot. Neural networks and their derivatives for history matching and reservoir optimization problems. *Computational Geosciences*, 18:549–561, 08 2014.

[19] N.V. Queipo et al. Surrogate modeling-based optimization of sagd processes. *Journal of Petroleum Science and Engineering*, 35(1):83–93, 2002.

[20] G. Zangl et al. Proxy Modeling in Production Optimization. volume All Days of *SPE Europec featured at EAGE Conference and Exhibition*, 06 2006. SPE-100131-MS.

[21] Golzari A. et al. Development of an adaptive surrogate model for production optimization. *Journal of Petroleum Science and Engineering*, 133:677–688, 2015.

[22] D. Salam et al. Production optimization strategy using hybrid genetic algorithm. 11 2015.

[23] A.F. Teixeira and A.R. Secchi. Machine learning models to support reservoir production optimization. *IFAC-PapersOnLine*, 52(1):498–501, 2019. 12th IFAC Symposium on Dynamics and Control of Process Systems, including Biosystems DYCOPS 2019.

[24] J. Bruyelle and D. Guérillot. Proxy Model Based on Artificial Intelligence Technique

for History Matching - Application to Brugge Field. volume Day 2 Tue, October 22, 2019 of *SPE Gas*, 10 2019. D021S010R004.

[25] Zhi Chai et al. An integrated closed-loop solution to assisted history matching and field optimization with machine learning techniques. *Journal of Petroleum Science and Engineering*, page 108204, 2020.

[26] S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural computation*, 9:1735–80, 12 1997.

[27] L. Deng and Y. Pan. Machine-Learning-Assisted Closed-Loop Reservoir Management Using Echo State Network for Mature Fields under Waterflood. *SPE Reservoir Evaluation & Engineering*, 23(04):1298–1313, 11 2020.

[28] Chuan Tian and R.N. Horne. Recurrent Neural Networks for Permanent Downhole Gauge Data Analysis. volume Day 1 Mon, October 09, 2017 of *SPE Annual Technical Conference and Exhibition*, 10 2017. D011S008R007.

[29] A.M. Tartakovsky et al. Physics-informed deep neural networks for learning parameters and constitutive relationships in subsurface flow problems. *Water Resources Research*, 56:e2019WR026731, 05 2020.

[30] C.G. Fraces and H. Tchelepi. Physics Informed Deep Learning for Flow and Transport in Porous Media. volume Day 1 Tue, October 26, 2021 of *SPE Reservoir Simulation Conference*, 10 2021. D011S006R002.

[31] A. Yewgat et al. Deep-crm: A new deep learning approach for capacitance resistive models. In *ECMORXVII: 17 th European Conference On The Mathematics Of Oil Recovery*, Online, France, September 2020.

[32] M. Maniglio et al. Physics Informed Neural Networks Based on a Capacitance Resistance Model for Reservoirs Under Water Flooding Conditions. In *Abu Dhabi International Petroleum Exhibition and Conference*, 11 2021. D021S027R001.

[33] Yan Chen et al. Efficient ensemble-based closed-loop production optimization. *Spe Journal*, 14:634–645, 2009.

[34] Jian Hou et al. A review of closed-loop reservoir management. *Petroleum Science*, 12:114–128, 2015.

[35] S.E. Buckley and M.C. Leverett. Mechanism of Fluid Displacement in Sands. *Transactions of the AIME*, 146(01):107–116, 12 1942.

[36] J.P. Chilès. *Geostatistics: Modeling Spatial Uncertainty.* Wiley, 2012.

[37] J. Bear. *Modeling Groundwater Flow and Contaminant Transport*. Springer, 2010.

[38] A. Albertoni and L. Lake. Inferring interwell connectivity only from well-rate fluctuations in waterfloods. *SPE Reservoir Evaluation & Engineering - SPE RESERV EVAL ENG*, 6:6–16, 02 2003.

[39] A. Géron. *Hands-on machine learning with Scikit-Learn and TensorFlow : concepts, tools, and techniques to build intelligent systems*. O'Reilly Media, 2019.

[40] Marucha L. et al. On the implementation of an algorithm for large-scale equality constrained optimization. *SIAM Journal on Optimization*, 8(3):682–706, August 1998.

[41] R.H. Byrd et al. An interior point algorithm for large-scale nonlinear programming. *SIAM Journal on Optimization*, 9(4):877–900, September 1999.

[42] R.M. Solgi. Genetic algorithm for Python, 2020. `https://pypi.org/project/geneticalgorithm/`.

