UNIVERSITY OF CALGARY

Steam injection distribution optimization in SAGD oil field using reinforcement learning and web-based GIS

by

Changlin Yang

A THESIS

SUBMITTED TO THE FACULTY OF GRADUATE STUDIES

IN PARTIAL FULFILMENT OF THE REQUIREMENTS FOR THE

DEGREE OF MASTER OF SCIENCE

GRADUATE PROGRAM IN GEOMATICS ENGINEERING

CALGARY, ALBERTA

APRIL, 2021

Abstract

Steam injection distribution optimization refers to the process of distributing certain amount of high temperature steam in steam assisted gravity drainage (SAGD) oil field to maximize the total oil production. In this thesis, two novel optimization methods are presented to solve the steam injection distribution optimization problem. The first optimization method is dynamic programming (DP) method, and the second optimization method is Q-learning method. In the two proposed methods, long short-term memory (LSTM) neural network is used to construct the prediction model to predict oil production of the wells. A web-based geographical information system (GIS) called Petroleum Explorer is developed to support the two proposed methods. Comparison experiments have been conducted using the real-world SAGD production data to test the performance of the proposed methods and the influence of parameter settings on the optimization result. The experiments demonstrate that LSTM model gives better prediction result than other five existing models and both optimization methods can improve the oil production of the oil field. The result also shows the performance of Q-learning method is better than the DP method.

Acknowledgement

Frist, I would like to express my deepest gratitude to my supervisor, Dr. Xin Wang, for accepting me into the research group and for the ongoing support and encouragement during the master program.

I would also like to thank my excellent colleagues and friends. Thank you, my brothers Ge Cui and Yuanchen Li for the selfless help when I first came here. Thanks to Rodrigo Sliver, Boting Qu, Zhenyi Tang, Fang Gong, Helen Pinto, Zheng Guo and Mohammadreza Rahimi for cheering me up when I was faced with difficult and for the great discussion when we worked together. Thank you, my dear friends Tian Jin, Cheng Huang and Eric Wang for all the precious time we spent together.

Finally, I would like to thank my parents, who always be there for me when I need help. Thank you all for the care and love no matter where I was.

Table of Contents

List of Tables

List of Figures

List of Symbols and Abbreviations

Symbols

I	Well steam injection data
O	Well oil production data
P	Total oil production volume
N	Total number of wells in the oil field
M	Total volume of steam in the oil field
sop_t	Steam-oil pair at *t*-th time interval
X_t	Time window of the well at *t*-th time interval
T	Timestep
s_t	State at *t*-th optimization step
a_t	Action at *t*-th optimization step
$r_t(s_t, a_t, s_{t+1})$	Instant reward at *t*-th optimization step
R_t	Iong-term reward at *t*-th optimization step
π	Policy
$V(s_t)$	Value function at state s_t
$Q(s_t, a_t)$	Q function at state s_t with action a_t
γ	Discount factor
α	Learning rate
ε	Epsilon

Abbreviations

SAGD	Steam assisted gravity drainage
LSTM	Long short-term memory
DP	Dynamic programming
GA	Genetic algorithm
RMSE	Root mean square error
MAE	Mean absolute error
PCC	Pearson correlation coefficient
GIS	Geographic information system
ARIMA	Autoregressive integrated moving average
ANN	Artificial neural network
ELM	Extreme learning machine
RNN	Recurrent neural network
ENN	Elman neural network
MDP	Markov decision process
UWI	unique well identifier
SOR	Steam-oil ratio
DQN	Deep Q-Network

Chapter 1

Introduction

1.1. Background

Oil sands of Alberta, Canada contains the third largest oil reserves in the world. The proven reserves equal about 165.4 billion barrels [1]. In 2017, the crude bitumen production of Alberta was about 2.8 million barrels per day. Capital investment in the upstream energy sector of Alberta was estimated at 26.5 billion dollar. Approximately eighty percent of the oil sands in Alberta are recovered through in-situ production method. In-situ method refers to oil production methods making use of steam and drilling techniques, and steam assisted gravity drainage (SAGD) is one of the most commonly used in-situ production methods in Alberta [2]. Efficient operation of the SAGD oil projects plays a vital part in improving the bitumen production and economic situation of Alberta. Therefore, properly optimizing the production process of SAGD oil field and improving the oil production is extremely important and it is highly appealing to people from oil and gas industry and scientific field.

During the SAGD process, a pair of parallel wells is the most important component. The upper well works as a steam injector and the lower well works as a producer which picks up reservoir water and heated oil [3]. When continuously injecting steam from the injector well, the oil heats up and diffuses into the surroundings, therefore forms a steam chamber. Temperature of the steam chamber gradually becomes equal to the temperature of injected steam. The cold oil receives the heat from injected steam when the steam condenses at the interface of the oil. Consequently, oil heated by high temperature steam is recovered by gravity through the producer well [4]. The process of SAGD is shown in Figure 1.1.

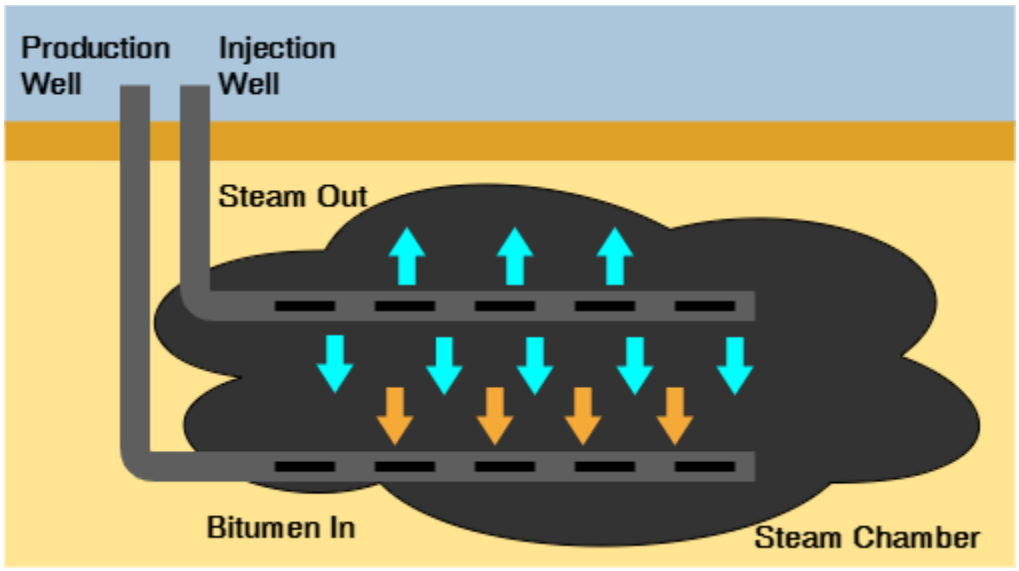


Figure 1.1: Steam assisted gravity drainage

Oil production prediction of the SAGD wells is a critical issue from the management perspective. The study has shown that prediction of the future oil production volume is highly related to steam injection and historical oil production volume of the well [5]. The steam injection and oil production data of

each well are non-linear time series data, and the oil production is widely believed to depend highly on its recent operations in terms of many factors, such as steam injection volume, steam injection temperature, pressure as well as many other geological properties [6]. Among them, steam injection volume is the most important factor [7].

Existing time series prediction methods have great potential in oil production prediction problems, such as autoregressive integrated moving average (ARIMA) [8], artificial neural network (ANN) [9], extreme learning machine (ELM) [10], recurrent neural network (RNN) [11], and Elman neural network (ENN) [12]. However, ARIMA model is not able to process multiple input variables in time series prediction [13]. Traditional ANN model and ELM model cannot take account of the temporal relationships of steam injection and oil production data [13]. Vanilla RNN model and ENN model have the gradient vanishing problem that memories from previous states could vanish during the training process [14]. Therefore, effective oil production prediction model should be implemented in the study to handle the temporal dependency in non-linear SAGD time series data and accurately predict oil production of the wells.

Careful planning on steam injection distribution is another critical task to improve the energy efficiency and production performance of the oil field. In SAGD, the steam is continuously injected through the injector to maintain the temperature of the steam chamber [15]. If the injection volume is too low, the oil recovery may not be maximized. On the other hand, excessive steam injection is very costly with significant greenhouse gas emissions. In addition, the steam

injection and oil extraction may lead to deformation on well casings and surface, which could cause serious environmental problems [16]. Furthermore, the total volume of steam generated per day is restricted due to the capacity limitation of the steam generation facilities.

Recently numerous studies have been conducted in oil production optimization problems using evolutionary algorithms such as non-dominated sorting genetic algorithm II [17] and particle swarm optimization algorithm [18]. However, to our best knowledge, currently no solutions are proposed for the steam injection distribution problem in SAGD oil field. Therefore, novel optimization methods should be proposed to solve the steam injection distribution problem to improve the oil production performance of the SAGD oil field.

Due to the complex field operation requirements, it is difficult for engineers to perform the proposed optimization methods without a proper software tool. Current desktop oil and gas data management system such as GeoCarta [19] and Accumap [20] are able to storage, visualize and analyze considerable amount of production data but they are not capable of offering customized steam injection optimization method for the oil and gas industry. Furthermore, a desktop software which only operates on PC platform is not convenient for engineers working in the field with mobile devices. Therefore, effective software system should be developed to deliver the customized optimization method on mobile devices.

1.2. Objectives

Based on the challenges listed above, the steam injection distribution optimization problem of this thesis is summarized as follow. Given a SAGD oil field with multiple wells, the main idea of the steam injection distribution optimization problem is to allocate certain amount of high temperature steam among the wells to maximize the total oil production of the oil field. During the optimization process, the total volume of injecting steam is limited by the capacity of steam generation facilities and steam injection volume of each well should meet certain ranges determined by the production history of the well.

The objective of this thesis is to propose efficient optimization method to solve the steam injection distribution optimization problem. A software system should also be developed for engineers to perform the proposed methods. To achieve this goal, accurate oil production prediction method based on historical steam injection and oil production data needs to be fulfilled. The specific objectives are listed below:

1. To accurately predict the future oil production of SAGD wells using state-of-the-art time series prediction model, aiming to provide solid prediction result for the steam injection distribution optimization process;
2. To propose efficient steam injection distribution optimization method that allocate the high temperature steam among the SAGD oil field and improve the overall amount of oil production;

3. To develop a web-based geographic information system (GIS) to work on diverse platforms and support the proposed steam injection distribution optimization methods.

1.3. Contributions

In this thesis, two optimization methods – dynamic programming (DP) method and Q-learning method are proposed to solve the steam injection distribution optimization problem in SAGD oil field. The DP optimization method is introduced as the first attempt to solve the optimization problem. The Q-learning optimization method is the second method and it achieves better optimization result than the DP method. Both two optimization methods are implemented together with genetic algorithm (GA) and long short-term memory (LSTM) prediction model to produce reliable prediction result for the optimization process. To provide a software system for engineers, a web-based GIS called Petroleum Explorer is developed to support the two proposed steam injection distribution optimization methods. The main contributions of this thesis are summarized as follows:

1. LSTM neural network models are proposed to predict the oil production of each SAGD well with its historical data. GA is used to identify the proper timestep for the input of LSTM model, then LSTM neural network models are built to capture the temporal dependency in time series data. ANN, ELM, ENN, EEMD-LSTM and EEL-ELM models are

implemented in the comparison experiment of this thesis. The result of the experiment shows that LSTM prediction model gives better prediction results than other five comparison prediction models;

2. DP is applied to optimize the steam injection distribution in SAGD oil field as the first attempt to solve this problem. First, the steam injection distribution optimization problem is properly formalized under DP scenario. With the result of LSTM prediction model, DP efficiently optimizes the steam injection distribution in SAGD oil field to improve the overall oil production of the oil field. A case study is conducted in the thesis to test the effect of the parameter setting on optimization process;

3. Q-learning is implemented in this thesis as the second optimization method. First, the problem is formalized again under Q-learning scenario. Monthly collected data and daily collected data are both applied in the experiments to test the performance of the Q-learning method. The result shows that the Q-learning method achieves better optimization performance than the DP method. Comparison experiments also carried out in the thesis to analyze the influence of parameter setting on the Q-learning optimization result;

4. A web-based GIS – Petroleum Explorer is developed to provide platform for engineers working in the oil field to implement and perform the two proposed steam injection distribution optimization methods. Case studies are conducted on two pads of real world SAGD project. The result shows

that the overall performance of the Q-learning method is better than DP method.

1.4. Thesis outline

The remaining part of this thesis is organized as follows.

In Chapter 2, related work of this thesis is introduced. First, the recent researches of time series prediction are listed. Then, the discussion about single-objective and multi-objective oil production optimization methods are given. After that, the motivation of using DP to solve the steam injection distribution optimization problem is discussed. At last, the information and researches about Q-learning is shown in the related work.

In Chapter 3, the structure of the proposed DP method is described. First, the steam injection distribution optimization problem is defined under DP scenario. Then, workflow of the DP method is discussed in detail. After that, experiments have been conducted to test the performance of different prediction models. Case studies have been implemented to show the performance of the DP method and the influence of parameter setting on the method.

In Chapter 4, the detail information about the Q-learning optimization method is discussed. First, the steam injection distribution optimization is defined under Q-learning scenario. Then, the workflow of the Q-learning method is introduced. Finally, comparison experiments are implemented to test the performance of the Q-learning method using monthly collected data and daily collected data. The

effect of parameter setting on the optimization result is also analyzed in the experiments.

In Chapter 5, Petroleum Explorer is introduced. The system architecture and database structure is shown to demonstrate the implementation of the system. And then, the workflows of dynamic programming function and Q-learning function are described.

Finally, the discussion and conclusion of this thesis are given in Chapter 6 of the thesis.

Chapter 2

Related works

Before introducing the proposed steam injection distribution optimization methods, some related works are listed below. First, the oil production and steam injection data in this thesis are time series data, and one objective of this thesis is to predict the oil production with time series data. Therefore, some recent time series prediction methods are presented in Section 2.1. Oil production optimization is an active research area. In Section 2.2, the studies on single-objective and multi-objective oil production optimization methods are discussed. In Section 2.3, the motivation of the proposed DP method is introduced. Finally, in section 2.4, the knowledge the Q-learning method is demonstrated.

2.1 Time series prediction

Accurately predicting oil production time series data is critical for oilfield management, oil production planning and economic cost control [7]. Different methods have been proposed to solve time series prediction problems.

ARIMA is a mathematical model that exploits historical input to predict future trends of the time series data. Yusof *et al.* apply ARIMA method to analyze monthly oil production data of Malaysia from January 2005 to May 2010 [26]. The result shows that the forecast predictability of oil production for three following months have been improved significantly. The method achieves reasonable prediction performance with simple model structure but does not capture the temporal dependency lies in time series data.

Scientific studies have also been published in the field of applying neural network models in time series prediction problems. A neural network model is constructed based on a collection of connected units called artificial neurons and the connections between the neurons will be trained based on the input data. Berneti and Shahbazian present a novel technique to forecast oil flow rate in wells of Persian Gulf oil fields with the combination of ANN and imperialist competitive algorithm [27]. The result demonstrates the effectiveness of the proposed method compared with traditional ANN model. As a variant ANN model, ELM is well known for the fast prediction speed and simple model structure. The parameters of hidden modes in ELM can be randomly assigned and the output weights of the model can be obtained in one calculation. Zhang *et al.* propose an ELM based prediction model to forecast wind speed of wind farms in China [10]. The experiment results demonstrate that the proposed model is able to predict time series wind speed data and achieve satisfactory prediction performance. Although ANN models capture the non-linear relations in historical

oil production data, they cannot consider the temporal dependency in time series data, which is important for the accuracy of predicting future oil production.

RNN is a commonly used model to solve time series prediction problems. It is a special architecture of ANN in which the connections between nodes form a directed graph along a time series [28]. It adopts internal memory to process the input sequence, which allows it to display temporal dependency in time series data. As a simple structure RNN, ENN has been implemented in time series prediction problem. An ENN is a type of neural network with the special context layer. At each time step, information from the hidden layer is saved in the context layer, which becomes memory of the neural network and works for time series prediction problem. Liu *et al.* propose an ENN method for multistep wind speed prediction problem [12]. Results show that the ENN model combined with secondary decomposition algorithm gives high-accuracy prediction result. LSTM neural network is a special kind of RNN which captures the long-term temporal dependency in time series data. It was put forward by Hochreiter and Schmidhuber in 1997 [21]. A common LSTM unit is made up of an input gate, an output gate and a forget gate [29]. The three gates control the information pass through the unit. With the capacity of learning long term dependency, LSTM has also been applied on time series prediction problems [22].

Most recently, Liu *et al.* proposed an ensemble empirical model decomposition based Long Short-Term Memory neural network model called EEMD-LSTM for oil production prediction problem [30]. The proposed model is applied to two oilfields in China and the results of experiment demonstrate the

effectiveness of the proposed model in oil production prediction. However, the proposed model cannot be directly applied to SAGD oil production prediction. First, only one time series, i.e., historical oil production data, is used as input data of the proposed EEMD-LSTM model. The oil production of SAGD wells depends on not only the historical oil production, but also the operational data, such as steam injection volume. Therefore, the input of prediction model should include both oil production and steam injection time-series data. Second, only one LSTM prediction model is constructed in EEMD-LSTM to predict the oil production of the whole block of the oil field. However, in order to optimize the steam injection of SAGD wells, LSTM prediction models need to be constructed for individual well in SAGD oil field to predict its oil production. With the prediction result of the multiple LSTM models, the optimization algorithm is then able to allocate the steam injection among different wells to maximize the oil production. Third, the proper setting of the timesteps of LSTM can reflect the temporal dependency among the input time series data, which is important for SAGD oil production prediction. However, in Liu *et al.* 2020, authors do not discuss how to determine the timesteps for the EEMD-LSTM models.

Some integration methods are also proposed to combine different prediction models and achieve superior prediction performance. For example, Chen *et al.* propose a two-layer integration wind speed prediction model called EEL-ELM [31]. The first layer of the EEL-ELM model consists of ENN, LSTM and ELM prediction model, and the second layer is made up of an ELM based nonlinear aggregated mechanism to aggregate the results from the first layer together. The

result shows that the EEL-ELM model produces better prediction results among the eight prediction methods. Hu *et al.* propose a wind speed prediction method with the combination of differential evolution algorithm, ELM and LSTM [32]. The results demonstrate that the proposed method gives better prediction result in statistic tests. However, the complex model structure needs more time for model training.

2.2 oil production optimization methods

Researches have been conducted in different oil production optimization problems with single-objective and multi-objective optimization methods [38]. Some studies solve the optimization problem using single-objective optimization methods which only optimizes oil production of the oil field to achieve the optimal optimization result. For example, Ray *et al.* propose a evolutionary algorithm approach for the gas lift optimization problem using non-dominated sorting genetic algorithm II [17]. Gas lift is an oil production technique that uses an external source of high-pressure gas to lift the well fluids in the oil field [18]. Results show the significant oil production improvement over the existing practical data. Hamedi *et al.* propose a particle swarm optimization technique for the gas lift allocation problem [18]. The proposed method is implemented in the real world oil field project and the result indicates the method improves the total oil production rate of the oil field. Namdar develops a water cycle optimization approach to solve the gas lift optimization problem [33]. The idea of water cycle optimization algorithm comes from water cycle in nature [34]. The comparison

experiment shows the proposed approach is able to speed up the optimization process.

Table 2.1: Summarization of the recent related optimization methods

Reviewed Works	Objective	Methodology	Variables	Result and improvement
Ray *et al.* [17]	To maximize the total daily gas lift production (Single-objective)	Non-dominated sorting genetic algorithm II	1. Gas used in each well 2. Relationship between well and well segment	1. Produce 35 barrels per day more for the 6 well problem 2. Produce 243 barrels per day more for the 56 well problem
Hamedi *et al.* [18]	To assign an optimum gas injection rate for each individual well (Single-objective)	Particle swarm optimization	1. Number of particles 2. Inertia weight	Produce 605 stock tank barrel oil per day more than the current produced oil
Namdar [33]	To increase the speed and accuracy of solving gas allocation optimization problem (Single-objective)	Water cycle optimization algorithm	1. Injected gas flow rate 2. Produced oil flow rate	1. Converge to an optimal point in 0.56 s with a population of 50 3. More efficient in lower oil production targets
Ameli *et al.*[35]	To increase the speed of Fast-SAGD oil recovery optimization process (Multi-objective)	Repetition inhibitory algorithm	1. Cumulative solvent-oil ratio 2. Recovery factor	1. Optimization speed 6.33 times faster 2. Termination of optimization algorithm 9.67 times faster
Ma *et al.* [36]	To improve oil production and reduce solvent usage for warm solvent injection operations (Multi-objective)	Non-dominated sorting genetic algorithm II	1. Cumulative solvent-oil ratio 2. Recovery factor	1. In good agreement with the actual simulate targets for all 50 cases. 2. Overestimate cumulative solvent-to-oil ratio for 1 / (recovery factor) between 1.8 to 2.0 3. Speed up the optimization process

There are also studies focus on multi-objective optimization methods which optimizes the combination of oil production and solvent-oil-ratio to reach the best oil production performance. For example. Ameli *et al.* propose a repetition inhibitory algorithm method to increase the optimization speed of Fast-SAGD optimization process [35]. Fast SAGD refers to oil production technique which

makes use of offset wells with cyclic steam stimulation. The proposed technique is tested in multiple Fast-SAGD studies. Result shows considerable improvement in the speed of the optimization process. Ma *et al.* design a Pareto-based multiple-objective optimization workflow using non-dominated sorting genetic algorithm II for warm solvent injection process [36]. In warm solvent injection, heated chemical solvent is injected into the reservoir to produce the diluted bitumen. In the hybrid workflow, ANN and response surface methodology are used to build the proxy model. Result shows the proposed method increases the optimization speed of the warm solvent injection process. To summarize aforementioned optimization studies, detailed information of the studies is shown in Table 2.1.

