Sequential decision making using Bayesian networks and reinforcement learning

to determine the hydrocarbon prospect exploration strategy

Laura Veerhoek

MSc Thesis



TU Delft

equinor

University of Stavanger

SEQUENTIAL DECISION MAKING USING BAYESIAN NETWORKS AND REINFORCEMENT LEARNING TO DETERMINE THE HYDROCARBON PROSPECT EXPLORATION STRATEGY

by

Laura Pauline Veerhoek

July 2022

to obtain the degree of Master of Science in Applied Mathematics
at the Delft University of Technology,
to be defended publicly on Thursday July 7, 2022 at 03:30 PM.

Project duration:	December 1, 2021 - July 7, 2022	
Thesis committee:		
Dr. D. Kurowicka	Delft University of Technology	Chair
Dr. Ir. G. F. Nane	Delft University of Technology	Supervisor
Dr. Ir. L.J.J. van Iersel	Delft University of Technology	Committee member
Prof. Dr. R. B. Bratvold	University of Stavanger	External supervisor
Dr. R.G. Hanea	Equinor	External supervisor

An electronic version of this thesis is available at http://repository.tudelft.nl/.
Copyright ©2022 by Laura Veerhoek. All rights reserved.

Cover image: photo taken by Jan-Rune Smenes Reite







ABSTRACT

In the process of drilling wells to produce hydrocarbons, an exploration strategy is used to determine which wells should be drilled and in which order. This strategy is vital, as a suboptimal drilling sequence will lead to more expenses and fewer gains. Furthermore, the wells considered in most exploration strategies are geologically dependent. Thus, a realistic model of these dependencies will be beneficial and contribute to a more reliable optimal drilling strategy.

Previous research has shown that modelling similarities between the geological properties of prospect wells in the same region and updating the drilling strategy dynamically after more information is available can add much value. However, the currently developed models are not realistic enough to use in practice. Previous models separated the process into two parts, first modelling the geological success with deterministic rewards and then applying dynamic programming to obtain the sequential drilling strategy.

This thesis proposes a new model with the addition of flexible uncertainty and dependency models between the geological characteristics in the hydrocarbon volume calculation of a reservoir. For this purpose, non-parametric Bayesian networks in combination with copulae are used. This model and the input format of the required data align with how data is gathered in the oil and gas industry. The dependencies are modelled based on simple expert assessments.

After the new model for the geological properties of prospect wells and their dependencies is constructed, a decision policy that hinges on the observations of previously drilled wells is constructed. Previously used dynamic programming becomes infeasible with continuous variables and with a large number of wells. Therefore, the reinforcement learning algorithm Q-learning is used, which is better equipped to handle these more complex models.

The analysis shows that Q-learning achieves similar results to dynamic programming. Afterwards, the new model is applied to a case based on real oil and gas industry data. The results show that this case does not lend itself to an improvement when the drilling strategy is sequentially updated as the decisions compared to the original drilling strategy did not change. This occurs because the prospects all have high positive expected rewards and should be drilled, regardless of the other wells' outcomes and dependencies.

Another case with synthetic data is explored with more favourable characteristics that, in theory, should lead to the result of the policy from the new model and Q-learning outperforming the current industry policy. However, this was not shown yet within the time constraints of this thesis, as the Q-learner should be run for a more extended period and tuned according to the case.

While the model is created for dynamic sequential decision making in the application of drilling wells, this method can easily be adapted to fit any other application where decisions should be updated sequentially, and the dependencies can be modelled through a Bayesian network.

In conclusion, the key contributions of this thesis are the creation of a new model for the dependencies between geological properties of a hydrocarbon reservoir and the use of the Q-learning reinforcement learning algorithm to compute the sequentially updated drilling strategy as an alternative to dynamic programming.

PREFACE

This master thesis presents my research results for sequential decision analysis using non-parametric Bayesian networks and reinforcement learning in the application of drilling hydrocarbon prospects.

I want to express my gratitude to everyone who helped me while working on my thesis. There are a few people I would like to mention here specifically.

First and foremost, I am incredibly grateful to my supervisor Gabriela Nane for her time, invaluable advice, and continuous support during my thesis. Without a doubt, she was always there for me and made time in her busy schedule to meet with me whenever I asked.
Next, I wish to thank Reidar Bratvold for all his effort and input during all of our meetings. His extensive oil and gas industry expertise was instrumental in helping me understand how the hydrocarbon drilling process takes place and how I can incorporate that knowledge into my model to make it more realistic.
I want to express my gratitude to my third supervisor, Remus Hanea, for his help and input along the way. Through him I got a real insight into the practices of drilling wells in the industry.
Furthermore, I also appreciate the expert input from Kjersti Breirem Neumann, who is employed as the lead advisor for prospect evaluation at Equinor. Her knowledge and willingness to explain the concepts of hydrocarbon exploration strategies within Equinor to me made her a very valuable expert.
Additionally, I would like to thank Dan Ababei for his technical help setting up the connection between R and Uninet as the software developer of Uninet.
I thank all my thesis committee members for taking time out of their day to judge my report and presentation. In particular, I would like to thank Dorota Kurowicka for chairing my thesis committee.
Finally, my appreciation goes to my friends and family for their emotional support during my thesis.

Laura Veerhoek
Delft, July 2022

CONTENTS

Abstract iii
Preface iv
1 INTRODUCTION 1
2 APPLICATION BACKGROUND THEORY 3
2.1 Problem setting . . . 3
2.2 Model to estimate the amount of recovered hydrocarbons . . . 3
3 RELATED WORK 5
4 THEORY 7
4.1 Bayesian network . . . 7
4.1.1 Types of nodes in a Bayesian network . . . 7
4.1.2 Structure of a Bayesian network . . . 7
4.2 Copulae . . . 9
4.2.1 Bivariate Gaussian copulae . . . 9
4.2.2 Learning correlation parameters from data . . . 10
4.2.3 Other applications of copulae . . . 11
4.3 Optimisation procedures . . . 11
4.3.1 Dynamic programming . . . 11
4.3.2 Q-learning . . . 12
5 METHODOLOGY 15
5.1 Combining previous work . . . 15
5.1.1 Discrete Bayesian network . . . 16
5.1.2 Dynamic programming . . . 17
5.2 Q-learning and discrete Bayesian network . . . 18
5.3 Extend model to more realistic situation and to fit real data . . . 19
5.3.1 Extended Bayesian network . . . 20
5.3.2 Data . . . 21
5.3.3 Q-learning on the extended model . . . 22
5.3.4 Assumptions for the model and sequential optimisation algorithm . . . 22
5.4 Extra synthetic data case . . . 24
5.5 Sensitivity analysis . . . 24
5.6 Implementation . . . 24
6 DISCRETE BAYESIAN NETWORK MODEL: DYNAMIC PROGRAMMING 25
6.1 Results from dynamic programming on three-well problem . . . 25
6.2 Results from dynamic programming on five-well problem . . . 25
6.3 Computational time analysis . . . 26
7 VALIDATION OF Q-LEARNING 29
7.1 Results from Q-learning with initial hyperparameters . . . 29
7.1.1 Comparison to dynamic programming policy . . . 29
7.1.2 Expected revenue . . . 30
7.1.3 Computational time . . . 30
7.2 Parameter tuning . . . 32
7.2.1 Reset after each episode . . . 32
7.2.2 Initialisation . . . 33
7.2.3 Epsilon . . . 34
7.2.4 Learning rate . . . 34
7.2.5 Results after tuning . . . 36
7.3 Remembering the best policy thus far . . . 38
7.4 Comparison of dynamic programming and Q-learning . . . 39
7.5 Five-well problem . . . 40

8 RESULTS OF APPLYING THE EXTENDED MODEL TO REAL DATA 42
9 EXTENDED MODEL ON SYNTHETIC DATA CASE 45
10 SENSITIVITY ANALYSIS ON DISCRETE NODES IN UNINET 47
11 CONCLUSION 49
12 DISCUSSION 51

Appendices

A PROBABILITIES USED IN THE THREE AND FIVE WELL PROBLEM TO VALIDATE Q-LEARNING 57
B CONSTRUCTING JOINT DISTRIBUTION 58
C RESULTS OF TUNING 61
D EXTENDED BAYESIAN NETWORK 66
E DATA 69
F SYNTHETIC DATA CASE 74

LIST OF FIGURES

Figure 4.1 Example of conditionally independent variables 8
Figure 4.2 Flowchart of the training process of Q-learning 13
Figure 5.1 Visualisation of the workflow of previous work of Bickel and Smith [2006] 15
Figure 5.2 Visualisation of the workflow of previous work of Martinelli et al. [2014] 15
Figure 5.3 Visualisation of the workflow of the first step in combining Bickel et al. [2008] and Martinelli et al. [2014] 16
Figure 5.4 Bayesian network structure illustration for five wells and three geological factors. 17
Figure 5.5 Visualisation of the workflow after including reinforcement learning 19
Figure 5.6 Visualisation of the workflow after including reinforcement learning and extending the model 20
Figure 5.7 Bayesian network structure for the extended model for three wells with three geological factors and six volume parameters. 21
Figure 5.8 Bayesian network structure for the extended model for three wells with two geological factors and six volume parameters of which the dependencies between three of them are considered. 23
Figure 6.1 The computational times against the number of factors for the three-well problem 27
Figure 6.2 The computational times against the number of wells for the situation where only one geological success factor is considered 28
Figure 7.1 performance metrics throughout the training process on the three-well problem with initial parameters. 31
Figure 7.2 Reset parameter tuning: performance metrics throughout the training process on the three-well problem. 33
Figure 7.3 Initialisation tuning: performance metrics throughout the training process on the three-well problem. 34
Figure 7.4 Learning rate tuning: performance metrics throughout the training process on the three-well problem. 35
Figure 7.5 Learning rate tuning (exponential decay vs constant): performance metrics throughout the training process on the three-well problem. 36
Figure 7.6 performance metrics throughout the training process on the three-well problem with tuned parameters. 37
Figure 7.7 Expected revenue based on the policy with regular Q-learning and with Q-learning where the best policy is remembered. . 38
Figure 7.8 Performance metrics throughout the training process with tuned parameters on the five-well problem. 41
Figure 8.1 Bayesian network structure for the extended model for three wells with three geological factors and six volume parameters of which the dependencies between three of them are considered. 42
Figure 8.2 Performance during the training process of Q-learning on the real data model with parameters as specified in Table 4.1 . . . 44
Figure 9.1 Performance during the training process of Q-learning on the extended model on the synthetic data case. 46

Figure 10.1 Bayesian network in Netica 47
Figure 10.2 Bayesian network in Uninet 47
Figure C.1 Reset parameter tuning: performance metrics throughout the learning process with parameters: epsilon = 0.1, learning rate = 0.2, optimistic initialisation. 61
Figure C.2 Initialisation parameter tuning: performance metrics throughout the learning process with parameters: epsilon = 0.1, learning rate = 0.2, random reset. 62
Figure C.3 Epsilon parameter tuning: performance metrics throughout the learning process with parameters: learning rate = 0.2, optimistic initialisation, random reset. 63
Figure C.4 Learning rate parameter tuning: performance metrics throughout the learning process with parameters: epsilon = 0, optimistic initialisation, random reset. 64
Figure C.5 Learning rate parameter tuning (exponential decay vs constant): performance metrics throughout the learning process with parameters: epsilon = 0, optimistic initialisation, random reset. 65
Figure D.1 Bayesian network for the extended model (full page view) . . 67
Figure D.2 Bayesian network structure for the extended model for the real data case (full page view). 68
Figure E.1 Geographical relationship between the six wells in the data set 72

LIST OF TABLES

Table 4.1 Example of Q-table 11
Table 4.2 Possible choices for the hyperparameters in Q-learning . . . 14
Table 7.1 Initial choices for parameters in Q-learning 29
Table 7.2 Tuned choices for parameters in Q-learning 36
Table 8.1 Hyperparameters for the real data case 42
Table 10.1 CPT of three-well fully connected network from Netica 48
Table 10.2 CPT of three-well fully connected network from Uninet 48
Table A.1 Marginal probabilities of success of the three geological factors in the five well problem. 57
Table A.2 Pairwise condition probabilities of success for the geological factor source P(S_i= success $|S_j$= success) 57
Table A.3 Pairwise condition probabilities of success for the geological factor reservoir P(R_i= success $|R_j$= success) 57
Table A.4 Pairwise condition probabilities of success for the geological factor trap P(T_i= success $|T_j$= success) 57
Table D.1 The functional node functions in extended Bayesian network 66
Table E.1 Data for the success probabilities of the geological factors. . . 69
Table E.2 Data for the volume parameters (Part 1) 70
Table E.3 Data for the volume parameters (Part 2) 71
Table E.4 Data for the recovery factor parameter 71
Table E.5 Data for dependencies between wells (descriptions) 72
Table E.6 Data for dependencies between wells (correlations) 72
Table E.7 Costs of drilling the six wells in the data set 73
Table E.8 Outcome of drilling the six wells in the data set 73
Table F.1 Marginal distributions for synthetic data case 74

1 INTRODUCTION

Choosing a drilling strategy is a crucial step in the decision analysis process of hydrocarbon exploration. Currently, the drilling strategy is determined far in advance and is based on pre-drill estimates for each prospect separately. Only prospects expected to have a high positive outcome are drilled during the exploration process. The outcome of a prospect is considered a success when they contain hydrocarbons that can be successfully extracted and thus result in positive revenue. A prospect is regarded as a failure if it does not contain hydrocarbons or the reservoir is faulty in some way that the hydrocarbons cannot be extracted. The order in which the prospects are drilled depends mainly on the prospects' geological location and what route the drilling rig will take both before and after completing the drills in a particular region. This drilling strategy is not altered along the way, regardless of the outcome of the prospects drilled already. However, prospects often lie in the same geological area and thus share similar geological properties. This information can be used to update the estimates for prospects that have not yet been drilled, which can be advantageous in many ways. Firstly, if the prospects in an area are a success, the probability of success in other prospects in the same area may increase, and uncertainty may be reduced. It can also lead to the inclusion of prospects not considered in the initial drilling strategy because they now have a higher expected value and have become more interesting to drill. In addition, if the wells fail, the probability of success in other wells that share similar characteristics, according to the data, may be reduced. This could result in the prospects planned for drilling at a later stage not being drilled. Thus, not unnecessarily drilling wells with a low reward or loss and a significant environmental impact. So, capturing and using this information would be valuable in making drilling wells more profitable and environmentally friendly.

The status quo in drilling strategy in the oil and gas industry, according to experts in the field, is not to consider the dependencies between prospects and to not update the drilling strategy dynamically after observing the outcomes of previously drilled wells. Nevertheless, previous research has shown its considerable value. Bickel and Smith [2006] and Bickel et al. [2008] have shown that considering the dependencies between wells in a geologically dependent region generates profit, even from an area where the wells are initially expected to be non-profitable. This was done by constructing a joint distribution for the geological factors that influence a prospect's success through expert assessments. This joint distribution was exploited using dynamic programming to compute which prospect to drill next, given the information of previously drilled wells. While this research covers the set-up of the problem very nicely, it is simplified to such an extent that it is too unrealistic for the industry to use in practice. This is because, for example, they cannot include a large number of wells and do not include uncertainty in calculating the expected recovered hydrocarbon volume. Martinelli et al. [2014] extended this method by using a discrete Bayesian network to model the dependencies between the success factors of the prospects. This allowed for more complex dependency relationships between prospects due to the flexibility of the network structure. They elicited expert opinion to fill the conditional probability tables required in a discrete Bayesian network. This is a more extensive expert elicitation process, making the model difficult to use in practice as such a detailed assessment of the dependencies may not be available.

This thesis aims to build on previous research and create a model that combines the current practices of using extensive pre-drill estimates in the oil and gas industry with a model that accounts for the geological dependencies between prospects in the same region. The model aims to be realistic and practical enough to be utilised by energy companies effectively. Nevertheless, it should be mentioned that the model can easily be adapted to solve any other sequential decision making problem.
For this purpose, non-parametric Bayesian networks and copulae are used to model the dependencies within a geological region of prospects. This allows for adding continuous parameters and their interdependencies that influence the volume of the hydrocarbons recovered from a prospect. Converting these parameters that in previous research were modelled to be deterministic as uncertain non-parametric distribution is more in line with the current practices of the pre-drill estimates in the oil and gas industry. This way, both whether the well was a success and how successful the well was are considered.
Additionally, the previously researched optimisation procedure to obtain the sequential drilling strategy of dynamic programming will be compared to a more flexible reinforcement learning algorithm called Q-learning. Dynamic programming computes a value function for each action in a state. A state in the application of drilling wells would be which wells have been drilled already and their outcome. Q-learning approximates this value function by interacting with the environment and updating its beliefs along the way. It does not require a complete model to be specified beforehand. Thus, it is suited for more complex problems where these probabilities might not be available.

Both simulated and real data will be used to validate this novel method of sequential decision analysis with uncertainty and dependency in both the modelling of the geological success and the modelling of the volume of hydrocarbons retrieved, which can be seen as the utility. In doing so, the following research question and sub-questions will be answered:

How can sequential hydrocarbon drilling be approached with a more realistic dynamic decision support tool so that it can be deployed in practice?

- What is the added value of Bayesian networks in doing so?
- What alterations need to be made to the optimisation procedure to accommodate for the extended and more realistic model proposed in this thesis?

This thesis was written in collaboration with Equinor, a Norwegian petroleum refinery company. All expert opinions and data are provided by them.

The structure of this thesis report is the following. In Chapter 2, background information on the application of drilling prospects in the oil and gas industry is described. Then related work on sequential decision making in the oil and gas sector is discussed in Chapter 3. Chapter 4 contains the theory behind all the mathematical concepts used during the analysis. The overall methodology of this study is described in Chapter 5. The results are separated into three chapters. Chapter 6 contains the results from dynamic programming and the validation of using discrete Bayesian networks for this problem. In Chapter 7, the alternative optimisation procedure Q-learning is validated and compared to the previously used dynamic programming approach. Chapter 8 contains the results of applying the extended model, which includes the uncertainty and dependencies in estimating the volume of a real data case. The model is applied to the synthetic data case in Chapter 9. After those results, a sensitivity analysis is given in Chapter 10. Chapter 11 gives a conclusion of the research and a recommendation for further uses. Finally, a discussion on what could be improved in this study is contained in Chapter 12.

2 APPLICATION BACKGROUND THEORY

This chapter describes the application and background knowledge required to understand the conclusions and analyses in more detail. The reader is referred to Sanni [2018] and Lake and Fanchi [2005] for a more complete explanation of hydrocarbon engineering.

2.1 PROBLEM SETTING

The problem setting is an offshore area in the sea that is believed to contain oil, gas, or a combination of the two. Energy companies, such as Equinor, are interested in harvesting these hydrocarbons as they will lead to energy sources that can then be sold to supply energy for the consumers. Within these offshore hydrocarbon filled areas, geologists use, for example, seismic data or satellite data to determine estimates for the geological features that may influence the success of a prospect. Based on these estimates, the location of the most profitable or successful prospects is determined. These estimates also provide a way to estimate the amount of hydrocarbons extracted from the well. The model for this estimate is described in the next section.

2.2 MODEL TO ESTIMATE THE AMOUNT OF RECOVERED HYDROCARBONS

The success of a well is separated into two parts: whether the well is a success and how much of a success it is. Whether it is a success depends on the area having certain geological factors. These factors are:

- A producible reservoir must be present. A reservoir is producible when the rock has favourable characteristics. The most important characteristics are porosity and permeability. A porous rock formation means there are small spaces between the rock where the hydrocarbons can be detained. The permeability of the rock is the capability to transmit fluids. This is necessary for the hydrocarbons to be extracted.
- The trap is sufficient, meaning the hydrocarbons are trapped in an underground structure. There are many ways that these traps can take their form. The two main types of traps are structural traps and stratigraphic traps or a combination of the two [Sanni, 2018, Ch.1.4]. Besides the trap geometry, a seal on top of the trap must be present to ensure the hydrocarbons are contained in the reservoir.
- The source of hydrocarbons is present and has migrated to the reservoir.

These three geological factors are modelled by their respective probabilities of being a success through a Bernoulli distribution. If all geological factors are a success, then the well is considered successful. Note that the three geological factors are made up of six reservoir properties that do not necessarily have to be discrete random variables.

The second part is to calculate how much of a success it was, in other words, how much hydrocarbons can be extracted from the well. For this, the volumetric method in the form of the Original Oil In Place (OOIP) formula as given in Equation 2.1 is used King [2020]. In the nominator, the amount of hydrocarbon in the reservoir is calculated. This is done in the following steps:

1. The volume of the rock in the reservoir, the gross rock volume, is calculated via Equation 2.2. The more information is available on the reservoir, the more accurate this calculation can be.
2. Within this rock, there may only be a part of the reservoir that contains hydrocarbons and has a high permeability. The net-to-gross thickness ratio converts the gross rock volume (thickness) to the producible rock volume.
3. The porosity, in the form of a fraction, is used to compute how much actual fluids are available in the reservoir.
4. The fluids can be a combination of water and hydrocarbons. Thus to get the volume of the hydrocarbons, the amount of fluid is multiplied by the hydrocarbon saturation.
5. The last step is to convert the volume of the hydrocarbons in the reservoir to the volume of hydrocarbons at the surface level. For this, the formation volume factor B is used. This volume at the surface level can be expressed as cubic metres or barrels, depending on the model's assumptions.

$$OOIP = \frac{V_{grossrock} \cdot (N/G) \cdot \phi \cdot S_{HC}}{B} \tag{2.1}$$

where $V_{grossrock}$ is the volume of the reservoir (m^3), N/G is the net-to-gross thickness ratio (fraction, so unitless), ϕ is the porosity (fraction, so unitless), S_{HC} is the hydrocarbon saturation (fraction, so unitless), and B is the formation volume factor

$$V_{grossrock} = Area \cdot Thickness \cdot geometric factor \tag{2.2}$$

where *Area*is the area of the reservoir (m^2), *Thickness* is the thickness of the reservoir (m), and the geometric factor is based on the reservoir's shape. The geometric factor removes the assumption that the reservoir is a rectangular prism and allows for more flexible and more accurate estimations of the volume of the reservoir.
During the process of extracting the oil, some losses may occur, so the volume of hydrocarbons after drilling a well, called the reserves, is the hydrocarbons from the reservoir at surface level times a recovery factor. This is given in Equation 2.3.

$$Reserves = OOIP \cdot RF \tag{2.3}$$

where RF is the recovery factor.
After the drilling process has been completed, the reserves of the well are sold for a specific price. The revenue of the well is the reserves times the price for which they can be sold per unit minus the cost of drilling and producing the well.

3 RELATED WORK

Sequential decision making in the oil and gas industry has been around for quite some time. Early on, the industry realised the value of information and the importance of learning. Hendricks and Kovenock [1989] has shown that not having a rational decision making tool leads to over-drilling in areas without hydrocarbons and under-drilling in areas where hydrocarbons are plentiful. However, it was first put into more concrete research set up in Bickel and Smith [2006]. They modelled the dependencies between the probability of success of multiple wells in the same area using the minimally informative joint distribution based on the marginal and pairwise conditional probability assessments given by experts. Using dynamic programming to derive the optimal policy from the model. Through a case with simulated data, they have shown that modelling dependencies between wells in a highly dependent geographical area where the wells individually are expected to not be profitable, and taking the information obtained after drilling into account in the drilling strategy will increase your expected revenue to be positive. So, using the model for dependencies value was created out of a non-valuable area.
Two years later Bickel et al. [2008] expanded that model by separating the probability of success into three geological factors that must all be successful for the well to succeed. This meant that the dependency structure between the specific geological factors could be more detailed.
Martinelli et al. [2014] expanded this even further and is closest to the work reported in this thesis. They use a discrete Bayesian network to model the dependencies between the geological risk factors. All conditional probability tables in the discrete Bayesian network were specified through expert assessments. Here also, dynamic programming was used to obtain the decision policy from the model.

Besides dynamic programming, no other algorithms have been used in oil and gas research to achieve the optimal sequential decision policy. In other applications, sequential decision analysis problems have been tackled and solved using ideas from optimal control engineering and reinforcement learning. Recht [2018] has constructed a concise overview of different types of problems and reinforcement learning algorithm solutions that can be used on sequential decision problems. Which algorithm to use depends mainly on the situation at hand. Generally, most sequential decision problems can be formulated as Markov decision problems (MPDs). Solving MDPs is extensively researched, see for example Howard [1960] for the first solution of solving MDPs with dynamic programming, or Otterlo and Wiering [2012] for reinforcement learning and MDPs, or Gosavi [2015] for solving MPDs through simulations, or Spaan [2012] for the situation where only partial information is observed in the environment.

The differences between previous research into sequential decision making in the oil and gas industry and the work reported in this thesis are threefold.
Firstly, the type of Bayesian network considered. Martinelli et al. [2014] uses a discrete Bayesian network. Utilising a non-parametric Bayesian network means using a mixture of discrete and continuous variables, which makes the model more realistic. Secondly, previous research estimated the recovered volume of hydrocarbons and assumed it to be deterministic without any uncertainty.
This thesis aims to separate the parameters used to compute the volume of hydrocarbons and model uncertainty and dependencies between those volume parameters

of various prospects. This means that not only the success probability of a well, but also how successful the well was is taken into account in the sequential decision analysis. Lastly, the optimisation procedure of dynamic programming is extended to deal with the larger and more complex model using a reinforcement learning algorithm called Q-learning.