[43] C. Temizel et al. Production Optimization through Voidage Replacement using Triggers for Production Rate. volume Day 3 Thu, December 08, 2016 of *SPE International Heavy Oil Conference and Exhibition*, 12 2016. D031S013R004.

[44] Schlumberger. *Eclipse Technical Descrpition*, 1st edition, 2017.

[45] Streamsim Technologies Inc. Streamsim technologies website. `https://www.streamsim.com/`.

[46] A. Yousef et al. A capacitance model to infer interwell connectivity from production and injection rate fluctuations. *SPE Reservoir Evaluation & Engineering - SPE RESERV EVAL ENG*, 9:630–646, 12 2006.

[47] M. Sayarpour et al. The use of capacitance–resistance models for rapid estimation of waterflood performance and optimization. *Journal of Petroleum Science and Engineering*, 69(3):227–238, 2009.

[48] M. J. Blunt. *Imperial College Lectures In Petroleum Engineering, Reservoir Engineering*. World Scientific Publishing Europe Ltd, 2017.

[49] R.M. Fonseca et al. Description of OLYMPUS reservoir model for optimization challenge, 2017. `https://www.isapp2.com/downloads/olympus-reservoir-model.pdf`.

[50] Schlumberger. Schlumberger Oilfield Glossary. `https://glossary.oilfield.slb.com/`.

[51] L. P. Dake. *Fundamentals of Reservoir Engineering: Volume 8*. Elsevier Science, 1983.

[52] M. Muskat and M. W. Meres. The Flow of Heterogeneous Fluids Through Porous Media. *Physics*, 7(9):346–363, September 1936.

[53] J. Onwunalu and L. Durlofsky. Application of a particle swarm optimization algorithm for determining optimum well location and type. *Computational Geosciences*, 14:183–198, 05 2010.

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

[55] L. Lake et al. Optimization Of Oil Production Based On A Capacitance Model Of Production And Injection Rates. volume All Days of *SPE Hydrocarbon Economics and Evaluation Symposium*, 04 2007. SPE-107713-MS.

[56] A. Yousef. *Investigating Statistical Techniques To Infer Interwell Connectivity From Production And Injection Rate Fluctuations*. PhD thesis, University of Texas at Austin, 2006.

[57] M. Sayarpour. *Development and Application of Capacitance-Resistive Models to Water/CO2 Floods*. PhD thesis, University of Texas at Austin, 2008.

[58] D. Weber et al. Improvements in Capacitance-Resistive Modeling and Optimization of Large Scale Reservoirs. volume All Days of *SPE Western Regional Meeting*, 03 2009. SPE-121299-MS.

[59] H. R. Jahangiri et al. A Data-Driven Approach Enhances Conventional Reservoir Surveillance Methods for Waterflood Performance Management in the North Sea. volume All Days of *SPE Intelligent Energy International Conference and Exhibition*, 04 2014. SPE-167849-MS.

[60] C. Temizel et al. Data-Driven Optimization of Injection/Production in Waterflood Operations. volume All Days of *SPE Middle East Intelligent Oil and Gas Symposium*, 05 2017. SPE-187468-MS.

[61] J. H. Holland. *Adaptation in Natural and Artificial Systems An Introductory Analysis with Applications to Biology, Control, and Artificial Intelligence*. Bradford Books, 1992.

[62] R. M. Fonseca et al. Improving the Ensemble Optimization Method Through Co-

variance Matrix Adaptation (CMA-EnOpt). volume All Days of *SPE Reservoir Simulation Conference*, 02 2013. SPE-163657-MS.

[63] M. Thiele and R. Batycky. Water injection optimization using a streamline-based workflow. *Proceedings - SPE Annual Technical Conference and Exhibition*, 10 2003.

[64] Holstein E.D. et al. *Petroleum Engineering Handbook, Vol. 5 - Reservoir Engineering and Petrophysics*. Society of Petroleum Engineers, 2007.

[65] D.W. Pollock. Semianalytical computation of path lines for finite-difference models. *Ground Water*, 26:743–750, 1988.

[66] M. Thiele and R. Batycky. Using streamline-derived injection efficiencies for improved waterflood management. *SPE Reservoir Evaluation & Engineering - SPE*, 9:187–196, 04 2006.

[67] W. Tailai et al. Waterflood management using two-stage optimization with streamline simulation. *Computational Geosciences*, 18:1–22, 08 2014.

[68] Han-Young Park and A. Datta-Gupta. Reservoir management using streamline-based flood efficiency maps and application to rate optimization. *Journal of Petroleum Science and Engineering*, 109:312–326, 2013.