2.3 Dynamic programming

All studies discussed in Section 2.2 focus on different optimization problems to improve the oil production in the field. To our best knowledge, the steam injection distribution optimization problem in SAGD oil field is still a new topic for SAGD oil production research. In steam injection distribution optimization problem, total steam injection volume of the field is restricted due to the capacity limitation of the steam generation facilities. The optimal solution for the problem is to determine the steam injection volume for each well so that the total oil production is maximized. Given the generated amount of high temperature steam, when the total oil production reaches the maximum value, the steam-oil ratio (SOR) correspondingly reaches the optimal value. Therefore, single-objective

optimization technique is used in the thesis to optimize the total oil production of the oil field.

DP is an algorithm used to solve a complex problem by breaking it down into a series of sub-problems, solving each sub-problem just once, and then a memory-based data structure is used to store all the solutions. It is mainly utilized when the same sub-problem needs to be solved repeatedly. Therefore, solved solutions of the sub-problems are stored in the data structure to avoid recomputing of the same problem. There are four key steps in DP method. First, the sub-problems under DP scenario should be properly defined. Then the relationship between the sub-problems should be recursively characterized. Next, the calculation order of the sub-problems should be confirmed. Finally, the optimal solution is generated based on the calculation order.

Recently, DP has been applied to various optimization problems to reach the optimal solution [39]. Kalaycı *et al.* propose a DP based optimization approach for appointment scheduling in banking system [23]. This approach aims to balance the number of customers each branch serves within a particular region and minimize waiting times of customers. Experiment shows the proposed method solves the optimization problem and maximizes the added value for bank at branches. Zhu *et al.* develop an optimization model to determine dynamic frequencies and timetables of buses based on dynamic programming to maximize the profits of bus carriers and passengers [25]. To study the optimization problem in the real world environment, several simulation models are implemented in the thesis to describe the characteristic of the bus scheduling system, such as the real-

time traffic flow speed model, the dynamic passengers bus taking model and the dynamic bus scheduling model. Case study is presented in a major Chinese city bus operation, the result demonstrates the efficiency of the proposed model compared with previous research.

Compared with alternative approaches such as genetic algorithm and particle swarm optimization algorithm, DP method is able to generate the global optimal solution and achieves better optimization result [40]. Second, it is well suited for multi-stage or sequential decision process [24]. Third, since the DP method shows the evolution of the dynamic process, it is possible to use practical knowledge to improve the optimization efficiency. Therefore, DP is used in the thesis to solve the steam injection distribution optimization problem.

However, in each step of DP, only the instant reward of the process will be considered by the algorithm. In steam injection distribution optimization problem, the instant reward refers to the instant oil production increase of the optimized wells by taking each optimization action. That is to say, the instant reward only represents the reward of current optimization step. However, the optimization of the problem is a continuous process, during the optimization process, the reward of the future optimization actions will also influence the optimization of the current step. Therefore, long-term reward of the problem, which counts on the influence of future optimization actions at each step is important for the oil field to achieve better optimization performance [52]. In order to generate stable global optimal solution in consideration of the long-term reward of each optimization

action, novel optimization method should be introduced to properly solve the steam injection distribution optimization problem.

2.4 Q-learning

Reinforcement learning is a type of machine learning algorithm which allows software agent taking actions to interact with the environment and to maximize the cumulative reward during the leaning process [49]. In reinforcement learning, the environment is typically stated in the form of a Markov decision process (MDP) [53]. A MDP is a discrete-time stochastic control process. It provides a mathematical framework for modeling decision making in situations where outcomes are partly random and partly under the control of an agent. Therefore, MDP is useful for studying optimization problems solved via reinforcement learning [54].

An overview of the MDP is shown in Figure 2.1. At tth step of a MDP episode, the process is in a state s_t, and the agent may choose any action a_t that is available in state s_t. The process responses at the next step by moving into a new state s_{t+1} and giving the agent a corresponding reward r_t. The next state s_{t+1} depends on the current state s_t and the action from the agent a_t. One MDP episode will end when the process reaches the termination condition.

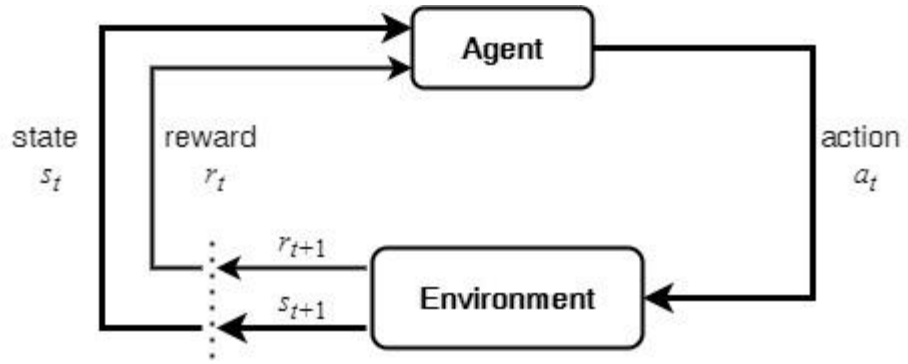


Figure 2.1: An overview of Markov decision process (cited from [62])

Q-learning is a model-free reinforcement learning algorithm that aims to find the best action to take given the outside environment [46]. During the Q-learning process, the Q-value is used to measure the quality of actions under different circumstances and the Q-value is updated when receiving the reward of each action [50]. When training the Q-learning model with certain number of MDP episodes, the Q-value of the actions under each state will be stored and updated in a table-based data structure called Q-table. The goal of updating the Q-table is to find an ideal policy for the agent: a function π that specifies the action $\pi(s_t)$ that the agent will choose when in state s_t. After completing the Q-learning process, the algorithm will generates the policy π by choosing the action with the highest Q-value in the Q-table under each state.

Q-learning is widely used in the optimization problems that focus on maximizing the expected value of total reward over a series of decision making process [47]. For example, Liu *et al.* propose a Q-learning based energy management strategy for a hybrid tracked vehicle to adapt to different driving conditions [57]. An onboard Markov Chain model is employed to simulate the

power demand of different route segment and the Q-learning based method is applied in the Markov Chain model to accelerate the convergence rate of the optimization process. Comparison results of two driving cycles show that the Q-learning energy management strategy can greatly improve efficiency of the fuel compared to DP method and other reinforcement learning methods. Yu *et al.* develop a Q-learning based algorithm to optimize the energy consumption of air conditioners and exhaust fans when balancing the indoor air quality and thermal comfort in the classroom [58]. The transition of indoor air quality and thermal comfort in the classroom are described in the Markov decision process. In order to analyse the performance of the optimization algorithm, the effects of outdoor environments and class conditions on the optimization result are examined in details. The experiment shows when compared with a fixed temperature of 26 °C, the proposed method can offer about 15% lower energy consumption on average and the concentration of carbon dioxide can be reduced by 13% in average. However, even though Q-learning method has been used to solve optimization problems in different field, the implementation of the Q-learning in oil production optimization problem still needs exploration.

Chapter 3

Dynamic programming optimization method

In this chapter, the dynamic programming optimization method is described to solve the steam injection distribution optimization problem. The structure of the chapter is organized as follow. In Section 3.1, the steam injection distribution optimization problem is formalized under the dynamic programming scenario. In Section 3.2, the workflow of the proposed dynamic programming optimization method is discussed in detail. Finally, in section 3.3, case studies are implemented in the thesis to test the performance of the proposed LSTM prediction model and dynamic programming optimization method.

3.1 Problem statement

In this thesis, two types of data, i.e., the steam injection data and oil production data of each well are used in the SAGD oil field, which are defined as follows.

Definition 1: Well Steam Injection Data. Given a SAGD well, the steam injection data of the well is a sequence of steam injection values for some time intervals, denoted as:

$$I = \{i_1, i_2, \dots, i_t, \dots, i_{T_i}\} \quad (1)$$

where T_i is total number of time intervals, i_t is the steam injection volume of the well at the *t-th* time interval. The time interval can be with different lengths, such as hour, day or month.

Definition 2: Well Oil Production Data. Given a SAGD well, the oil production data of the well is a sequence of oil production values for some time intervals, denoted as:

$$O = \{o_1, o_2, \dots, o_t, \dots, o_{T_o}\} \quad (2)$$

where T_o is total number of time intervals, o_t is the oil production volume of the well at the *t-th* time interval.

Definition 3: Total Oil Production Volume. Given a set of *N* wells in the oil field sharing the same steam generation facility, the total oil production at the *t-th* time interval of the *N* wells $P_N(M)$ is the oil production summation of the *N* wells, denoted as:

$$P_N(M) = \sum_{k=1}^{N} o_t^k \quad (3)$$

where o_t^k is the oil production volume of the *k-th* well at the *t-th* time interval, *M* is the total amount of steam generated by the facility at the *t-th* time interval.

Definition 4: Dynamic Programming Steam Injection Distribution Optimization Problem. At the *t-th* time interval, given the total volume of steam is M m^3, the dynamic programming steam injection distribution optimization problem is to determine the steam injection for each well so that the total oil production at the *t-th* time interval is maximized, with respect to the steam injection volume of the well should less than M m^3 and steam injection volume of individual well should meet minimum and maximum constraints, denoted as:

$$argmax\ P_N(M), \qquad wrt. \sum_{k=1}^{N} i_t^k \leq M\ , \qquad \theta_1^k * i_{t-1}^k \leq i_t^k \leq \theta_2^k * i_{t-1}^k \qquad (4)$$

Where $P_N(M)$ is the total volume of oil production in the oil field, N is the number of wells in the oil field, o_t^k and i_t^k are the oil production volume and steam injection volume of the *k-th* well at the *t-th* time interval. θ_1^k and θ_2^k are value factors of the *k-th* well which set up the minimum and maximum constraints of individual well. The steam injection volume at *t-th* time interval i_t^k should be within a certain range and depends on the steam injection volume of previous time interval i_{t-1}^k and this is based on the general practice in the field.

The DP steam injection optimization problem becomes distributing the injecting steam among different wells to reach the maximum oil production of the oil field given M m^3 steam and N wells. However, considering the relationship between o_t^k and $i_t^k$is unknown, the future oil production for each well should be predicted based on the historical steam injection and oil production volume. LSTM neural network is used in this thesis to construct the oil prediction model

to consider the influence of historical steam injection and oil production value when predicting the future oil production. The input data of the model is defined as follows.

Definition 5: Steam-Oil Pair. Given a SAGD well, a steam-oil pair at the *t-th* time interval sop_t is a combination of steam injection at the *t-th* time interval i_t and oil production volume at the *t*-1-*th* time interval o_{t-1} denoted as:

$$sop_t = \{i_t, o_{t-1}\}, \qquad t > 1 \tag{5}$$

To represent the temporal dependency among the data, the number of steam-oil pairs should be considered by the prediction model. Therefore, two concepts, time window and timestep, are defined.

Definition 6: Time Window. Given a SAGD well, the time window X_t of the well is defined as total steam-oil pairs considered as input data of the prediction model when predicting the oil production volume at the *t-th* time interval.

$$X_t = \{sop_i \mid f(t, w^l) \leq i \leq f(t, w^h)\} \tag{6}$$

Where w^l and w^h are two integers determining the number of constituent steam-oil pairs included in the time window of the well at the *t-th* time interval, and function *f* is given below where *w* is the representation of w^l or w^h.

$$f(t, w) = \begin{cases} w & if\ w > 0 \\ t + w & otherwise \end{cases} \tag{7}$$

Definition 7: Timestep. Given the time window of a SAGD well at the *t-th* time interval, the timestep *T* is defined as the number of steam-oil pairs in the time window.

$$T = w^h - w^l + 1 \tag{8}$$

The definition of w^l and w^h significantly increases the flexibility in specifying time windows. If w^l and w^h are assigned with positive integers, the earliest and latest steam-oil pair of the time window will be fixed. In the other case, if w^l and w^h are assigned with non-positive integers, the earliest and latest steam-oil pair of the time window will be changing as the time interval *t* changes.

For each well in the oil field, a corresponding prediction model with selected time step is constructed to predict short-term oil production volume. Considering future short-term oil production volume is most relevant to recent historical data, when timestep for the model is set as five, time window for the model at the *t-th* time interval will include five most recent *sop* pairs as input data. w^l and w^h are the earliest and latest steam-oil pair of the time window, i.e., -4 and 0, as shown in Figure 3.1. In this way, time window is directly related to timestep and it is convenient to compare prediction result of models constructed with different timestep values.

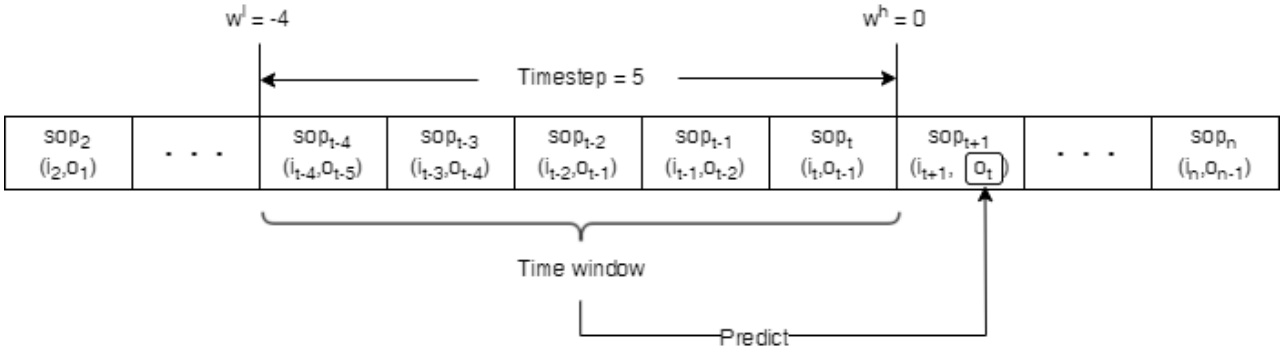


Figure 3.1: An example of time window and timestep when timestep is five

The DP optimization method includes four steps: data preprocessing, determining timesteps using GA, building LSTM models for oil production prediction and optimizing steam injection using DP. First, the raw oil production and steam injection data is preprocessed for the prediction model. The timesteps for different wells to achieve the best prediction performance could be different because the historical steam injection and oil production records vary for each well. Therefore, timestep for prediction model is determined for each well in the oil field using GA in the second step. After determining timesteps using GA, LSTM prediction model is construct to predict the short-term oil production. Finally, with the oil production prediction result, DP is used to optimize steam injection in the oil field. The overview of the method is shown in Figure 3.2. Based on the DP optimization method, the web-based GIS - Petroleum Explorer is developed in the thesis. In the following, each step of the DP method and the development of Petroleum Explorer will be discussed in detail.

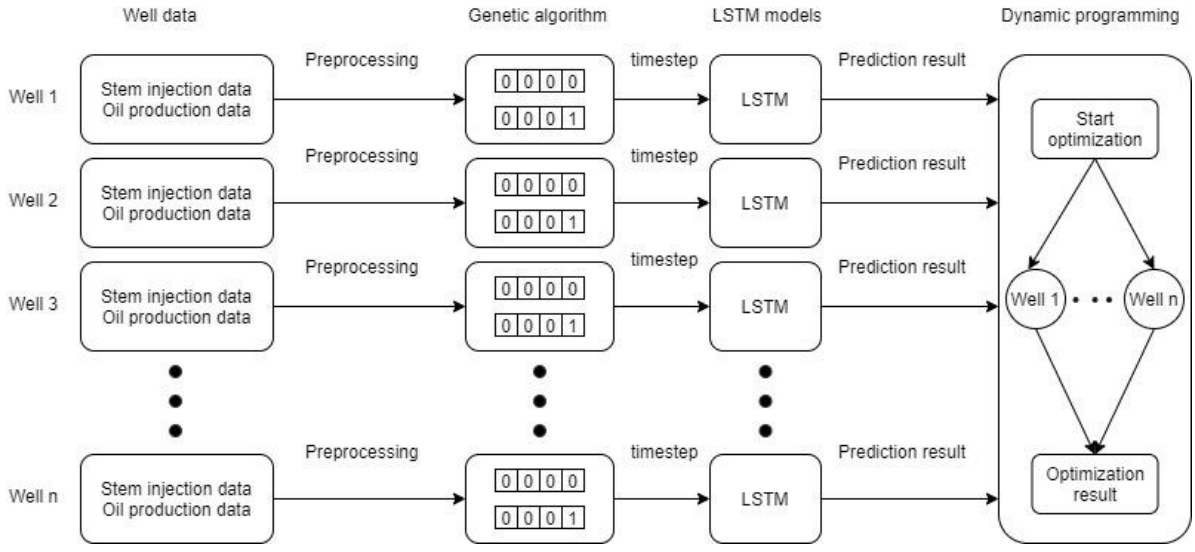


Figure 3.2: The overview of the dynamic programming optimization method

3.2 Workflow of the dynamic programming optimization method

3.2.1 Data Preprocessing

The data preprocessing stage consists of two steps. The first step of the data preprocessing method is to handle missing values in the raw data. In this thesis, the missing values of steam injection data and oil production data are filled with a global constant *Unknown* and the tuple are ignored when building the prediction model. After that, Z-Score normalization method is used to transform the oil production and steam injection data into a same scale.

3.2.2 Determining timesteps using genetic algorithm

Timestep for the prediction model of each well should be decided because proper length of timestep for the prediction model will keep the temporal dependency among the input data and improve the prediction accuracy [29]. Determining the best timestep for different wells is a challenging issue, since candidate timesteps could be numerous and trying all possible timesteps for each well could be extremely time consuming. GA is utilized to select the most appropriate timestep for each well because GA is equipped with selection, crossover and mutation operations which means it has more chance to find the global optimal solution than other evolutionary algorithms such as population extremal optimization[37, 38]. Figure 3.3 depicts the process to determine the timestep by using GA.

Input:
N: Nnumber of initial timesteps
I : Number of iterations

Output:
Selected timestep for the well T

Algorithm:
/ produce initial timesteps */*
(1) $k = 1$;
(2) *Generate initial timesteps* $X^k = \{x_1^k, x_2^k, ..., x_N^k\}$;
(3) *Calculate fitness*(x_i^k), *for* $i = 1, ..., N$;
(4) ***While*** $(k \leq I)$
/ produce new generation */*
(5) $k = k+1$;
(6) *Pick two timesteps* x_p *and* x_q *from* X^{k-1};
(7) *Produce two new timesteps* x_p^{new} *and* x_q^{new} *by crossover* x_p *and* x_q;
(8) *Peform mutation operation on* x_p^{new} *and* x_q^{new};
(9) *Calculate fitness*(x_p^{new}) *and fitness*(x_q^{new});
(10) *Insert* x_p^{new} *and* x_q^{new} *into* X^{new};
(11) *Pick N best timesteps from* X^{k-1} *and* X^{new} *to form* X^k;
(12) T = *the best timestep in* X^k;
(13) ***end While***
(14) *Return T*

Figure 3.3: Determining timesteps using genetic algorithm

For the initialization of GA, a certain number of candidate timesteps for the LSTM model of the well are randomly generated as shown in line (2). Next, models with candidate timesteps are evaluated according to Root-Mean-Square Error fitness function. Roulette wheel selection is performed afterwards to select timesteps from candidate timesteps for crossover and mutation as shown in line (6). Ordered crossover is then used to create child timesteps from two selected parent timesteps as shown in line (7). After that, shuffle mutation is used to select a consecutive subset of each child timestep and shuffle them randomly to create new timestep as shown in line (8). The generated new timesteps are added into

the candidate timesteps to produce new population. Models with new timesteps are evaluated again using Root-Mean-Square Error fitness function. The process is repeated for the defined number of iterations. At the end, a timestep with highest fitness score is selected as the best solution for the LSTM model of the given well.

3.2.3 Building LSTM models for oil prediction

In this thesis, The oil prediction model is constructed using LSTM neural network and the structure of a LSTM cell is shown in Figure 3.4.

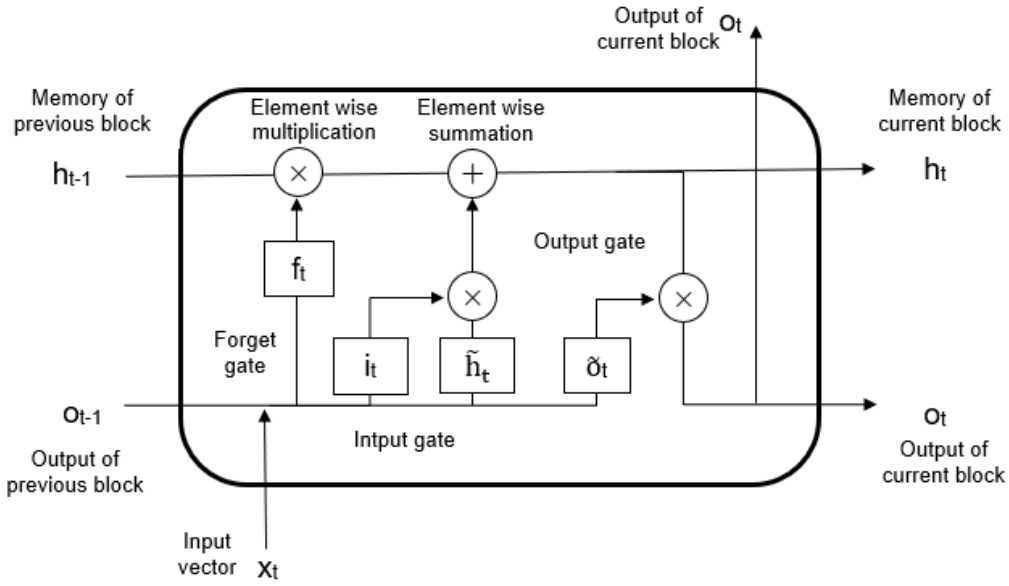


Figure 3.4: A cell of LSTM neural network

The most critical part of a LSTM is state of the cell, the straight line located in the upper section of the figure. The LSTM employs gates to remove or add information to state of the cell Gates are made up of a sigmoid layer and a pointwise multiplication which optionally let information through. Equations of the gates are listed below.

$$f_t = \sigma\big(W_f * [o_{t-1}, x_t] + b_f\big) \quad (9)$$

$$i_t = \sigma(W_i * [o_{t-1}, x_t] + b_i) \quad (10)$$

$$\tilde{h}_t = tanh(W_{\tilde{h}} * [o_{t-1}, x_t] + b_{\tilde{h}}) \quad (11)$$

$$h_t = f_t * h_{t-1} + i_t * \tilde{h}_t \quad (12)$$

$$\tilde{o}_t = \sigma(W_{\tilde{o}} * [o_{t-1}, x_t] + b_{\tilde{o}}) \quad (13)$$

$$o_t = \widetilde{o_t} * \tanh(h_t) \quad (14)$$

A cell of LSTM has a forget gate, an input gate and an output gate to control state of the cell. The forget gate is used to get rid of the previously stored value from historical state and the formula is shown in Equation (9) where W_f and b_f are weights and bias that govern behavior of the forget gate. The forget gate concatenate $[o_{t-1}, x_t]$ and multiply it by W_f. As shown in Equation (9),the forget gate produces a vector of numbers f_t with values between zero and one. The vector generated by forget gate will be multiplied by state from the previous cell h_{t-1} to decide which part of the information should be thrown away. For example, if one value of f_t is zero then the cell should delete corresponding part of information in h_{t-1}.