Bayesian networks have also been utilised in the oil and gas industry for other research applications. For example, Martinelli et al. [2011] modelled dependent wells to perform an exhaustive static search of an optimal set of wells to be drilled simultaneously to know with high certainty that the area is either wet or dry. Bayesian networks are also used by Hanea et al. [2013] to estimate parameters such as the permeability of the rock in the reservoir simulation process as an alternative to the Ensemble Kalman Filter.

4 THEORY

This chapter, divided into three main sections, provides theoretical background information on all models and mathematical aspects used in this thesis. How these models are combined to answer the research questions will be described in the methodology in Chapter 5. The first section covers the theory for Bayesian networks. The second section contains the theory of copulae. The last section describes two optimisation techniques, dynamic programming and Q-learning, that can be used to obtain a sequential decision policy.

4.1 BAYESIAN NETWORK

A Bayesian network, also known as a Bayesian belief net, is a graphical structure representing the dependencies between various random variables. Specifically, the graphical structure is a directed acyclic graph. A directed graph is a set of nodes connected through directed arcs. A directed graph is acyclic if there is no directed cycle in the graph.
In a Bayesian network, the nodes represent random variables, and the arcs portray dependencies between the nodes Hanea [2008].

4.1.1 Types of nodes in a Bayesian network

The nodes inside of a Bayesian network can be of many different types. A high-level distinction can be made between chance nodes and functional nodes.

Chance nodes represent random variables. The most commonly used ones are discrete nodes, which model discrete random variables with a finite number of states. Most often, continuous random variables are discretised in order to use discrete nodes in a Bayesian network Chen et al. [2017]. This is because modelling dependencies between discrete nodes is quite simple by using conditional probability tables. For nodes representing continuous variables, the dependencies cannot be modelled using conditional probability tables anymore; rather, more advanced mathematical concepts such as copulae can be used. Section 4.2 explains what copulae are. Lucas et al. [2007] analyses two alternative methods to copulae to model the dependencies in a Bayesian network with a mixture of discrete and continuous nodes. Besides parametric discrete and continuous nodes, a non-parametric distribution can also be represented. This type of node mitigates the need for fitting a distribution to the acquired data.

Functional nodes are nodes where the random variable is a function of the distribution of its parent nodes. For example, a functional node could be the product of two chance nodes.

4.1.2 Structure of a Bayesian network

The collection of the nodes and the arcs between the nodes in a Bayesian network is referred to as the structure of a Bayesian network. An arc between two nodes

implies that there is a direct dependency between them. Certain combinations of arcs encode additional information about (conditional) (in)dependencies. For example, in Figure 4.1 nodes B and C do not have an arc between them, so there is no direct dependence. However, they both have node A as a parent. Thus, B and C are conditionally independent given A. This concept of the structure characterising (conditional) (in)dependencies is called d-separation, for more information see Geiger et al. [1990a], Geiger et al. [1990b], or Scanagatta et al. [2019].

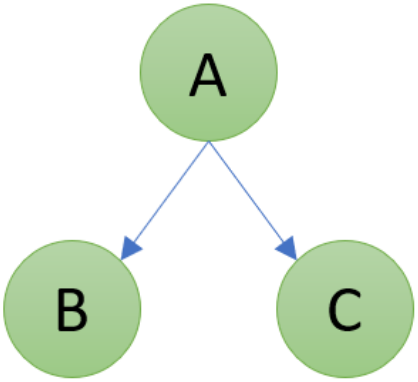


Figure 4.1: Example of conditionally independent variables

Thus, the structure of a Bayesian network is a representation of (conditional) (in)dependencies between certain variables in the network. The conditional independence between variables lead to a factorisation of the joint distribution. The joint distribution of n random variables, noted as X_i with $i \in \{1, .., n\}$, can always be factorised using the chain rule of probability:

$$P(X_1, X_2, ..., X_n) = \prod_{i=1}^{n} P(X_i | X_1, ..., X_{i-1})) \tag{4.1}$$

The joint distribution of the random variables in a Bayesian network can be represented as follows:

$$P(X_1, X_2, ..., X_n) = \prod_{i=1}^{n} P(X_i | parents(X_i))) \tag{4.2}$$

In the example of Figure 4.1 the conditional independency represented by the structure means the joint distribution of the three random variables can be written as: $P(A, B, C) = P(A)P(B|A)P(C|A)$.

The number of possibilities for structures in a Bayesian network is quite large. For just five nodes, the number of possible structures is already 29.281. Thus, a strategy must be used to determine which structure best suits our data. There are multiple ways to tackle this, depending on the type of information available. If there is data available, a structure learning algorithm can be used. There are three types of structure learning algorithms. Constraint-based learning algorithms use statistical tests to find conditional independence constraints within the data. A commonly used algorithgm is the PC algorithm proposed in Colombo et al. [2014]. An alternative is a score-based learning algorithm, which maximises a score assigned to the structure. This score can be based on various criteria, such as the likelihood the data fits the structure and the size of the network. An example of a score-based learning algorithm is Hill Climbing, the reader is referred to Russell [2010] for more detail. Lastly, a hybrid learning algorithm combines constraint-based and score-based learning algorithms. First, the options of structures that the score-based algorithm can choose

from are reduced by applying a constraint-based strategy. Within those structures, the optimal structure is chosen using a score-based learning algorithm. An example is the Max-Min Hill Climbing algorithm presented by Tsamardinos et al. [2006]. Constantinou [2019] has performed a comparison on various properties different structure learning algorithms have.

If no or limited data is available, the structure is determined either through previously established (in)dependencies available in literature or through expert assessments.

Given the network structure, the dependency still needs to be represented. For discrete Bayesian networks, the dependencies are defined using conditional probability tables. This table specifies the probabilities of all the states of a child node, given all the possible states the parent nodes could be in. When nodes represent continuous random variables, there are multiple options. One of which is to use (conditional) copulae assigned to the arcs of the network. In the next section, copulae and their functionality is explained.

4.2 COPULAE

Formally, a d-dimensional copula is a joint distribution function defined on an n-dimensional unit hypercube with standard uniform marginal distributions Nelsen [2007]. A copula can be used to model the dependency structures between random variables by isolating the dependency structure from the marginal distributions. This means the dependency structure can be investigated and/or specified separately from the marginal distributions. This makes the model easier to apply to real-life situations as the marginal and dependency data are obtained separately. Copulae were first introduced by means of Sklar's theorem in Sklar [1959] and later more clearly introduced in Sklar [1973] and Schweizer and Sklar [1974]. The reader is referred to Rüschendorf [2013] for a concise representation of the proof of this theorem. Using Sklar's theorem and some uniform transformations the general form of a bivariate copula of variables X and Y is

$$C(u, v) = F_{X,Y}(F_X^{-1}(u), F_Y^{-1}(v)), \tag{4.3}$$

where u and v are realisations of the standard uniform distribution. F_X^{-1} is the inverse CDF of variable X and analogously F_Y^{-1} is the inverse CDF of variable Y. $F_{X,Y}$ is the joint CDF of variables X and Y.
The general form of a bivariate copula conditional on a third random variable, or a set of random variables, $\mathbf{Z}$, called the conditioning set, is

$$C(u, v|\mathbf{z}) = F_{X,Y|Z}(F_{X|\mathbf{Z}}^{-1}(u|z), F_{Y|\mathbf{Z}}^{-1}(v|z)) \tag{4.4}$$

where u, v, and z are realisations of the standard uniform distribution. $F_X|\mathbf{Z}^{-1}$ is the inverse CDF of variable X conditional on $\mathbf{Z}$ and analogously $F_Y|\mathbf{Z}^{-1}$ is the inverse CDF of variable Y conditional on $\mathbf{Z}$. $F_{X,Y|\mathbf{Z}}$ is the joint CDF of variables X and Y conditional on $\mathbf{Z}$.

There are parametric families of copulae, such as Gaussian copulae or Archimedean copulae.

4.2.1 Bivariate Gaussian copulae

Uninet is the software designed for non-parametric Bayesian networks that use copulae assigned to edges to model the dependencies. In this thesis, only (un)conditional

bivariate Gaussian copulae are considered, as that is the only copula Uninet can deal with at the moment. The bivariate normal Gaussian copula is given as

$$C(u,v) = \boldsymbol{\phi}_\rho(\phi^{-1}(u), \phi^{-1}(v)), \tag{4.5}$$

where $\boldsymbol{\phi}_\rho$ is the joint CDF of the bivariate normal distribution with zero mean and correlation coefficient ρ and ϕ^{-1} is the inverse of the standard univariate normal CDF.

4.2.2 Learning correlation parameters from data

In non-parametric Bayesian networks, as specified in Uninet, a bivariate Gaussian copula with a (conditional) rank correlation as parameter is assigned to each arc between two probabilistic nodes, say X and Y. When these two variables have parent nodes in the network, these may be inside the conditioning set $\mathbf{Z}$. Whether the parent node is conditioned on depends on the sampling order of the network. In theory, the joint distribution is the same regardless of the factorisation. In practice, this may not be the case for graphs that are not fully connected due to sequential estimation of the rank correlation coefficients.

Once the structure of a Bayesian network has been specified, either through expert input or through a structure learning algorithm as discussed in section 4.1.2, it is known which (un)conditional rank correlations need to be specified. Ideally, these rank correlations are learned from data. For many applications, this is useful and possible. However, there are cases where it is not possible because the data is unavailable, and then expert judgement can be used. This means that experts are asked for their opinions, and those opinions are used as an estimate. There are many ways that this expert elicitation process to obtain the rank correlations can be conducted. For example, having experts give exceedence probabilities from which the rank correlations are deduced, see Morales et al. [2008a] and Cooke and Goossens [2000]. An alternative is to ask pairwise Pearson correlation coefficients and deduce the (conditional) rank correlations from that for which the ideas are summarised in Morales et al. [2008b]. Which of these methods are chosen can depend on many things, but mostly on the type of input the experts are willing to provide. The second option is described here in more detail.

The pairwise Pearson correlation coefficients for N random variables, provided by experts, can be represented by the upper diagonal of the correlation matrix as presented in Equation 4.6.

$$\begin{pmatrix} \rho_{1,2} & \rho_{1,3} & \cdots & \rho_{1,N} \\ & \rho_{2,3} & \cdots & \rho_{2,N} \\ & & \ddots & \vdots \\ & & & \rho_{N-1,N} \end{pmatrix} \tag{4.6}$$

To go from the pairwise Pearson correlation coefficients to the (un)conditional rank correlations, the following steps need to be taken. First, the partial correlation coefficients can be computed from the pairwise correlation coefficients as the partial and conditional correlations are the same for the joint normal distribution. This is done by using Equation 4.7 recursively. Thus, the conditional correlations are obtained. Then, the known relationship for the Gaussian copula as given in 4.8 is used to obtain the ranked conditional correlations, where ρ is the Pearson correlation coefficient, and r is the rank correlation coefficient.

$$\rho_{1,2|3,4,\ldots,n} = \frac{\rho_{1,2|4,\ldots,n} - \rho_{1,3|4,\ldots,n} \cdot \rho_{2,3|4,\ldots,n}}{\sqrt{(1-\rho^2_{1,3|4,\ldots,n}) \cdot (1-\rho^2_{2,3|4,\ldots,n})}} \tag{4.7}$$

$$\rho = 2\sin\left(\frac{\pi}{6}r\right) \iff r = \frac{6}{\pi}\arcsin\left(\frac{\rho}{2}\right) \tag{4.8}$$

4.2.3 Other applications of copulae

Copulae can be used in many applications where dependencies between variables are considered. Examples are risk analysis in various sectors such as the financial sector McNeil et al. [2015], research in the field of health care such as in Quinn et al. [2007], and models for climate change as in Mesbahzadeh et al. [2019]. Copulae will be used in the analysis of this paper to model dependencies between properties of prospects in a similar geological region.

4.3 OPTIMISATION PROCEDURES

Once a model has been established, an optimisation procedure to obtain the optimal sequential decision policy must be performed. This section will explain two methods considered in this analysis: dynamic programming and Q-learning. The explanation here will be kept very general, and in the methodology, specific alterations are discussed to fit the application in the oil and gas industry.

4.3.1 Dynamic programming

The first option would be to use dynamic programming. Dynamic programming is a method that breaks down a larger problem into smaller problems that can be more easily solved. These smaller problems combined will lead to the solution of the whole problem Bertsekas [2005]. Dynamic programming in sequential decision-making uses the Bellman equation as given in Equation 4.9 Sutton and Barto [2018]. This states that the value of action a in action space A in state s in state space S is the expected value of the sum of the immediate reward of action a and the future rewards of the next state s' and its optimal action a' at a discounted rate. The value of the state-action pair (s, a) is called the Q-value. The value of the discount factor γ is application dependent. All of the Q-values of the state-action pairs can be combined in a Q-table, which is a table with each row being a different state, and the columns being the possible actions. An example of how that would look is given in Table 4.1.

$$Q(s,a) = \mathbb{E}(\underbrace{r_a}_{\text{immediate reward}} + \underbrace{\gamma \cdot Q(s',a')}_{\text{discounted future rewards}}) \tag{4.9}$$

		Action		
		a1	a2	...
State	s1	Q(s1, a1)	Q(s1, a2)	...
	s2	Q(s2, a1)	Q(s2, a2)	...
	⋮	⋮	⋮	⋱

Table 4.1: Example of Q-table

The dynamic programming algorithm computes the Q-values recursively. Note that to compute the expected value the whole model and the probabilities to go from one state to another must be known. From the Q-values the policy, or most optimal action, can be computed through the general equation 4.10.

$$\text{Policy in state } s = \text{argmax}_a(Q(s,a)) \tag{4.10}$$

Using dynamic programming creates the optimal policies given the model it is based on as it brute force calculates the best action. It is important to note that dynamic programming is only possible for a small and discrete state and action space due to the curse of dimensionality. This is further discussed in Powell [2007]. Thus, it is no longer suitable when the problem becomes too large.

4.3.2 Q-learning

Q-learning is a reinforcement learning algorithm. This means there is an agent which learns the outcome of actions in a specific state through interacting with the environment and gathering experiences. Q-learning approximates Q-values from this interaction with the environment. The Q-values in Q-learning are the same as described in dynamic programming in section 4.3.1 and are given in Equation 4.9. Q-learning is a tabular learning algorithm, meaning its approximation for the value functions can be represented as a table, known as a Q-table. This Q-table takes the same form as for dynamic programming.
Q-Learning was the first reinforcement learning that guaranteed convergence to the optimal policy. It was initially proposed in Watkins [1989] and its convergence proof was refined in Watkins and Dayan [1992]. A simple version of the proof can be found in Melo [2001].
The Q-learning algorithm can be described by iteratively encountering the environment and updating the value functions in the state and with the action it just took. Thus, the algorithm learns which action to take in a specific state based on both the instant rewards and future possibly discounted, rewards. The way the Q-value in the Q-table is updated after experiencing more is given in Equation 4.11. This updating method uses weighted averages of the old experiences and the new experience in combination with estimated future experiences.
The main difference between Q-learning and dynamic programming is that Q-learning creates policies solely based on the rewards it receives by interacting with its environment. In contrast, for dynamic programming computes the Q-values through its knowledge of the model.

$$Q^{new}(s_t, a_t) = (1 - \underbrace{\alpha}_{\text{learning rate}}) \cdot Q^{old}(s_t, a_t) + \underbrace{\alpha}_{\text{learning rate}} \cdot \left(\underbrace{r_t}_{\text{reward}} + \underbrace{\gamma}_{\text{discount factor}} \cdot \underbrace{\max_a Q(s_{t+1}, a)}_{\text{estimate of optimal future value}} \right) \tag{4.11}$$

The general flow of the training process of the Q-learning algorithm is visualised in a flowchart in Figure 4.2. It starts by first initialising the Q-table, and observing the current state s. Then an action a needs to be chosen, either through exploration where a random action is chosen, with probability ϵ, or exploitation where the best action is chosen, with probability $1 - \epsilon$. If epsilon is above zero this procedure is called epsilon greedy. Then, the action is taken and the reward r and next state s' are observed and used to update the Q-table. In some applications, there may be an end state where the episode is finished. For example, in a game, you take certain actions and then die, so no further actions can be taken and the game is over. If you can still continue, the next state s' becomes the current state s and the process of choosing an action and updating is repeated. Otherwise, the episode is finished and either a reset occurs and a new episode is started or the training process is finished.

Another important thing to note is that Q-learning is applied to Markov decision problems with the Markov property. This means that the transition from one state to the next only depends on the current state and the action taken with some stochasticity involved Wiering and Van Otterlo [2012]. Applications not satisfying this

problem, either need a different state representation or cannot and should not be solved using Q-learning.

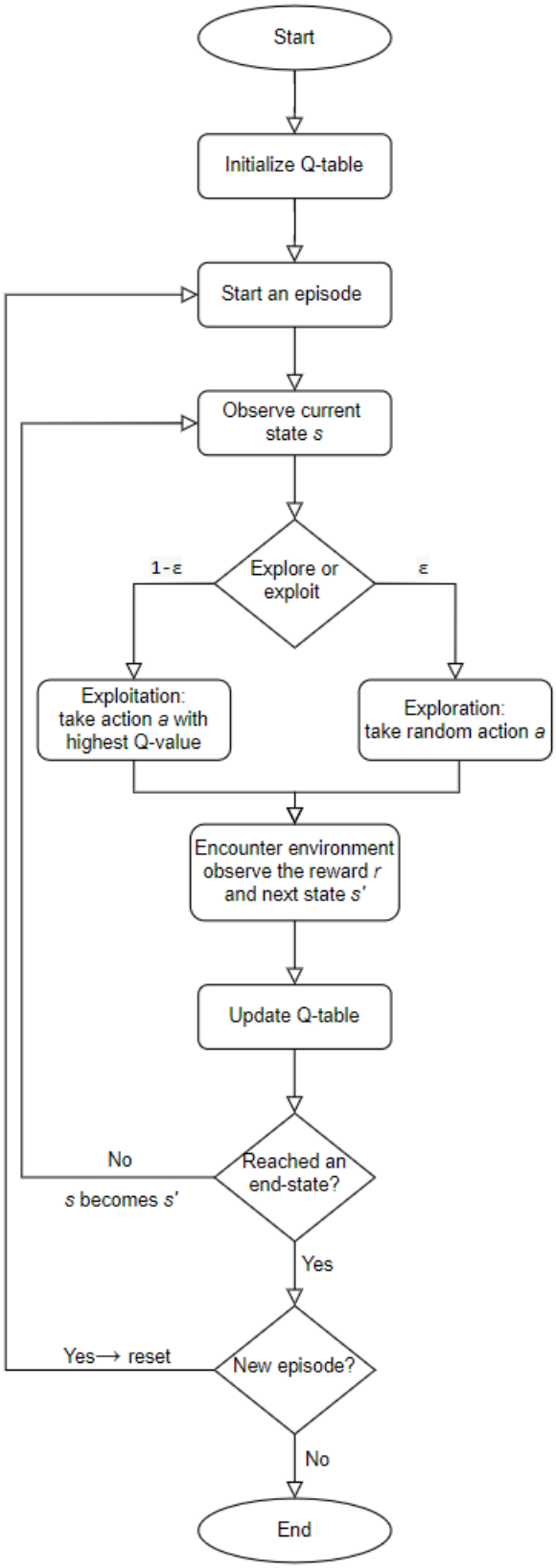


Figure 4.2: Flowchart of the training process of Q-learning

In the Q-learning algorithm there are there are four parameters that can be tuned to change and improve performance.
First off, the initialisation of the Q-values in the Q-table. The two most commonly used initialisation techniques are zero initialisation, where all Q-values are zero at the start, and optimistic initialisation, where all Q-values are initialised to the maximum possible value. Optimistic initialisation and its effect on the exploration strategy has been shown to result in a higher percentage of optimal actions compared to zero initialisation with the epsilon-greedy approach Sutton and Barto [2018].
Secondly, how the iterations are reset. For every iteration, the agent could either start in a starting state or in a random state, depending on the application. Starting in a starting state would mimic more real-life trial and error learning situations. Randomly resetting might be more beneficial if the non-likely states also should have representative Q-values.
During the training process, for every action the agent takes, he can either exploit his knowledge and choose the best action he knows thus far or explore a random other action. This is called the exploration-exploitation trade-off and is materialised in the epsilon value in Q-learning. Epsilon lies between zero and one and represents the probability of exploring a random action Sutton and Barto [2018]. The exploration-exploitation strategy can also be made possible through other options, such as the optimistic initialisation mentioned before Sutton and Barto [2018].
Lastly, the learning rate can be altered between zero and one. The learning rate is the weight attributed to how meaningful a new experience is compared to the old experience. The learning rate has a considerable influence on the convergence speed of the Q-learner Even-Dar et al. [2003]. In the convergence proof of Q-learning, presented in Watkins and Dayan [1992] and Melo [2001], a condition is set on the learning rate. Namely, the learning rate λ at time t must satisfy the conditions in Equation 4.12.

$$\sum_t \lambda_t = \infty \qquad \text{and} \qquad \sum_t \lambda_t^2 < \infty \tag{4.12}$$

An example of a learning rate satisfying these conditions is $\lambda_t = \frac{1}{t}$. However, this means that a learning rate of zero is approached relatively fast as it is already 0.001 after only 1000 iterations, and then the Q-values do not update much anymore. In theory, letting the training process run forever, they would converge according to the proof, but this is not realistic. Therefore, in practice, this condition is often violated.

An overview of the parameters in the Q-learning algorithm that can be tuned is given in Table 4.2

Parameter	Choice
Reset	From start state, random state
Initialisation	Optimistic, zero
Epsilon	[0,1]; constant, decaying
Learning rate	(0,1); constant, decaying

Table 4.2: Possible choices for the hyperparameters in Q-learning

5 METHODOLOGY

This chapter describes the research methodology used to answer the research questions and create a dynamic support tool for sequential exploration decisions. It is separated into four sections corresponding to the steps taken in this research. Then, a sensitivity analysis is described. Finally, a section explaining the implementation can be found at the end of this chapter.

5.1 COMBINING PREVIOUS WORK

The first step in answering the research questions is to combine the work of Bickel et al. [2008] and Martinelli et al. [2014] into one model. A visual representation of their processes is given in Figure 5.1 and 5.2 to make the workflow of both methodologies clearer. Both methodologies can be split into two parts: creating a model, which is visualised in the green box, and using an algorithm on this model to obtain a sequential drilling policy, visualised in the purple box.

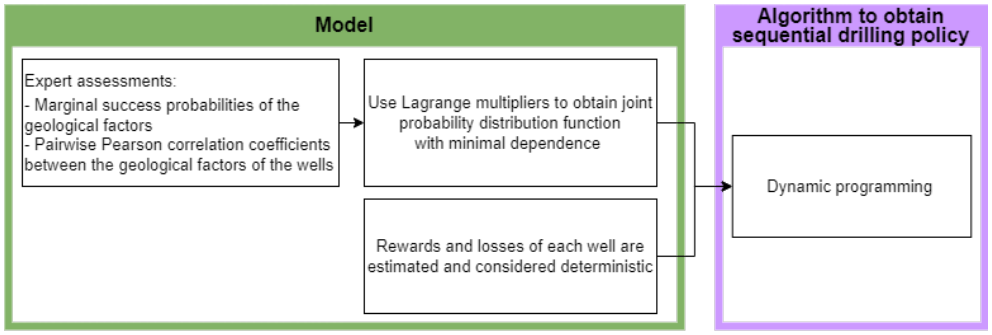


Figure 5.1: Visualisation of the workflow of previous work of Bickel and Smith [2006]

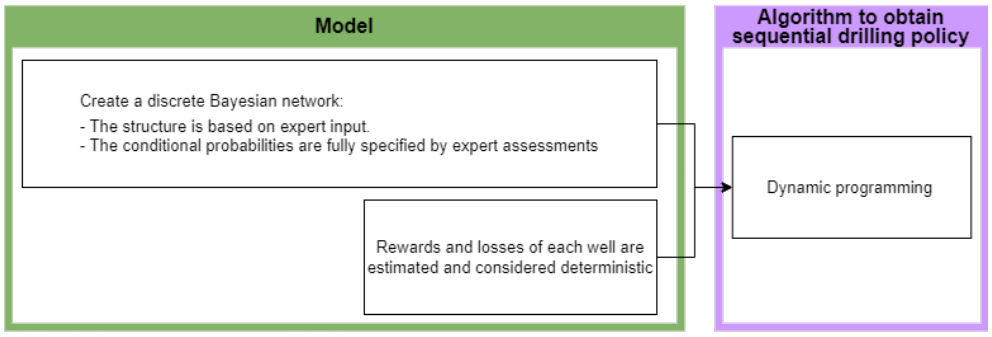


Figure 5.2: Visualisation of the workflow of previous work of Martinelli et al. [2014]

The combination of the two methodologies takes place in the model creates part. Specifically, the set up of Bickel et al. [2008] is used, with an additional step of constructing a Bayesian network with a specified structure and conditional probability tables defined through the joint distribution. This workflow is visualised in Figure 5.3. This combination is done to validate the use of Bayesian networks and their

ability to capture all information and result in the same policy as directly using the joint distribution as Bickel et al. [2008] does. The method for obtaining the joint distribution is duplicated to Appendix B for completeness. Note that for the numerical optimisation algorithm the Broyden-Fletcher-Goldfarb-Shanno method with initial values takes to be zero was used as this yielded the same outcome as the optimisation performed by the Solver in Excel as used in Bickel et al. [2008].
The results from these analysis are given in Chapter 6.
As the first steps are the same as in Bickel et al. [2008], the same input data for the marginal success probabilities, the pairwise Pearson correlation coefficients, and the rewards and losses of each well are used. These can be found in Appendix A. The three geological factors, source, reservoir, and trap, are considered in this analysis. Both the settings with three wells and with five wells are considered.