[69] M. R. Thiele. Streamline Simulation. 8th International Forum on Reservoir Simulation, 06 2005.

List of Figures

List of Tables

List of Symbols

Due to the complex combination of field and SI units, which is peculiar to oil and gas engineering, both the dimension and the typical units for each symbol are shown. The units reported below are not the only possibility and reservoir engineering often uses a combination of different systems. When the unit is variable (e.g. for errors or control vectors) no unit is reported.

Latin symbols

Variable	Description	Dimension	Typical unit
a	Correlation length	-	-
A	Surface area	L^2	m^2
A_e	Effective surface area	L^2	m^2
$\boldsymbol{b}$	Bias vector	-	-
B_g	Gas formation volume factor	-	$\mathrm{rm}^3/\mathrm{Sm}^3$
BHP	Bottom-hole pressure	$\mathrm{ML}^{-1}\mathrm{T}^{-2}$	bar
B_o	Oil formation volume factor	-	$\mathrm{rm}^3/\mathrm{Sm}^3$
B_w	Water formation volume factor	-	$\mathrm{rm}^3/\mathrm{Sm}^3$
$\boldsymbol{c}$	Constraints vector	-	-
$\boldsymbol{C}$	Correlation matrix	-	-
c_t	Rock compressibility	$\mathrm{M}^{-1}\mathrm{LT}^2$	Pa^{-1}
CWI	Cumulative water injection	L^3	stb or Sm^3
d	Discount factor	-	-
D	Datum	L	m
$\mathbb{E}$	Expected value	-	-
E_d	Local displacement efficiency	-	-
E_s	Sweep efficiency	-	-

f	Generic function	-	-
f_{ij}	Connectivity	-	-
f_j	Fractional flow of phase j	-	-
$\mathcal{F}^{NN}$	Neural network operator	-	
g	Gravitational acceleration constant	LT^{-2}	$\mathrm{m/s^2}$
$\boldsymbol{g}$	Reservoir simulator equations vector	-	-
G	Maximum number of generations	-	-
h	Distance for variogram	-	-
$\boldsymbol{h}$	State of neural network cell	-	-
H	Net pay zone thickness	L	m
IE	Injector efficiency	-	-
J	Objective function	-	-
$\boldsymbol{K}$	Absolute permeability	L^2	mD or m^2
k_r	Relative permeability	-	-
L	Generic length	L	m
	Loss function	-	-
$\boldsymbol{L}$	Lower triangular matrix	-	-
m	Quadratic approximation of function	-	-
MSE	Mean-squared error	-	-
N	Number of candidates	-	-
	Initial oil in place	L^3	stb
$\mathcal{N}$	Generic PDE operator	-	-
n_{corr}	Number of correlation variables	-	-
N_{data}	Number of elements in dataset	-	-
N_e	Number of members in ensemble	-	-
N_G	Gravity number	-	-
N_{in}	Number of injection wells	-	-
N_p	Number of production wells	-	-
N_P	Oil produced	L^3	stb or Sm^3
N_{PD}	Dimensionless oil produced	-	-
N_{ph}	Number of phases	-	-
NPV	Net present value	-	\$

$NRMSE$	Normalized root mean-squared error	-	-
N_s	Number of streamlines	-	-
N_t	Number of time steps	-	-
NTG	Net-to-gross	-	-
N_z	Number of vertical cells	-	-
$\bar{p}$	Average reservoir pressure	$\mathrm{ML^{-1}T^{-2}}$	bar
$\boldsymbol{\mathcal{P}}$	Probability	-	-
P	Number of individuals in population	-	-
$\boldsymbol{P}$	GA population vector	-	-
P_c	Capillary pressure	$\mathrm{ML^{-1}T^{-2}}$	bar
p_h	Hydrostatic pressure	$\mathrm{ML^{-1}T^{-2}}$	bar
PI	Productivity index	$\mathrm{M^{-1}L^4T}$	stb/(d·psi)
P_o	Pressure of oil	$\mathrm{ML^{-1}T^{-2}}$	bar
$PORV$	Volume of grid cell	$\mathrm{L^3}$	$\mathrm{m^3}$ or $\mathrm{ft^3}$
P_w	Pressure of water	$\mathrm{ML^{-1}T^{-2}}$	bar
p_{wf}	Bottom hole flowing pressure	$\mathrm{ML^{-1}T^{-2}}$	bar
p_{wh}	Wellhead pressure	$\mathrm{ML^{-1}T^{-2}}$	bar
$\boldsymbol{q}$	Generic reservoir output vector	-	-
$\bar{q}$	Range of output dataset	-	-
$\hat{\boldsymbol{q}}$	Neural network approx. of output	-	-
$\tilde{\boldsymbol{q}}$	Generic output from dataset	-	-
Q	Generic flow rate	$\mathrm{L^3T^{-1}}$	stb/d or $\mathrm{Sm^3/d}$
$q_{in,i}$	Water injection rate	$\mathrm{L^3T^{-1}}$	stb/d or $\mathrm{Sm^3/d}$
q_j	Darcy velocity of phase j	$\mathrm{LT^{-1}}$	m/s
q_o	Oil Darcy velocity	$\mathrm{LT^{-1}}$	m/s
$q_{o,j}$	Oil production rate	$\mathrm{L^3T^{-1}}$	stb/d or $\mathrm{Sm^3/d}$
q_t	Total Darcy velocity	$\mathrm{LT^{-1}}$	m/s
q_w	Water Darcy velocity	$\mathrm{LT^{-1}}$	m/s
$q_{w,j}$	Water production rate	$\mathrm{L^3T^{-1}}$	stb/d or $\mathrm{Sm^3/d}$
r_e	Drainage area radius	L	m
$ReLU$	Rectified linear unit function	-	-
R_f	Recovery factor	-	-