Once the neural network forgets the previous record, the model needs to update the input through input gate as shown in Equation (10). In the input gate, i_t is also a vector of values between zero to one. In order to update the input of neural network, the model needs to produce a novel digital vector of values and adds it to the previous state of the cell. Equation (11) is shown as below where $\tilde{h}_t$

is the new vector of numbers between zero to one and $W_{\tilde{h}}$ and $b_{\tilde{h}}$ are weights and bias of $\tilde{h}_t$. In order to output $\tilde{h}_t$ as vector of numbers between zero to one, the tanh function is used in computing $\tilde{h}_t$ because results of sigmoid layer could not be negative. Values in $\tilde{h}_t$ will only increase when sigmoid layer is used in creating $\tilde{h}_t$. The results of tanh function can be either positive or negative, therefore $\tilde{h}_t$ is allowed to increase and decrease. So as to compute h_t, i_t will be multiplied by $\tilde{h}_t$. Then the new cell state is computed by adding up $f_t * h_{t-1}$ and $i_t * \tilde{h}_t$ as shown in Equation (12).

Finally, the model needs to calculate the predicted oil production value of this cell by using the output gate. As shown in Equations (13) and (14), $W_{\tilde{o}}$ and $b_{\tilde{o}}$ are weights and bias of the output gate. The model runs a sigmoid layer to decide which part of the information of the current cell will be output. State of the cell is processed by the tanh function and then multiplied by results of sigmoid layer. After that, the model outputs the predicted oil production value of this LSTM cell o_t.

The LSTM prediction model is constructed with an input layer, a hidden layer and an output layer. Two ANN cells is used to build input layer to receive steam-oil pairs from time window X_t. Then, a set of LSTM neurons is used to construct the hidden layer of the model. Only one ANN neuron is served as output layer to summarize the data sequence from hidden layer. The structure of the LSTM prediction model is shown in Figure 3.5. In order to prevent the model from overfitting on training set, dropout regulation is used between hidden layer and

output layer. To further prevent overfitting, early stop method is also applied in the proposed model to stop the training process by setting patience value. Prediction models with selected timesteps for the wells will be used in the following DP process at the next step.

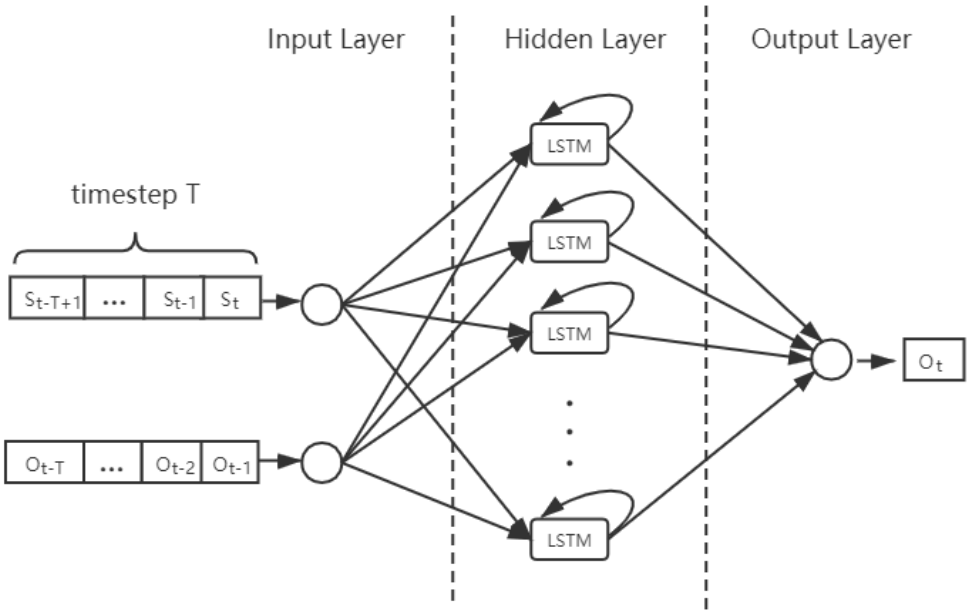


Figure 3.5: Structure of the LSTM prediction model

3.2.4 Optimizing steam injection using dynamic programming

Steam injection distribution optimization in SAGD oil field is complex because the overall volume of generated steam is limited, and steam injection volume of individual well should be within a certain range. In this thesis, the objective is to maximize the total oil production by optimizing steam injection of different wells in the oil field. It can be solved by breaking it down into smaller sub-problems, i.e., maximizing oil production of a group of wells in the oil field.

As mentioned, DP is a method for solving a complex problem by decomposing the problem into sub-problems. It can be applied to optimization problems which have two important characteristics: optimal substructure and overlapping subproblems [39, 40]. Optimal substructure refers an optimal solution contains optimal sub-solutions. Overlapping subproblems means a recursive algorithm would visit the same subproblems repeatedly. In our case, the optimization process starts from optimizing steam injection volume of a single well and ends when it optimizes all wells in the oil field. The problem has optimal sub-solutions which is to optimize steam injection distribution with different numbers of the optimized wells. Besides, the algorithm re-calculates the steam injection volume of each well when optimizing different numbers of wells. Therefore, the steam injection distribution optimization problem has the optimal substructure and overlapping subproblems, and the proposed DP algorithm is able to converge to the global optimal solution for the steam injection distribution optimization problem every time when the algorithm optimized all the SAGD wells in the oil field .

Given the total volume of steam is M m^3, the sub-problem is to determine the steam injection for each well so that total oil production $P_n(m)$ is maximized, with respect to the steam injection volume of the well less than m m^3 (here $m \leq M$), and steam injection volume of individual well meets minimum and maximum injection constraints at the *t-th* time interval. In each sub-problem, considering the total volume of injecting steam is m m^3 the number of wells in the current wells group is n and $P_n(m)$ as the total volume of oil production of the current sub-

problem. The minimum injection constraint of the *k-th* well ($\theta_1^k * i_{t-1}^k$) is represented as min_k and the maximum injection constraint of the *k-th* well ($\theta_2^k * i_{t-1}^k$) is represented as max_k.

The optimization algorithm starts from one well and then recursively optimize all wells in the oil field. First, the algorithm sets *n* as 1 which means only optimizing one well in this scenario and then gradually changes the value of *m* ($min_1 \leq m \leq max_1$). After that, the algorithm calculates the value of total oil production $P_n(m)$ in this sub-problem and saves it into the memory table. The main idea of steam injection distribution optimization using DP is that the algorithm calculates $P_n(m)$ based on the results which are already saved in the memory table. In this way, $P_n(m)$ can be easily obtained if $P_{n-1}(d)$ is known, and $P_{n-1}(d)$ means optimizing a group of *n-1* wells with *d* m^3 steam ($d \leq m$). The recursive calculation equation of this problem is shown as below, where $p_n(m_n)$ is the oil production of the *n*-th well if injecting m_n m^3 ($min_n \leq m_n \leq max_n$) steam into this well:

$$P_n(m) = \max_{min_n \leq m_n \leq max_n} \{p_n(m_n) + P_{n-1}(m - m_n)\}, \quad n > 1 \qquad (15)$$

$$P_1(m) = \max_{min_1 \leq m \leq max_1} p_1(m_1) \qquad (16)$$

The algorithm gradually increases *n* and calculate $P_n(m)$. When *n* reaches to the total number of wells *N* and the steam injection volume *m* reaches *M*, $P_N(M)$ is obtained as the optimized oil production with *M* m^3 steam in *N* different wells in the oil field. Pseudocode of the optimization process is shown in Figure 3.6. For the *N* wells in the oil field, each subproblem should be viewed as an indispensable step and perform the optimization with the maximum *M* m^3 steam

to obtain the optimal sub-solution. As mentioned, obtaining $P_n(m)$ is based on the recursive calculation result of $P_{n-1}(d)$, therefore the time complexity for this step is $O(M^2)$. Since this step is performed N times, the time complexity for the DP algorithm is $O(NM^2)$.

Input:

Predicted oil production result from LSTM

Number of wells in the oil field N

Total volume of steam M.

Output:

Optimized total oil production $P_N(M)$.

Algorithm:

(1) *Obtain the oil production result from LSTM prediction model*

(2) *Initialize memory table to save the calculation result of each step*

(3) *Repeat (for n wells in the oil field,* $1 \leq n \leq N$)

(4) *Repeat (for m as the total volume of steam,* $0 \leq m \leq M$)

(5) *Calculate* min_n *and* max_n *for each well*

(6) $P_n(m) = \max_{min_n \leq m_n \leq max_n} \{p_n(m_n) + P_{n-1}(m - m_n)\}$

(7) *Save the calculation result in table*

(8) *Return $P_N(M)$ as the optimized total oil production*

Figure 3.6: The process of dynmaic programming

3.3 Case studies

3.3.1 Data description and system specification

The Suncor MacKay River of Athabasca oil sands is a SAGD oil field with twelve well pads located in northeastern Alberta, Canada. It started from 2002 and 262 wells are currently under operating. The capacity of the oil field is 38,000

barrels per day. In case studies, Pad C and Pad D from Suncor MacKay River oil field and a set of observation wells near the Suncor MacKay River are chosen to apply the DP optimization method.

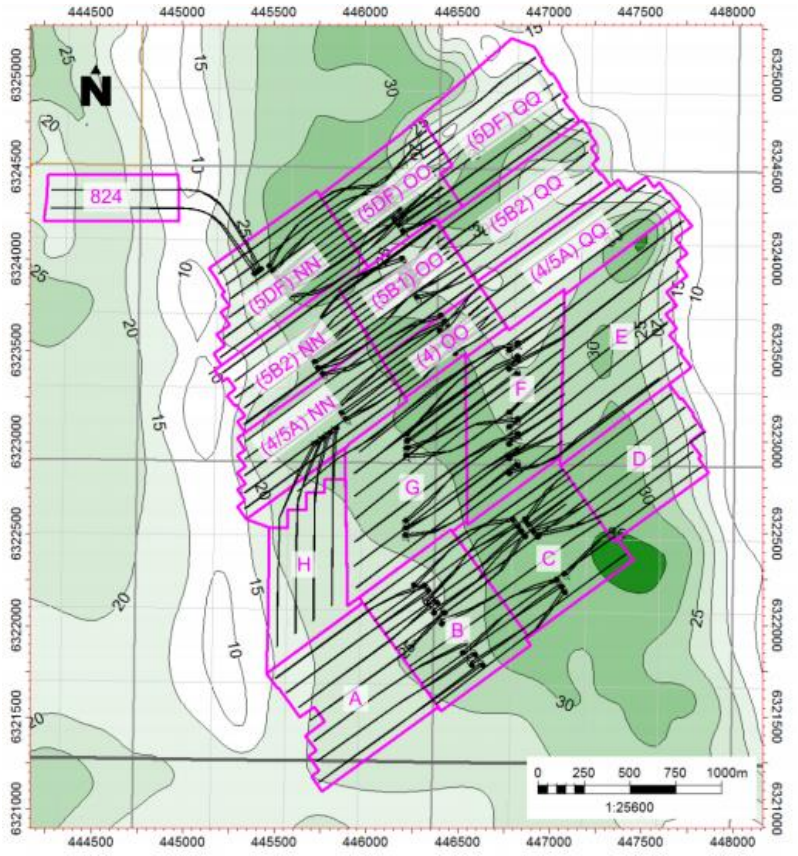


Figure 3.7: Suncor Mackay River oil field (taken from [44])

Figure 3.7 shows the Suncor MacKay River oil field. The three datasets used in the experiments are listed as follow. Dataset 1 is monthly production data of Pad C (six wells) in Suncor MacKay River from August 1997 to December 2011. Dataset 2 is monthly production data of Pad D (five wells) in Suncor MacKay River from August 1997 to December 2011. Dataset 3 is the daily production data of three observation wells near Suncor MacKay River from August 1 1993 to

October 12 1998 [45]. The experiments were conducted on a 1.60 GHz Core i5 PC with 8 GB of RAM. All methods were implemented in Python 3.6.6 and Keras 2.2.4. All values reported are the average of the 10 runs.

3.3.2 The performance analysis of the dynamic programming method

To study the impact of control parameters on the optimization performance, a comparison experiment is conducted on the two pads with three study cases. Each case was ran for 10 times, therefore there are totally 60 trials for the comparison experiments on the two pads. The control parameters are set between ±5% ~ ± 20% of the steam injection volume of the well in the previous time interval. In Case 1, the parameters θ_1 and θ_2 are set as 0.95 and 1.05. In Case 2, θ_1 and θ_2 are set as 0.90 and 1.10. In Case 3, θ_1 and θ_2 are set as 0.85 and 1.15. Table 3.1 - 3.3 show the result of the comparison experiments.

Table 3.1: Optimization performance on Dataset 1 using dynamic programming

Pad C	Oil production (m^3) in Case 1	SOR in Case 1	Oil production (m^3) in Case 2	SOR in Case 2	Oil production (m^3) in Case 3	SOR in Case 3
Jun-2011	24644.55	1.538	24848.87	1.525	**24887.64**	**1.523**
Jul-2011	24742.58	1.826	25097.15	1.800	**25220.78**	**1.791**
Aug-2011	24586.66	1.610	24681.35	1.603	**24794.63**	**1.596**
Sep-2011	23786.62	1.642	23896.4	1.634	**23942.34**	**1.631**
Oct-2011	23395.96	1.773	23409.33	1.772	**23636.13**	**1.755**
Nov-2011	23497.29	2.047	23656.8	2.033	**23758.50**	**2.025**
Total	144653.66	1.737	145589.9	1.726	**146240.02**	**1.718**

Table 3.2: Optimization performance on Dataset 2 using dynamic programming

Pad D	Oil production (m^3) in Case 1	SOR in Case 1	Oil production (m^3) in Case 2	SOR in Case 2	Oil production (m^3) in Case 3	SOR in Case 3
Jun-2011	10492.26	0.860	10536.54	0.857	**10577.08**	**0.853**
Jul-2011	10933.47	1.544	10847.41	1.556	**10942.42**	**1.542**
Aug-2011	11097.75	1.809	11181.62	1.796	**11287.18**	**1.779**
Sep-2011	11060.41	1.913	11116.43	1.904	**11125.70**	**1.902**
Oct-2011	10675.59	2.094	10693.62	2.091	**10760.97**	**2.078**
Nov-2011	10274.53	1.965	10258.17	1.968	**10394.86**	**1.942**
Total	64534.01	1.700	64633.79	1.697	**65088.21**	**1.685**

Table 3.3: Optimization performance on Dataset 3 using dynamic programming

Daily	Oil production (m^3) in Case 1	SOR in Case 1	Oil production (m^3) in Case 2	SOR in Case 2	Oil production (m^3) in Case 3	SOR in Case 3
06-Oct-1998	116.30	4.539	116.54	4.530	**118.55**	**4.453**
07-Oct-1998	109.10	4.782	109.33	4.772	**112.86**	**4.623**
08-Oct-1998	101.76	5.099	101.95	5.090	**109.42**	**4.742**
09-Oct-1998	103.61	4.788	107.39	4.620	**107.65**	**4.608**
10-Oct-1998	98.86	5.214	103.55	4.978	**105.34**	**4.894**
11-Oct-1998	94.79	5.483	95.01	5.470	**100.16**	**5.189**
Total	624.42	4.964	633.77	4.891	**653.98**	**4.740**

As shown in tables, the best optimization result of the two pads is obtained in Case 3 with the highest oil production and the lowest SOR. It is because when θ_1 and θ_2 are set as 0.85 and 1.15, the optimization method is able to allocate less steam into wells with low production potential and distribute more steam into wells with high production potential. The result of the experiment proves the

effectiveness of the DP method and shows the impact of the control parameters on the optimization performance.

3.3.3 Comparison experiment of the prediction models

The objective of this experiment is to compare the prediction results of different prediction models. Five different prediction models are implemented and compared with LSTM in the experiment. The five methods include ANN [9], ELM [10], ENN [12], EEMD-LSTM [30] and EEL-ELM [31].

To evaluate the result of different prediction models, the following three evaluation metrics are used for experiments on Pad C and Pad D of the oil field. The dataset is spilt into training data (from August 1997 to November 2008) and testing data (from December 2008 to June 2016). In the evaluation metrics, o_i is the actual oil production, $\hat{o}_i$ is the predicted oil production and n is number of records in testing data.

1) Root mean squared error (RMSE) is the square root of the average of squared errors:

$$\mathrm{RMSE} = \sqrt{\frac{\sum_{i=1}^{n}(o_i - \hat{o}_i)^2}{n}} \tag{17}$$

2) Mean absolute error (MAE) is the summation of difference between two corresponding variables divided by the total number of observations:

$$\mathrm{MAE} = \sqrt{\frac{\sum_{i=1}^{n}|o_i - \hat{o}_i|}{n}} \tag{18}$$

3) Pearson correlation coefficient (PCC) is the covariance of the two variables divided by the product of their standard deviations:

$$\mathrm{PCC} = \frac{cov(o_i, \hat{o}_i)}{\sigma o_i \sigma \hat{o}_i} \tag{19}$$

In this experiment, the detailed adjustable parameters of different models are given in Table 3.4.

Table 3.4: Parameter setting of different models

Model	ANN	ELM	ENN	EEL-ELM	EEMD-LSTM	LSTM
Data scaler	Standard	Standard	Standard	Standard	Standard	Standard
Number of hidden layers	1	1	1	1	1	1
Number of neurons in the hidden layer	10	10	10	10	10	10
Training batch size	10	N/A	10	10	10	10
Training epoch	100	N/A	100	100	100	100

For the EEL-ELM model, the first layer is built based on ELM, ENN and LSTM, the parameters setting of the three models for the first layer is the same as the separate model implemented in the comparison experiment. The proposed LSTM prediction models are constructed based on the structure described in Section 3 with the selected timestep from the optimization experiment. The results of comparison experiment in term of three evaluation metrics are shown in Tables 3.5 to 3.10.

Table 3.5: Comparison of different models in terms of RMSE on Pad C

RMSE ($100\ m^3$)	ANN	ELM	ENN	EEL-ELM	EEMD-LSTM	LSTM
Well 1	7.396	8.791	8.126	8.742	7.610	**7.142**
Well 2	8.522	19.894	7.708	30.558	10.712	**7.189**
Well 3	9.613	10.015	11.062	10.488	9.187	**8.919**
Well 4	8.145	8.656	10.583	9.642	10.688	**7.107**
Well 5	8.704	**7.633**	10.993	9.246	9.537	8.611
Well 6	10.894	10.295	10.077	10.388	13.399	**9.940**

Table 3.6: Comparison of different models in terms of RMSE on Pad D

RMSE ($100\ m^3$)	ANN	ELM	ENN	EEL-ELM	EEMD-LSTM	LSTM
Well 1	7.616	9.239	9.750	9.547	7.907	**7.002**
Well 2	7.880	9.068	9.544	9.699	7.120	**5.909**
Well 3	4.643	5.314	5.346	6.627	4.495	**3.875**
Well 4	7.398	8.382	9.400	8.076	7.514	**6.628**
Well 5	5.304	6.146	8.241	7.334	5.299	**4.438**

Table 3.7: Comparison of different models in terms of MAE on Pad C

MAE ($100\ m^3$)	ANN	ELM	ENN	EEL-ELM	EEMD-LSTM	LSTM
Well 1	5.357	5.649	6.117	5.630	5.857	**5.007**
Well 2	7.100	10.880	5.299	14.115	8.736	**4.737**
Well 3	8.348	8.371	8.423	8.531	7.241	**7.607**
Well 4	7.120	7.376	8.790	8.274	8.722	**6.149**
Well 5	6.822	**5.984**	9.334	7.274	6.579	6.662
Well 6	8.831	7.941	**7.623**	7.976	9.776	7.966

Table 3.8: Comparison of different models in terms of MAE on Pad D

MAE ($100\ m^3$)	ANN	ELM	ENN	EEL-ELM	EEMD-LSTM	LSTM
Well 1	6.075	8.237	8.416	8.495	6.056	**5.410**
Well 2	6.607	7.532	6.912	7.402	5.705	**4.481**
Well 3	3.696	4.577	3.763	5.581	3.455	**2.770**
Well 4	5.921	6.845	7.054	6.029	5.838	**4.891**
Well 5	3.996	4.891	6.937	5.715	4.397	**3.230**

Tables 3.5 to 3.8 compare the performance of the prediction models in terms of RMSE and MAE on the two pads. Comparison results show that the proposed LSTM prediction models have lower RMSE and MAE values than other models in majority of cases, which means the proposed LSTM prediction model outperforms other five models on oil production prediction.