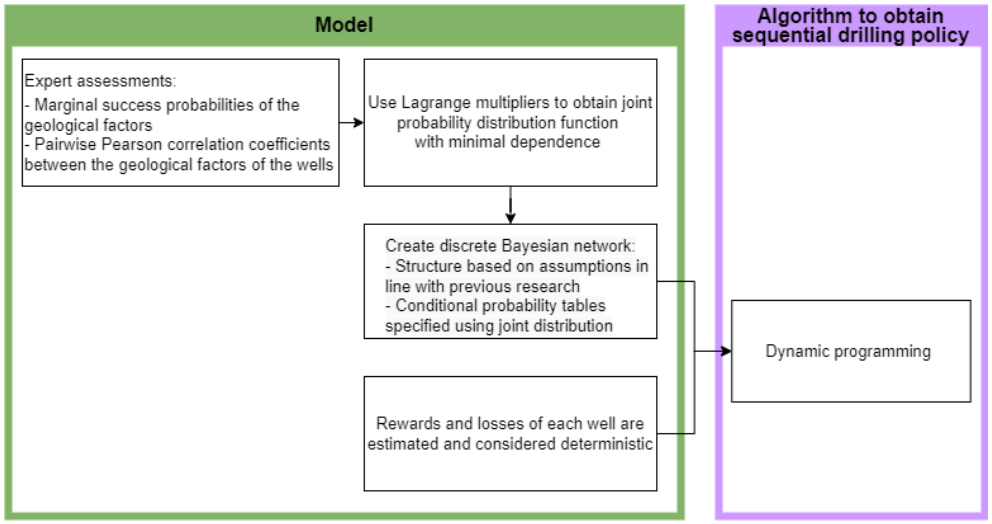


Figure 5.3: Visualisation of the workflow of the first step in combining Bickel et al. [2008] and Martinelli et al. [2014]

5.1.1 Discrete Bayesian network

This section describes how the discrete Bayesian network is defined. The structure of the Bayesian network is constructed based on the following assumptions:

- The geological factors of one well are independent. For example, the type of seal on the reservoir does not say whether hydrocarbons have migrated to the area.
- The wells are independent conditional on the three geological factors.
- A geological factor at one well may be dependent/correlated with the state of the geological factor at other wells.

For a five-well model with three geological factors, these assumptions lead to the Bayesian network structure of three sub-networks for the three geological risk factors with an arc going from each geological risk factor at well i to the node 'success of well i', $i \in \{1,2,3,4,5\}$. Figure 5.4 shows a visual representation of the three-factor five-well Bayesian network.
The green nodes are discrete probability nodes representing the state of a geological factor at a specific well. These have two states: success or failure. Each factor has its own fully connected Bayesian sub-network, meaning there is an arc between each variable in the sub-network. This corresponds to assuming no (conditional) independencies between the wells for each geological factor which keeps the setting as general as possible. If more information about the group of wells and their

(in)dependencies is known, alternatives could be considered. For example, Martinelli et al. [2014] has considered a few other structures, such as including common factors that influence the success of the geological factors at the wells making the success of a geological factor at one well independent of the success of that geological factor at another well conditional on the common factor.
The blue nodes depict functional nodes of the success of the wells. In line with the theory, a well is only a success if all three of its geological factors are a success. Given that the geological factors are independent, the probability of success of well i is equal to the product of the probabilities that the geological risk factors are a success.

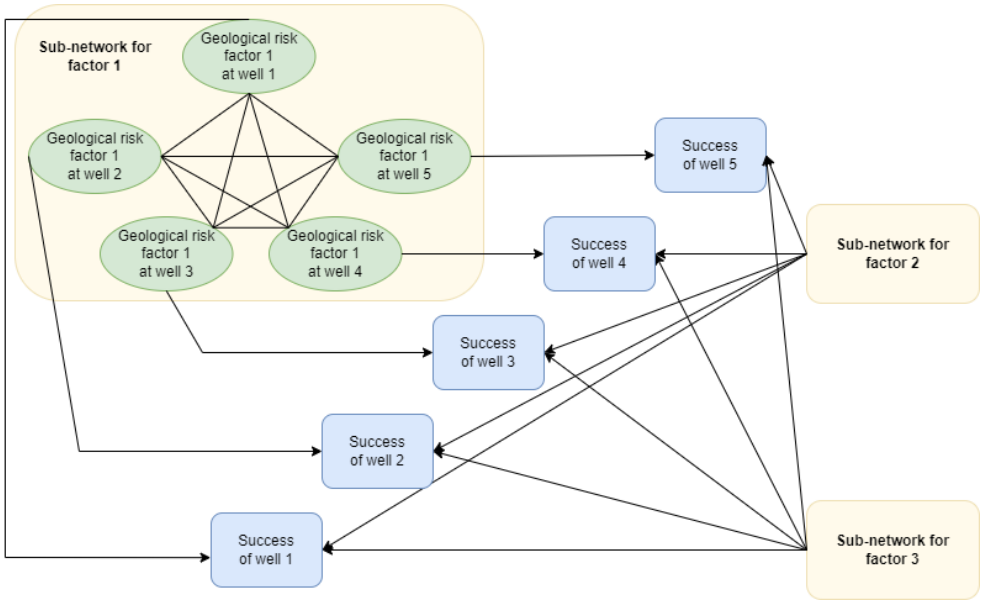


Figure 5.4: Bayesian network structure illustration for five wells and three geological factors.

After the structure has been established, the joint distribution obtained from the Lagrange multipliers with the assumption of minimal dependence as in Bickel et al. [2008] is used to specify the conditional probability tables in the Bayesian network.

5.1.2 Dynamic programming

After the Bayesian network has been created dynamic programming is applied to obtain the optimal sequential drilling policy given the model. The solution of dynamic programming is optimal given the model as it needs a fully specified model and covers each case with its respective probabilities. See section 4.3.1 for an explanation on how dynamic programming works. The states s in states space S for the application of drilling N wells considering three geological factors are given as in Equation 5.1.

$$s = \left(\begin{pmatrix} f_{1,1} \\ f_{2,1} \\ f_{3,1} \end{pmatrix}, \quad \begin{pmatrix} f_{1,2} \\ f_{2,2} \\ f_{3,2} \end{pmatrix}, \quad \ldots, \quad \begin{pmatrix} f_{1,N} \\ f_{2,N} \\ f_{3,N} \end{pmatrix} \right) \tag{5.1}$$

where $f_{i,j}$ is the state of factor j at well i, which can be either a success, a failure, or unobserved due to the fact that the well has not been drilled yet. This is encoded as in Equation 5.2.

$$f_{i,j} = \begin{cases} 1 & \text{if the state of factor } j \text{ at well } i \text{ is a success} \\ 0 & \text{if the state of factor } j \text{ at well } i \text{ is a failure} \\ -1 & \text{if the state of factor } j \text{ at well } i \text{ is a unobserved} \end{cases} \tag{5.2}$$

The actions in the action space are $A = (a_1, a_2, ..., a_N)$, where a_i is the action of drilling well i.
The value of the discount rate γ for the Q-values calculations is chosen as 0.99. This is adopted from Bickel et al. [2008] with the same application, where it is explained as a discount rate of 9% if the wells each take six weeks to drill.

Based on the Q-table, as a result of dynamic programming, the policy for each state can be determined via Equation 5.3. First, the action with the highest Q-value for state s is chosen. If none of the Q-values are positive, the action to stop drilling is taken.

$$\text{Policy in state } s = \begin{cases} \text{argmax}_a(Q(s,a)) & \text{if } Q(s,a) > 0 \ \forall a \in A \\ \text{stop drilling} & \text{if } Q(s,a) \leq 0 \ \forall a \in A \end{cases} \tag{5.3}$$

On the results of performing the dynamic programming algorithm to the three-well and the five-well problem, some analyses are performed. First, the obtained policy is compared to the policy in Bickel et al. [2008], which is directly based on the joint distribution. If these policies are the same, then using a Bayesian network for this application is validated. Second, the expected revenue based on the discrete Bayesian network dynamic programming policy is computed. The steps to compute this expected revenue are the following:

1. Obtain a list of all states the group of wells could be in, so leaving out the states where some wells are unobserved, call these the end-state of the wells.
2. Compute for each end-state the probability of that end-state occurring via the joint distribution. The sum of these probabilities is equal to one.
3. For each end-state, apply the policy on the group of wells and obtain the observations as they are in the end-state. Compute the total loss or reward gained from the policy at that end-state.
4. Weigh the rewards for each end-state with the probability of that end-state. This gives a measure of the expected revenue based on the policy.

This computed expected revenue is compared to the expected revenue of other policies. These alternative policies are, for example, the policy of drilling all wells with a positive intrinsic value or the policy where only successful wells are drilled as if the outcome of each well is known beforehand.
Additionally, a time complexity analysis of the increasing number of wells and the number of geological factors is conducted.

5.2 Q-LEARNING AND DISCRETE BAYESIAN NETWORK

The next step is to replace the dynamic programming algorithm with the reinforcement learning algorithm Q-learning. This is done to see whether a reinforcement learning algorithm will get the same performance as dynamic programming while some of the limitations of dynamic programming, such as the curse of dimensionality, are mitigated. Training through the Q-learning algorithm based on a Bayesian network is done by simulating the network. Once an action is chosen, the network is simulated. The network is then conditioned on the simulated values of the drilled well. The sample of the Bayesian network to obtain the outcome of the next action will thus be a conditional sample.

The Q-learning algorithm is applied to the problem setting and Bayesian network as described before. This workflow is visualised in Figure 5.5.

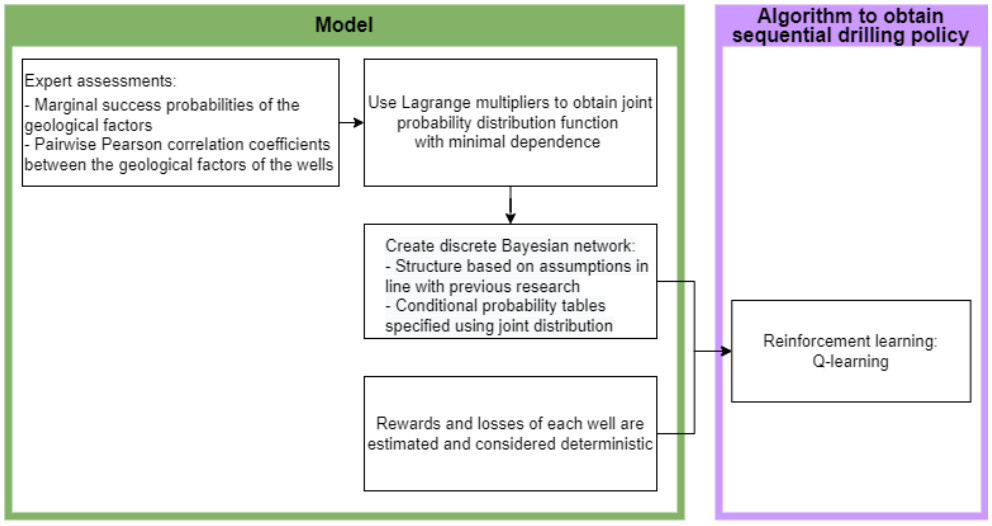


Figure 5.5: Visualisation of the workflow after including reinforcement learning

Afterwards, it is time to evaluate the performance of the Q-learning algorithm. For this, some validation metrics are used. The most straightforward way is to compare the policy derived from the Q-learning algorithm with the policy derived from dynamic programming. Three metrics are used for this purpose:

- The percentage of policy decisions that are the same as in the dynamic programming policy. In other words, given the state that the group of wells is in, is the action according to the Q-learner policy the same as that of the dynamic programming policy?
- The mean of the differences between the Q-values approximated by Q-learning and the Q-values computed in dynamic programming.
- The standard deviation of the differences between the Q-values approximated by Q-learning and the Q-values computed in dynamic programming.

Lastly, the expected revenue based on the Q-learning policy is computed and compared to the expected revenue of the dynamic programming policy. If dynamic programming policy would not be available, this expected revenue metric can still be computed and its value would still be representative of the performance.
In this thesis, only the computational time, ease of implementation, intuitiveness, and usefulness in practice are considered as additional evaluation tools for the models.

The results of the before mentioned analysis on Q-learning in combination with a discrete Bayesian network as a model can be found in Chapter 7.

5.3 EXTEND MODEL TO MORE REALISTIC SITUATION AND TO FIT REAL DATA

After Q-learning is validated, the model is extended to resemble the real-life situation more closely. The main extension to the model is, instead of estimating the volume of hydrocarbons (and by proxy the rewards) and considering it deterministic, separating the volume calculation into six continuous parameters that each have their probability distribution which can be dependent on the parameters of other wells. These six parameters are the same as in the OOIP formula described in Chapter 2. This extension prompts the conversion from a discrete Bayesian network

to a non-parametric Bayesian network. The dependencies between the nodes in the non-parametric Bayesian network are modelled through copulae.
For the reinforcement learning algorithm, Q-learning is used, just as before. Ideally, a reinforcement learning algorithm that is equipped to handle continuous states is utilised. However, these alternative reinforcement learning algorithms were not feasible for this project due to the limitations of the implementations of Uninet, the software for non-parametric Bayesian networks, as it can only connect to the programming language R.
In order to combine the continuous distributions and the discrete Q-learning algorithm a discretisation scheme is used, which is explained in more detail later.
The workflow of this extended model is visualised in Figure 5.6. The results of this model are given in Chapter 8.

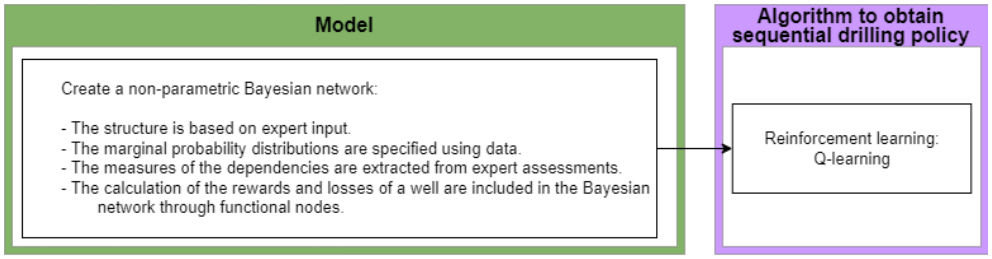


Figure 5.6: Visualisation of the workflow after including reinforcement learning and extending the model

5.3.1 Extended Bayesian network

The Bayesian network presented before is extended by the addition of six volume parameters and functional nodes containing the calculation of the revenue. The structure of the Bayesian network for the extended model is based on assumptions, just as before. The part of the geological factors and their dependencies in the network stays the same. The assumptions for the geological risk factors as in the smaller model are also applied to the volume parameters. So, the volume parameters of one well are independent from each other. The volume parameters of different wells may be dependent. Conditional on the volume parameters and the geological factors the wells are independent. The calculation of the revenue is based on the formulas described in Chapter 2 and represented in the Bayesian network through functional nodes. This leads to the Bayesian network for three wells with three geological factors and six volume parameters as displayed in Figure 5.7. See Appendix D for a full page view of the Bayesian network and the formulas used for every functional node in Table D.1. This structure can easily be extended to accommodate more than three wells.

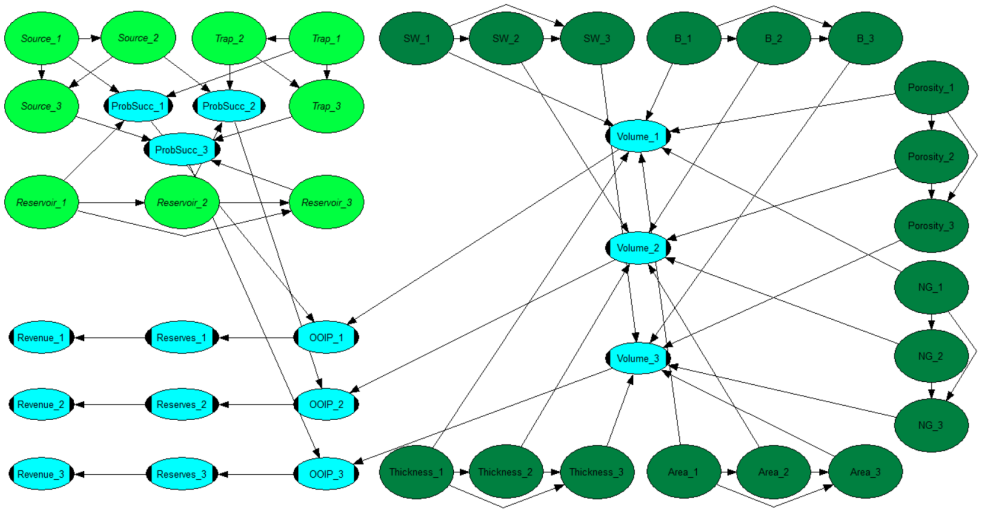


Figure 5.7: Bayesian network structure for the extended model for three wells with three geological factors and six volume parameters.

To use this Bayesian network the marginal distributions and the (conditional) rank correlations must be specified. To make the case as realistic as possible, and with that show, it can be used in practice, real data is used for this. The next section described what data is used and how it is processed.

5.3.2 Data

The data used in this thesis are fully anonymized and provided by Equinor and fully anonymized and can be found in Appendix E. The data set consists of six wells in the same geological region. These wells will get the names Albert, Martin, Vetle, Harald, Stig, and Bernt. For each of these wells, the estimated probabilities of success for the geological factors are given. These are separated into six probabilities: trap geometry, trap seal, source presence, source migration, reservoir presence, and producibility. These six probabilities can be combined pairwise into the probabilities of success of trap, source, and reservoir. This combination is done by taking the product of two of the probabilities. For example, the probability of trap being successful is the product of the probability of trap geometry being a success and the probability of trap seal being a success. Note that the calculations of these probabilities are based on the assumption that the geological factors are all independent.

For each of the wells, the marginal distribution of most volume parameters in the OOIP formula, as described in Chapter 2, is given in terms of quantiles. The distribution of the formation volume factor B is computed in Equinor but was not on its own provided in the dataset, so it was retrieved by dividing the hydrocarbon pore volume parameter by the total in-place volume parameter.
The recovery factor, which is also computed in Equinor but not given explicitly in this dataset, is chosen to be constant and computed by dividing the mean of the recoverable volume divided by the mean of the total in-place volume. The aforementioned method for obtaining the formation volume factor and the recovery factor were provided by the expert at Equinor.

The quantiles specified are the 1%, 10%, 20%, 30%, 40%, 50%, 60%, 70%, 80%, 90%, and the 99% quantile.
To go from the quantiles to a full non-parametric distribution the minimally informative distribution is used, which linearly interpolated between quantiles, just as in structured expert judgement as described in Cooke et al. [1991]. This means that between the quantiles the cumulative distribution function is linearly interpolated.

The support of the distribution is obtained by considering the 1% and 99% quantiles and hereafter a 10% overshoot. This rule of thumb is used for constructing distributions based on experts quantile assessments in the Classical Model for Structured Expert Judgement Cooke et al. [1991]. The obtained ranges were then truncated based on information on the underlying quantities. For example, a porosity fraction of above one or a negative area would not be possible.

Equinor currently does not model the dependencies between prospects, so the data set did not include measurements of dependencies between the prospects. The dependency measures between the prospects were obtained by asking an expert with domain expertise in that geological region. They provided descriptions and corresponding pairwise correlations for the various parameters. Their estimates are given in Appendix E. As some correlations depend on the location, information about the location was also provided by the expert. Wells Harald, Stig, and Bernt lie closely to each other. Further away lies the pair Martin and Vetle. In between, but closer to Martin and Vetle lies Albert on his own. A visualisation of this is given in Appendix E. To transform these pairwise Pearson correlation coefficients to the (conditional) rank correlations necessary for the non-parametric Bayesian network the properties of the normal copula were used. This is explained in the Theory in section 4.2.2. Alternatives to this method mentioned in the Theory are not considered due to the way the expert was able to give assessments.
The data set also contains the true outcomes of the wells and the cost of the wells. Both of these are also given in Appendix E.
Note that the initial data set provided was much larger and only the data relevant to the analysis of this thesis is reported.

5.3.3 Q-learning on the extended model

The extended model includes more parameters, which means the state representation of the Q-learning algorithm needs to be adapted to accommodate for this. See Equation 5.4 for the extended state representation

$$.s = \left(\begin{pmatrix} f_{1,1} \\ f_{2,1} \\ f_{3,1} \\ p_{1,1} \\ p_{2,1} \\ p_{3,1} \\ p_{4,1} \\ p_{5,1} \\ p_{6,1} \end{pmatrix}, \begin{pmatrix} f_{1,2} \\ f_{2,2} \\ f_{3,2} \\ p_{2,2} \\ p_{2,2} \\ p_{3,2} \\ p_{4,2} \\ p_{5,2} \\ p_{6,2} \end{pmatrix}, \quad ..., \quad \begin{pmatrix} f_{1,N} \\ f_{2,N} \\ f_{3,N} \\ p_{N,N} \\ p_{2,N} \\ p_{3,N} \\ p_{4,N} \\ p_{5,N} \\ p_{6,N} \end{pmatrix} \right) \tag{5.4}$$

where $f_{i,j}$ is the state of factor j at well i, and $p_{i,j}$ is the state of parameter j at well i. Analogous to the smaller model, $f_{i,j}$ and $p_{i,j}$ take the values zero, one, and minus 1 according to Equation 5.2.

5.3.4 Assumptions for the model and sequential optimisation algorithm

From the data and the specified structure of the Bayesian network, a large model is created with both discrete and continuous random variables. Therefore, the continuous variables in Uninet were discretized into two classes: below median and above the median. These were represented by the value of the 33% and 66% quantile, as these two quantiles split the range into 3 equal size sections where the middle inter quantile range will half the time be accounted to the lower class and half the time to the higher class. These values will be used to condition the Bayesian network on.

Besides discretizing the continuous variables in the Bayesian network, some more concessions need to be done in order to let Q-learning run in a reasonable time for this project. Namely, the following:

- Only three of the six wells are considered. These are Harald, Stig and Bernt as these wells are close to each other and thus are strongly correlated. This is also an interesting case in practice, as the question arises whether after learning information about the first well drilled, the other two should still be drilled or whether the drilling of that region should be aborted.
- The variable *Reservoir* is left out of the Bayesian network as in this region, the probability of the reservoir being available and of good quality is one.
- The dependencies between the variables will not be considered for all variables. The five variables with the highest dependence are chosen. These are trap, source, porosity, hydrocarbon saturation, and net-to-gross ratio. This choice was made as weaker dependency between parameters means that the addition of the dependency to the model will influence the policy less than the addition of stronger dependencies.

This leads to the Bayesian network for the real data case having the structure as presented in Figure 5.8.

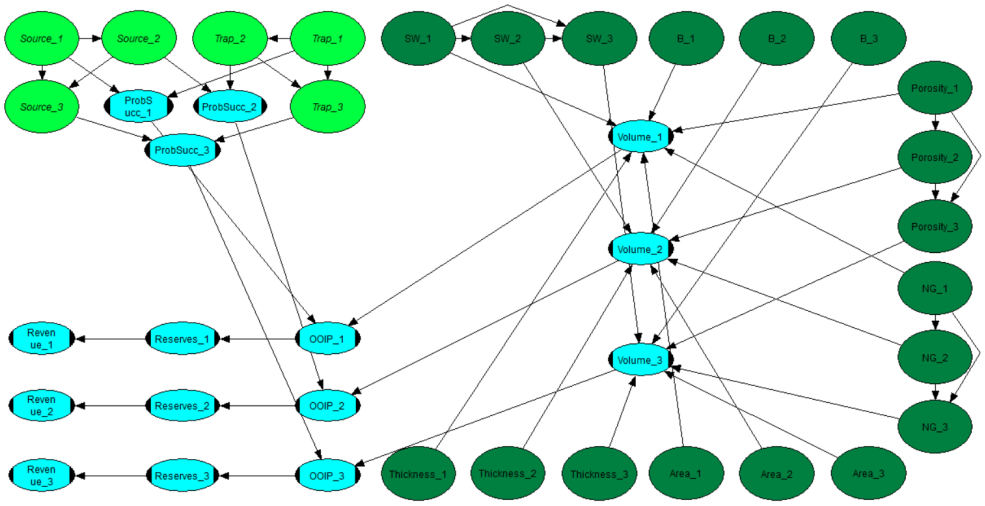


Figure 5.8: Bayesian network structure for the extended model for three wells with two geological factors and six volume parameters of which the dependencies between three of them are considered.