r_{in}	Cost of water injection	-	\$/stb
r_o	Price of oil	-	\$/stb
R_s	Gas-oil solubility ratio	-	-
r_w	Cost of water production	-	\$/stb
r_{wb}	Wellbore radius	L	m
s	Generic streamline	-	-
S_j	Saturation of phase j	-	-
S_k	Skin factor	-	-
S_o	Oil saturation	-	-
S_{or}	Residual oil saturation	-	-
S_w	Water saturation	-	-
S_{wi}	Initial water saturation	-	-
t	Time	T	day (d)
T	Time domain	T	day (d)
t_D	Dimensionless time	-	-
t_k	Generic moment in time	T	day (d)
$\boldsymbol{u}$	Control variable vector	-	-
$\tilde{\boldsymbol{u}}$	Generic input from dataset	-	-
U	Allowable values for $\boldsymbol{u}$	-	-
v	Generic velocity	LT^{-1}	m/s
V_{mob}	Mobile oil volume	L^3	stb or Sm^3
V_p	Pore volume of reservoir	L^3	m^3 of ft^3
VRR	Voidage replacement ratio	-	-
v_{sh}	Shock wave speed	LT^{-1}	m/s
v_{shD}	Dimensionless shock wave speed	-	-
$\boldsymbol{W}$	Weight matrix	-	-
WAF	Well allocation factor	-	-
WC	Water cut	-	-
w_i	Weight parameter	-	-
WOR	Water-oil ratio	-	-
$\boldsymbol{x}$	State variable vector	-	-
	Neural network input	-	-

x_D	Dimensionless space	-	-
$\boldsymbol{y}$	Neural network output	-	-
$\boldsymbol{Z}$	Random variable	-	-

Greek symbols

Variable	Description	Dimension	Typical unit
α	Step length	-	-
γ	Variogram model	-	-
Δp	Pressure drop	$\mathrm{ML^{-1}T^{-2}}$	bar
Δt	Time step	T	day (d)
θ	Neural network parameters	-	-
	Generic angle	-	rad
λ	PDE parameter	-	-
λ_i	Lagrange multiplier	-	-
λ_j	Mobility of phase j	$\mathrm{M^{-1}LT}$	1/(Pa·s)
μ	Viscosity	$\mathrm{ML^{-1}T^{-1}}$	Pa·s
ξ	Streamline coordinate	L	m
ρ	Density	$\mathrm{ML^{-3}}$	$\mathrm{kg/m^3}$
σ	Activation function	-	-
	Standard deviation	-	-
τ	Time of flight	T	day (d)
τ_j	Time constant	T	day (d)
ϕ	Porosity	-	-

Acknowledgements

This thesis is the result of a 6-month internship in Eni S.p.A's Reservoir Department, and it would not have been possible without their generous support. In particular, my sincere thanks go to my company supervisors, Dr. Giorgio Fighera and Dr. Emanuele Vignati, for their guidance throughout this project. Their insights, suggestions and encouragement made the past year a period of both academic and personal growth.

I would also like to express my gratitude to Professor Enrico Zio and Dr. Ahmed Shokry, for their invaluable support throughout the project. Their assistance and patience was critical to the development, as well as the writing, of this thesis.