Table 3.9: Comparison of different models in terms of PCC on Pad C

PCC	ANN	ELM	ENN	EEL-ELM	EEMD-LSTM	LSTM
Well 1	0.948	0.929	0.938	0.927	0.948	**0.949**
Well 2	0.919	0.606	0.930	0.412	0.917	**0.934**
Well 3	0.867	0.878	0.827	0.875	0.860	**0.883**
Well 4	0.666	0.667	0.666	0.621	0.750	**0.782**
Well 5	0.778	**0.839**	0.706	0.776	0.725	0.788
Well 6	0.808	**0.836**	0.797	0.815	0.589	0.799

Table 3.10: Comparison of different models in terms of PCC on Pad D

PCC	ANN	ELM	ENN	EEL-ELM	EEMD-LSTM	LSTM
Well 1	0.895	0.897	0.888	**0.900**	0.865	0.896
Well 2	0.867	0.829	0.853	0.794	0.857	**0.892**
Well 3	0.787	0.742	0.791	0.582	0.849	**0.858**
Well 4	0.654	0.609	0.548	0.582	0.685	**0.716**
Well 5	0.673	0.741	0.562	0.673	0.717	**0.745**

Tables 3.7 and 3.10 compare the performance of the prediction models in terms of PCC on the two pads. The result shows that the proposed LSTM prediction model has higher PCC value than other models in most of cases, which also indicates the proposed LSTM prediction model has better performance.

The results of the experiment show that overall, the proposed LSTM model achieves better prediction performance compared to other five prediction models in terms of three evaluation metrics. The traditional ANN model and ELM model are not able to capture the temporal dependency appearing in time series oil production data. Although the ENN model is equipped with the context layer which serves as the memory structure of the network, the memory structure of the model is simple and not able to handle the complex time series oil production data. The EEL-ELM model is a combined model which aggregating ELM, ENN and LSTM models. However, in the experiment, the prediction performance of ELM and ENN models are notably lower than the proposed LSTM model. Aggregating three separate model decreases the performance of the prediction model.

Therefore, the prediction performance of EEL-ELM model in oil production prediction problem is worse than the proposed LSTM model. The EEMD-LSTM model uses ensemble empirical model decomposition to decompose the oil production time series data into several intrinsic mode functions and utilizes dynamic time warping to detect the similarity and stability of different intrinsic mode functions. However, the input of the EEMD-LSTM model only includes historical oil production data and does not consider the influence of steam injection in SAGD oil field. The EEMD-LSTM also does not have the proper setting of timesteps to reflect the temporal dependency among the input time series data. Thus, the prediction performance of EEMD-LSTM model in SAGD oil field is worse than the LSTM prediction model proposed in this thesis. Compared to these models, the proposed LSTM prediction model uses backpropagation through time technique to update weights of the network and it avoids gradient vanishing problem by making use of the internal gate structures. Furthermore, it considers the influence of steam injection when predicting the future oil production and reveals the temporal dependency among the input time series data by proper setting of timesteps. Therefore, the proposed LSTM prediction model achieves better prediction performance.

Chapter 4

Q-learning optimization method

The idea of the proposed Q-learning optimization method is described in Chapter 4. The outline of this chapter is shown as follow. In Section 4.1, the steam injection distribution optimization problem is formalized under the Q-learning scenario. In Section 4.2, the workflow of the proposed Q-learning optimization method is explained. Finally, case studies are implemented in Section 4.3 to compare the optimization performance of the DP method and Q-learning method.

4.1 Problem statement

Steam injection distribution optimization problem refers to the process of distributing limited amount of steam among multiple wells in SAGD oil field to maximize the total oil production. In order to solve the steam injection distribution optimization problem considering long-term reward using Q-learning, the problem should be formalized as MDP problem. The definition of the steam

injection data and oil production data in MDP problem is the same as the definition in dynamic programming problem. When injecting steam into the SAGD well, steam injection amount of different wells in the oil field describes the situation and injection history of the oil field. Therefore, the state of the MDP is defined as follow:

Definition 8: State. Given *N* wells in a SAGD oil field, the state of the steam injection distribution optimization problem at tth MDP step is a set of cumulative steam injection amount of all wells in the oil field, denoted as:

$$s_t = \{c_t^1, c_t^2, \dots, c_t^k, \dots, c_t^N\} \tag{20}$$

where c_t^k is the cumulative steam injection amount of the kth well in the oil field at tth MDP step.

Actions for reinforcement learning model should be properly defined by considering number of wells to be injected and steam injection amount for different wells at each time step. The amount the steam is continuous, which makes the problem becomes a continuous task. However, in reality, engineers usually set and adjust the amount of steam by a certain unit, e.g. 100 m^3 at a time. Therefore, the amount of injected steam is a discrete numerical value.

Given *N* wells in the oil field, if the action is defined as operating multiple wells in the oil field with a unit, the number of actions becomes 2^N, which generates huge number of actions and greatly increase the computational complexity of the algorithm. Therefore, the action in MDP problem is restricted to operate one well at a time and defined as follows.

Definition 9: Action. Given N wells and M m^3 steam in the oil field. The action of the steam injection distribution optimization problem at tth MDP step is injecting v m^3 high temperature steam into the kth well, denoted as:

$$a_t \doteq c_t^k + v, \qquad min^k \leq c_t^k \leq max^k, \quad \sum_{k=1}^{N} c_t^k \leq M \tag{21}$$

where c_t^k is the steam injection amount of the kth well in the oil field at tth MDP step. $\doteq$ means action a_t injects v m^3 steam into c_t^k. min^k is the minimum cumulative amount of steam injection for the kth well, and max^k is the maximum cumulative amount of steam injection for the kth well. The steam injection amount of all the wells in the oil field should not exceed the available amount of steam generated by the facilities. What is more, the steam injection amount of each well in the oil field should be within certain range. Therefore, c_t^k should be within $[min^k, max^k]$, and total volume of cumulative steam injection amount of all the wells should not exceed M m^3.

After taking an action a_t at state s_t, the steam is injected into the kth well and state s_t transits to state s_{t+1}. The oil production of the injected well under the two states is different, therefore, the action causes the increase or decrease of the oil production, the difference of the oil production is used below to define the instant reward of the problem.

Definition 10: Instant reward. After taking action a_t at state s_t, the instant reward of the steam injection distribution optimization problem at tth MDP step is the difference of the instant oil production after taking action a_t, denoted as:

$$r_t(s_t, a_t, s_{t+1}) = o_{t+1}^k - o_t^k \tag{22}$$

where o_t^k is the instant oil production of the kth well at state s_t before taking action a_t and o_{t+1}^k is the instant oil production of the kth well at state s_{t+1} after taking action a_t.

As mentioned, the long-term reward should be introduced to improve the optimization performance by motivating the agent to consider the reward from the future. Discount factor plays a vital part when defining the long-term reward. A lower discount factor motivates the agent to count more on the reward of the current action but not rewards from the future [51]. Whereas, a higher discount means the agent takes more credit on the future reward of the action but not the current reward. The long-term reward should be future oil production increase of the current MDP action.

Definition 11: Long-term reward. The long-term reward of the steam injection distribution optimization problem for a MDP episode is the discounted sum of the instant reward starting from state s_t and following the policy π, denoted as:

$$R_t = V(s_t) = E(\sum_{t=0}^{n} \gamma^t r_t) \ , \tag{23}$$

Where $V(s_t)$ is also called value function and γ is the discount factor satisfying $0 \leq \gamma \leq 1$, t is the current step of a MDP episode and n is the end step of the MDP episode.

In the steam injection distribution optimization problem, the environment is the SAGD oil field simulated by well steam injection data and well oil production data. The agent should be considered as engineers who intend to optimize the steam injection distribution of the field. The environment observes the actions from the agent and gives corresponding reward as reflection.

The objective of the steam injection distribution optimization problem is to choose a policy π that maximizes the long-term reward defined in Equation (23). There should exist an optimal value function $V^*(s_t)$ that has the highest value for the state as shown in Equation (24).

$$V^*(s_t) = \max_{\pi} V(s_t) \tag{24}$$

A policy that maximizes the value function mentioned above is called an optimal policy and denoted as π^*. The policy should be the steam injection distribution strategy that allocate the high temperature steam among the oil field to achieve the maximum oil production. Q-learning is a reinforcement learning method to generate the optimal policy π^* of a MDP using the Q-function. The Q function $Q(s_t, a_t)$ represents the quality of a certain action given a state. In other words, the Q function gives the expected increase of the oil production when starting at s_t, performing a_t and following π. The relationship between the $V^*(s_t)$ and $Q(s_t, a_t)$ is shown in Equation (25).

$$V^*(s_t) = \max_{a_t} Q(s_t, a_t) \tag{25}$$

The optimal Q-function is denoted as $Q^*(s_t, a_t)$ that gives the maximum total increase of the oil production for the agent when starting at s_t, taking action a_t

and following π^*. That is to say, at state s_t, the maximum expected total increase of the oil production $V^*(s_t)$ is same as value of $Q^*(s_t, a_t)$ following the optimal policy π^*. With the statement discussed above, the steam injection distribution optimization problem under Q-learning scenario is defined as follow.

Definition 12: Q-learning Steam Injection Distribution Optimization Problem. The Q-learning steam injection distribution optimization problem is to extract an optimal policy π^* by choosing the action a_t that gives maximum increase of the oil production at state s_t.

$$\pi^*(s_t) = arg \max_{a_t} Q^*(s_t, a_t) \tag{26}$$

The Q-learning method includes three steps: data preprocessing, building LSTM models for oil production prediction to simulate the environment and optimizing steam injection using Q-learning. The workflow of the Q-learning method is shown in Figure 4.1. First, the historical oil production and steam injection data is preprocessed for the optimization model. Then, LSTM neural network models are built with GA to predict the oil production of the wells and simulate the environment of the problem. Finally, Q-learning is applied to optimize the steam injection among different wells to maximize the total oil production of the SAGD oil field.

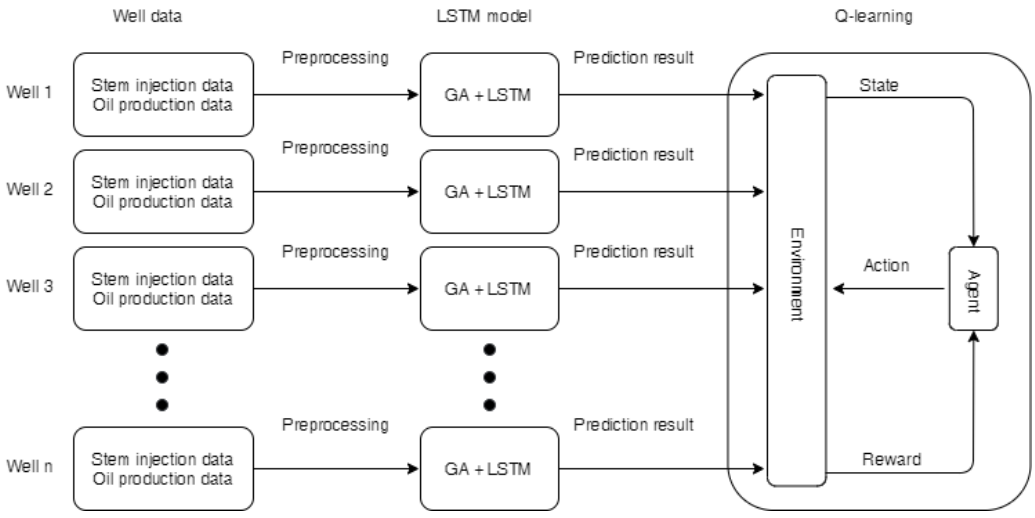


Figure 4.1: The workflow of the Q-learning optimization method

4.2 Workflow of the Q-learning optimization method

4.2.1 Data Preprocessing

The data preprocessing step of the research includes two procedures. First, missing values of historical steam injection and oil production data are filled with global constant *Unknown*, and preprocessed data tuple with *Unknown* values are ignored when feeding the data into the optimization model. Second, Z-Score normalization method is used to transform the steam injection and oil production data into a same scale.

4.2.2 Building LSTM models for oil production prediction

Before starting the Q-learning process, environment of the problem should be simulated using historical steam injection and oil production data. In this research the environment is simulated by the combination of genetic algorithm and LSTM neural network model to provide basic information for calculating the

reward of each action. For each SAGD well, a LSTM prediction model is constructed to receive the historical data and output the target oil production amount. The input data of a LSTM model are formalized as steam-oil pair to represent the production performance of the corresponding SAGD well. The structure of LSTM prediction model is determined using genetic algorithm.

The LSTM prediction model is constructed with an input layer, a hidden layer and an output layer. Two ANN cells is used to build input layer to receive the preprocessed data. And then, a set of LSTM neurons is used to construct the hidden layer of the model. Finally, one ANN neuron is served as output layer to generate the result from hidden layer. The detail of the LSTM prediction model is described in the Chapter 3 of the thesis.

After training the LSTM models with historical oil production data, the environment of the Q-learning problem is simulated by the LSTM prediction result. The environment plays a vital part to provide feedback for the software agent. During the Q-learning process, the agent receives positive or negative reward after taking each action. With the reward, the agent learns the value of each action and make better decision when picking action under the same state. In the study, action is injecting certain amount of steam into the SAGD well which increases the volume of injected high temperature steam, therefore the reward should be the difference of oil production after taking each action. If the oil production increases after taking the action, the reward should be positive, if the oil production decreases after taking the action, the reward should be negative. In order to know the reward of taking each action, the fully trained LSTM model

should be applied to predict the oil production of each well with different steam injection amount. With the prediction result of the model, the reward of taking each action is obtained based on the predicted oil production difference.

4.2.3 Optimizing steam injection using Q-learning

With the prediction result from LSTM models, the optimal policy π^* could be obtained through Q-learning process. The $Q(s_t, a_t)$ discussed in Equation (25) can be expressed recursively when considering the relationship between the Q-value of current state s_t and action a_t with the Q-value of next state s_{t+1} and action a_{t+1} as shown in Equation (27). It is also called Bellman equation [55] which shows the maximum future reward at current state s_t is the reward that the agent received at the current step r_t plus the maximum future reward of the next state s_{t+1}.

$$Q'(s_t, a_t) = r_t + \gamma \max_{a_{t+1}} Q(s_{t+1}, a_{t+1}) \tag{27}$$

The main idea of Q-learning is iteratively updating the Q-value as shown in Equation (28) to generate the optimal policy π^* that maximizing the total oil production. When updating the Q-value, α is the learning rate that controls how much the difference between previous and new Q-value should be updated to approximate the optimal policy.

$$Q(s_t, a_t) = Q(s_t, a_t) + \alpha[r_t + \gamma \max_{a_{t+1}} Q(s_{t+1}, a_{t+1}) - Q(s_t, a_t)] \tag{28}$$

During the Q-learning process, Q-value of each step is stored in Q-table. The rows and columns of the Q-table represents the states and actions of the Q-learning. In each Q-learning episode, in order to keep the steam injection amount

of all the wells within the limitation, the value of different wells in initial state are set as the minimum injection amount regarding to the production history. As mentioned, one MDP episode starts with the initial state and ends when the total steam injection amount of all the wells exceeds the maximum amount of steam generated in the oil field or steam injection amount of any single well exceeds the steam injection limitation of that well.

When updating the Q-table, the purpose is to improve the accuracy of the Q-value to generate the reliable steam injection strategy that maximizing the total oil production. If the Q-value is not accurate at all, the strategy will be generated based on unrealistic number and means pointless for the real-world oil production process. When choosing actions to improve the accuracy of Q-value, there are two strategies to choose actions which are choosing random action (inject steam into a random well) and choosing greedy action (inject steam into the well which generates the most oil production). On the one hand, if the agent always choose random action to improve the Q-value, it will be difficult for the algorithm to improve the accuracy. On the other hand, if the agents always choose action base on greedy policy, the algorithm will convergence to local optimal solution. Therefore, the key of improving the accuracy of Q-value is to balance the random action and greedy action.

Epsilon-greedy solution is the method to balance whether the algorithm should take a random action to improve the current accuracy of Q-value or it should choose the greedy action to get the most reward based on the current accuracy of Q-value. The epsilon-greedy solution is shown in Equation (29).

$$a_t = \begin{cases} max\ Q(s_t, a_t), & p_1 = 1 - \varepsilon \\ random(a_t), & p_2 = \varepsilon \end{cases} \quad (29)$$

Where a_t is the action should be taken at tth MDP step, p_1 is the probability of taking the greedy action and p_2 is the probability of taking random action, ε is the parameter that adjust p_1 and p_2. When updating the Q-table, the epsilon-greedy solution chooses the action based on the probability distribution of p_1 and p_2, so it is able to improve the accuracy of Q-value while avoid falling into corner cases repeatedly. Therefore, the algorithm can generated the steam injection strategy based on accurate Q-value and maximize the total oil production of the oil field.

After taking each action, the steam injection amount of the corresponding well increases and the problem transfers into the next state. The algorithm calculates the reward of the action and update it in the Q-table. The initial approximations of Q-learning will most likely be completely random. However, after running the model for a given number of episodes, the training results are saved in the Q-table, and the model is capable of obtaining the optimal policy using greedy solution to pick the best action recorded in Q-table [56]. Therefore, the Q-learning optimization method generates the maximum total oil production of the oil field by following the optimal policy. Pseudocode of the Q-learning process is shown in Figure 4.2.

Input:

Predicted oil production result from LSTM

Output:

Optimal steam injecyion policy π^*

Algorithm:

(1) *Initialize* $Q(s_t, a_t)$ *based on the historical data*

(2) *Repeat* (*for each steam injection distribution episode*):

(3) *Initialize* s_t *as the initial steam injection amount of each well*

(4) *Repeat* (*for each action of injecting steam*):

(5) *Choose* a_t *from* s_t *using* ε*-greedy solution derived from* $Q(s_t, a_t)$

(6) *Take action* a_t*, observe* r_t*,* s_{t+1} *based on the prediction result from LSTM*

(7) $Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha[r_t + \gamma \max_{a_{t+1}} Q(s_{t+1}, a_{t+1}) - Q(s_t, a_t)]$

(8) $s_t \leftarrow s_{t+1}$

(9) *Until* s_t *terminates when the steam injection reaches the limiation*

(10) *Generate optimal policy* π^**using greedy solution dericed from* $Q(s_t, a_t)$

Figure 4.2: Pseudocode of the Q-learning algorithm

In Figure 4.2, the input of Q-learning method is the prediction result of LSTM models. The output of the method is the optimal steam injection distribution policy that allocate the high temperature steam among the SAGD wells in the oil field and achieves the maximum total oil production. At the beginning of the algorithm, the Q-learning method initializes the Q-values in Q-table based on the input data as shown in line (1). After that, the algorithm starts to update the Q-value based on each MDP episode as shown in line (2). During each MDP episode, the algorithm first initializes the state of the problem and then taking actions to update the Q-value using epsilon- greedy solution as shown in line (3) – (9). When

the updating of Q-value is finished, the algorithm derives the optimal steam injection distribution optimization strategy based on the Q-table as shown in line (10). With the optimal policy, the agent then distributes the high temperature steam in the SAGD oil field to generate the maximum amount of oil production.

In the following, an example is used to illustrated the updating process of the Q-table. In this example, three wells are in a SAGD oil field. The minimum injection limitation of each well is 100 m^3, the maximum injection limitation of each well is 300 m^3 and the maximum amount of injecting steam of the oil field is 500 m^3. Assuming that there are three actions available during the Q-learning process, which are injecting 100 m^3 steam into Well1 (Action 1), injecting 100 m^3 steam into Well2 (Action 2) and injecting 100 m^3 steam into Well3 (Action 3). At the beginning of the Q-learning method, the algorithm first initializes the state as $\{c_1^1 = 100\ m^3, c_1^2 = 100\ m^3, c_1^3 = 100\ m^3\}$, which means the initial steam injection volume of the three wells are set as 100 m^3 according to the minimum steam injection limitation of the well. The algorithm starts to update the Q-table by performing an action, updating the Q-value and changing to the next state. After taking an action, the steam is injected into the corresponding well and increase the cumulative steam injection volume of the well. With the new cumulative steam injection volume, and algorithm finds the corresponding oil production from prediction result of LSTM model. And then, the algorithm calculates the reward by returning the difference of oil production and adds into the Q-value.

In the example, three episodes are listed to update the Q-table. The order of actions performed in the episodes is [2, 1, 2, 1, 3, 2], [2, 1, 2, 2, 2], [1, 2, 3, 3, 3, 1]. The updating process of the first episode is shown in Figure 4.3. After taking each action, the steam is injected into the corresponding well. The algorithm updates the Q-value with the reward calculated based on LSTM prediction result. When the episode reaches the termination condition (the total volume of injected steam exceeds 500 m^3), the current episode ends and then the next episode starts. After updating the Q-table with certain number of episodes, the optimal policy that produces the maximum amount of oil production is generated based on the Q-value stored in the Q-table. The policy is a series of actions with the maximum Q-value at each state.

Actions

States

Q-table	Inject 100 m^3 steam into Well1	Inject 100 m^3 steam into Well2	Inject 100 m^3 steam into Well3
{100,100,100}	0	40	0
{100,200,100}	50	0	0
{200,200,100}	0	30	0
{200,300,100}	40	0	0
{300,300,100}	0	0	30
{300,300,200}	0	−100	0

Reaching termination condition

Figure 4.3: An example of updating the Q-table

4.3 Case Studies

4.3.1 Data description and system specification

In case study, the datasets used are the same as Section 3.3.1. There are three different datasets. Dataset 1 is monthly production data of Pad C (including six wells) in Suncor MacKay River oil field from August 1997 to December 2011. Dataset 2 is monthly production data of Pad D (including five wells) in Suncor MacKay River oil field from August 1997 to December 2011. Dataset 3 is the daily production data of three observation wells near Fort McMurray from August 1 1993 to October 12 1998. The experiments were conducted on a 1.60 GHz Core i5 PC with 8 GB of RAM. All methods were implemented in Python 3.6.6 and Keras 2.2.4. The oil production reported are the average of the 10 runs.