It is important to note that once the implementation limitations are lifted and an alternative reinforcement learning method is used the full model can be utilised and even expanded to capture even more aspects that may influence the decision process.
Lastly, some of the constants needed in the analysis were chosen from literature:

- The value of the geological factor will be set to 0.5. For the wells in this area, it is known that the trap type is a fault trap. According to an example from Kjemperud [NA] this results in a geological factor of 0.5.
- The price per barrel of hydrocarbons will be chosen as the price per barrel of oil, which is on average approximately 80$ in 2021 TradingEconomics [2022].
- The production costs per barrel vary greatly according to the country. On average it is about 30$ UCube by Rystad Energy [2015]

5.4 EXTRA SYNTHETIC DATA CASE

In addition to the real data case provided by Equinor, another synthetic data case is explored that has more favourable characteristics than the real data case. This case consists of three wells. Two of them have a positive expected revenue and one has a negative expected revenue. According to the current drilling strategy of the oil and gas industry, the two wells with a positive expected revenue will be drilled and the other one not. However, there are strong dependencies between the three wells. Thus, for some outcomes of the other two wells, the negative expected value becomes positive, and thus it should be included in the drilling strategy. This case is modelled with a non-parametric Bayesian network with the same structure as in the real data case as presented in Figure 5.8. Thus, only dependencies between five of the variables are considered. These are trap, source, porosity, hydrocarbon saturation, and net-to-gross ratio. The specified marginal distributions can be found in Appendix F. The pairwise Pearson correlations as dependency measures are all taken to be 0.7. The fixed costs per well are the costs in the real data case multiplied by 1000 so that they are in the same order of magnitude as the revenue. The rest of the choices for parameters or values are the same as in the real data case. On the model for this synthetic case, Q-learning is performed, which results in a sequential drilling policy. This policy is analysed through the expected revenue and compared to other policies, analogous to what is done for the real data case. This extra synthetic data case is analysed in Chapter 9.

5.5 SENSITIVITY ANALYSIS

As non-parametric Bayesian networks with copulae to model the dependencies are created for continuous random variables, a sensitivity analysis is performed on the use of discrete random variables in a non-parametric Bayesian network. The non-parametric Bayesian network and its probabilities and behaviours within that are compared to a discrete Bayesian network modelled in Netica, which is software designed specifically for discrete Bayesian networks. This is done in Chapter 10.

5.6 IMPLEMENTATION

To implement the above-mentioned procedures R version 4.1.2 is used.
The numerical optimisation algorithm used to obtain the joint distribution through Lagrange multipliers was the Broyden-Fletcher-Goldfarb-Shanno method. This was performed using the implementation in the lbfgs function in the nloptr package in R Johnson [2022]. The discrete Bayesian networks are constructed using the package bnlearn. This provided the necessary tools to construct a discrete Bayesian network given its structure and conditional probability tables. The non-parametric Bayesian networks are modelled in software designed specifically for non-parametric Bayesian networks named Uninet. Uninet is driven by the functions in the Uninet Engine. For both Uninet and the Uninet Engine an academic license was obtained. Trough the R library RDCOMClient a connection between R and the Uninet Engine was established. This allowed for creating, conditioning, and sampling from non-parametric Bayesian networks from within R. This was necessary as then analyses could be run autonomously, without intermediate steps having to be taken by the user. In order to apply the reinforcement learning algorithm Q-learning to a non-parametric Bayesian network in Uninet, Q-learning has been self-implemented in R.

6 DISCRETE BAYESIAN NETWORK MODEL: DYNAMIC PROGRAMMING

This chapter combines the models from Bickel et al. [2008] and Martinelli et al. [2014] to obtain a discrete Bayesian network from marginal and pairwise probabilities. The dynamic programming algorithm uses this Bayesian network to get an optimal dynamic sequential drilling strategy. This sequential drilling strategy will be compared to the results from Bickel et al. [2008] to validate that the Bayesian network with its specified structure can capture all information as described in Chapter 5. For this the three-well problem with three geological factors is used. Additionally, this analysis is used to validate the Q-learning algorithm in the next chapter. At the end of this chapter, a computational time analysis can be found on the dynamic programming algorithm.

6.1 RESULTS FROM DYNAMIC PROGRAMMING ON THREE-WELL PROBLEM

Dynamic programming calculates the Q-value for each state-action pair explicitly. As explained in Chapter 4, the policy derived from this Q-table will be considered the optimal policy based on the model and problem at hand. The calculated expected revenue based on this policy, for which the steps of the calculation are explained in Chapter 4, is 8.38 Million US Dollars. The value of 8.38 MUSD is slightly higher than the Q-value in the state where no wells have been drilled and the optimal action is taken, which is 8.21 MUSD. This slight difference is due to the discounted future rewards in the Q-value calculation. The performance of an expected value of 8.38 MUSD is considered to be optimal according to the model. This also shows that from a situation where all wells have a negative intrinsic value, value can be created when utilising the information of learning from experience and updating the drilling strategy along the way.
Nevertheless, if the policy was based on knowing the states of each well for sure and only drilling profitable wells, the expected revenue would be 44 MUSD.
Another important thing to mention is that you will never obtain the expected revenue in practice. There are only five possible outcomes of drilling wells in the three-well problem when using the problem as presented in Bickel et al. [2008]: -55, -20, 10, 40, and 105. These have a respective (rounded) probability of occurring: 0.23, 0.21, 0.27, 0.12, 0.17.
Lastly, it takes around 2 seconds to compute the whole Q-table for the three-well problem using dynamic programming. In the last section of this chapter, a more in-depth analysis of the computational time of dynamic programming is given.

6.2 RESULTS FROM DYNAMIC PROGRAMMING ON FIVE-WELL PROBLEM

Increasing the number of wells from three to five gives the five-well problem. The dynamic programming algorithm was applied to obtain a sequential drilling policy on the model of this five-well problem. The expected revenue from this dynamic programming policy is 21.81 MUSD. The highest Q-value of the state where no wells

have been drilled yet is 21.16 MUSD, meaning the discounting of future rewards reduces the expected revenue by over half a million dollars. The expected revenue for the five-well problem is higher than that of the three-well problem, which makes sense as two more wells can be drilled and thus can lead to higher revenue. The policy of only drilling successful wells as if the outcome of the well would be known beforehand leads to an expected revenue of 71.6 MUSD. Furthermore, it takes approximately two hours to run the dynamic programming algorithm on the five-well problem and obtain a sequential drilling policy.

6.3 COMPUTATIONAL TIME ANALYSIS

In this section, computational time analysis of dynamic programming is performed. The dynamic programming algorithm is run on increasingly large problems, and the time it takes to compute the optimal policy is saved. It is important to note that dynamic programming was implemented in R, in contrast to Bickel et al. [2008] where Excel was used. The implementation of dynamic programming in R is quite a lot slower than the original implementation in Excel due to the properties of the programming languages. The time it takes to run the dynamic programming algorithm on the model across various inputs, different numbers of wells, and different numbers of geological factors, is given in Figure 6.1 and Figure 6.2. From these figures, some interesting conclusions can be drawn. First, the computational time grows exponentially both in an increased number of wells and in an increased number of geological factors. Also, it costs relatively more time to consider an extra geological factor than it costs to consider an extra well in dynamic programming. This makes sense because the number of states is the number of wells to the power number of states.

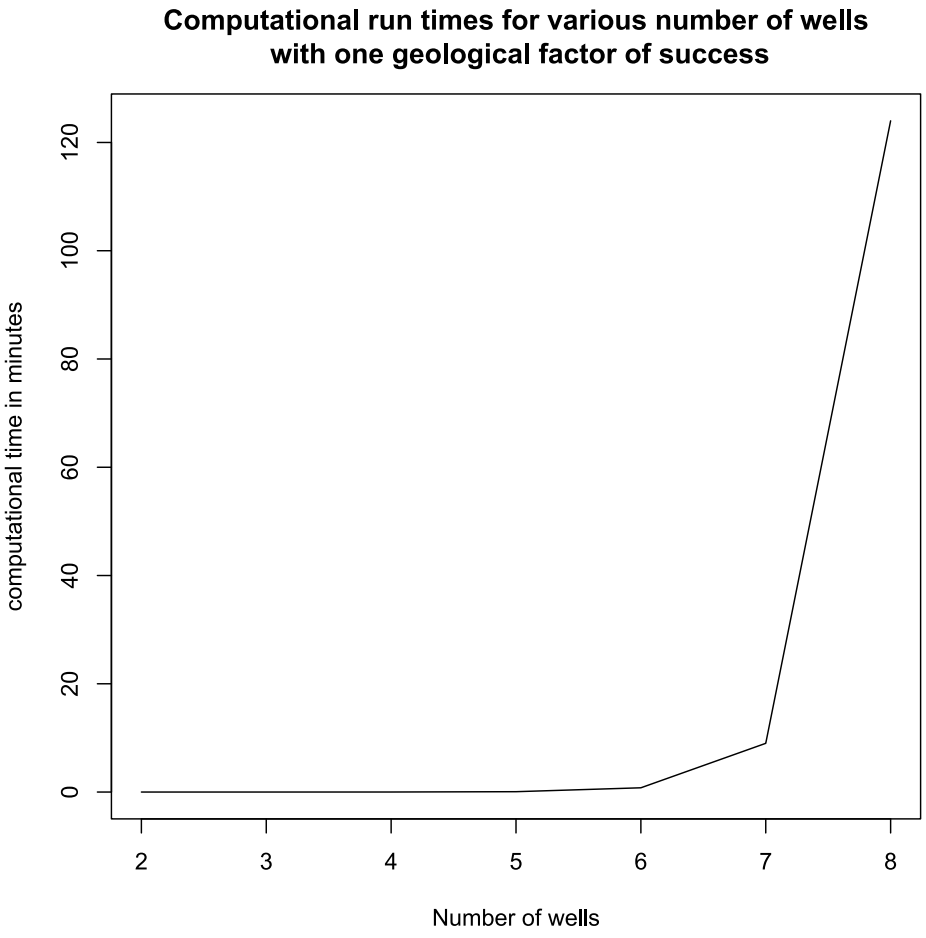


Figure 6.1: The computational times against the number of factors for the three-well problem

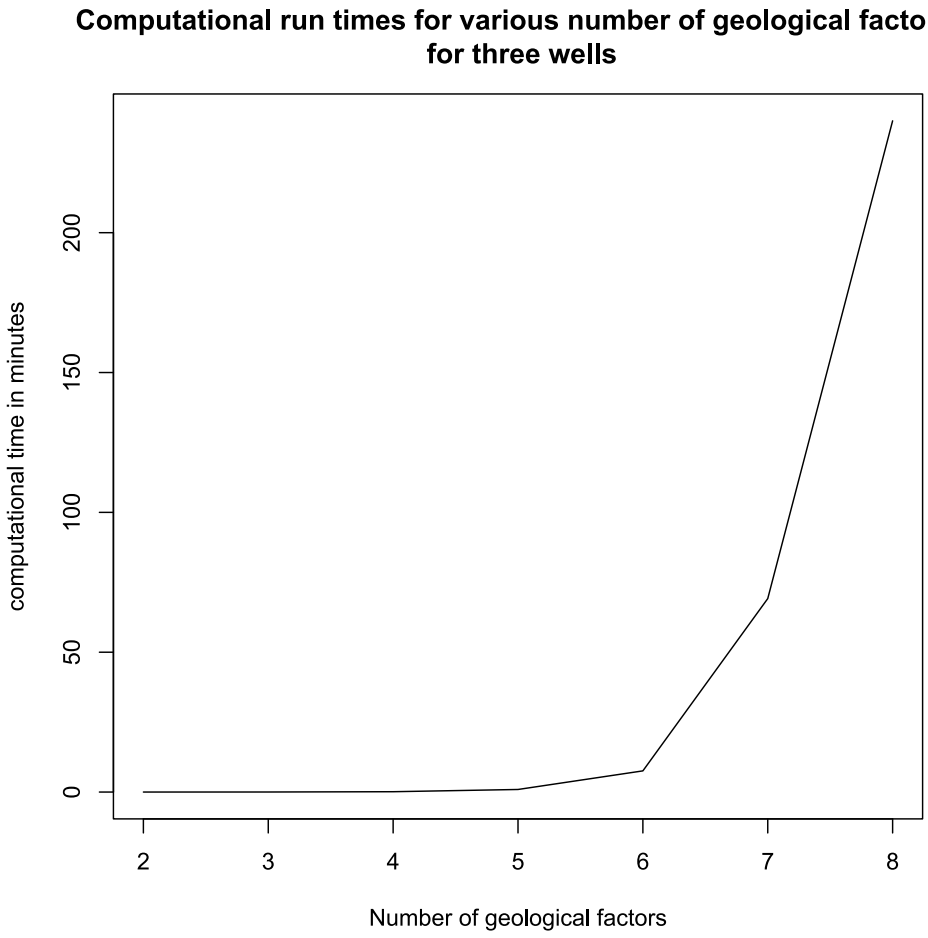


Figure 6.2: The computational times against the number of wells for the situation where only one geological success factor is considered

7 VALIDATION OF Q-LEARNING

This chapter will validate the Q-learning algorithm for the smaller discrete well-application. For this purpose, the problem with three wells and three geological factors as discussed in Bickel et al. [2008] is used. The corresponding marginal and pairwise conditional probabilities are given in Appendix A. The geological factors are modelled as binary discrete random variables. The smaller problem is used to compare the results from Q-learning to the results from dynamic programming. Additionally, this will be used to gather an understanding of the parameters in the Q-learning algorithm and how these should be chosen for this type of application. After the three-well problem has been tuned and analysed, the problem will be extended to five wells to investigate whether the same behaviour and parameters translate well to a larger problem.

7.1 RESULTS FROM Q-LEARNING WITH INITIAL HYPER-PARAMETERS

First, the Q-learning algorithm is used to train an agent using some commonly used choices for the parameters, these are given in Table 7.1.

Parameter	**Choice**
Reset	Reset to start state
Initialisation	Optimistic initialisation
Epsilon	0.2
Learning rate	0.2

Table 7.1: Initial choices for parameters in Q-learning

When training the agent for 20.000 episodes repeated ten times using these initial choices, the values of the four performance metrics discussed in Chapter 5 throughout the training process are visualised in Figure 7.1. Note that in the visualisations of the performance metrics over time, the darker coloured line represents the average of the ten training runs. The shaded interval represents the range of all the metrics across the ten runs at that iteration.

7.1.1 Comparison to dynamic programming policy

First, the correctness of the Q-learner's policy is analysed by comparing it with the optimal policy of dynamic programming using the three metrics discussed in Chapter 5.
The metric considering the percentage of state-action pairs where the Q-learner makes the same decision as in the dynamic programming result, and the way this metric evolves during the training process is given in Figure 7.1a. At the start of training, but after initialising optimistically, around 15% of the policies the Q-learner suggests are correct. That it starts at 15% instead of 0% can be attributed to the fact that after initialisation, all actions are equally favourable; thus, it is the same as choosing a random policy. The learning process can be described as slow but steady. There is an almost linear learning curve that, after 20.000 iterations,

ends at 64%, which means that at the end, 64% of the policy is the same as the dynamic programming policy. Near the end of the 20.000 episodes, the increase in performance per episode decreases. Following this trend, the model would still increase in performance more when increasing the number of iterations, but whether the policy would be the same as dynamic programming is uncertain.

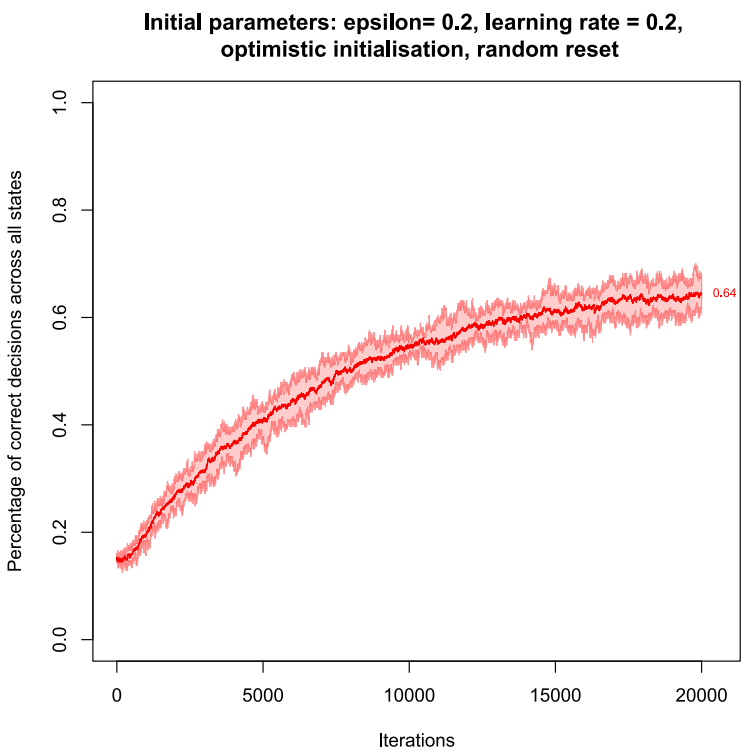
Another way of analysing how well the Q-learning algorithm learns is to compare the approximated Q-values for each state-action pair with the true Q-value calculated during the dynamic programming algorithm. For this purpose, the distribution of the differences, not taking the absolute values, is analysed. At the start, this distribution is heavily skewed, with most differences being around 100 MUSD due to the initialisation of each Q-value at 105 MUSD. During the learning process, this distribution becomes symmetric around the mean. In Figure 7.1b the mean of the differences is given across the training period, and the standard deviations are presented in Figure 7.1c. Observe that the standard distribution seems to be getting bigger at the beginning of the training process. This can be explained by the fact that the spread is increasing as the Q-values are starting to become lower and more distant from the initialised value of 105, while some Q-values of states which have not been encountered often are still around 105 MUSD, so the range becomes bigger. Eventually, the distribution becomes symmetric around 20 MUSD with a standard deviation of 36. This is still very much off from the true Q-values as most of those fall in the range from -30 to 30 MUSD, and a difference of 20 MUSD for a state action pair could change the policy and thus the revenue procured from the well.

7.1.2 Expected revenue

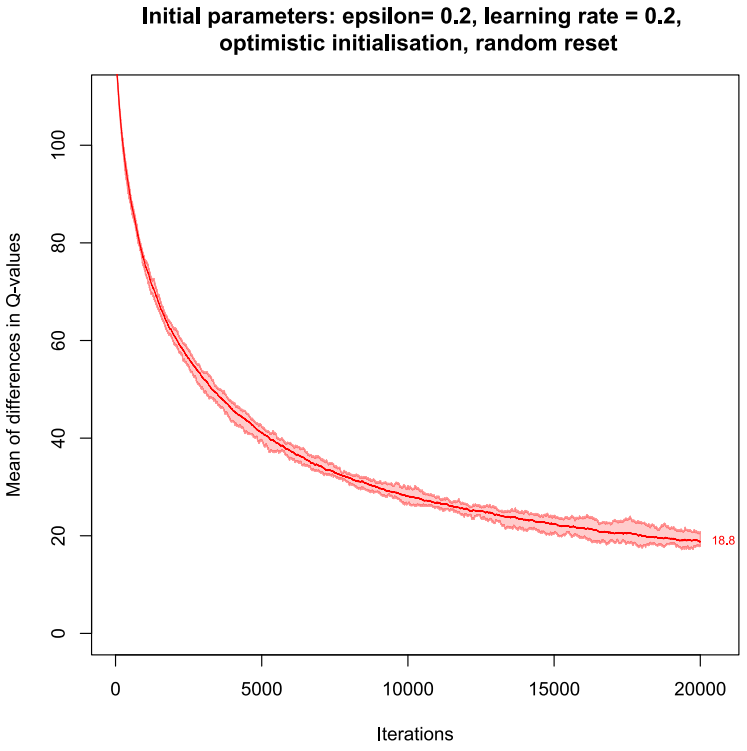
How the difference in policy between Q-learning and dynamic programming affects the performance, the expected revenue of the policy is computed. Figure 7.1d shows the evolution during the training process of this metric. After initialisation, it starts out negative, around -5 MUSD. After 1000 iterations, it starts to increase, and after 5000 iterations, it is on average around 0. After the 20.000 iterations, the expected revenue ends up around 5.2 MUSD. This shows that the Q-learner's policy results in value creation when considering updating the drilling policy sequentially, as the individual wells would not be drilled. However, compared to the dynamic programming policy with an expected revenue of 8.38 MUSD, there is still room for improvement. On average, so for the darker coloured line, the Q-learner's performance will not reach the performance level of dynamic programming. However, when considering the shaded interval, that almost touches the dynamic programming performance line, so during the learning process, there were runs where at specific episodes, the policy at that time performed almost as good as the dynamic programming policy. This prompts an addition to the algorithm: remembering the best policy thus far according to the expected revenue and using that policy in the end.

7.1.3 Computational time

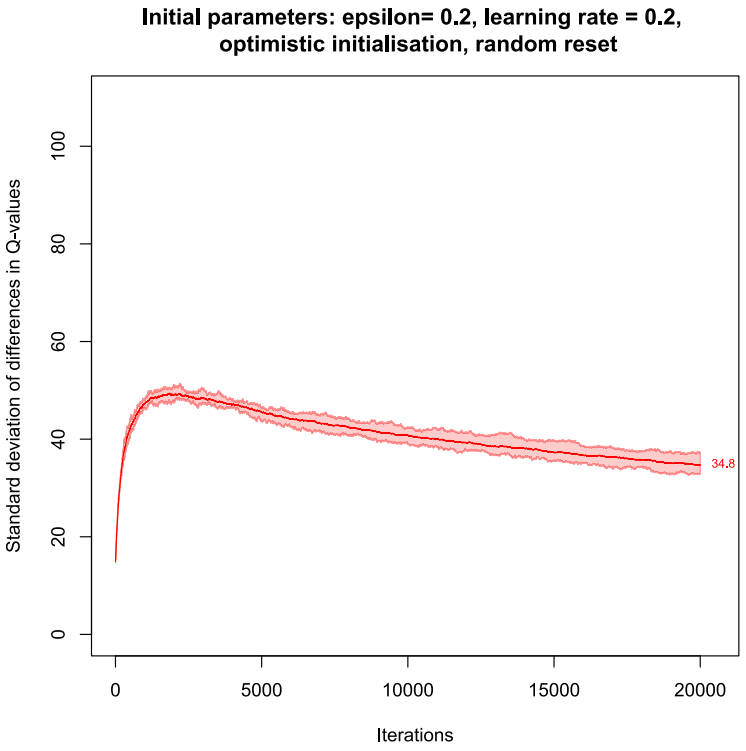
Lastly, the computational time of the Q-learning algorithm is considered. A Q-learning training process of 20.000 iterations, including the performance metrics calculation, takes around three minutes. This is longer than dynamic programming, which takes only a few seconds to run.



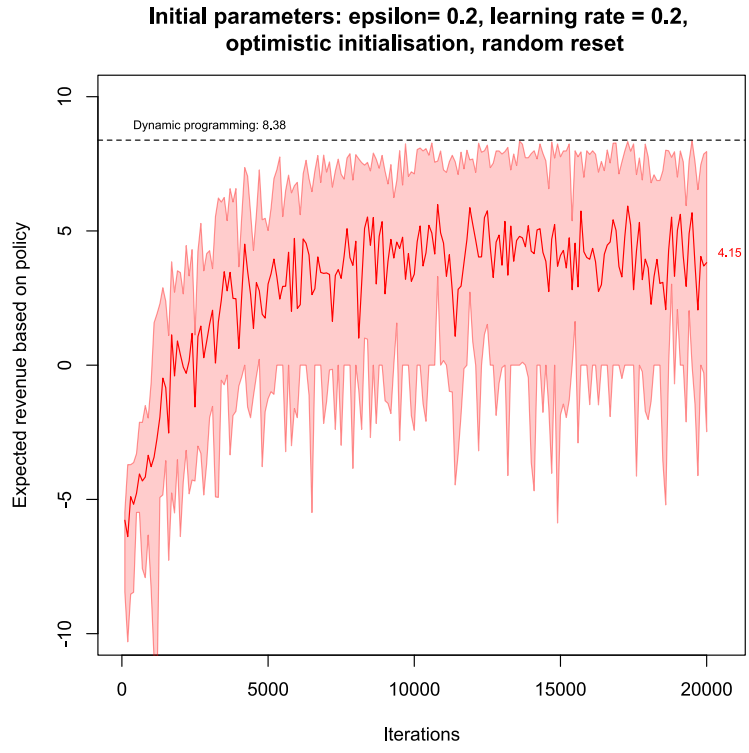
(a) Percentage of correct decisions across all states



(b) Mean of the differences between the true Q-values and the approximated Q values



(c) The standard deviation of the differences between the true Q-values and the approximated Q-values



(d) Expected revenue based on the policy derived from Q-learning

Figure 7.1: performance metrics throughout the training process on the three-well problem with initial parameters.