4.3.2 The performance analysis of the Q-learning method

To study the optimization performance of the Q-learning method, experiments are conducted in the three datasets to compare the optimization result of dynamic programming method and Q-learning method. The two optimization methods are both implemented in the experiment under three datasets. In Dataset 1 and Dataset 2, the two methods are implemented to optimize the oil production from June 2011 to November 2011, the steam injection range of the wells is set as ± 10%. In Dataset 3, the two methods are implemented to optimize the oil production from October 6 1998 to October 11 1998, the steam injection range is also set as ± 10%. Table 4.1 shows the parameter setting of the two optimization methods.

Table 4.1: Parameter setting of the two optimization methods

Method	Dynamic programming	Q-learning
Optimization range	± 10%	± 10%
Discount factor	N/A	1
Learning rate	N/A	0.4
Epsilon	N/A	0.5

In the following, Table 4.2 – 4.10 and Figure 4.4 – 4.9 show the comparison experiment result on three different dataset. For each dataset, first the steam injection distribution under different circumstances are given in tables and pie chart, then the total oil production under different circumstances are shown in the figure.

Table 4.2: Steam injection distribution on Dataset 1 before optimization

Steam injection volume (m^3)	Well 1	Well 2	Well 3	Well 4	Well 5	Well 6
Jun-2011	7090.7	3402.6	5977.4	4424.3	8595.1	8409.6
Jul-2011	8789.9	5318.6	6872.4	5191.2	9370.7	9634.9
Aug-2011	5017.2	1752.9	7146.4	5629.5	10057.2	9973.3
Sep-2011	1448.6	1475.2	7937.2	5268.6	11538.2	11384.6
Oct-2011	1669	1531.1	8277.3	6356.2	12450.3	11200.3
Nov-2011	3557.1	1508.8	9245.1	7684.7	13239.8	12870.3

Table 4.3: Steam injection distribution on Dataset 1 after dynamic programming

Steam injection volume (m^3)	Well 1	Well 2	Well 3	Well 4	Well 5	Well 6
Jun-2011	7400	3600	6400	3600	9300	7500
Jul-2011	7900	5300	7400	5700	8400	10400
Aug-2011	5400	1500	7700	1500	9000	10800
Sep-2011	1400	1500	8600	1500	12500	10400
Oct-2011	1700	1500	9000	1500	11500	12200
Nov-2011	3800	1500	10000	1500	14400	12100

Table 4.4: Steam injection distribution on Dataset 1 after Q-learning

Steam injection volume (m^3)	Well 1	Well 2	Well 3	Well 4	Well 5	Well 6
Jun-2011	7700	3700	5900	4300	8200	8000
Jul-2011	9600	5800	6600	5100	8800	9200
Aug-2011	5500	1900	7600	5400	9600	9500
Sep-2011	1500	1600	8700	5600	10900	10700
Oct-2011	1800	1600	8700	6500	11400	11400
Nov-2011	3900	1600	10100	8000	12500	12000

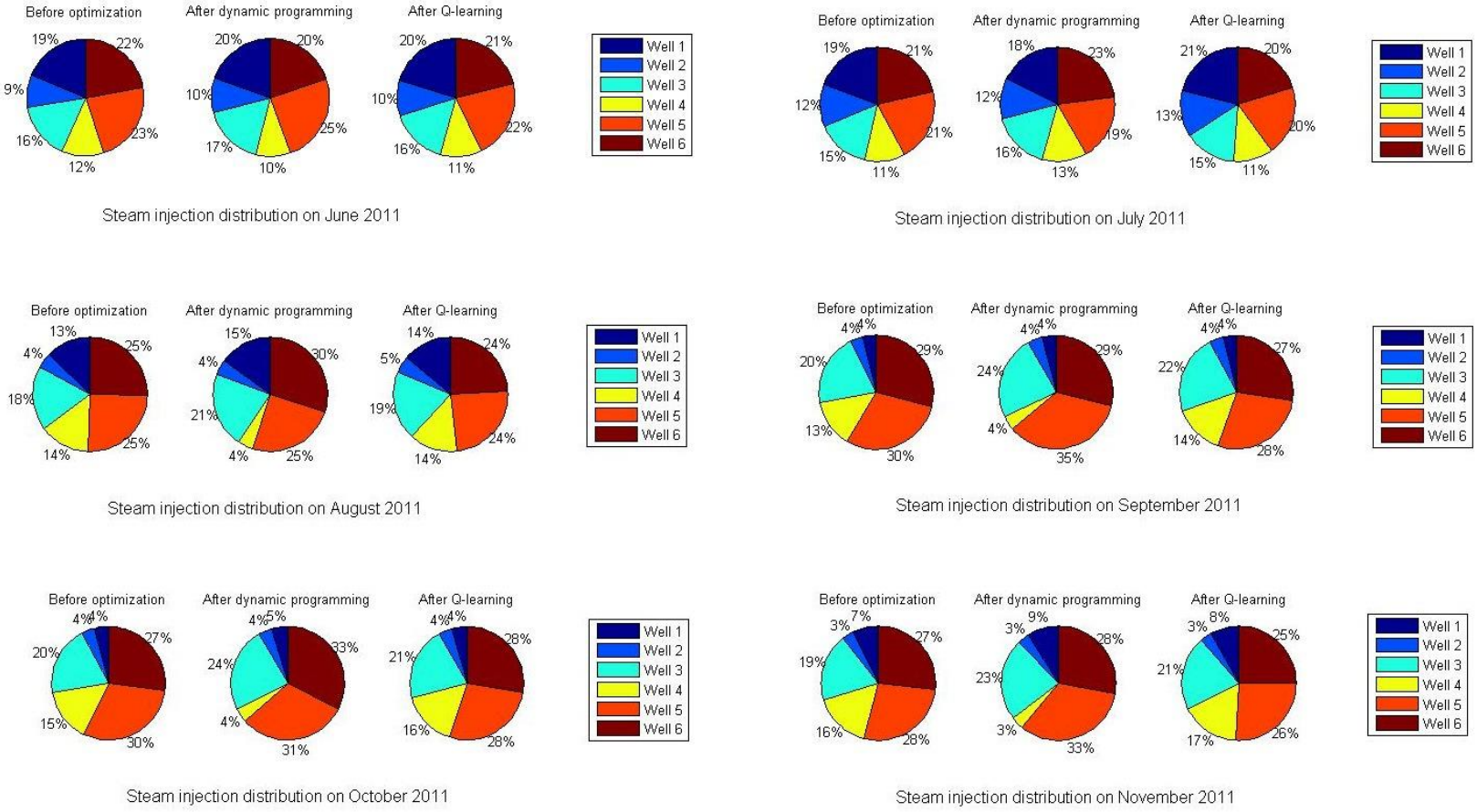


Figure 4.4: Optimization result of steam injection distribution on Dataset 1

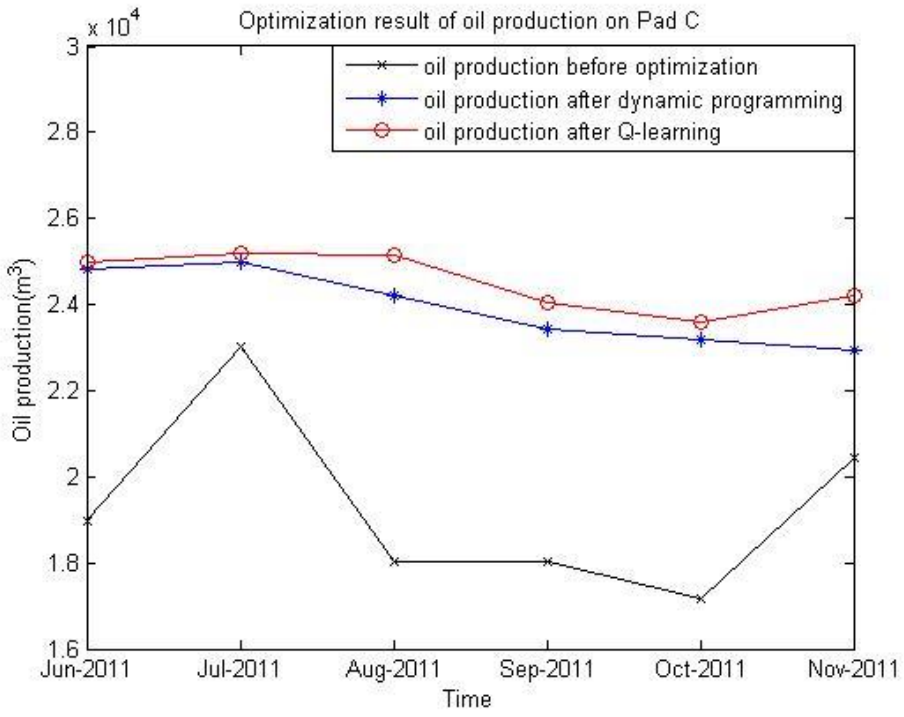


Figure 4.5: Optimization result of oil production on Dataset 1

As shown in Table 4.2 – 4.4 and Figure 4.4, the monthly steam injection distribution of Pad C from June 2011 to November 2011 is not the same under different circumstances. For example, in June 2011, the percentage of steam injection distribuion for the six wells before optimization is [19%, 9%, 16%, 12%, 23%, 22%], after dynamic programming is [20%, 10%, 17%, 10%, 25%, 20%], after Q-learning is [20%, 10%, 16%, 11%, 22%, 21%]. For the dynamic programming method, the steam injection volume of Well 1, Well 2, Well 3, Well 5 are increased, the steam injection volume of Well 4 and Well 6 are decreased. For the Q-learning method, the steam injection volume of Well 1, Well 2 are increased, the steam injection volume of Well 3, Well 4, Well 5 and Well 6 are decreased. However, as shown in Figure 4.5, both changes of the steam injection

distriution increases the total oil production of Pad C. The optimized oil production of Q-learning method is higher than that of dynamic programming method every month.

Table 4.5: Steam injection distribution on Dataset 2 before optimization

Steam injection volume (m^3)	Well 1	Well 2	Well 3	Well 4	Well 5
Jun-2011	2113.9	2110.6	2115.2	1364.9	1321.6
Jul-2011	4235.1	4814.7	3881.2	1900.2	2045.2
Aug-2011	4677.2	5686.1	4519	2473.7	2720.8
Sep-2011	5112.2	6360.6	3276.4	3145.5	3268.5
Oct-2011	5837.1	7232.1	2257.6	3668.2	3362.3
Nov-2011	2797.2	7012.7	2579	4051.1	3745

Table 4.6: Steam injection distribution on Dataset 2 after dynamic programming

Steam injection volume (m^3)	Well 1	Well 2	Well 3	Well 4	Well 5
Jun-2011	2200	2200	1900	1400	1300
Jul-2011	3900	5100	4100	1900	1800
Aug-2011	4200	6100	4300	2600	2800
Sep-2011	4600	6300	3500	3300	3400
Oct-2011	5200	7800	2300	3900	3100
Nov-2011	2500	7300	2700	4300	3300

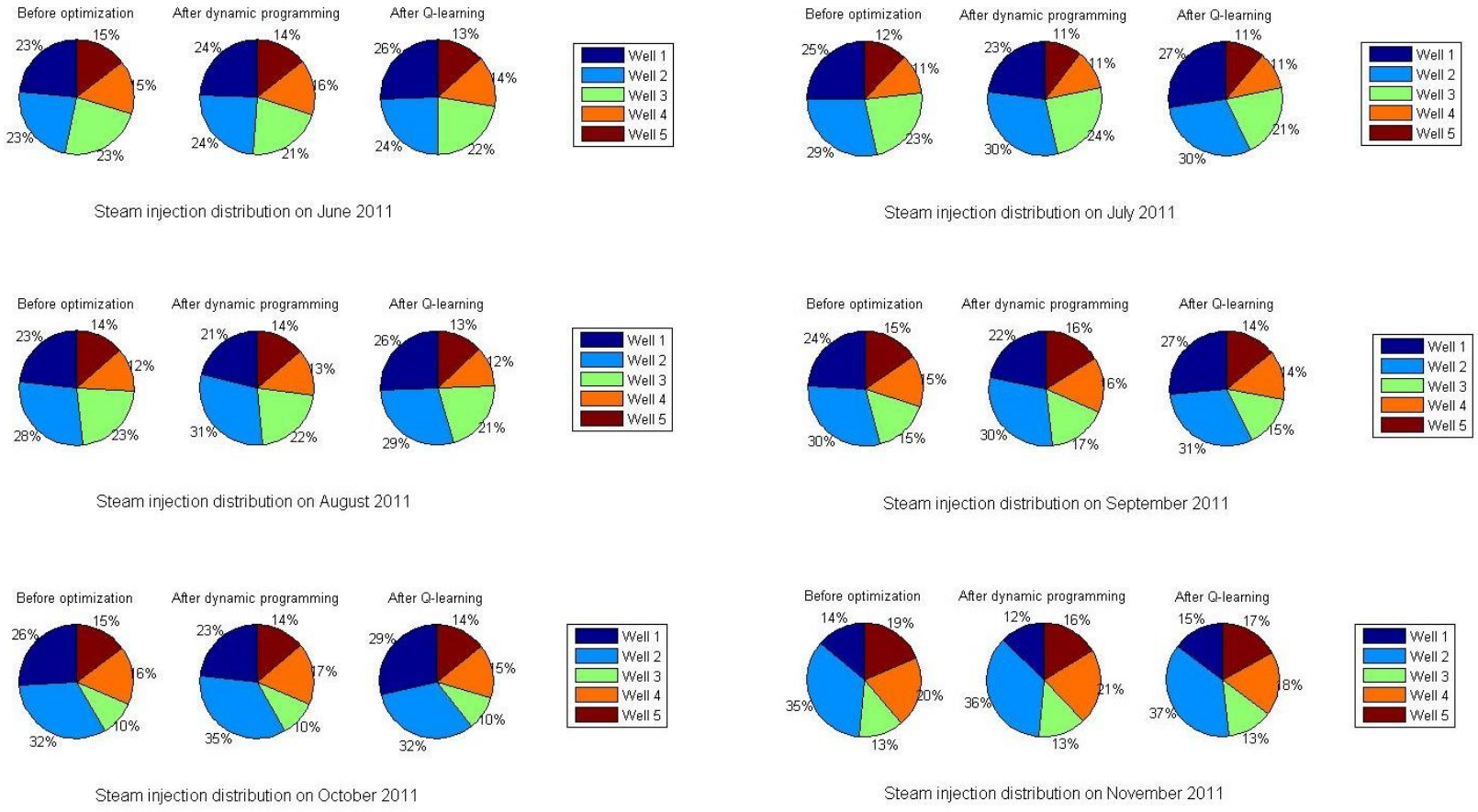


Figure 4.6: Optimization result of steam injection distribution on Dataset 2

Table 4.7: Steam injection distribution on Dataset 2 after Q-learning

Steam injection volume (m^3)	Well 1	Well 2	Well 3	Well 4	Well 5
Jun-2011	2300	2200	2000	1300	1200
Jul-2011	4600	5000	3500	1800	1900
Aug-2011	5100	5800	4200	2300	2600
Sep-2011	5600	6500	3100	2900	3000
Oct-2011	6400	7100	2200	3400	3200
Nov-2011	3000	7400	2600	3700	3400

Figure 4.6: Comparison of steam injection distribution on Dataset 2

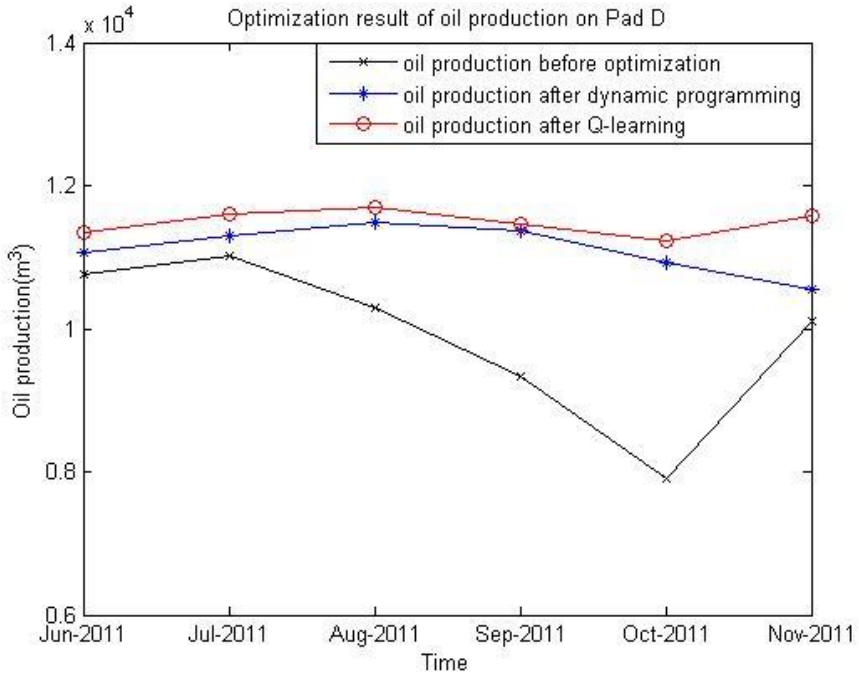


Figure 4.7: Optimization result of oil production on Dataset 2

As shown in Table 4.5 – 4.7 and Figure 4.6, the monthly steam injection distribution of Pad D from June 2011 to November 2011 is not the same in

different scenario. For example, in June 2011, the percentage of steam injection distribuion for the five wells before optimization is [23%, 23%, 23%, 15%, 15%], after dynamic programming is [24%, 24%, 21%, 16%, 14%], after Q-learning is [26%, 24%, 22%, 14%, 13%]. For the dynamic programming method, the steam injection volume of Well 1, Well 2, Well 4 are increased, the steam injection volume of Well 3 and Well 5 are decreased. For the Q-learning method, the steam injection volume of Well 1, Well 2 are increased, the steam injection volume of Well 3, Well 4, Well 5 are decreased. However, as shown in Figure 4.7, both changes of the steam injection distriution increases the total oil production of Pad D. The monthly total oil production of Q-learning method is better than that of dynamic programming method.

Table 4.8: Steam injection distribution on Dataset 3 before optimization

Steam injection volume (m^3)	Well 1	Well 2	Well 3
6-Oct-1998	287.8	107.7	132.4
7-Oct-1998	297	76.2	148.5
8-Oct-1998	294.5	49.8	174.6
9-Oct-1998	296.3	50.3	149.5
10-Oct-1998	295.2	49.8	170.5
11-Oct-1998	294.6	49.9	175.2

Table 4.9: Steam injection distribution on Dataset 3 after dynamic programming

Steam injection volume (m^3)	Well 1	Well 2	Well 3
6-Oct-1998	266	117	144
7-Oct-1998	277	82	162
8-Oct-1998	274	53	191
9-Oct-1998	279	54	163
10-Oct-1998	276	53	186
11-Oct-1998	275	53	191

Table 4.10: Steam injection distribution on Dataset 3 after Q-learning

Steam injection volume (m^3)	Well 1	Well 2	Well 3
6-Oct-1998	294	102	131
7-Oct-1998	299	75	147
8-Oct-1998	298	52	168
9-Oct-1998	300	55	141
10-Oct-1998	301	48	166
11-Oct-1998	300	51	168

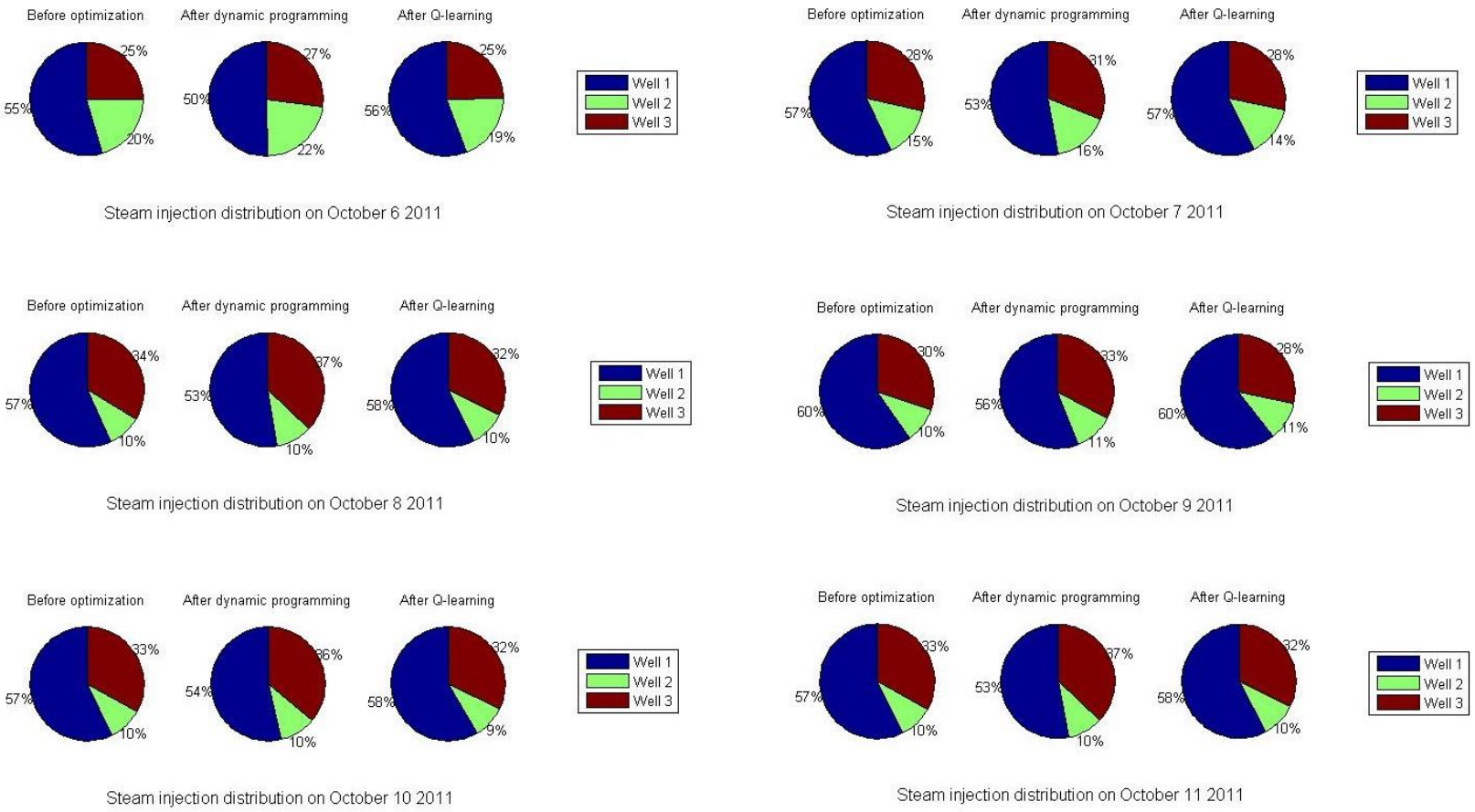


Figure 4.8: Optimization result of steam injection distribution on Dataset 3

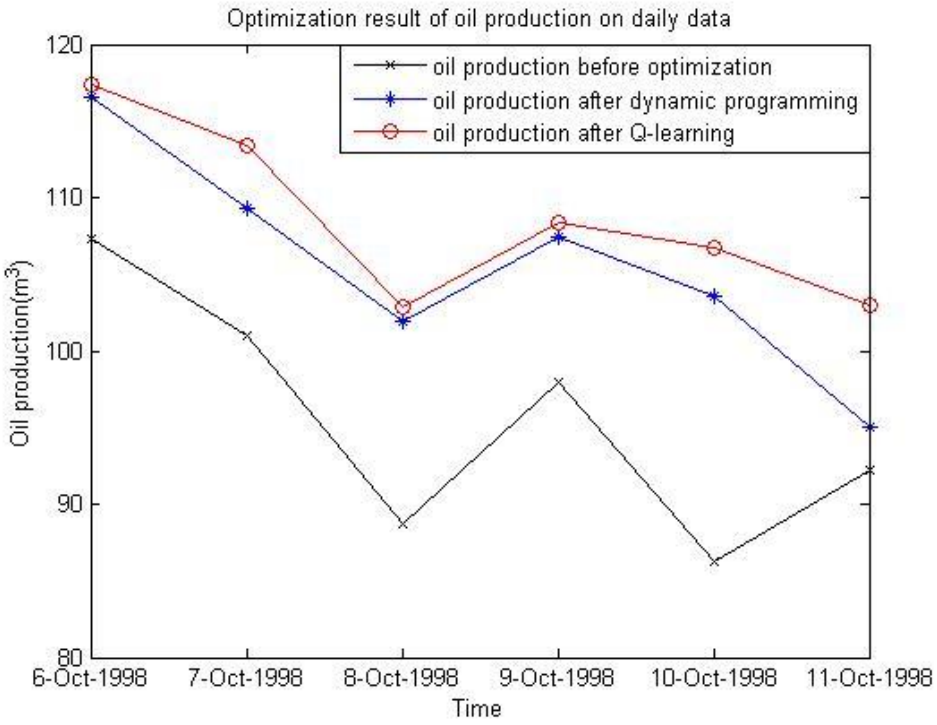


Figure 4.9: Optimization result of oil production on Dataset 3

As shown in Table 4.8 – 4.10 and Figure 4.8, the daily steam injection distribution of the three observation wells from October 6 1998 to October 11 1998 is not the same in different scenario. For example, at October 6 1998, the percentage of steam injection distribuion for the three wells before optimization is [55%, 20%, 25%], after dynamic programming is [50%, 22%, 27%], after Q-learning is [56%, 19%, 25%]. For the dynamic programming method, the steam injection volume of Well 2, Well 3 are increased, the steam injection volume of Well 1 is decreased. For the Q-learning method, the steam injection volume of Well 1 is increased, the steam injection volume of Well 2 is decreased. However, as shown in Figure 4.9, both changes of the steam injection distriution increases the total oil production of the three observation wells. The daily total oil

production of Q-learning method is better than that of dynamic programming method.

There are three conclusions of the experiment as shown in figures. First, both the dynamic programming method and Q-learning method can improve the total oil production of the SAGD oil pad. Second, the optimization performance of Q-learning method is superior than that of dynamic programming method under all datasets. It is because dynamic programming method generates the optimal solution only considering the instant reward of the problem. Third, the Q-learning method is able to produce superior optimization result on both daily data and monthly data. Therefore, the Q-learning optimization method can be applied to various production data with different time scale. Result of the experiment further proves the efficiency of the proposed Q-learning method.

4.3.3 The influence of parameter setting on the optimization result

In order to analyze the influence of parameter setting on the experiment, three comparison experiments are conducted to evaluate the impact of three important parameters, i.e., discount factor, learning rate and epsilon. Production data in October 9 1998 from Dataset 3 is used to implement the comparison experiment. In each experiment, three cases are implemented to show the influence of parameter setting with different steam injection limitation. In Case 1, the steam injection range is set as ± 5%. In Case 2, the steam injection range is set as ± 10%.

In Case 3, the steam injection range is set as ± 15%. Figure 4.10 – 4.12 show the result of the comparison experiment.

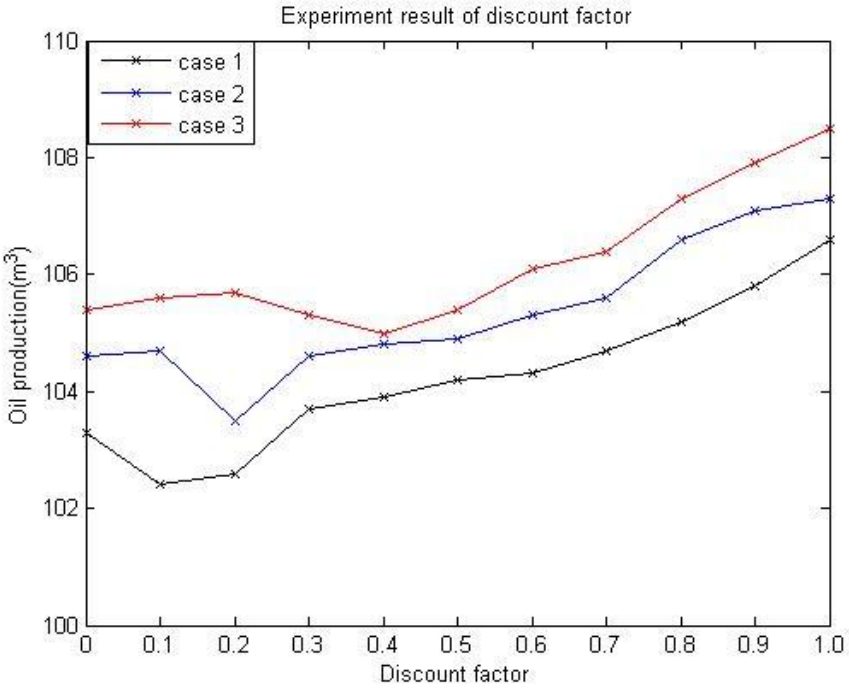


Figure 4.10: Optimization result of oil production with different discount factor

Discount factor plays a vital part when defining the long-term reward. In the experiment, discount factor is changing from 0 to 1.0 to analyze the impact on optimization result. A lower discount factor motivates the agent to count more on the reward of the current action and a higher discount means the agent takes more credit on the reward of the future actions. As shown in Figure 4.10. First, the maximum oil production of the three cases are obtained when discount factor is set as 1.0. Second, the oil production of the three cases are gradually increasing with the increase of discount factor. It is because long-term reward is important for increasing the oil production of the oil field. When discount factor increases

from 0 to 1.0, the long-term reward relies more on the reward from the future actions, therefore increase the oil production optimization result.

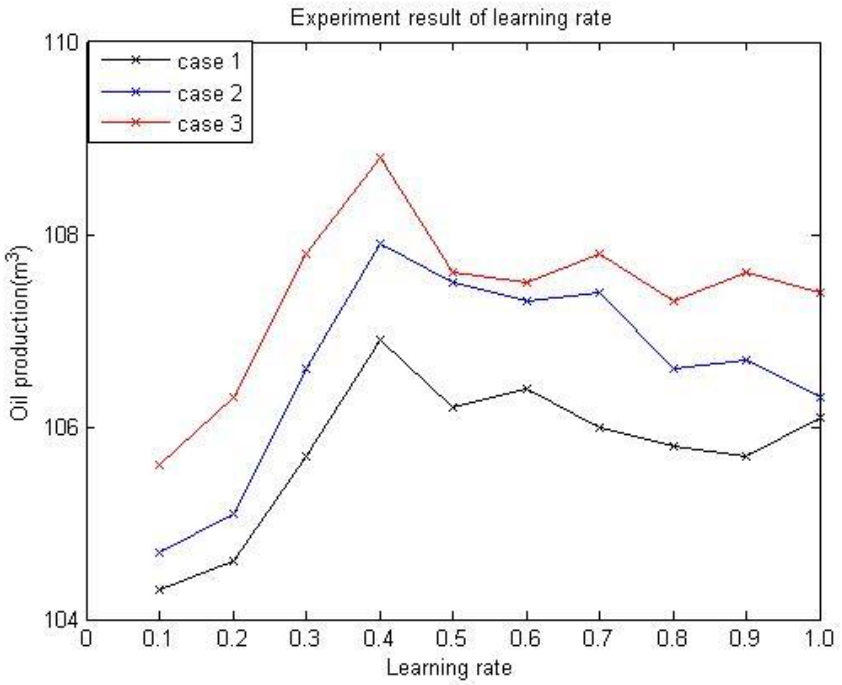


Figure 4.11: Optimization result of oil production with different learning rate

During the Q-learning process, learning rate controls how much the difference between previous and new Q-value should be updated to approximate the optimal policy. In the experiment, learning rate is changing from 0 to 1.0 to analyze the impact on optimization result. If the learning rate is set as 0, the Q-value would not be updated during the training process. As shown in Figure 4.11, first, when learning rate is set as 0, no outcome is generated in the experiments. It is because no updated is made on the Q-value, therefore the Q-learning method cannot generate reasonable optimal policy based on the Q-table. Second, the optimized oil production reaches the maximum value when learning rate is set as 0.4 and slightly decrease when the learning rate increases after 0.4. It is because

the optimization result achieves the optimal solution when the learning rate reaches 0,4 under this circumstances. After that, the increase of learning rate leads to over training of the Q-learning model and then produce lower performance.

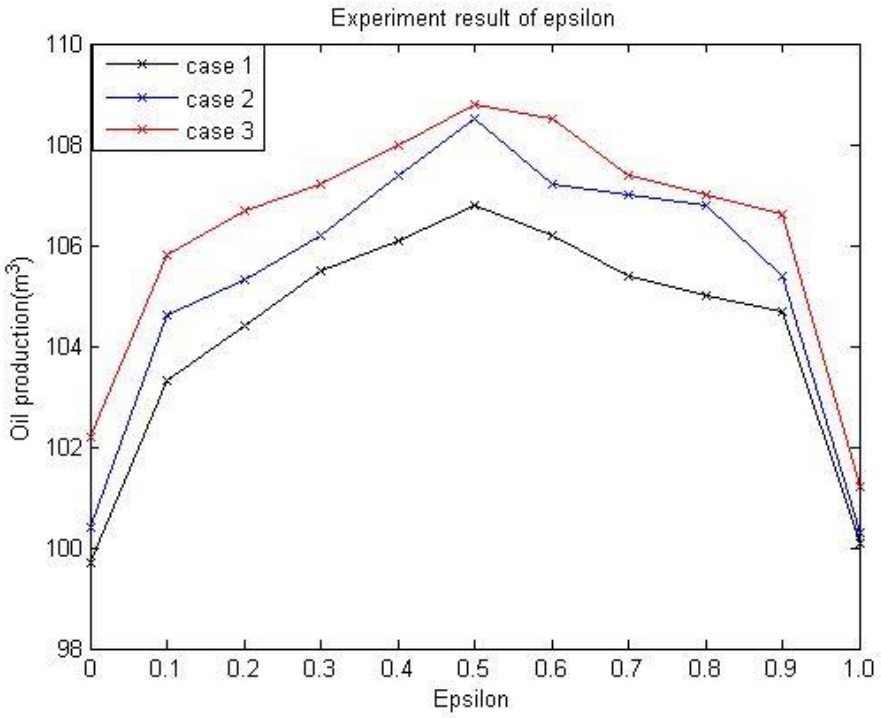


Figure 4.12: Optimization result of oil production with different epsilon

Epsilon is the factor to balance whether the algorithm should take a random action to improve the current accuracy of Q-value or it should choose the greedy action to get the most reward based on the current accuracy of Q-value. In the experiment epsilon is changing from 0 to 1.0 to analyze the impact on optimization result. As shown in Figure 4.12, the oil production optimization result reaches the maximum value when the epsilon is set as 0.5. It is because when epsilon is set as 0.5 in this problem, the algorithm achieves the best balance when choosing the random action and the greedy action, therefore produces the

maximum oil production optimization result with the generated optimal policy. When the epsilon is lower than 0.5, the algorithm relies more on random action and make the method hard to convergence. When the epsilon is higher than 0.5, the algorithm chooses more greedy action and make it hard to improve the accuracy of the Q-value.

Chapter 5

Web-based geographic information system

As introduced before, a web-based GIS called Petroleum Explorer is developed to support the two proposed optimization methods as a data visualization platform for the reservoir engineers working in the oil field. In this chapter, the detail design of the Petroleum Explorer is described. The organization of Chapter 5 is shown as follow. In Section 5.1, the system architecture of the Petroleum Explorer is discussed in detail. In Section 5.2, the database structure of the proposed system is shown with figure. In Section 5.3, the workflow and optimization result of the dynamic programming optimization function are described. Finally, Section 5.4 shows the workflow and result of the Q-learning optimization function.

5.1 System architecture

The proposed Petroleum Explorer system consists of four parts: frontend, backend databased and python server. The architecture of Petroleum Explorer is shown in Figure 5.1.

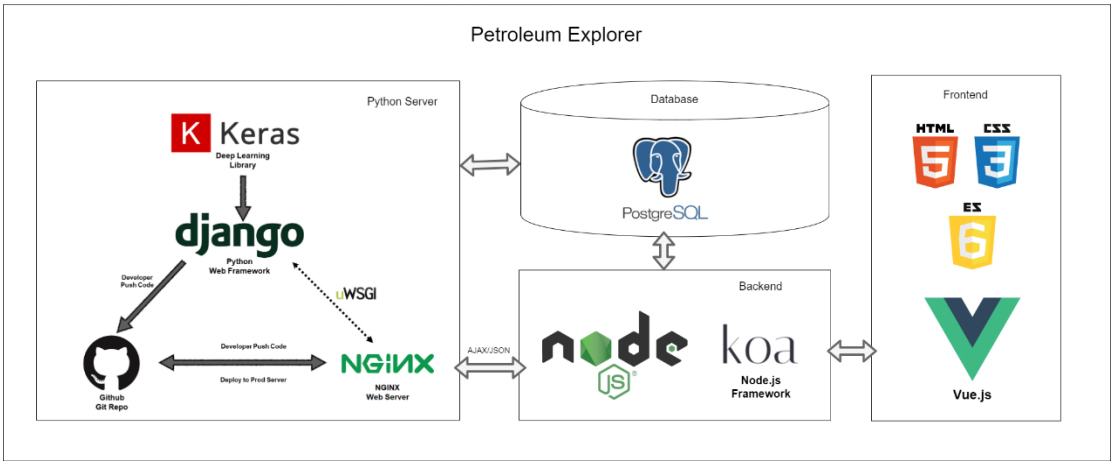


Figure 5.1: The architecture of Petroleum Explorer

The frontend of Petroleum Explorer is developed with Vue.js which is a state-of-art web development framework [41]. To display the website on different platforms without distortion the viewpoint meta tag of the website should be set up properly. The viewpoint of a website is the area which can be viewed by the user, and it varies on the screen size of the device which displays the website [42]. The viewpoint meta tag is the CSS tag to adjust the website to fit the screen size of different devices. In PE, the viewpoint meta tag of the website is set up using the progressive JavaScript Framework Vue.js to automatically transform the resolution of website on a variety of devices [43]. Therefore, Petroleum Explorer can be successfully accessed by different desktops and mobile devices. As shown in Figure 5.2 which displays well data of real world SAGD oil field on the website,

the interface of the Petroleum Explorer includes User Login, Menu, Google Map and Map Legend. User Login is the role-based access control, different role of the system can only access part of the functions provided in the website. Menu shows the functions implemented in the system including Search, Polygon Selection, Data Mining, Data Visualization, Annual Progress Report, Table and Export. The proposed optimization methods is developed under the Date Mining section of the system. Google Map is implemented with JavaScript Google map API, and it is the map view to show the wells data in the oil field. The category of the wells data are distinguished by the Map Legend located at the bottom left corner of the map.

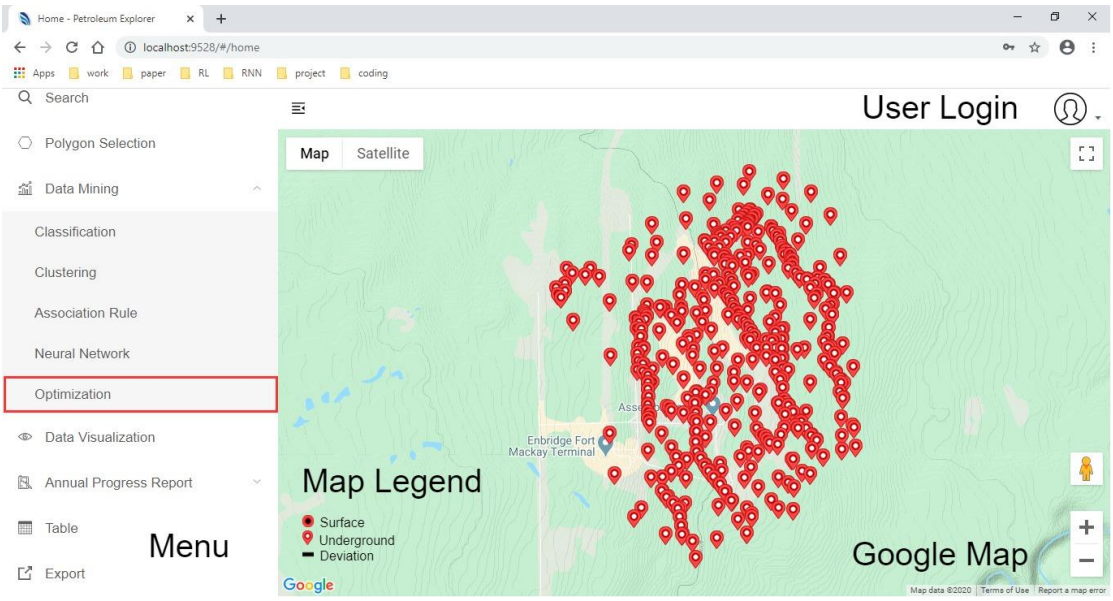


Figure 5.2: Interface of the Petroleum Explorer frontend

The backend of Petroleum Explorer is located in the remote server on the cloud and developed with Node.js and Koa framework using JavaScript. All the data used for the HTTP request are extracted from the database constructed using PostgreSQL. The backend can process all requests sent from the frontend if no

machine learning functions involved in the request. If the request includes the application of machine learning method such as neural network, the request will be further send to the python server for processing. The python server of the system is developed using Django framework. Keras deep learning library is installed on the python server for the setting up of neural network models. The backend communicates with the frontend and python server using HTTP request. Result of the program is then sent back to the frontend and shown on the webpage.

5.2 Database structure

The database of Petroleum Explorer is created using PostgreSQL which contains well data, injection data, production data and statistic data. The structure of database is shown in Figure 5.3. The well data includes spatial and non-spatial data of each well. The spatial data is the geographical location of wells and non-spatial data includes unique well identifier (UWI), well type, well depth, well status and operation company of the field. The injection data includes the injection log of the well such as injection type, injection date and injection volume. The production data includes the production record of the well such as oil production, gas production, water production and production date. The statistic data includes the statistic calculation results of the injection hour, injection volume, production volume and SOR. In the Petroleum Explorer system, data from different categories are related based on the primary key w_id.