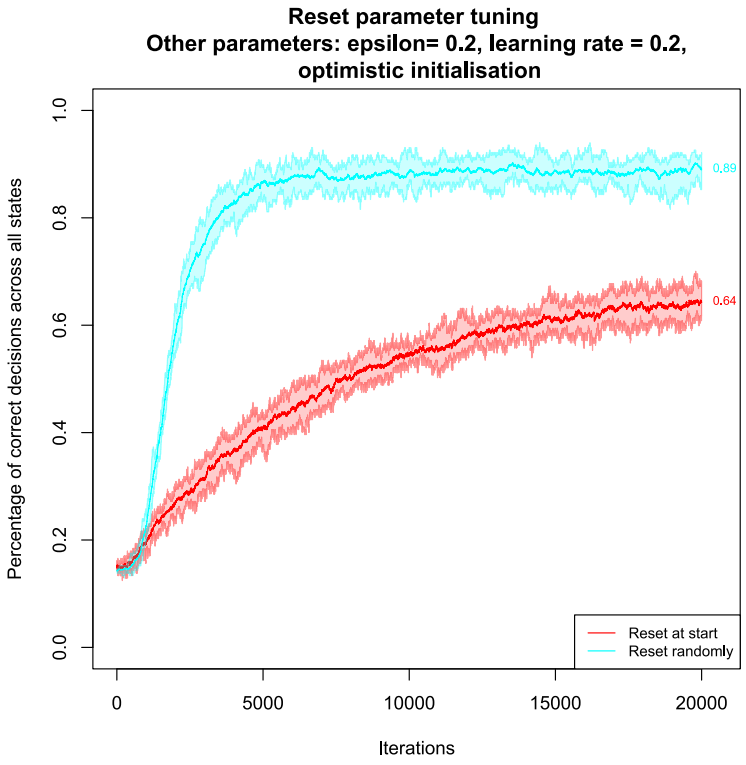
The next steps can be deduced from training the agent using Q-learning on some initial parameters. First, the parameters for Q-learning need to be tuned to see how these may change the results. Secondly, the addition of remembering the best policy thus far, according to the expected return, is considered.

7.2 PARAMETER TUNING

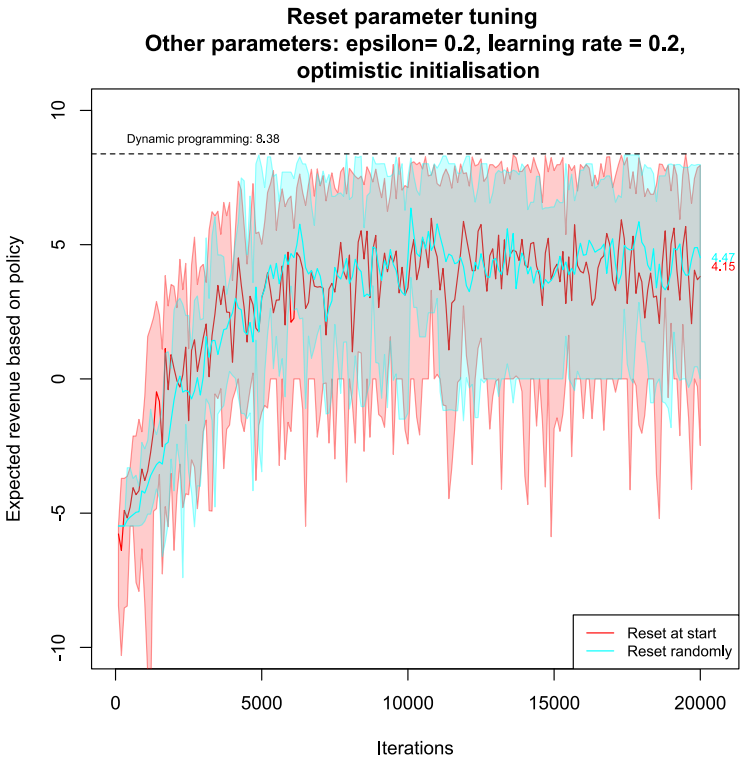
In this section, the results of the parameter tuning are discussed. Tuning of one parameter is done by keeping the other parameters the same and analysing the differences in performance metrics. For each of the parameters tuned, each of the four performance metrics is calculated. As this results in many figures, only figures containing visualisation from which conclusions were drawn are given in this section. A complete overview of the figures can be found in Appendix C.

7.2.1 Reset after each episode

The reset after each episode can be either to the start state, which is the state where no wells have been drilled yet or to a random state. Which one to choose depends on the application. To analyse the effect that the type of reset has on the performance of determining a sequential drilling strategy, all other parameters are kept the same and equal to the initial parameters displayed in Table 7.1, and only the reset parameter is changed. The results of the percentage of correct decisions throughout the training process are visualised in 7.2a. It is observed that the percentage of correct policy decisions for randomly resetting after each episode, which is the blue line, is much higher than for resetting to the starting state, which is the red line. Thus, resetting to a random state leads to a policy closer to the dynamic programming policy. However, the expected revenue, visualised in Figure C.2 between the two reset options is very similar. This shows that even though a policy is closer to the correct policy, it does not necessarily have the same performance. Different policies can still lead to similar results in expected revenue. This could be explained by some states having a very small probability of occurring. Resetting to a starting state results in those states being visited less often than when resetting to a random state. When a state is encountered less often, the Q-learner has not had enough experience to estimate the Q-value correctly. However, these states may have such a low probability of occurring that the effect of the policy not being correct on the expected revenue is very slim. Nevertheless, the dynamic programming policy is considered the correct policy, and a policy closer to that is preferred. Therefore, resetting to a random state is the better option.



(a) Percentage of correct decisions during learning

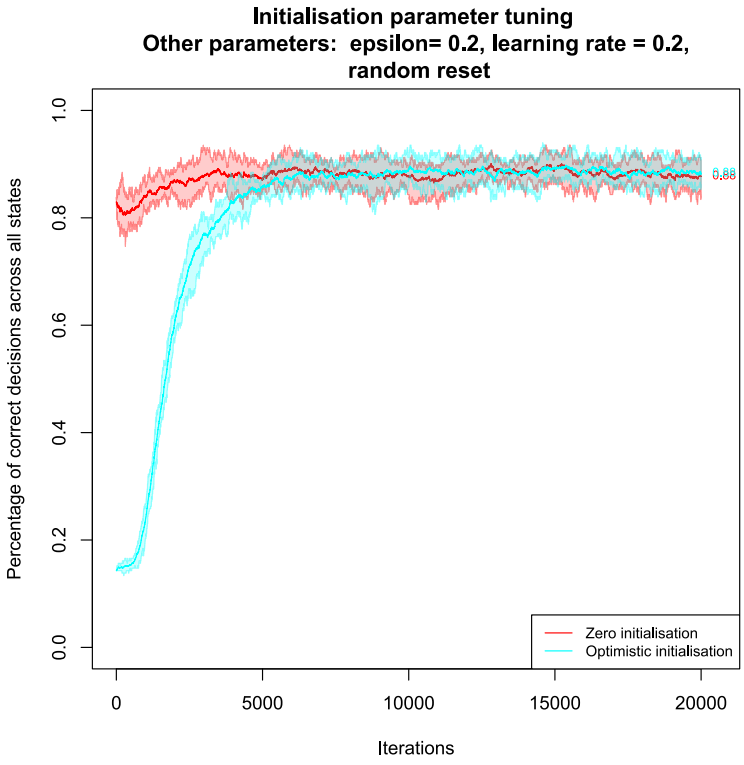


(b) Expected revenue throughout the training process

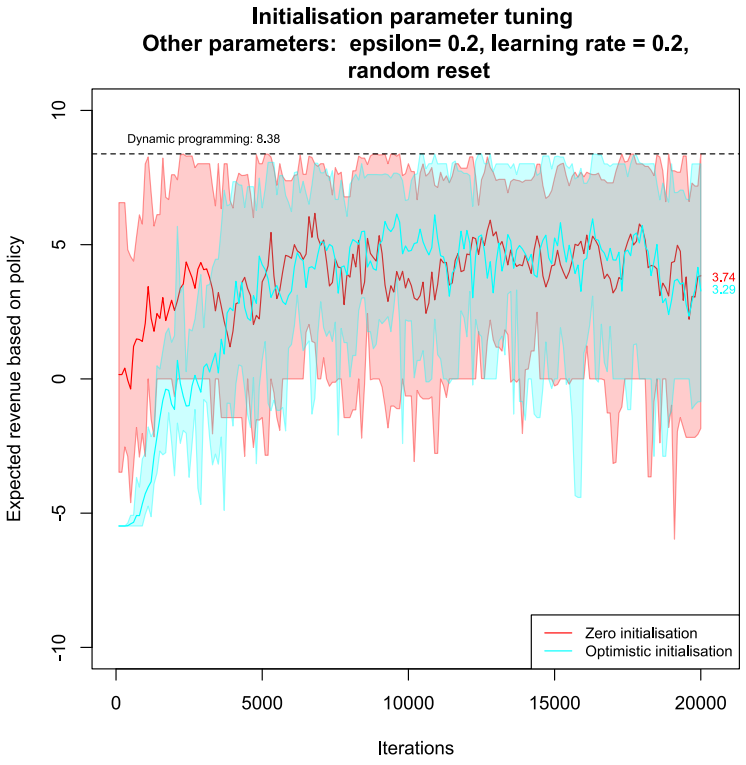
Figure 7.2: Reset parameter tuning: performance metrics throughout the training process on the three-well problem.

7.2.2 Initialisation

The initialisation of the Q-table can also have a significant impact on the outcome of the Q-learning algorithm. The two options considered in this analysis are zero initialisation and optimistic initialisation. The value for optimistic initialisation in the three-well problem will be 105 MUSD, which is the sum of the rewards when all three wells are successful. No true Q-value will be above this level. Both initialisation techniques perform very similarly on all metrics. The main difference is that it seems like the zero-initialisation gets to the optimal solution faster, which is visible in Figure 7.3. The red line, representing zero initialisation, starts higher than the blue line, depicting optimistic initialisation. After around 5000 episodes, the two lines coincide and have the same performance. This makes sense for the application of drilling wells as the Q-values for the state action pairs all fall within a range of -35 and 20. Thus, initialising it to zero means the Q-values are already closer to their true values. With optimistic initialisation, the Q-values are all positive during the beginning of the approximation of the Q-values. This means that the policy would be to drill a well. Zero initialisation leads to no positive Q-value, so the policy at the beginning would be never to drill. Of the dynamic programming policy, 83% of the policy decisions in states are to stop drilling. This means that the correct policy severely favours the decision not to drill. This is advantageous when using zero initialisation as fewer policy changes need to be made to get to the optimal policy. Note that this cannot be generalised and is very application dependent. Nonetheless, as mentioned in Chapter 4, optimistic initialisation generally performs better or at least as good as zero initialisation. Because of this, and wanting to keep the method as generalisable as possible, the optimistic initialisation procedure is chosen as the tuned parameter.



(a) Percentage of correct decisions during learning



(b) Expected revenue throughout the training process

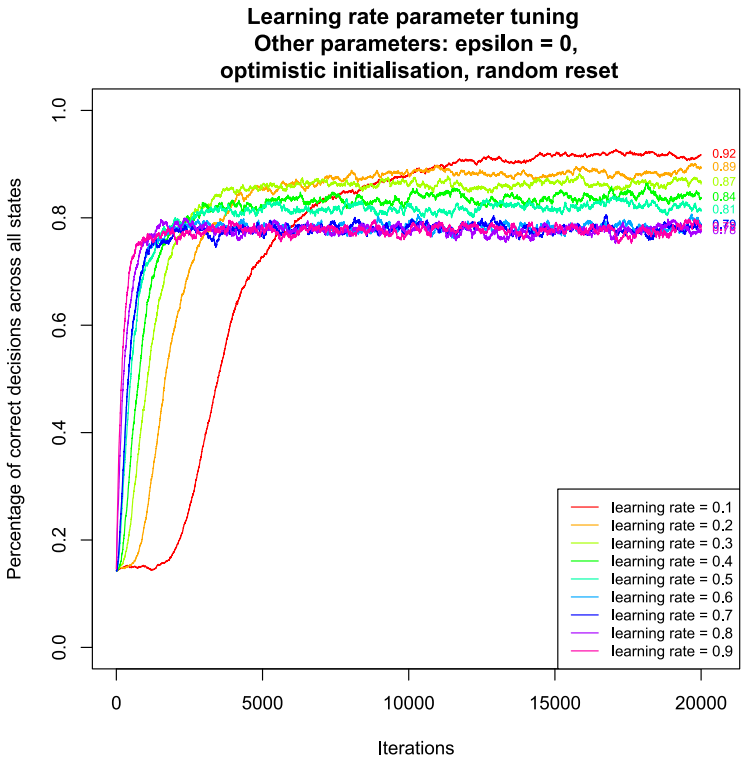
Figure 7.3: Initialisation tuning: performance metrics throughout the training process on the three-well problem.

7.2.3 Epsilon

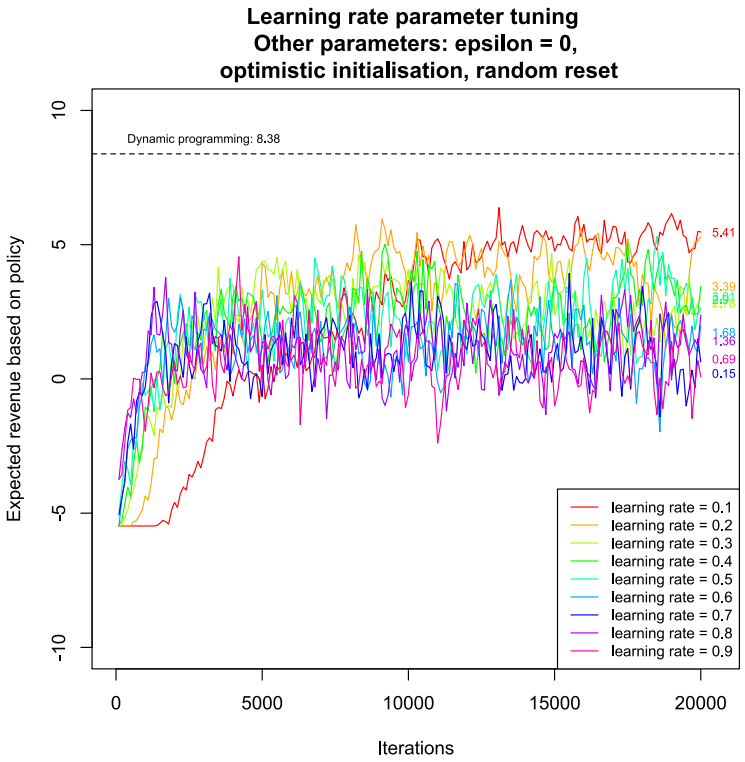
The exploration-exploitation trade-off in machine learning materialises in the epsilon value in Q-learning. When conducting a grid search considering $\epsilon \in \{0, 0.1, 0.2, ..., 0.9\}$ the metrics show no difference in performance between the different values for epsilon. Using optimistic initialisation already guarantees a nice exploration and exploitation trade-off as mentioned in Chapter 4. Thus, it makes sense that the value of epsilon does not affect the performance much.

7.2.4 Learning rate

In this section, the tuning of the learning rate is performed. A higher learning rate means that the Q-values update quickly, and training can be done quicker. However, learning at a faster pace results in a decrease in performance. First, the case where the learning rate is constant is considered. A grid search is performed: learning rate $\in \{0.1, 0.2, ..., 0.9\}$. Figure 7.4 shows the trade-off between training speed and performance. Here the shaded bounds of the minimum and maximum of the ten runs are left out to make the picture clearer. These are included in Figure C.4 in the appendix. It is clear that a lower learning rate takes longer to attain a constant level in a performance metric, but that level is higher than for higher learning rates. The percentage of correct policy decisions is 92% on average, with an expected revenue of 5.4 MUSD. The learning rate considered the best option is 0.1.



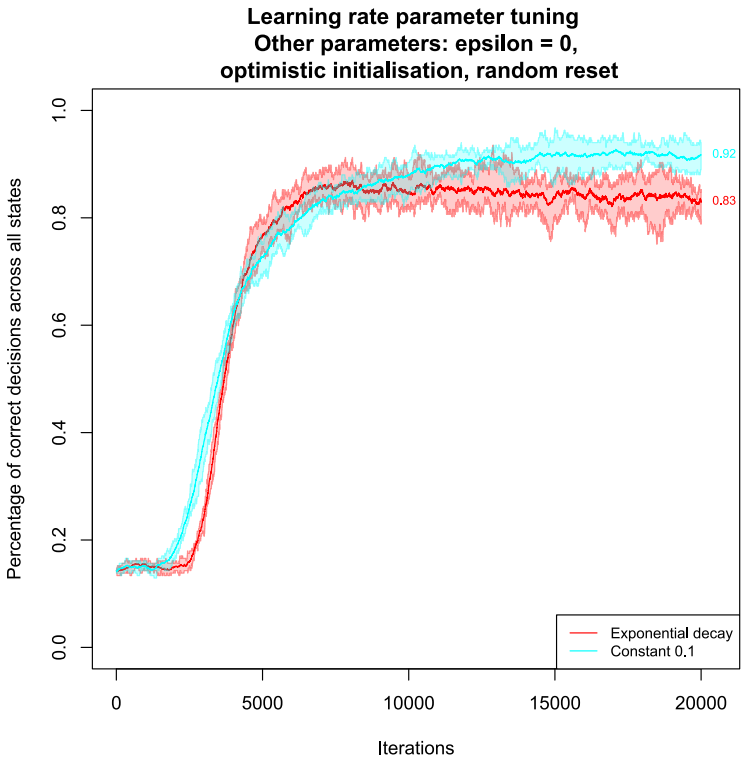
(a) Percentage of correct decisions throughout the training process



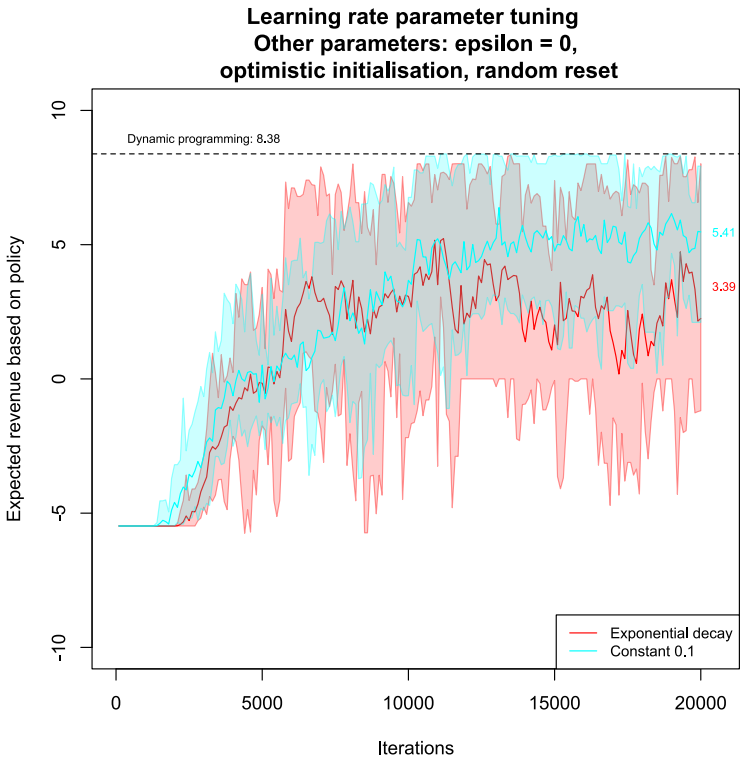
(b) Expected revenue throughout the training process

Figure 7.4: Learning rate tuning: performance metrics throughout the training process on the three-well problem.

Lastly, the learning rate can also be decaying. As mentioned in section 4.3.2, the learning rate must satisfy certain conditions for the proof of convergence of Q-learning to hold. However, for this choice, the Q-learner takes extremely long to run. Even though the condition of the learning rate for the convergence proof is already violated, an analysis is conducted to see the difference in performance for a decaying learning rate in comparison to a constant learning rate. For the exponentially decaying learning rate the following formula was used learning rate$_t$ = $10^{-5} + (0.4 - 10^{-5}) * e^{(-1.5*10^{-4}*t)}$. This choice was made as the learning rate would start around 0.4, decay exponentially, and after 20.000 iterations, the learning rate would be close to zero. The comparison between a constant learning rate of 0.1 and the exponentially decaying learning rate mentioned before is given in Figure 7.5. From this, it is clear that the constant learning rate both has a higher expected value and a higher amount of policy decisions that are the same as dynamic programming. Therefore, the results from this comparison do not prompt further analysis of the various other possibilities for the decaying learning rate.



(a) Percentage of correct decisions throughout the training process



(b) Expected revenue throughout the training process

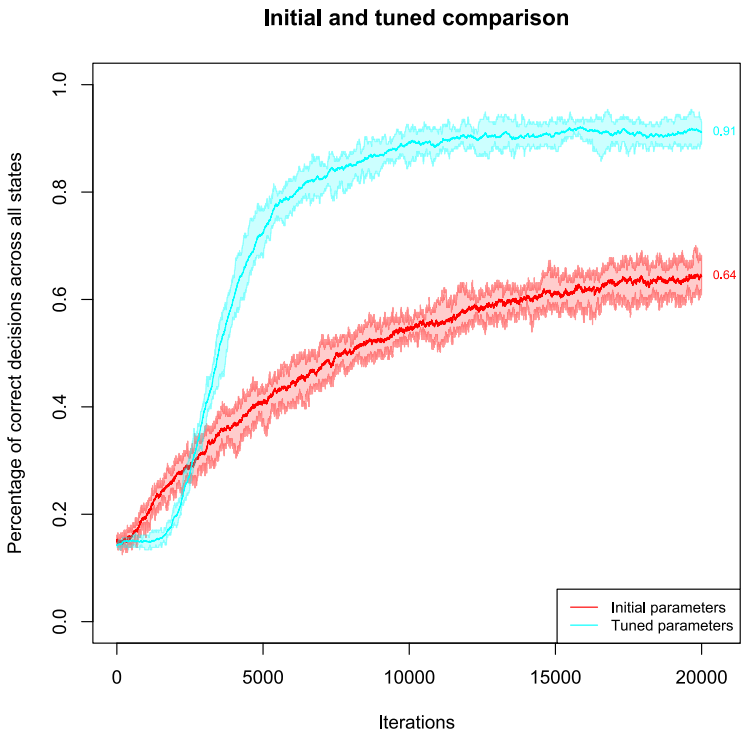
Figure 7.5: Learning rate tuning (exponential decay vs constant): performance metrics throughout the training process on the three-well problem.

7.2.5 Results after tuning

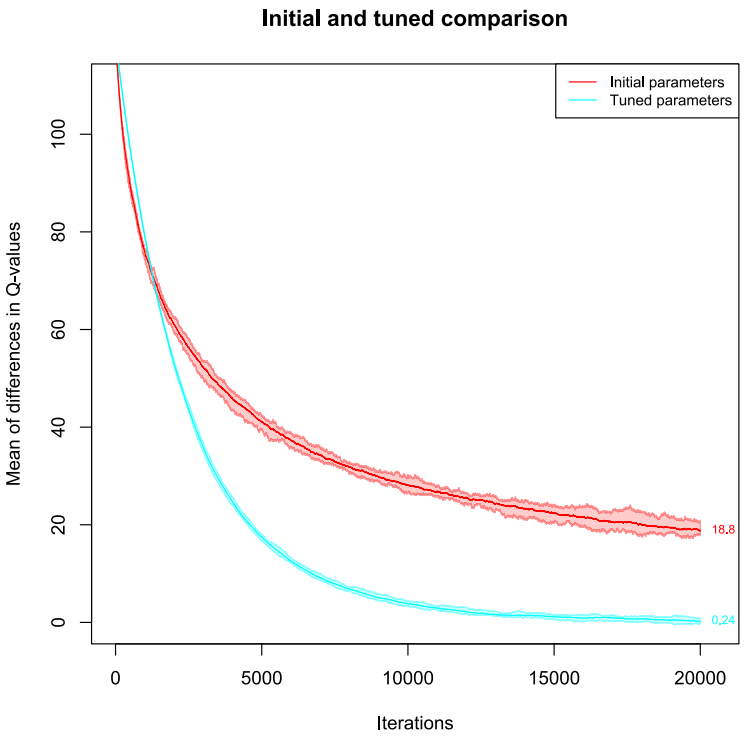
After all tuning analyses have been performed, the tuned parameters are given in Table 7.2. These choices were then used to train a Q-learner on the three-well problem. The results of their performance in comparison to the non-tuned/initial performance is given in Figure 7.6. Observe that the Q-learner does not improve much in performance for the first 2000 iterations. After that, the performance rapidly increases and comes close to the dynamic programming policy.