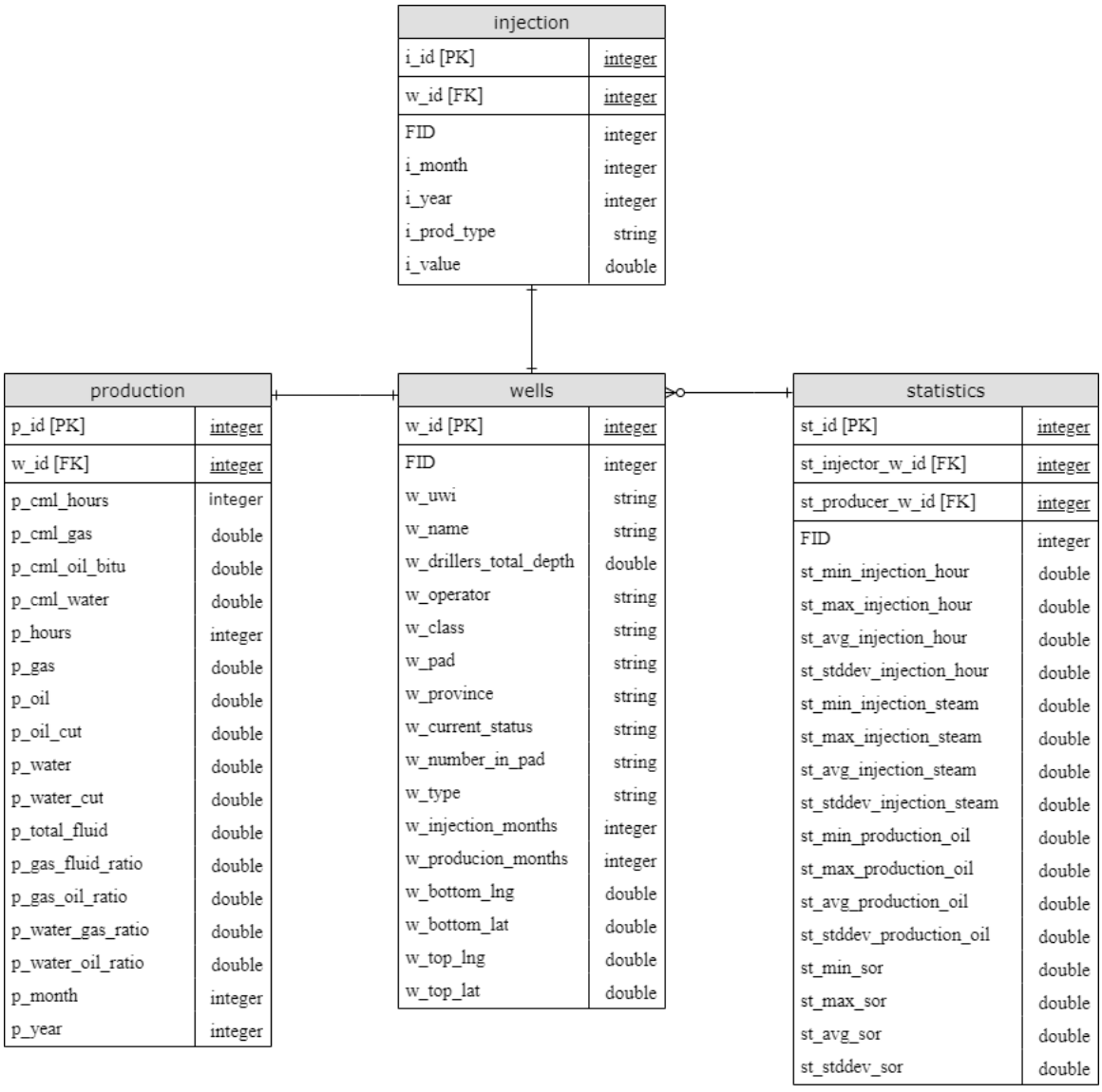


Figure 5.3: The structure of database

5.3 Dynamic programming workflow

The interface of the optimization function in the system is shown in Figure 5.4. The first step of the optimization procedure is to select the oil field and oil pad from the top of the menu. After the selection of field and pad, the next step is

to enter the initial population of timestep and iteration number for the GA. Timestep is the number of steam-oil pairs in the time window to train the prediction model and it plays a critical role for accurate prediction of the LSTM model. With the proper initial value for the GA, the final step of the experiment is to set the optimization parameters. The monthly steam injection volume of each well must be within certain range. Therefore, injection constraints θ_1 and θ_2 which are explained in Section 3 should be properly set by dragging the slider bars on the menu. The steam injection and oil production data of the case study was recorded monthly, so the optimization month range is decided by selecting the start month and end month at the bottom of the menu. Once finishing all the input setting, the experiment can be executed by clicking Run button.

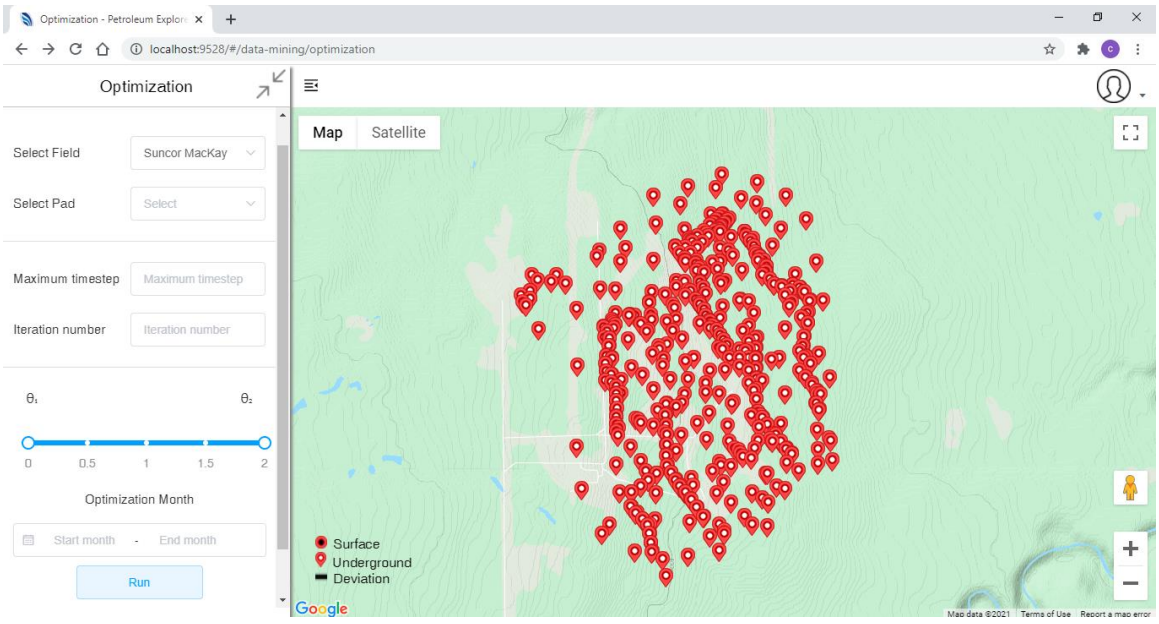


Figure 5.4: Dynamic programming optimization function

The workflow of dynamic programming function is shown in Figure 5.5. After clicking the Run button, the request is sent to the python server through the

backend. The historical data will be pulled out from the database and preprocessed. With the processed data, GA generates the LSTM prediction model with the best timestep for each well. When constructing the LSTM prediction model, eighty percent of the data is used as training data and the other twenty percent is used as testing data. With prediction result of LSTM models, the dynamic programming algorithm optimizes steam injection distribution of the wells. After the algorithm optimizes the steam injection distribution of all the wells, the maximum oil production of the field is obtained. At last, the optimization result is sent back to the frontend and displayed.

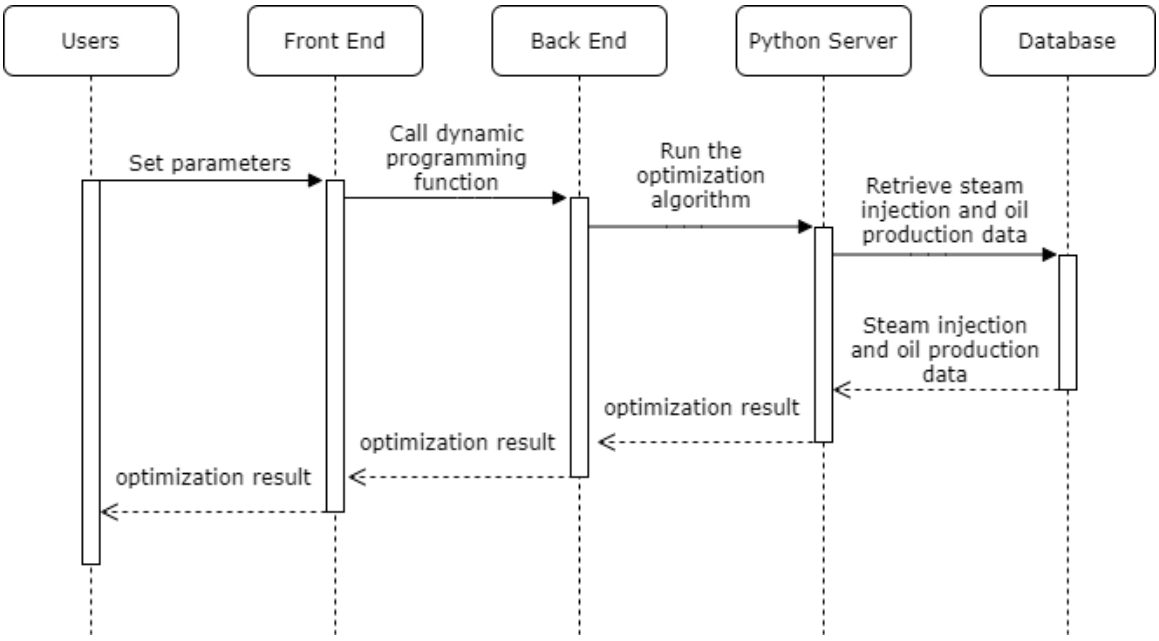
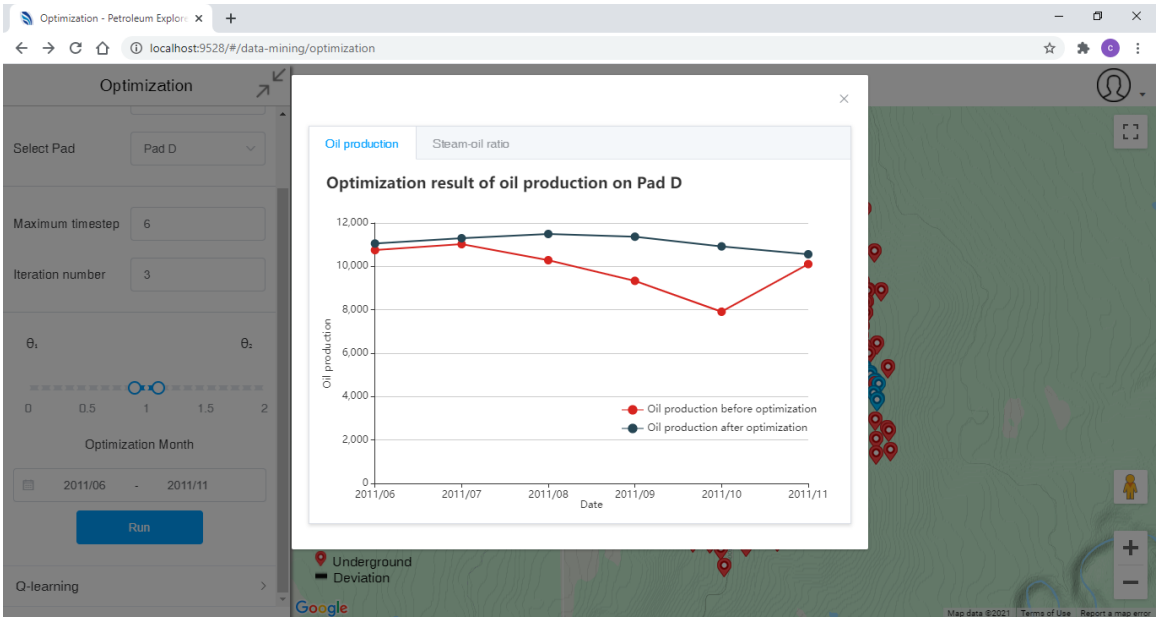


Figure 5.5: The workflow of dynamic programming function

The proposed method is applied to optimize the steam injection of the Pad C and Pad D of Suncor MacKay River oil field from June 2011 to November 2011 with the same amount of injecting steam before optimization. For the parameter

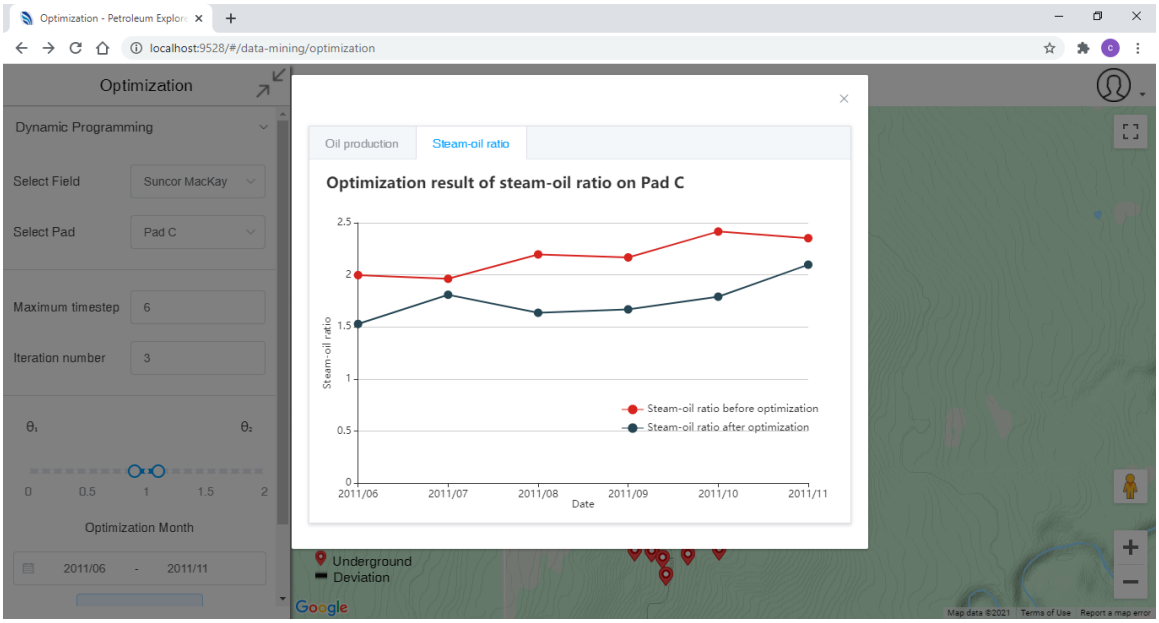
setting of GA, the initial population of timestep is set as 8 to make sure the GA has enough initial population to perform selection and crossover. Number of iterations is set as 3 to select the global optimal timestep values without using excessive time. θ_1 and θ_2 in the formalized problem are set as 0.90 and 1.10 to regulate the minimum and maximum steam injection value of the wells. It means at the current time interval, the steam injection volume of each well is within ± 10% of that at previous time interval. The optimization results of Pad C and Pad D are shown in Figure 5.6 and Figure 5.7.

(a) Pad C

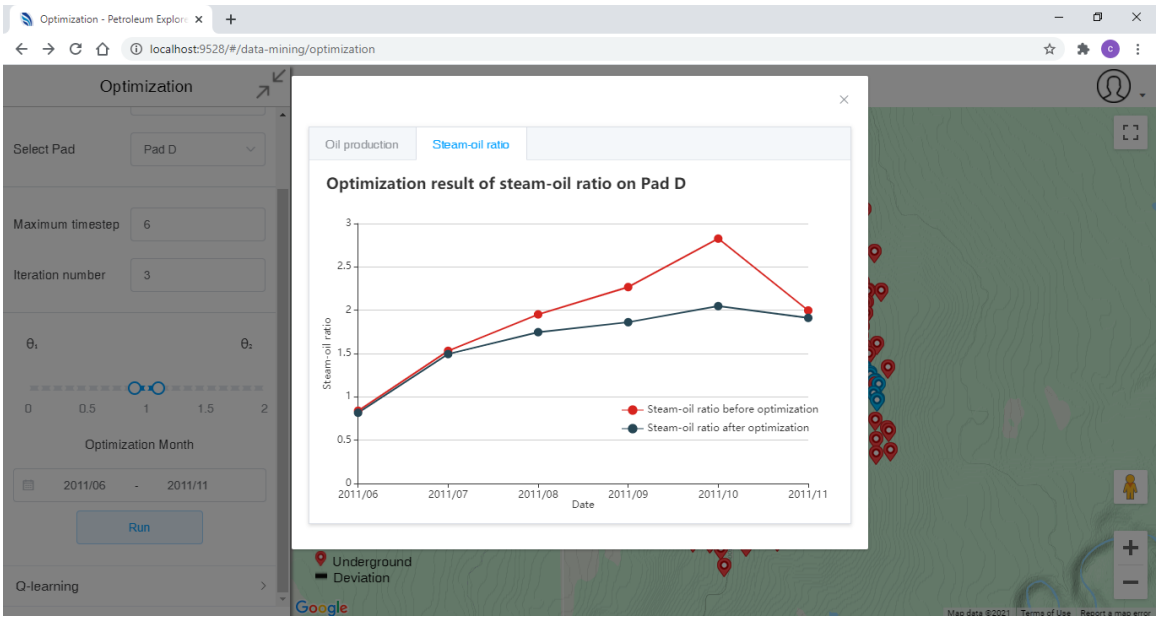


(b) Pad D

Figure 5.6: Result of dynamic programming optimization on oil production



(a) Pad C



(b) Pad D

Figure 5.7: Result of dynamic programming optimization on SOR

As shown in Figure 5.6 and Figure 5.7, from June 2011 to November 2011, oil production of the two pads are compared with the actual field production during that time. It is observed that the monthly oil production is improved and monthly SOR is decreased when the proposed optimization method is applied in the project. Without considering effects from other factors, oil production of the Pad C in the six months is increased by 24.07% which is 27841.20 m^3, oil production of the Pad D in the six month is increased by 12.25% which is 7279.80 m^3. The proposed method has the great potential to optimize the total injecting steam volume of the pads and improve the oil production of the oil field based on the sample operational output of the implementation.

5.4 Reinforcement learning workflow

The interface of the optimization function in the system is shown in Figure 5.8. The first step of the optimization procedure is to select the oil field and oil pad from the top of the menu. After the selection of field and pad, the next step is to set the learning rate, discount factor and epsilon of the experiment. The monthly steam injection volume of each well must be within certain range. Therefore, steam injection limitation should be properly set by dragging the slider bars on the menu. The steam injection and oil production data of the case study was recorded monthly, so the optimization month range is decided by selecting the start month and end month at the bottom of the menu. Once finishing all the input setting, the experiment can be executed by clicking Run button.

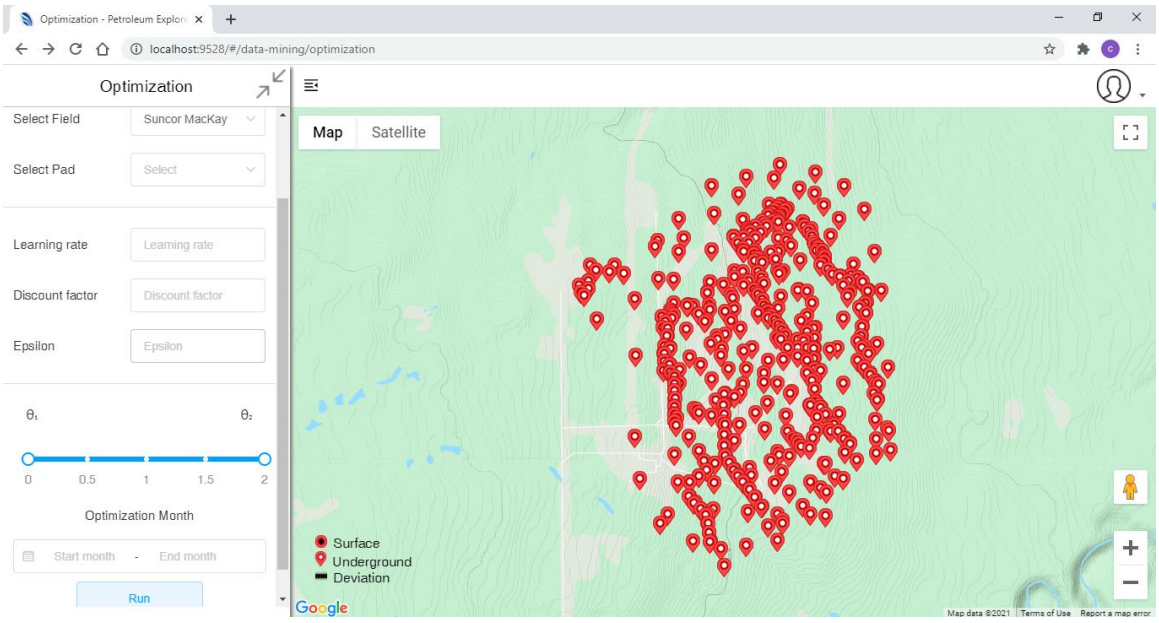


Figure 5.8: Q-learning optimization function

The workflow of Q-learning function is shown in Figure 5.9. After clicking the Run button, the request is sent to the python server through the backend. The

historical data will be pulled out from the database and preprocessed. With the processed data, LSTM prediction model will be constructed using GA. When constructing the LSTM prediction model, eighty percent of the data is used as training data and the other twenty percent is used as testing data. With prediction result of LSTM models, the Q-learning model will be constructed and the state, action, reward of the problem will be formalized. After training the Q-learning model with certain episodes, the steam injection strategy will be obtained and therefore generate the maximum oil production of the well. At last, the optimization result is sent back to the frontend and displayed.

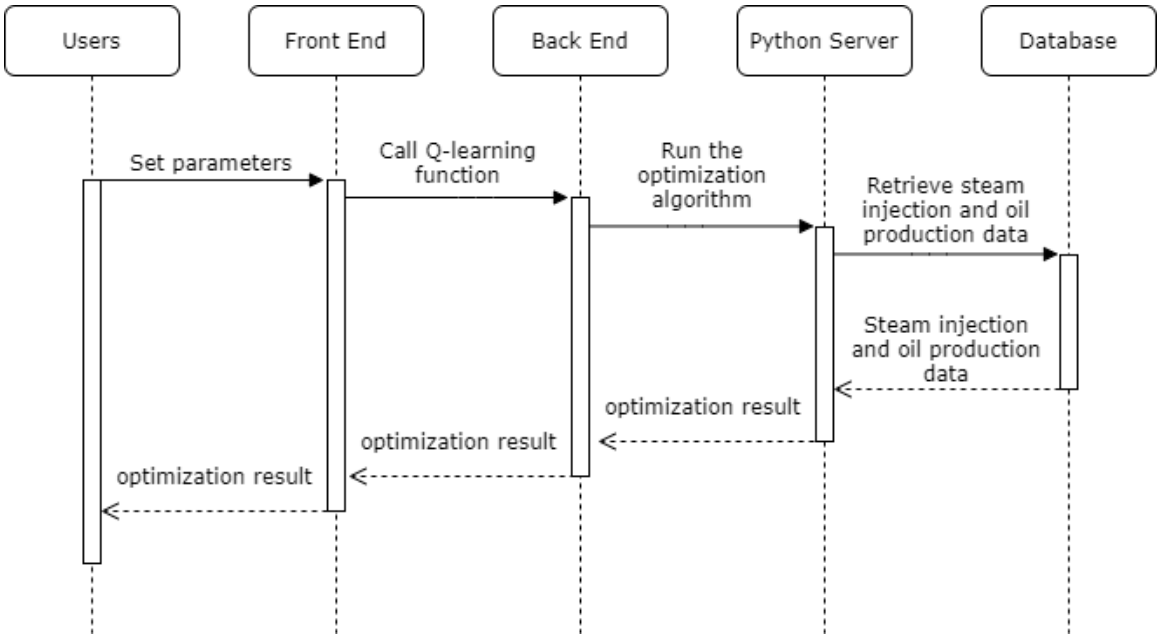
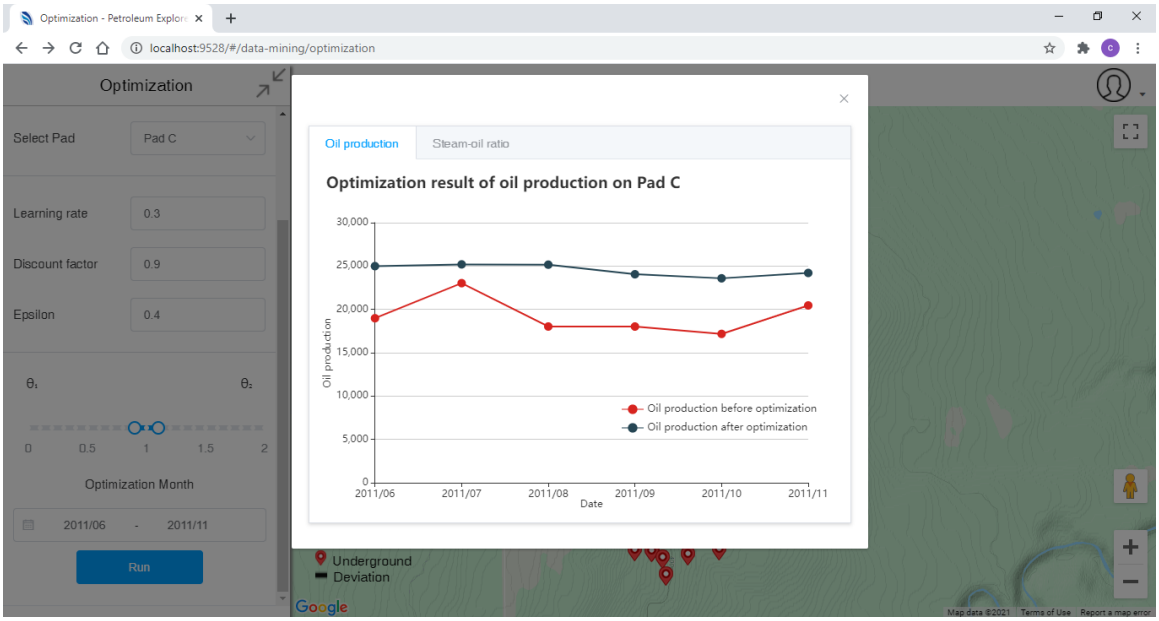


Figure 5.9: The workflow of Q-learning function

The proposed method is applied to optimize the steam injection of the Pad C and Pad D of Suncor MacKay River oil field from June 2011 to November 2011 with the same amount of injecting steam before optimization. The learning rate is

set as 0.4 to control the convergence speed of the algorithm. The discount factor is set as 1 to control the importance of future rewards during the optimization process. The epsilon is set as 0.5 to balance the greedy action and random action when choosing the action to perform. The steam injection limitation is set as ± 10% to regulate the minimum and maximum steam injection amount of the wells. It means at the current time interval, the steam injection volume of each well is within ± 10% of that at previous time interval. The optimization results of Pad C and Pad D are shown in Figure 5.10 and Figure 5.11.



(a) Pad C

(b) Pad D

Figure 5.10: Result of Q-learning optimization on oil production

(a) Pad C

(b) Pad C

Figure 5.11: Result of Q-learning optimization on SOR

As shown in Figure 5.10 and Figure 5.11, from June 2011 to November 2011, oil production of the two pads are compared with the actual field production during that time. It is observed that the performance of Q-learning function is better than dynamic programming function in this case. With the same steam injection limitation as the dynamic programming function, oil production of the Pad C in the six months is increased by 27.23% which is 31505.20 m^3, oil production of the Pad D in the six month is increased by 15.96% which is 9486.20 m^3. The result shows that the Q-learning method achieves better optimization performance than the dynamic programming method.

Chapter 6

Conclusions and future work

6.1 Conclusions

In this thesis, two optimization methods are proposed for the steam injection distribution optimization problem in SAGD oil field to distribute certain amount of steam in SAGD oil field to achieve the maximum cumulative oil production. LSTM prediction models are constructed with historical steam injection data and oil production data, GA is used to select the proper timestep for each LSTM prediction model.

In the first method, with the prediction result of the LSTM model, DP is applied to optimize the steam injection distribution of the oil field to maximize the total oil production. In the second method, state, action and reward of the steam injection distribution optimization problem are formalized under Q-learning scenario. With the prediction result of the LSTM model, Q-learning optimization algorithm is trained to generate the steam injection strategy that produce maximum amount of oil production.

A web-based GIS – Petroleum Explorer is implemented for the engineers in the oil field as a platform to run the two optimization methods on different devices and show the optimization result. The system architecture, database design and workflow of proposed DP method and Q-learning method are also shown in the thesis.

Besides, the experiments conducted in Chapter 3 and Chapter 4 demonstrate following conclusions:

1. To address the motivation of using LSTM prediction model, five comparison prediction models are implemented in the thesis to predict the oil production with the same input data. The proposed LSTM prediction model is constructed with selected timesteps and the LSTM model outperforms other five comparison prediction models.

2. Case studies are implemented in the thesis to compare the performance of the two proposed method. The result shows both dynamic programming method and Q-learning method can improve the oil production of the oil field under different cases. The performance of Q-learning method is better than dynamic programming method in both monthly data experiment and daily data experiment.

3. Different optimization cases are conducted in the experiment to test the effect of parameter setting on the optimization process. The result shows that proper setting of the parameters has a huge impact on the performance of the two proposed methods.

4. Experiments are implemented in Petroleum Explorer using real world SAGD data to show the workflow of two proposed method. Result of case studies indicates the dynamic programming method improves the oil production of the two pads by 24.07% and 12.25%, Q-learning method improves the oil production of the two pads by 27.23% and 15.96%. The experiment further proves the superiority of Q-learning method.

6.2 Future work

There are still several improvements which can be applied in the optimization method but were not explored in the thesis due to time limitation. Some of them are listed below:

1. Oil production depends on both reservoir properties and operations. In this thesis, the LSTM oil prediction model is built only based on the historical steam injection data and oil production data. Our future plans are to take reservoir properties such as porosity, permeability and other operational parameters such as steam pressure and steam temperature into consideration of the LSTM prediction model. Additionally, short-term oil production prediction sometimes is not enough for the purpose of long-term oil field management. For the maximization of long term oil production of the oil field, long term oil production prediction through LSTM neural network or autoencoder model would be explored.

2. When calculating the reward of each action in the Q-learning method, only oil production values are used in the algorithm to provide feedback for the agent. However, in the real-world production process, multiple operational parameters such as porosity, temperature and permeability should also be considered by the engineers when optimizing the steam injection [59]. Butler equation shows the correlation between the oil production rate with many other production factors such as porosity, permeability, thermal diffusivity and viscosity [60]. For the future work of the thesis, Butler equation will be introduced in the Q-learning optimization method to take more factors of the production process when calculating the reward.

3. In Q-learning, the updated Q-values are stored in the Q-table. In this way, Q-learning is applicable to the environment where all possible states can be stored in standard computer RAM memory. However, when the number of states overwhelms the capacity of Q-table, the Q-learning approach will be not feasible. To perform the optimization process in more complex and sophisticated learning environment, state-of-the-art optimization methods should be used in the thesis for the steam injection distribution optimization problem. Deep Q-Network (DQN) is the reinforcement learning method that use neural network to approximate the Q-value function [61]. In DQN, the next action is determined by the maximum output of the prediction network. To avoid the divergence when updating the DQN, two separate neural networks are used to

calculate the predicted value and the target value. The target network has the same architecture as the prediction network but with frozen parameters. For every few iterations, the parameters from the prediction network are copied to the target network. It leads to more stable training because it keeps the target function fixed for a while. In this way, no table structure will be used in DQN and the Q-value will be stored in two neural network models. Therefore, DQN can optimize the steam injection distribution optimization problem in a more complex and sophisticated environment.

4. To improve the efficiency of the software system, the application of high-performance software platform will be explored and implemented. For example, Apache Spark is a distributed working platform for big data processing, which uses in-memory caching and optimized query execution to process fast queries [48]. When running the algorithm on Apache Spark platform, the platform requires a cluster manager and a distributed storage system to accelerate the processing speed. For future work of this thesis, Apache Spark platform could be applied in the experiments to speed up the optimization process.

Bibliography

[1] Government of Alberta [Internet]. Oil sands facts and statistics. [cited 2020 Apr 23]. Available from: https://www.alberta.ca/oil-sands-facts-and-statistics.aspx

[2] Government of Alberta [Internet]. Oil sands 101. [cited 2020 Apr 23]. Available from: https://www.alberta.ca/oil-sands-101.aspx

[3] JWN Energy [Internet]. Temporarily shutting in oilsands SAGD wells might not be as harmful as you think. [cited 2019 Jun 12]. Available from: https://www.jwnenergy.com/article/2016/5/temporarily-shutting-oilsands-sagd-wells-might-not-be-harmful-you-think/

[4] Jimenez J. The field performance of SAGD projects in Canada. In: International Petroleum Technology Conference. 2008; Kuala Lumpur, Malaysia. 4: p. 2645-2656.

[5] Dong XH, Liu HQ, Chen ZX, Qi P. An empirical correlation to predict the SAGD recovery performance. Journal of Computational Methods in Sciences and Engineering. 2017. 17(3): p. 333-345.

[6] Liu H, Cheng LS, Huang SJ, Jia P, Chen MQ. Evolution characteristics of SAGD steam chamber and its impacts on heavy oil production and heat

consumption. International Journal of Heat and Mass Transfer. 2018. 121: p. 579-596.

[7] Ian DG, Stephen RL. Energy efficiency and emissions intensity of SAGD. Fuel. 2014. 115: p. 706-713.

[8] Ho SL, Xie M. The use of ARIMA models for reliability forecasting and analysis. Computers & Industrial Engineering. 1998. 35(1–2): p. 213-216.

[9] Mohammad A, Mohammad E, Amin S, Seyed MJM. Evolving artificial neural network and imperialist competitive algorithm for prediction oil flow rate of the reservoir. Applied Soft Computing. 2013. 13(2): p. 1085-1098.

[10] Zhang C, Zhou J, Li C, Fu W, Peng T. A compound structure of ELM based on feature selection and parameter optimization using hybrid backtracking search algorithm for wind speed forecasting. Energy Conversion and Management. 2017. 143: p. 360-376.

[11] Yona A, Senjyu T, Funabashi T. Application of recurrent neural network to short-term-ahead generating power forecasting for photovoltaic system. In: 2007 IEEE Power Engineering Society General Meeting. 2007; Tampa, FL, USA. 1: p. 1-6.

[12] Liu H, Tian HQ, Liang XF, Li YF. Wind speed forecasting approach using secondary decomposition algorithm and Elman neural networks. Apply Energy. 2015. 157: p. 183-194.

[13] Nguyen HH, Chan CW. Applications of data analysis techniques for oil production prediction. Engineering Applications of Artificial Intelligence. 2005. 18(5): p. 549-558.

[14] Cho K, Van MB, Gulcehre C. Learning phrase representations using RNN encoder-decoder for statistical machine translation. In: The 2014 Conference on Empirical Methods on Natural Language Processing. 2014; Doha, Qatar. 1: p. 1724-1734.

[15] Shin H, Polikar M. Optimizing the SAGD process in three major Canadian oil-sands areas. In: SPE Annual Technical Conference and Exhibition. 2005; Dallas, Texas, USA. 3: p. 1389-1400.

[16] Yuan Z, Liu PC, Zhang SF, Li XL, Shi LX, Jin RF. Experimental study and numerical simulation of nitrogen-assisted SAGD in developing heavy oil reservoirs. Jouranl of Petroleum Science and Engineering. 2018. 162: p. 325-332.

[17] Ray T, Sarker R. Multiobjective Evolutionary Approach to the Solution of Gas Lift Optimization Problems. In: 2006 IEEE International Conference on Evolutionary Computation. 2006; Vancouver, BC, Canada. 1: p. 3182–3188.

[18] Hamedi H, Rashidi F, Khamehchi E. A Novel Approach to the Gas-Lift Allocation Optimization Problem. Petroleum Science and Technology. 2011. 29(4): p. 418-427.

[19] Divestco [Internet]. [cited 2020 Feb 23]. Available from: http://www.divestco.com/

[20] HIS [Internet]. [cited 2020 Feb 23]. Available from: http://www.ihs.com/

[21] Hochreiter S, Schmidhuber J. Long short-term memory. Neural Computation. 1997. 9(8): p. 1735-1780.

[22] Liu S, Liao G, Ding Y. Stock transaction prediction modeling and analysis based on LSTM. In: 2018 13th IEEE Conference on Industrial Electronics and Applications. 2018; Wuhan, China. 1: p. 2787-2790.

[23] Kalaycı S, Arslan S. A dynamic programming based optimization approach for appointment scheduling in banking. In: 2017 International Conference on Computer Science and Engineering. 2017; Antalya, Turkey. 1: p. 625-629.

[24] Giegerich R, Meyer C, Steffen P. A discipline of dynamic programming over sequence data. Science of Computing Science. 2004. 51(3): p. 215-263.

[25] Zhu W, Li R. Research on dynamic timetables of bus scheduling based on dynamic programming. In: Proceedings of the 33rd Chinese Control Conference. 2014; Nanjing, China. 1: p. 8930-8934.

[26] Yusof NM, Rashid RSA, Mohamed Z. Malaysia crude oil production estimation: an application of ARIMA model. In: 2010 International Conference on Science and Social Research. 2010; Kuala Lumpur, Malaysia. F32: p. 1255-1260.

[27] Berneti S, Shahbazian M. An imperialist competitive algorithm artificial neural network method to predict oil flow rate of the wells. International Journal of Computer Applications. 2011. 26(10): p. 47-50.

[28] Selvin S, Vinayakumar R, Gopalakrishnan EA, Menon VK, Soman KP. Stock price prediction using LSTM, RNN and CNN-sliding window model. In: 2017 International Conference on Advances in Computing, Communications and Informatics (ICACCI). 2017; Udupi, India. 1: p. 1643-1647.

[29] Adam K, Smagulova K, James AP. Memristive LSTM network hardware architecture for time-series predictive modeling problems. In: 2018 IEEE Asia Pacific Conference on Circuits and Systems. 2018; Chengdu, China. 1: p. 459-462.

[30] Liu W, Liu WD, Gu J. Forecasting oil production using ensemble empirical model decomposition based Long Short-Term Memory neural network. Journal of Petroleum Science and Engineering. 2020. 189: 107013.

[31] Chen MR, Zeng GQ, Lu KD, Weng J. A Two-Layer Nonlinear Combination Method for Short-Term Wind Speed Prediction Based on ELM, ENN, and LSTM. IEEE Internet of Things Journal. 2019. 6(4): p. 6997-7010.

[32] Hu YL, Chen L. A nonlinear hybrid wind speed forecasting model using LSTM network, hysteretic ELM and Differential Evolution algorithm. Energy Conversion and Management. 2018. 173: p. 123-142.

[33] Namdar H. Developing an improved approach to solving a new gas lift optimization problem. Journal of Petroleum Exploration and Production Technology. 2019. 9: p. 2965–2978.

[34] Sadollah A, Eskandar H, Bahreininejad A, Kim JH. Water cycle algorithm with evaporation rate for solving constrained and unconstrained optimization problems. Applied Soft Computing. 2015. 30: p. 58-70.

[35] Ameli F, Mohammadi K. A novel optimization technique for Fast-SAGD process in a heterogeneous reservoir using discrete variables and repetition inhibitory algorithm. Journal of Petroleum Science and Engineering. 2018. 171: p. 982-992.

[36] Ma ZW, Leung JY. Design of warm solvent injection processes for heterogeneous heavy oil reservoirs: A hybrid workflow of multi-objective optimization and proxy models. Journal of Petroleum Science and Engineering. 2020. 191: 107186.

[37] Jankauskas K, Papageorgiou LG, Farida SS. Fast genetic algorithm approaches to solving discrete-time mixed integer linear programming problems of capacity planning and scheduling of biopharmaceutical manufacture. Computers & Chemical Engineering. 2019. 121: p. 212-223.

[38] Chen MR, Zeng GQ, Lu K. A many-objective population extremal optimization algorithm with an adaptive hybrid mutation operation. Information Sciences. 2019. 498: p. 62-90.

[39] Hansen EA, Bernstein DS, Zilberstein S. Dynamic programming for partially observable stochastic games. In: AAAI'04: Proceedings of the 19th national conference on Artifical intelligence. 2004; San Jose, California, USA. 1: p. 709-715.

[40] Li XH, Hui N, Shen HF, Fu YJ, Zhang LP. A robust mosaicking procedure for high spatial resolution remote sensing images. ISPRS Journal of Photogrammetry and Remote Sensing. 2015. 109: p. 108-125.

[41] Vue [Internet]. [cited 2020 Feb 23]. Available from: https://vuejs.org/index.html/

[42] CSS Day [Internet]. A tale of two viewports — part two. [cited 2020 Feb 23]. Available from: https://www.quirksmode.org/mobile/viewports2.html

[43] Google developers [Internet]. Has A Viewport Meta Tag With width Or initial-scale. [cited 2020 Feb 23].
Available from:
https://developers.google.com/web/tools/lighthouse/audits/has-viewport-meta-tag/

[44] Suncor Energy [Internet]. 2016. Suncor MacKay River Project 2016 AER Performance Presentation. Calgary, Canada.

[45] Alberta Energy Regulator [Internet]. [cited 2020 Aug 10]. Available from: https://www.aer.ca/providing-information/data-and-reports/activity-and-data.html

[46] Low SE, Ong P, Cheah KC. Solving the optimal path planning of a mobile robot using improved Q-learning. Robotics and Autonomous Systems. 2019. 115: p. 143-161.

[47] Chai S, Lau VKN. Joint Rate and Power Optimization for Multimedia Streaming in Wireless Fading Channels via Parametric Policy Gradient. IEEE Transactions on Signal Processing. 2019. 67(17): p. 4570-4581.

[48] Apache Spark [Internet]. [cited 2020 Jun 28]. Available from: http://spark.apache.org/

[49] Shimokawa T, Suzuki K, Misawa T, Okano Y. Predicting investment behavior: An augmented reinforcement learning model. Neurocomputing. 2009. 72(16-18): p. 3447-3461.

[50] Luo B, Liu D, Huang T, Wang D. Model-Free Optimal Tracking Control via Critic-Only Q-Learning. IEEE Transactions on Neural Networks and Learning Systems. 2016. 27(10): p. 2134-2144.

[51] Wei Q, Liu D, Shi G. A novel dual iterative Q-learning method for optimal battery management in smart residential environments. IEEE Transactions on Industrial Electronics. 2014. 62(4): p. 2509 – 2518.

[52] Guevara JL, Patel RG, Trivedi JJ. Optimization of Steam Injection for Heavy Oil Reservoirs Using Reinforcement Learning. In: SPE International Heavy Oil Conference and Exhibition. 2018; Kuwait City, Kuwait. SPE-193769-MS.

[53] Liang L, Ye H, Li GY. Spectrum Sharing in Vehicular Networks Based on Multi-Agent Reinforcement Learning. IEEE Journal on Selected Areas in Communications. 2019. 37(10): p. 2282-2292.

[54] Bellman R. A Markovian Decision Process. Journal of Mathematics and Mechanics. 1957. 6(5): p. 679–684.

[55] Bellman R. The Theory of Dynamic Programming. RAND Corporation. 1954. P-550.

[56] Xu Y, Xia J, Wu H, Fan L. Q-Learning Based Physical-Layer Secure Game Against Multiagent Attacks. IEEE Access. 2019. 7: p. 49212 – 49222.

[57] Liu T, Wang B, Yang C. Online Markov Chain-based energy management for a hybrid tracked vehicle with speedy Q-learning. Energy. 2018. 160: p. 544-555.

[58] Yu KH, Chen YA, Jaimes E, Wu WC, Liao KK, Liao JC, Lu KC, Sheu WJ, Wang CC. Optimization of thermal comfort, indoor quality, and energy-saving in campus classroom through deep Q learning. Case Studies in Thermal Engineering. 2021. 24: 100842.

[59] Butler RM, Mcnab GS, Lo HY. Theoretical studies on the gravity drainage of heavy oil during in-situ steam heating. In: The 29th Canadian Chemical Engineering Conference. 1979; Ontario, Canada. 59(4): p. 455-460.

[60] Butler RM. A New Approach To The Modelling Of Steam-Assisted Gravity Drainage. Journal of Canadian Petroleum Technology. 1985. 24(3): p. 42-51.

[61] Pan J, Wang X, Cheng Y, Yu Q. Multisource Transfer Double DQN Based on Actor Learning. IEEE Transactions on Neural Networks and Learning Systems. 2018. 29(6): p. 2227 – 2238.

[62] Sutton RS, Barto AG. Reinforcement Learning: An Introduction. 2018. Cambridge, MA, USA.

Appendix

Publication during the program

Yang Changlin, Wang Xin. A steam injection distribution optimization method for SAGD oil field using LSTM and dynamic programming. ISA Transactions. 2020.