Parameter	**Choice**
Reset	Reset to random state
Initialisation	Optimistic initialisation
Epsilon	0
Learning rate	0.1

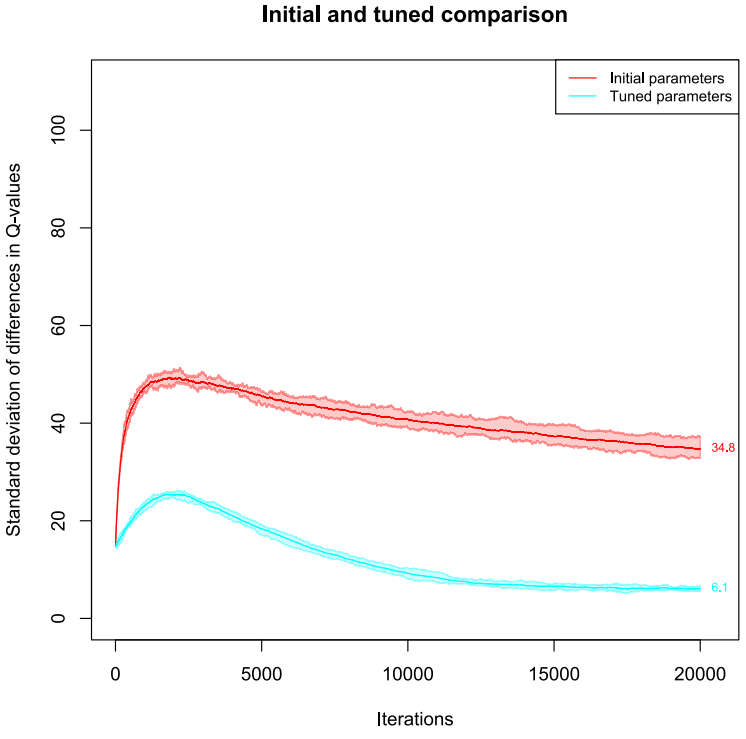
Table 7.2: Tuned choices for parameters in Q-learning



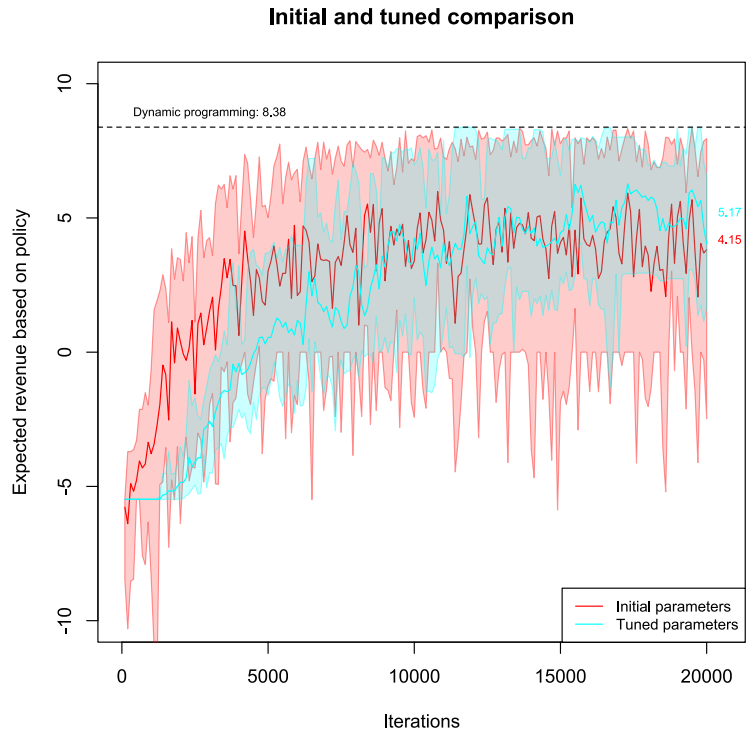
(a) Percentage of correct decisions across all states



(b) Mean of the differences between the true Q-values and the approximated Q values



(c) The standard deviation of the differences between the true Q-values and the approximated Q-values



(d) Expected revenue based on the policy derived from Q-learning

Figure 7.6: performance metrics throughout the training process on the three-well problem with tuned parameters.

7.3 REMEMBERING THE BEST POLICY THUS FAR

That the maximum bounds of the expected revenue based on the Q-learning policy touch the performance of dynamic programming prompted the idea of utilising the common practice in optimisation procedures of remembering the best policy thus far. After the Q-learner has learned most of what there is to learn during the training process, it still goes through more iterations as there is no clear stopping criterion. During these iterations, the Q-table still gets adapted, and thus the policy also changes. These changes can be either good when it increases the expected revenue or bad when it decreases the expected revenue. Due to the stochasticity in the environment, these changes will, with a constant learning rate, never stop. Considering the distribution of the differences, and the fact that the mean of the differences converges to zero as seen in Figure 7.6b, the Q-table has converged to the true Q-table in expected value. The performance during the training process of the Q-learning algorithm when remembering the best policy thus far results in the expected revenue performance given in Figure 7.7. Here it is clear that remembering the performance, in red, leads to higher expected revenue on average: 8.02 MUSD instead of 5.17 MUSD, for regular Q-learning, depicted in blue. Note that these two lines are based on the same ten training cycles. The upper bound for the red shaded area touches the dynamic programming performance line. The confidence bounds are more narrow for remembering the best policy.

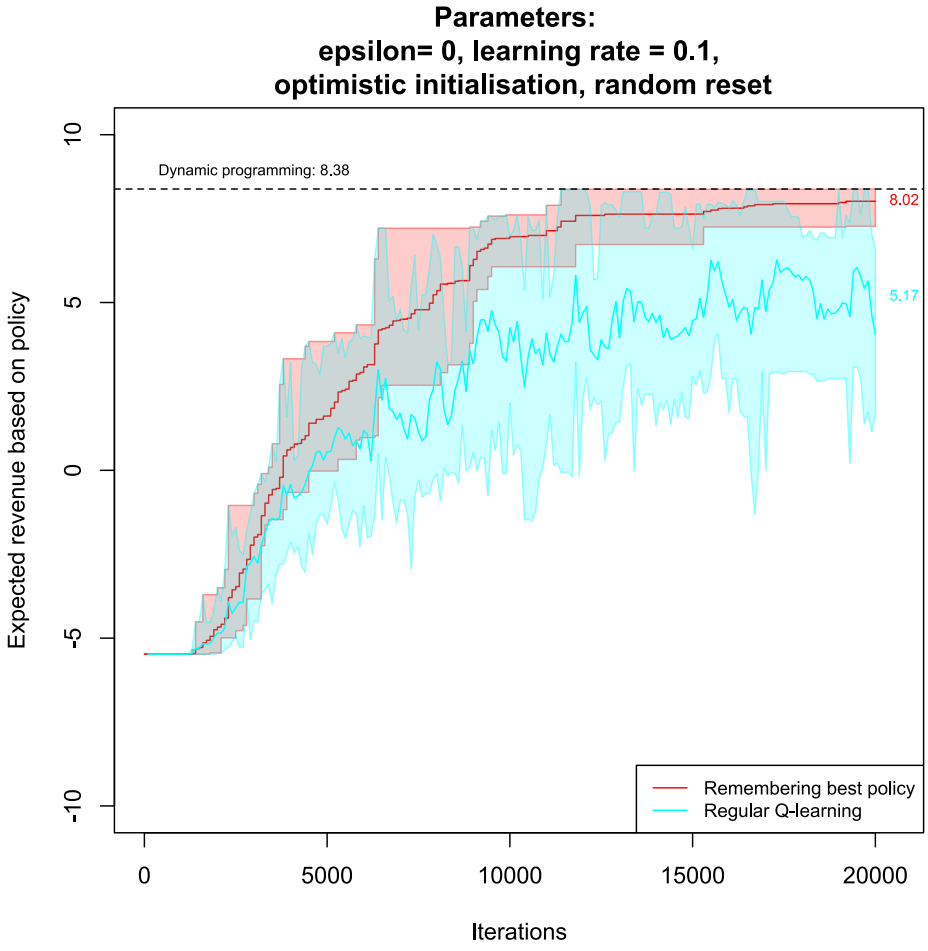


Figure 7.7: Expected revenue based on the policy with regular Q-learning and with Q-learning where the best policy is remembered.

7.4 COMPARISON OF DYNAMIC PROGRAMMING AND Q-LEARNING

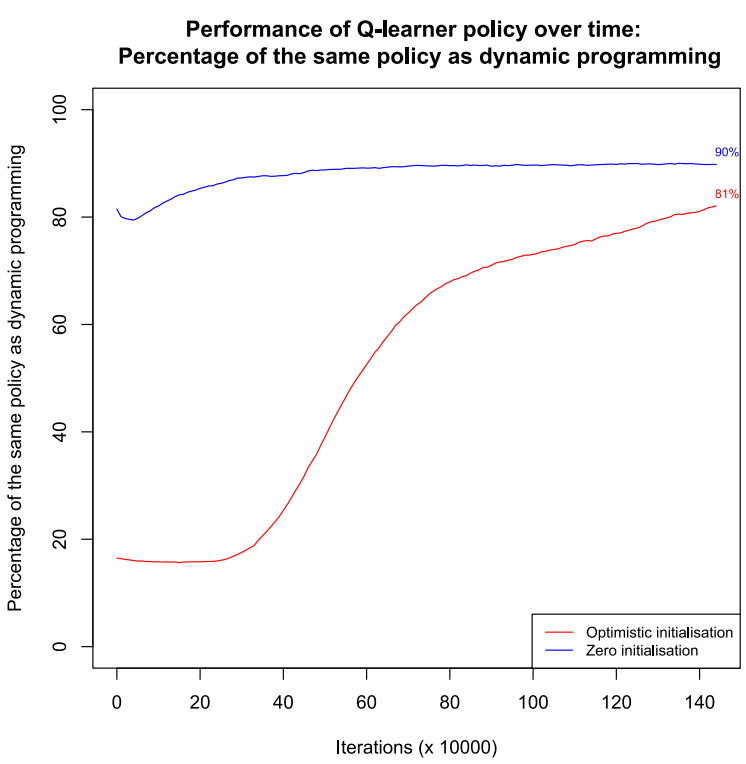
Now that the Q-learning result has been optimised, it can be compared to the dynamic programming algorithm.
First of all, it is clear that Q-learning approximates the Q-values well. When remembering the best policy, the average difference between the expected revenue of the policies based on Q-learning and dynamic programming is 0.36 MUSD. Out of the ten training cycles performed, one reached 8.38 MUSD, and two others were at 8.37 MUSD. Thus, when running the training process multiple times, it is likely that the same performance as dynamic programming would be achieved.
After training the Q-learner, its policies are, on average, 92% the same as the policy based on dynamic programming. The policy differences arise when the differences in Q-values are larger, or at least large enough to tip the scales and favour a different action. The differences between the approximated Q-values and true Q-values from dynamic programming after the agent has been trained are symmetric around zero with a standard deviation of 6.

When considering the time it takes to run both the Q-learning algorithm and the dynamic programming algorithm, it is clear that Q-learning takes longer than dynamic programming. To be precise, dynamic programming takes 2 seconds and Q-learning about 2 minutes. However, it is not recommended to use Q-learning for a simple problem that can be solved with dynamic programming. Dynamic programming, namely, will for sure return the most optimal solution according to the model in less time than Q-learning will. As mentioned in Chapter 4, Q-learning has been proven to converge to the dynamic programming solution in the end, which is also shown in the results here, where the mean of the differences between the approximated Q-values and the true Q-values approaches zero. Nevertheless, dynamic programming has a substantial limitation on the type of problems it can handle. As the number of states the model can take increases to a relatively large problem, dynamic programming becomes infeasible. Q-learning can still handle this bigger model, albeit while taking a long time to run. When improving and thus increasing the model size even further by, for example, modelling variables as continuous variables, Q-learning can also reach its limitations. The next step would be deep Q-learning, which can handle a fully continuous state input. See section 6.3 for computational time analysis of dynamic programming when enlarging the problem.
Note that it is essential to recognise that states that are less often to occur also are modelled less accurately. This is because more iterations would be needed to model them all well. However, this would not increase the expected revenue much.
The uncertainty in which state the well will end up makes for a more unstable Q-learning algorithm. This is compensated by increasing the number of episodes Q-learning is run for.

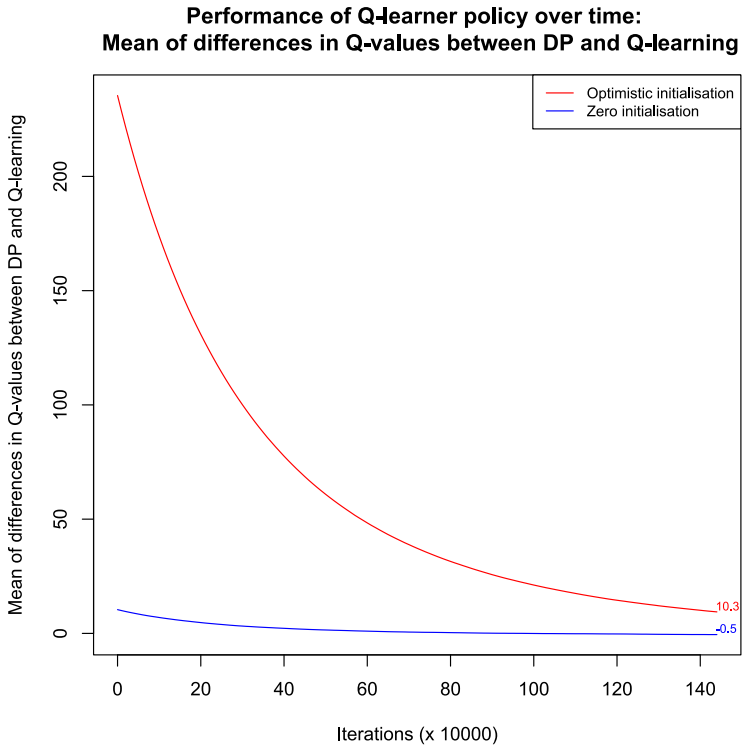
7.5 FIVE-WELL PROBLEM

The next step is to see whether the same type of performance can be achieved with the tuned parameters for the five-well problem. The same three geological factors are used for the five-well problem as in Bickel et al. [2008]. The exact marginal and conditional probabilities are given in Appendix A. Next, the hyperparameters are set to the tuned parameters for the three-well problem as given in Table 7.2. The number of iterations that the Q-learner was trained for is 1440000. The performance results of the Q-learner during the training process according to the four defined measures of performance are given in Figure 7.8. Note that only the performance of one training process is given instead of ten as in the three-well problem because the time it takes to run one training process for 1440000 iterations is around 50 hours. This is important to remember because it can be the case that different training runs have either a better or worse performance. Thus, no robust conclusions can be drawn from this single result.
Also, note that the type of problem is precisely the same as the three-well problem, just with five wells. Because of the significant computation times, the comparison between optimistic and zero initialisation was repeated. Previously, optimistic initialisation was chosen as a better option as the performance is the same but with a more general success guarantee. On the other hand, zero initialisation has, in this type of problem, the advantage of converging more quickly towards the optimal policy.

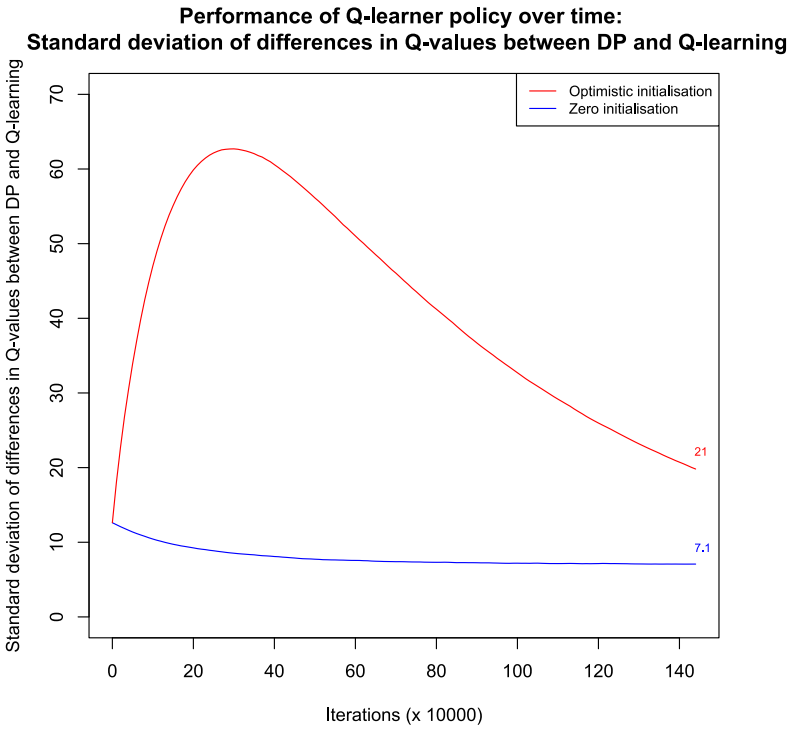
The following conclusions can be drawn from the results in Figure 7.8.
First, zero initialisation, depicted by the blue line, scores better on the four performance metrics than optimistic initialisation in red. This behaviour is similar to the behaviour in the three-well problem. The difference here is that the Q-learner is not trained for enough iterations for the performance of the two initialisation techniques to coincide.
Secondly, the percentage of policy decisions that are the same as dynamic programming is 95% for this specific run of the Q-learner. The Q-table converges to the true Q-table, and the differences are symmetrically distributed around zero with a standard deviation of 7.
Lastly, when considering the expected revenue, as shown in Figure 7.8d, the performance level of dynamic programming at 21.81 MUSD is not attained. However, here it is important to recall that this is only one training run of the Q-learner. Possibly a different run would reach the dynamic programming level. Furthermore, it is also necessary to mention that the parameters were not tuned on the five-well problem but rather on the three-well problem, so tuning could still be performed in order to improve the performance.



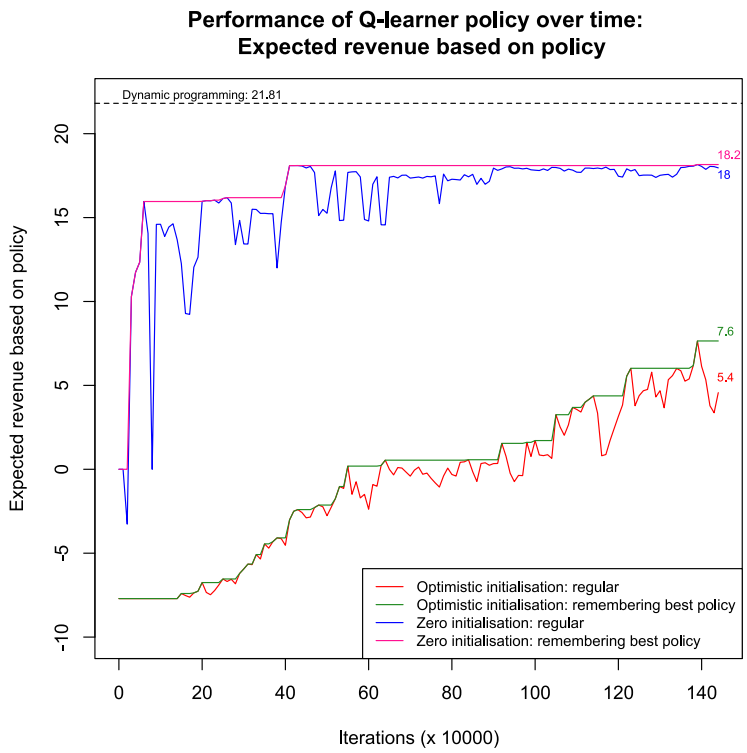
(a) Percentage of correct decisions across all states



(b) Mean of the differences between the true Q-values and the approximated Q values



(c) The standard deviation of the differences between the true Q-values and the approximated Q-values



(d) Expected revenue based on the policy derived from Q-learning

Figure 7.8: Performance metrics throughout the training process with tuned parameters on the five-well problem.

8 RESULTS OF APPLYING THE EXTENDED MODEL TO REAL DATA

After Q-learning has been shown to have similar performance to dynamic programming, it is applied to the more complex extended model as described in the methodology in section 5.3 based on real data, which also models the volume parameters. For this purpose of clarity, this Bayesian network is repeated in Figure 8.1, with the full page view in Appendix D. The marginal distributions and (un)conditional rank correlations are specified through the data and expert assessments provided by Equinor. After the model has been established, Q-learning is applied to the model and a sequential drilling strategy is obtained. For this purpose, the hyperparameters of Q-learning are set according to Table 8.1. The choices for the hyperparameters were derived from the analyses performed in the previous chapters, where multiple options were considered. Finally, the options with the best performance and computational time were chosen. Note that zero initialisation was chosen to be used here because of the results in the five-well problem, where the Q-learner with zero initialisation outperforms optimistic initialisation in reasonable time. As zero initialisation is paired with an exploration parameter epsilon, this is chosen to be 0.2.

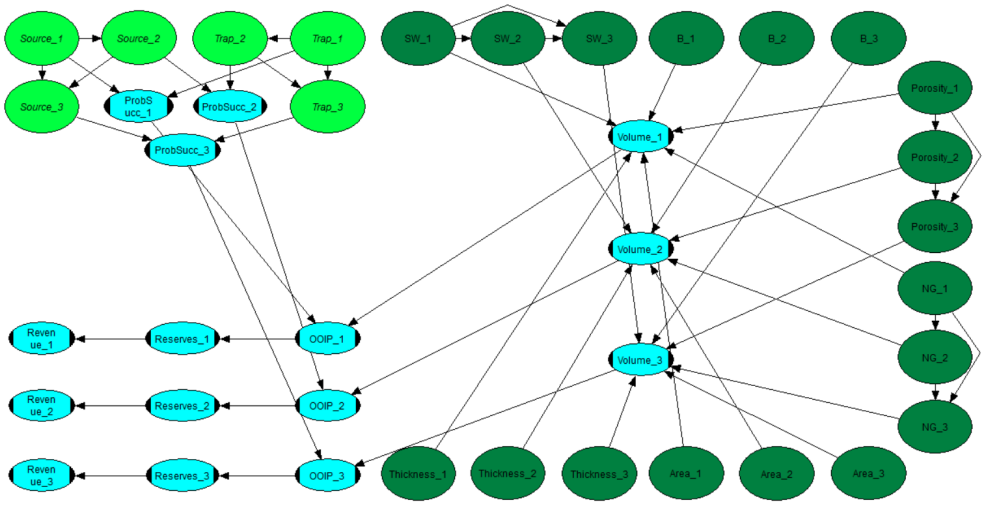


Figure 8.1: Bayesian network structure for the extended model for three wells with three geological factors and six volume parameters of which the dependencies between three of them are considered.

Parameter	**Choice**
Reset	Reset to random state
Initialisation	Zero initialisation
Epsilon	0.2
Learning rate	0.1
Remembering best policy thus far	Yes

Table 8.1: Hyperparameters for the real data case

The expected value based on the policy of the Q-learner on the real data case during the learning process of 500.000 iterations is visualised in Figure 8.2. The expected revenue at the start of the training process is zero, which is in line with the initialisation of the Q-table at zero. With that Q-table, the policy is to not drill any prospects so no profit or loss occurs. The expected value rapidly increases as the training of the Q-learner proceeds, and after 100.000 iterations, it stays fairly constant around the value of 135 MUSD. After 500.000 iterations, with remembering the best policy, the expected value is 135.24 MUSD. From this, it is clear that it is not necessary to train the Q-learner for 500.000 iterations to attain the expected value of around 135 MUSD, but instead stopping after 100.000 iterations leads to the same performance. No dynamic programming solution is available for the expected revenue performance can be compared to. However, it can still be compared to the expected value of the policy of knowing everything and drilling only successful wells that generate revenue. This would give an expected value of 139.56 MUSD, which is an increase of about 4 MUSD compared to the Q-learner's policy. However, this is a substantial difference compared to the rise visible in the three-well problem, where it increased from 8 to 40 MUSD, or the five-well problem, where it increased from 21 to 72 MUSD, it is relatively a small increase. This could be explained because of the input data used. The prospects in the dataset all have a high individual expected revenue as they are included in the drilling procedure and would only be included if Equinor expects them to be a success with a large enough probability and with a significant enough return. In contrast, the three-well problem had all negative individual expected revenues. The order of magnitude between the cost and reward is also quite skewed in the real data set. The costs are about 10 MUSD per well if the well is a failure, and the rewards lie around 1500 MUSD if it is a success. As the order of magnitude is very different, it may point to a mistake in the cost calculation or some factor being missed in the revenue calculation. However, it may also be the case that this is just how it is for some prospects. Further analysis of and expert input on the cost calculation process is recommended for future uses.
The policy of drilling all prospects, regardless of previous outcomes, also leads to an expected value of 135.24 MUSD. Thus, in this case, where the rewards are much larger in quantity than the losses and the probabilities of success are all relatively high, adapting the strategy dynamically and learning from the outcomes of the previously drilled wells adds only modest value, even considering high correlations between the particular geological factors.

It takes approximately 2.5 days to run the Q-learner for 500.000 iterations.

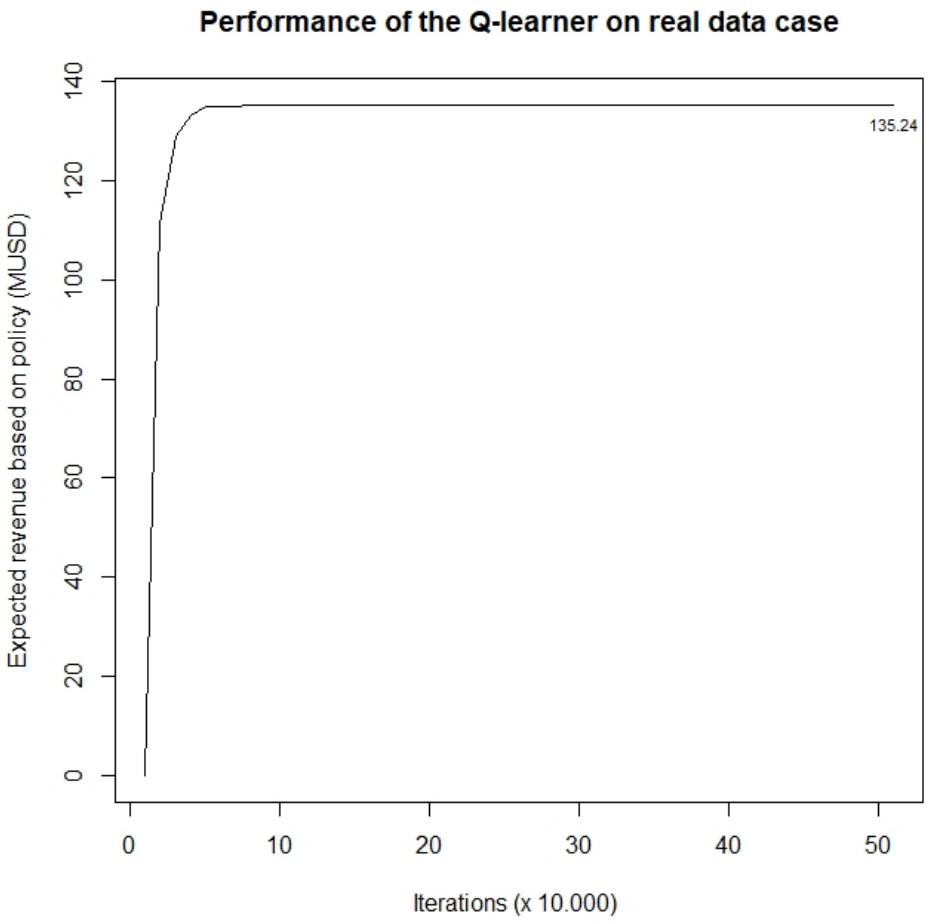


Figure 8.2: Performance during the training process of Q-learning on the real data model with parameters as specified in Table 4.1

9 EXTENDED MODEL ON SYNTHETIC DATA CASE

In Chapter 8, it has been shown that there are some real data cases where the addition of dynamically updating the drilling strategy does not create value. Therefore, a synthetic data case based on the real data case with more favourable characteristics as explained in more detail in Section 5.4 is used. The Q-learning algorithm was run with the same hyperparameters used in the real data case presented in Table 8.1.

The expected value based on the policy during the training process of 100.000 iterations is visualised in Figure 9.1. The expected revenue increases and eventually settles at an expected revenue of 345 MUSD. The individual expected values of the three wells are 184 MUSD, -324 MUSD, and 383 MUSD. The current drilling strategy of the oil and gas industry is to only drill wells with a positive expected value, thus well one and well three in this case. This policy would lead to an expected revenue of 567 MUSD. Thus, the sequentially updating decision policy from the Q-learning algorithm has a lower expected revenue than the current policy.
Theoretically, this synthetic case has all the necessary characteristics for the Q-learner to obtain a policy with a higher expected revenue than the industry's policy. However, in practice, these are not the results that are obtained. This can have various reasons. For example, perhaps the Q-learner should be run for a more extended period. Given that a well is drilled, there are 32 possible outcomes, each represented by a different state. In the three-well and five-well problems there were only eight possible outcomes. This difference stems from the number of factors or parameters and their dependencies considered as that increased from three to five. Therefore, it is logical for the Q-learner to need more iterations to get a reasonable estimate of the Q-values. The 200.000 iterations performed in this analysis are apparently not enough lead to a policy where at least well one and well three are drilled as they have a positive expected revenue and well two is drilled in some instances of success in the other wells. Moreover, a considerable improvement could also be seen when tuning the hyperparameters to this specific problem as that has been shown to increase the performance of Q-learning in the smaller problem in Chapter 7. Unfortunately, these additional analyses were not performed during this project due to time constraints.

It takes two days for 200.000 iterations and the performance calculations to run. This is longer than the computational time of the real data case, but this is only due to the calculation of the expected revenue being performed once every 100 iterations, whereas, in the real data case, it was performed once every 10.000 iterations. This increase in the frequency for computing the expected revenue is used to make sure the policy with the highest expected revenue was not missed.

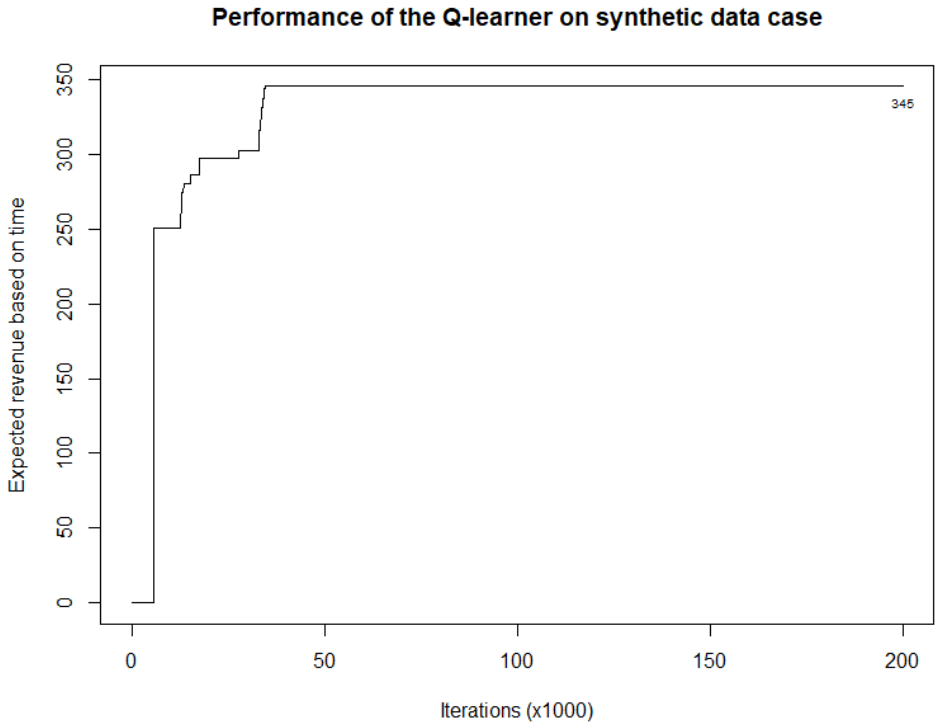


Figure 9.1: Performance during the training process of Q-learning on the extended model on the synthetic data case.

10 SENSITIVITY ANALYSIS ON DISCRETE NODES IN UNINET

This section analyses the effects the use of discrete variables in Uninet, built for continuous variables, has on the conditional probabilities between the random variables representing geological factors of wells. For this purpose, a Bayesian network with three nodes is used, where the graph structure is fully connected. This graph was modelled in Uninet with a non-parametric Bayesian network and in Netica with a discrete Bayesian network. Netica is software built specifically for discrete Bayesian networks.
The wells considered are called well A, well B, and well C. Their marginal probabilities of success are respectively 0.730, 0.792, and 0.735. Their upper diagonal pairwise Pearson correlation matrix is given in Equation 10.1. The structure of the Bayesian networks will be the same in both Netica and Uninet: fully connected with well A being a root node, well B having well A as a parent, and well C being a leaf node with both other wells as their parents.

$$\begin{pmatrix} \rho_{1,2} & \rho_{1,3} \\ & \rho_{2,3} \end{pmatrix} = \begin{pmatrix} 0.15 & 0.27 \\ & 0.31 \end{pmatrix} \tag{10.1}$$

For the discrete model in Netica, the conditional probability tables (CPTs) were calculated using the joint distribution. This joint distribution is obtained in the same way as before in Bickel et al. [2008] and Bickel and Smith [2006] with the assumption of minimal dependence. This method is described in Appendix B. The CPTs constructed through this are given in Table 10.1. This Bayesian network is visualised in Figure 10.1.

For the discrete model using copulae in Uninet, the marginals were specified as Bernoulli random variables with their respective probabilities of success. The (un)conditional rank correlations specified for the bivariate normal copulae on the edges of the Bayesian network are computed from the Pearson correlation matrix in the same way described in Section 4.2.2. This leads to the non-parametric Bayesian network as visualised in Figure 10.2. The CPT extracted from the non-parametric Bayesian network through analytical conditioning in the Uninet software is given in Table 10.2.

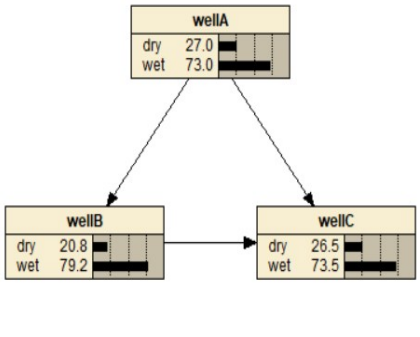


Figure 10.1: Bayesian network in Netica

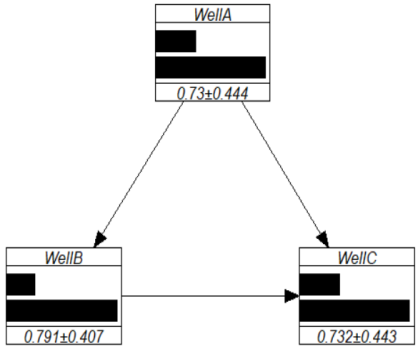


Figure 10.2: Bayesian network in Uninet

Now that the models are set up, they can be compared. The first noticeable difference lies in the marginal distributions in the non-parametric Bayesian network of

Uninet. Even though the marginal distributions are specified exactly for each well, the marginals visualised in the graph are off by 0.001 and 0.003 for respectively well B and well C. Well A does seem to have the same marginal distribution as specified. This may be due to how Uninet obtains the analytical conditioning calculations. When Uninet enters the analytical conditioning mode, it transforms a standard normal distribution into a uniform distribution, and that uniform distribution to the discrete Bernoulli distribution using the inverse cumulative distribution function of the Bernoulli random variable. This transformation is only guaranteed to be rank preserving if the distribution function is both invertible and continuous. The Bernoulli distribution function is not continuous, as it is discrete, so problems may arise in this aspect. See the Uninet help guide for more information on the process of analytical conditioning. As there may be complications from using discrete random variables in a non-parametric Bayesian network with copulae, it is interesting to see what those complications may be for our application. Therefore, the CPTs of the discrete Bayesian network in Netica and the non-parametric Bayesian network in Uninet are compared.
The differences between the two CPTs are as large as 17 percentage points. Generally, the probability of the child well being wet conditional on the parent wells being dry is higher in the Uninet model than in Netica. On the other hand, the probability of the child being wet conditional on the parent wells being wet is lower than in Netica. Overall, considering Uninet uses a model designed for continuous random variables for discrete random variables, it does a decent job at modelling the dependencies in a way that is similar to Netica. Note that this is only one specific case, and the specific differences between the two models will vary case by case. Therefore, more extensive sensitivity analysis and comparison should be performed to make any statements regarding the actual differences and consequences of using discrete random variables in the non-parametric Bayesian network in Uninet. These differences will be kept in mind for now, but the discrete variables in the non-parametric Bayesian network are still used.

Well A = wet	Well A = dry
0.73	0.27

(a) CPT of well A

Well A	Well B = wet	Well B = dry
wet	0.83	0.17
dry	0.69	0.31

(b) CPT of well B

Well A	Well B	Well C = wet	Well C = dry
wet	wet	0.85	0.15
dry	wet	0.64	0.36
wet	dry	0.6	0.4
dry	dry	0.32	0.68

(c) CPT of well C

Table 10.1: CPT of three-well fully connected network from Netica

Well A = wet	Well A = dry
0.73	0.27

(a) CPT of well A

Well A	Well B = wet	Well B = dry
wet	0.813	0.187
dry	0.793	0.207

(b) CPT of well B

Well A	Well B	Well C = wet	Well C = dry
wet	wet	0.801	0.199
dry	wet	0.671	0.329
wet	dry	0.651	0.349
dry	dry	0.493	0.507

(c) CPT of well C

Table 10.2: CPT of three-well fully connected network from Uninet

11 CONCLUSION

This thesis aimed to create a model that can be used as a dynamic decision support tool for determining the drilling strategy when drilling hydrocarbons. The goal was that this model would accommodate for adding more uncertainty, dependence, and continuity in the calculation of the utility function to mimic reality more closely. To accomplish this goal, the following research questions were considered.

How can sequential hydrocarbon drilling be approached with a more realistic dynamic decision support tool so that it can be deployed in practice?

- What is the added value of Bayesian networks in doing so?
- What alterations need to be made to the optimisation procedure to accommodate the more complex model?

The goals set out at the start of this thesis have been accomplished. A model that can use real-life data as input and return a policy that dynamically updates based on new information has been developed based on the foundations laid by previous researchers. The developed method consists of two parts: the model for the uncertainties and dependencies and the algorithm that uses this model to obtain a sequential decision policy. For the first part, non-parametric Bayesian networks are used. The Bayesian network contains both discrete and continuous non-parametric nodes for the application of drilling prospects. These networks are very flexible as both the structure and the variables in the network can be easily altered. The dependencies between the variables in the Bayesian network are modelled using bivariate Gaussian copulae. The model can be built based on both data and expert assessments. Because of the mathematical properties of the Gaussian copula, the expert assessments for the dependencies are simple and intuitive as they can be given in the form of Pearson correlation coefficients. For the second part, it has been shown that the reinforcement learning algorithm Q-learning can lead to a policy with the same performance as the policy obtained from dynamic programming on a smaller problem. For the larger and more realistic problems, two cases were considered. The first was a real data case provided by Equinor that only contains very profitable wells, and thus they should all be drilled, regardless of the outcome of other wells. This meant that sequentially updating the policy was not beneficial. The second larger case explored contained more favourable characteristics and, in theory, would outperform other policies. However, given the time constraints, the Q-learning algorithm could not be tuned to this case and could not be run for an extended period. Therefore, the results did not reflect the capabilities of this thesis's proposed method but rather highlighted some of the limitations of using Q-learning.

In conclusion, the key contributions of this thesis to the sequential decision analysis procedure in the application of drilling wells are twofold. First, a new model for the dependencies between geological properties of a hydrocarbon reservoir in the form of a non-parametric Bayesian network is created that is more flexible and realistic. Secondly, the use of a Q-learning reinforcement learning algorithm trained through simulations of the Bayesian network has been explored as an alternative to dynamic programming for obtaining a sequential drilling strategy.

Based on the result of this study, it is recommended that the idea of combining Bayesian networks with reinforcement learning to solve sequential decision making problems is researched further and deployed into practice. Recommendations for further research topics can be found in Chapter 12.

Although this thesis tackled the issue of drilling wells, the developed model can easily be applied in other contexts where decisions should be updated sequentially, and the dependencies can be modelled through a Bayesian network.

12 DISCUSSION

Even though this thesis research completed the goals set at the start, improvements can always be made. These improvements can be carried out through further research. The upgrades are split into two categories: mathematical enhancements to make the model more accurate and additions to the model to make it more realistic for the application of drilling wells.

First, some extensions to improve the model from a mathematical standpoint are discussed.
The most considerable improvement can be made by switching from a discrete tabular reinforcement learning to a deep reinforcement learning algorithm which can handle continuous variables to obtain the decision policy. Tabular reinforcement learning, such as Q-learning, must encounter every state in the environment multiple times before accurately estimating the best action. This means that the algorithm does not know what to do in a new state and takes a random decision. In comparison, deep reinforcement learning can relate the new state to a previously encountered state through similarities in the states and use that information to make an educated guess about the best action. This switch would mean that the model no longer needs to be discretised, and the complete information from the continuous nodes can be exploited. The reader is referred to [Wiering and Van Otterlo, 2012, Chapter 7] for an overview of reinforcement learning algorithms that can deal with continuous nodes.
Another improvement could be made by separating the non-parametric Bayesian network with both continuous and discrete nodes into two Bayesian networks. One Bayesian network would be non-parametric and encompass the modelling of the dependencies between the continuous parameters that influence the estimation of the volume recovered. The second Bayesian network would be discrete and model the dependencies between the geological factors. This would mitigate the differences that came forward during the sensitivity analysis of using discrete nodes and copulae, which are made for continuous nodes, to model the dependencies. At the same time, the advantages a non-parametric Bayesian network offers in modelling continuous non-parametric variables are preserved. Because both the creation of the model and the obtaining of the decision policy are currently implemented in R, this separation is relatively easy to do all from within R. Note that this separation is only possible because the discrete and continuous parts of the network constructed in this thesis are only connected through functional nodes, and thus, this improvement can not be generalised to all other applications which have other network structures.
Finally, the discretisation scheme used for Q-learning can be investigated and extended. For example, moment matching can be used to obtain a different grid. It is also recommended that a sensitivity analysis is performed on the various choices for discretisation schemes and which values are used as a representative value to condition the network. Due to implementation constraints, this was out of the scope of this thesis.

Besides these general mathematical improvements, additions to the model can be made so that it is more realistic. These improvements are aimed specifically toward the application of sequential decision analysis in drilling prospects. Still, they can be used as inspiration for what else can be included in the Bayesian network for

other applications. These recommendations can be easily added to the Bayesian network model as it is currently constructed and implemented.

First, the cost analysis of drilling wells is modelled quite simplistically through a fixed cost per well plus a variable production cost per barrel of hydrocarbons. This may be adapted to fit the more elaborate cost scheme of drilling wells. For example, the costs may be dependent on the geological orientation of the wells and thus include transportation costs. Or some geological aspect of the reservoir may influence the ease, hence the cost of producing the hydrocarbons.

Second, the three considered geological factors can be split up into six original geological factors (favourable porosity, permeability, trap geometry, trap seal, source presence, and source migration). It should be emphasised that this is only beneficial if additional information on the dependencies between these separated geological factors is available.

Lastly, different types of wells could be modelled, such as information wells or production wells. Information wells would give more information about the region but cannot be produced anymore. On the other hand, production wells would extract more hydrocarbons but provide less information about the geological aspects.

BIBLIOGRAPHY

Bertsekas, D. P. (2005). *Dynamic Programming Optimal Control, Volume I, Third edition*. Athena Scientific.

Bickel, J. E. and Smith, J. E. (2006). Optimal sequential exploration: A binary learning model. *Decision Analysis*, 3(1):16–32.

Bickel, J. E., Smith, J. E., and Meyer, J. L. (2008). Modeling dependence among geologic risks in sequential exploration decisions. *SPE Reservoir Evaluation & Engineering*, 11(02):352–361.

Chen, Y.-C., Wheeler, T. A., and Kochenderfer, M. J. (2017). Learning discrete bayesian networks from continuous data. *Journal of Artificial Intelligence Research*, 59:103–132.

Colombo, D., Maathuis, M. H., et al. (2014). Order-independent constraint-based causal structure learning. *J. Mach. Learn. Res.*, 15(1):3741–3782.

Constantinou, A. C. (2019). Evaluating structure learning algorithms with a balanced scoring function. *arXiv preprint arXiv:1905.12666*.

Cooke, R. et al. (1991). *Experts in uncertainty: opinion and subjective probability in science*. Oxford University Press on Demand.

Cooke, R. M. and Goossens, L. H. (2000). Procedures guide for structural expert judgement in accident consequence modelling. *Radiation Protection Dosimetry*, 90(3):303–309.

Even-Dar, E., Mansour, Y., and Bartlett, P. (2003). Learning rates for q-learning. *Journal of machine learning Research*, 5(1).

Geiger, D., Verma, T., and Pearl, J. (1990a). d-separation: From theorems to algorithms. In *Machine Intelligence and Pattern Recognition*, volume 10, pages 139–148. Elsevier.

Geiger, D., Verma, T., and Pearl, J. (1990b). Identifying independence in bayesian networks. *Networks*, 20(5):507–534.

Gosavi, A. (2015). Solving markov decision processes via simulation. In *Handbook of Simulation Optimization*, pages 341–379. Springer.

Hanea, A. M. (2008). Algorithms for non-parametric bayesian belief nets.

Hanea, A. M., Gheorghe, M., Hanea, R., and Ababei, D. (2013). Non-parametric bayesian networks for parameter estimation in reservoir simulation: a graphical take on the ensemble kalman filter (part i). *Computational geosciences*, 17(6):929–949.

Hendricks, K. and Kovenock, D. (1989). Asymmetric information, information externalities, and efficiency: the case of oil exploration. *The RAND Journal of Economics*, pages 164–182.

Howard, R. A. (1960). Dynamic programming and markov processes.

Johnson, S. G. (2022). lbfgs: Low-storage bfgs in nloptr: R interface to nlopt. https://rdrr.io/cran/nloptr/man/lbfgs.html (Accessed: 15-05-2022).

King, G. (2020). Estimation of stock tank oil originally in-place, stooip using the volumetric method, introduction to petroleum and natural gas engineering, the pennsylvania state university. https://www.e-education.psu.edu/png301/node/607 (Accessed: 12-05-2022).

Kjemperud, A. (NA). Prospect evaluation. http://www.ccop.or.th/ppm/document/CHEXV3/CHEXV3DOC05_kjemperud.pdf (Accessed: 16-05-2022).

Lake, L. W. and Fanchi, J. R. (2005). Petroleum engineering handbook: General engineering.

Lucas, P., Gámez, J. A., and Cerdan, A. S. (2007). *Advances in Probabilistic Graphical Models*, volume 213. Springer.

Martinelli, G., Eidsvik, J., Hauge, R., and Førland, M. D. (2011). Bayesian networks for prospect analysis in the north sea. *AAPG bulletin*, 95(8):1423–1442.

Martinelli, G., Eidsvik, J., Hokstad, K., and Hauge, R. (2014). Strategies for petroleum exploration on the basis of bayesian networks: A case study. *SPE Journal*, 19(04):564–575.

McNeil, A. J., Frey, R., and Embrechts, P. (2015). *Quantitative risk management: concepts, techniques and tools-revised edition*. Princeton university press.

Melo, F. S. (2001). Convergence of q-learning: A simple proof. *Institute Of Systems and Robotics, Tech. Rep*, pages 1–4.

Mesbahzadeh, T., Miglietta, M., Mirakbari, M., Soleimani Sardoo, F., and Abdolhoseini, M. (2019). Joint modeling of precipitation and temperature using copula theory for current and future prediction under climate change scenarios in arid lands (case study, kerman province, iran). *Advances in Meteorology*, 2019.

Morales, O., Kurowicka, D., and Roelen, A. (2008a). Eliciting conditional and unconditional rank correlations from conditional probabilities. *Reliability Engineering & System Safety*, 93(5):699–710.

Morales, O., Kurowicka, D., and Roelen, A. (2008b). Eliciting conditional and unconditional rank correlations from conditional probabilities. *Reliability Engineering & System Safety*, 93(5):699–710.

Nelsen, R. B. (2007). *An introduction to copulas*. Springer Science & Business Media.

Otterlo, M. v. and Wiering, M. (2012). Reinforcement learning and markov decision processes. In *Reinforcement learning*, pages 3–42. Springer.

Powell, W. B. (2007). *Approximate Dynamic Programming: Solving the curses of dimensionality*, volume 703. John Wiley & Sons.

Quinn, C. et al. (2007). The health-economic applications of copulas: methods in applied econometric research. *Health, Econometrics and Data Group (HEDG) Working Paper*, 7:22.

Recht, B. (2018). A tour of reinforcement learning: The view from continuous control. *arXiv preprint arXiv:1806.09460*.

Rüschendorf, L. (2013). Copulas, sklar's theorem, and distributional transform. In *Mathematical risk analysis*, pages 3–34. Springer.

Russell, S. J. (2010). *Artificial intelligence a modern approach*. Pearson Education, Inc.

Sanni, M. (2018). *Petroleum engineering: principles, calculations, and workflows*, volume 237. John Wiley & Sons.

Scanagatta, M., Salmerón, A., and Stella, F. (2019). A survey on bayesian network structure learning from data. *Progress in Artificial Intelligence*, 8(4):425–439.

Schweizer, B. and Sklar, A. (1974). Operations on distribution functions not derivable from operations on random variables. *Studia Mathematica*, 52(1):43–52.

Sklar, A. (1973). Random variables, joint distribution functions, and copulas. *Kybernetika*, 9(6):449–460.

Sklar, M. (1959). Fonctions de repartition an dimensions et leurs marges. *Publ. inst. statist. univ. Paris*, 8:229–231.

Spaan, M. T. (2012). Partially observable markov decision processes. In *Reinforcement Learning*, pages 387–414. Springer.

Sutton, R. S. and Barto, A. G. (2018). *Reinforcement learning : an introduction*. Westchester Publishing Services.

TradingEconomics (2022). Petroleum geology: An introduction. https://geoinfo.nmt.edu/faq/energy/petroleum/Petroleum_geology_intro.pdf (Accessed: 16-05-2022).

Tsamardinos, I., Brown, L. E., and Aliferis, C. F. (2006). The max-min hill-climbing bayesian network structure learning algorithm. *Machine learning*, 65(1):31–78.

UCube by Rystad Energy, . (2015). What it costs to produce oil. Visualized b CNN at https://money.cnn.com/interactive/economy/the-cost-to-produce-a-barrel-of-oil/index.html?iid=EL (Accessed: 20-05-2022).

Watkins, C. J. and Dayan, P. (1992). Q-learning. *Machine learning*, 8(3):279–292.

Watkins, C. J. C. H. (1989). Learning from delayed rewards.

Wiering, M. A. and Van Otterlo, M. (2012). Reinforcement learning. *Adaptation, learning, and optimization*, 12(3):729.

APPENDICES

A PROBABILITIES USED IN THE THREE AND FIVE WELL PROBLEM TO VALIDATE Q-LEARNING

Prospect number	Marginal probability of success of Reservoir	Marginal probability of success of Trap	Marginal probability of success of Seal
1	0.73	0.81	0.59
2	0.77	0.87	0.73
3	0.73	0.83	0.87
4	0.65	0.81	0.63
5	0.55	0.57	0.57

Table A.1: Marginal probabilities of success of the three geological factors in the five well problem.

i\j	2	3	4	5
1	0.8	0.77	0.7	0.6
2		0.78	0.72	0.61
3			0.68	0.61
4				0.65
5				

Table A.2: Pairwise condition probabilities of success for the geological factor source P(S_i= success $|S_j$= success)

i\j	2	3	4	5
1	0.95	0.9	0.92	0.69
2		0.9	0.91	0.65
3			0.9	0.65
4				0.7

Table A.3: Pairwise condition probabilities of success for the geological factor reservoir P(R_i= success $|R_j$= success)

i\j	2	3	4	5
1	0.87	0.94	0.80	0.80
2		0.95	0.80	0.76
3			0.72	0.65
4				0.85

Table A.4: Pairwise condition probabilities of success for the geological factor trap P(T_i= success $|T_j$= success)

B CONSTRUCTING JOINT DISTRIBUTION

This section states the procedure to go from marginal and pairwise probability assessments of one geological factor and combine them into a joint distribution under the assumption of minimal dependence. This is a direct duplication of the method described in [Bickel et al., 2008, p. 3-4], and is only given here for easy reference because this method is used in this thesis.

Let $\mathbf{w} = (w_1, w_2, ..., w_N)$ be a vector of N binary random variables representing the success or failure of the geological factor w at the N wells. The notation is that $w_i = 1$ if geological factor w is a success at well i, and $w_i = 0$ if it is a failure. The marginal probabilities are denoted as $p_i = p(w_i = 1)$. The pairwise joint probabilities are denoted as $p_{ij} = p(w_i = 1, w_j = 1) = p(w_i = |, w_j = 1) \cdot p(w_j = 1)$, which can be calculated by combining the marginal probability and a pairwise conditional probability.
The joint distribution, represented by $\pi(\mathbf{w})$, is selected to minimise the amount of dependence in the joint distribution, which is the same as being maximally conservative about the amount of information one well outcome provides about other locations. The dependence is measured by mutual information defined as the relative entropy of Kullback-Liebler distance between $\pi(\mathbf{w})$ and the reference distribution $\pi_0(\mathbf{w})$ that assumes independence, given in Equation B.2. The mutual information can be found in Equation B.1.

$$D(\pi, \pi_0) = \sum_{\mathbf{w}} \pi(\mathbf{w}) \ln \left(\frac{\pi(\mathbf{w})}{\pi_0(\mathbf{w})} \right) \tag{B.1}$$

$$\pi_0(\mathbf{w}) = \prod_i (p_i)^{w_i} (1 - p_i)^{1 - w_i} \tag{B.2}$$

To represent the constraints on the joint distribution, the following functions are defined:

$$\Omega_0(\mathbf{w}) = 1 \tag{B.3}$$
$$\Omega_i(\mathbf{w}) = w_i \qquad \text{for all } i \tag{B.4}$$
$$\Omega_{ij}(\mathbf{w}) = w_i w_j \qquad \text{for all } i, j \tag{B.5}$$

Let $E_\pi(f(\mathbf{w}))$ denote the expectation of some function $f(\mathbf{w})$ when $\mathbf{w}$ has distribution π. Using this the optimisation problem for determining the joint probabilities can be written as

$$\min_{\pi} \sum_{\mathbf{w}} \pi(\mathbf{w}) \ln \left(\frac{\pi(\mathbf{w})}{\pi_0(\mathbf{w})} \right) \tag{B.6}$$

subject to:

$$E_\pi[\Omega_0(\mathbf{w})] = 1 \tag{B.7}$$

$$E_\pi[\Omega_i(\mathbf{w})] = p_i \qquad \text{for all } i \tag{B.8}$$

$$E_\pi[\Omega_{ij}(\mathbf{w})] = p_{ij} \qquad \text{for all } i, j \tag{B.9}$$

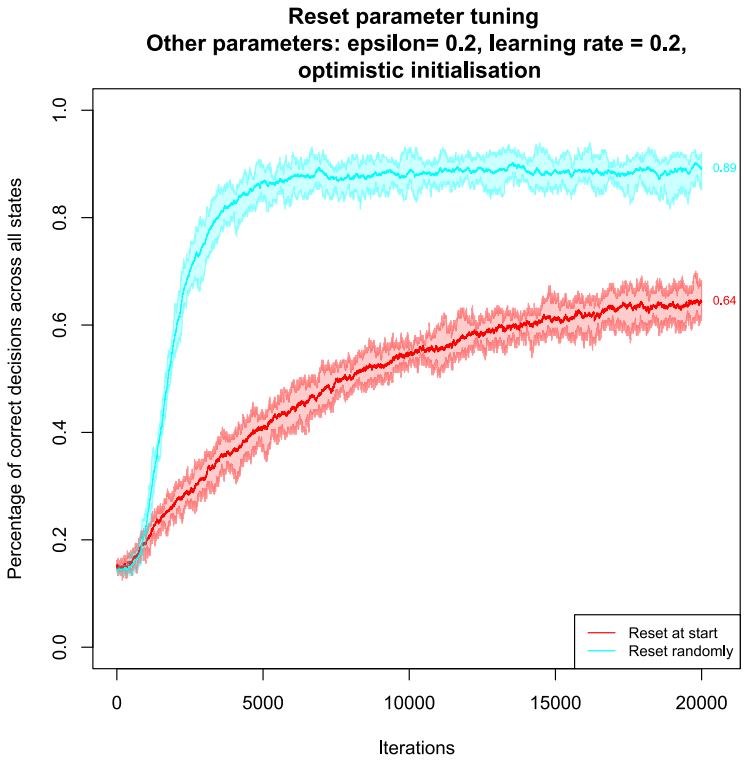
The constraints require the probabilities to sum to one. It also forces the joint distribution to match the specified marginal and pairwise joint probabilities. The objective function is convex in π and the constraints are linear in π. Thus, any locally optimal solution is a global optimum.
The dual of this optimisation problem is easy to solve. The duality arguments give the optimal joint distribution of the form

$$\pi^*(\mathbf{w}, \boldsymbol{\lambda}) = \pi_0(\mathbf{w}) \cdot \exp(-1 + \lambda_0 + \sum_i \lambda_i \Omega_i(\mathbf{w}) + \sum_{i,j} \lambda_{ij} \Omega_{ij}(\mathbf{w})). \tag{B.10}$$

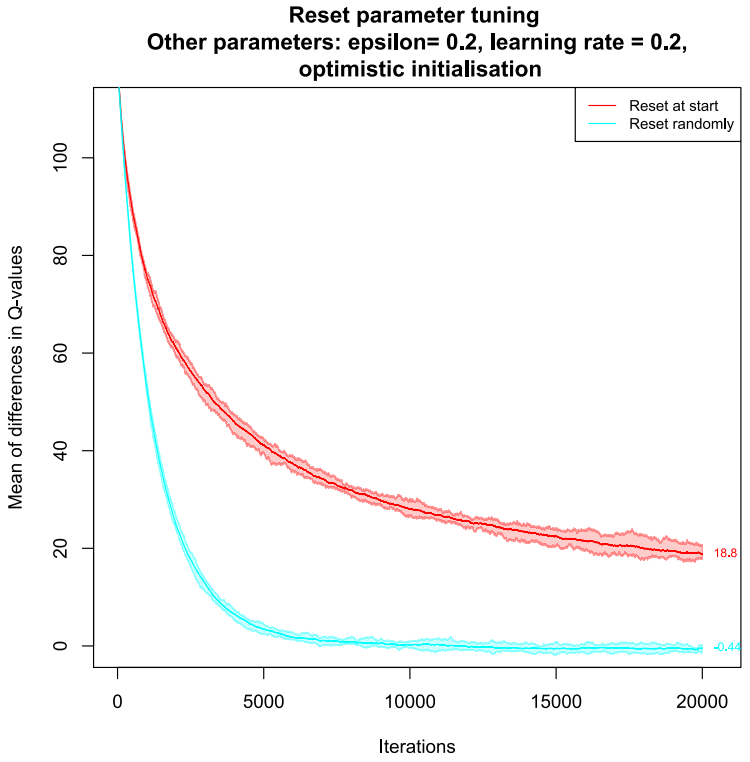
where λ_0, λ_i, and λ_{ij} denote the Lagrange multipliers associated with the constraints in Equations B.7 - B.9. Thus, the solutions to the unconstrained maximisation problem as given in Equation B.11, in combination with Equation B.10 gives the joint distribution for one geological factor.

$$\max_\lambda \left(-\sum_{\mathbf{w}} + \lambda_0 + \sum_i \lambda_i p_i + \sum_{i,j} \lambda_{ij} p_{ij} \right) \tag{B.11}$$

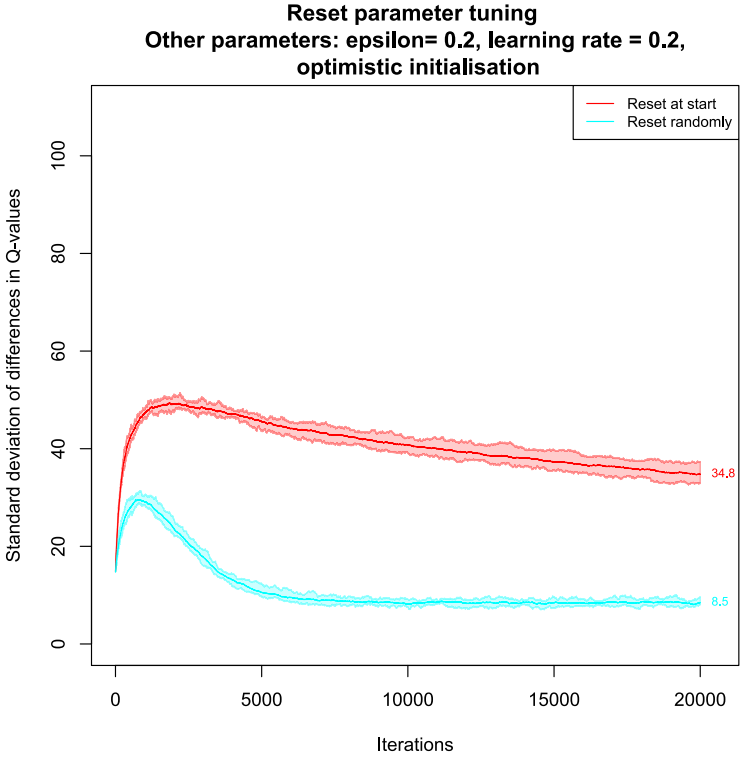
C | RESULTS OF TUNING



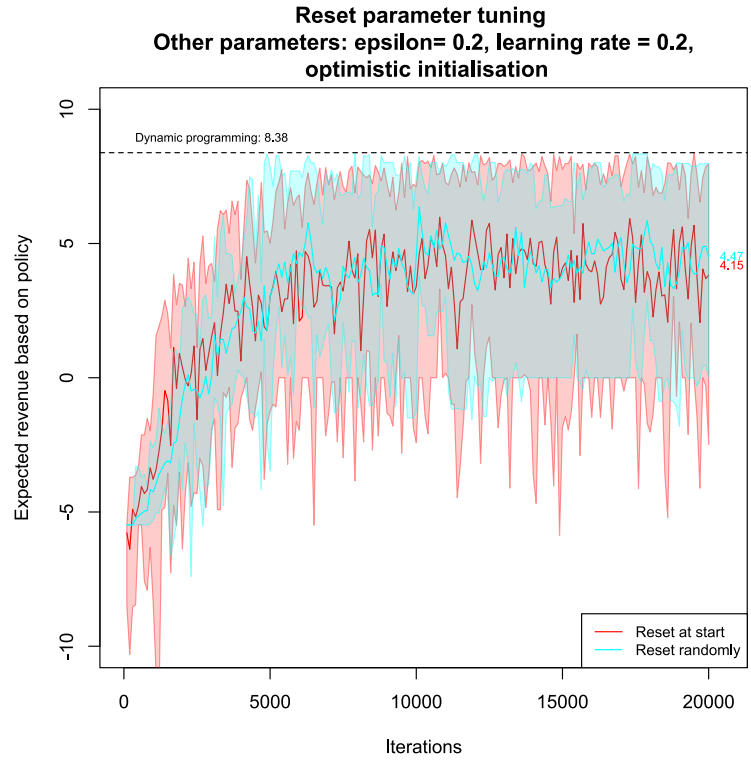
(a) Percentage of correct decisions across all states



(b) Mean of the differences between the true Q-values and the approximated Q-values

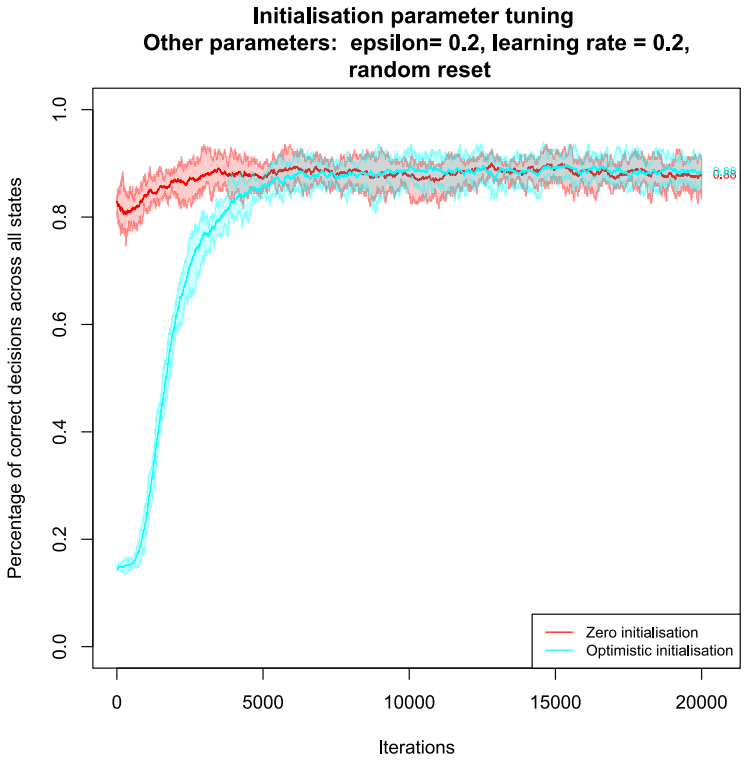


(c) The standard deviation of the differences between the true Q-values and the approximated Q-values

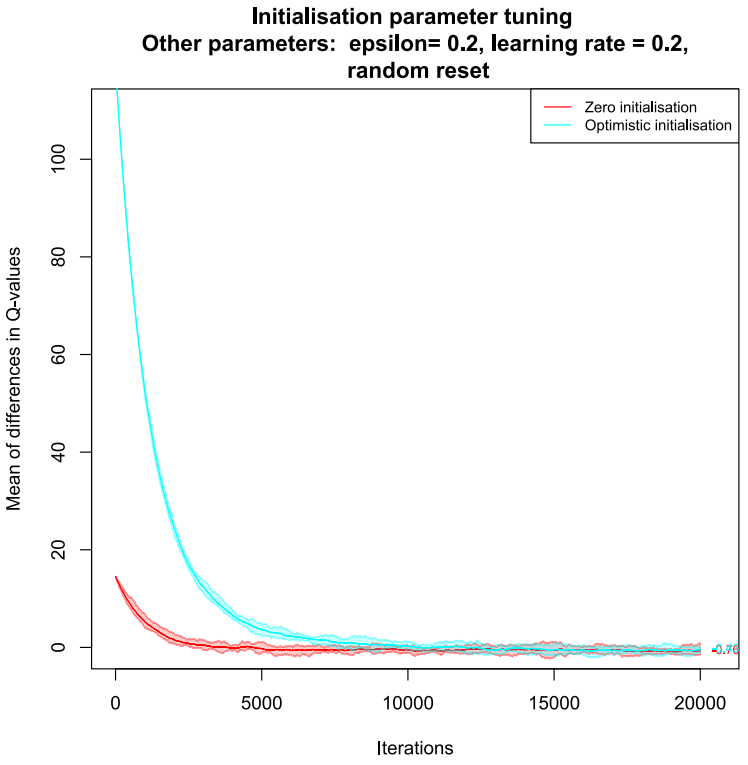


(d) Expected revenue based on the policy derived from Q-learning

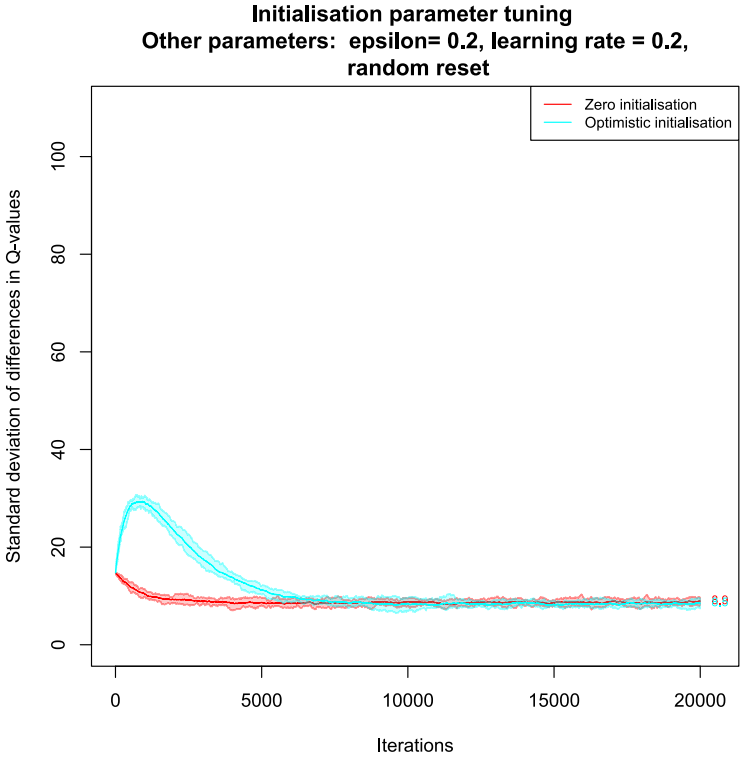
Figure C.1: Reset parameter tuning: performance metrics throughout the learning process



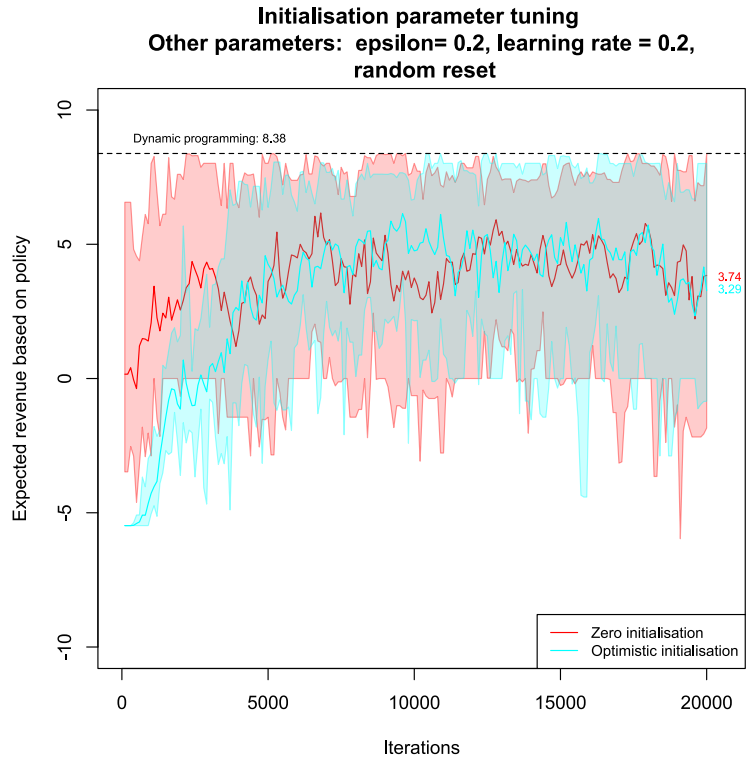
(a) Percentage of correct decisions across all states



(b) Mean of the differences between the true Q-values and the approximated Q-values

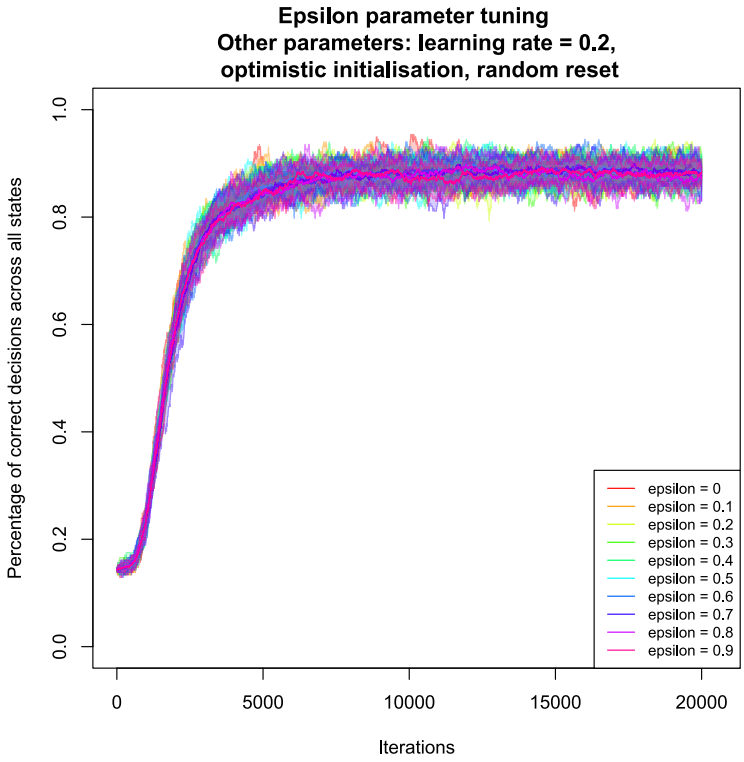


(c) The standard deviation of the differences between the true Q-values and the approximated Q-values

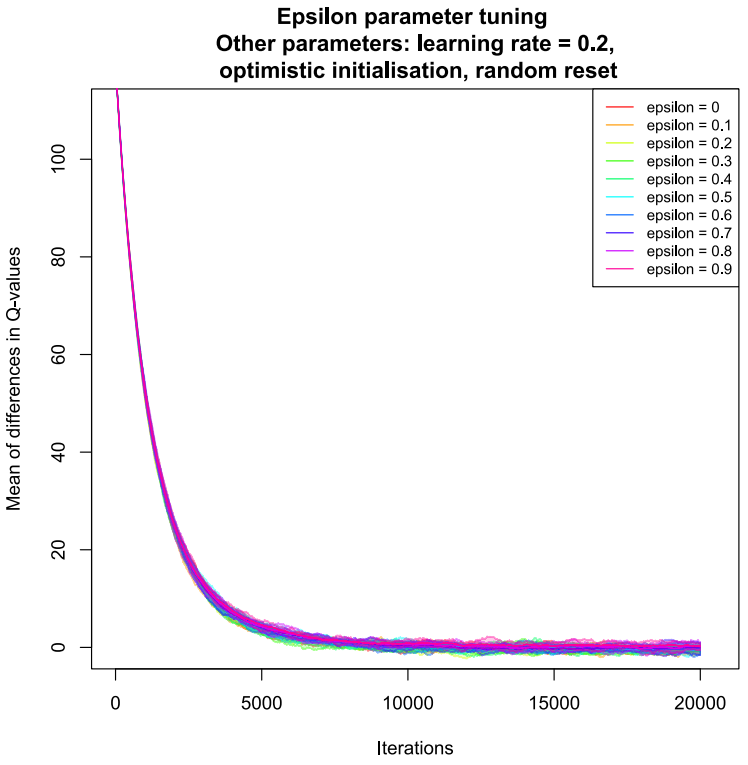


(d) Expected revenue based on the policy derived from Q-learning

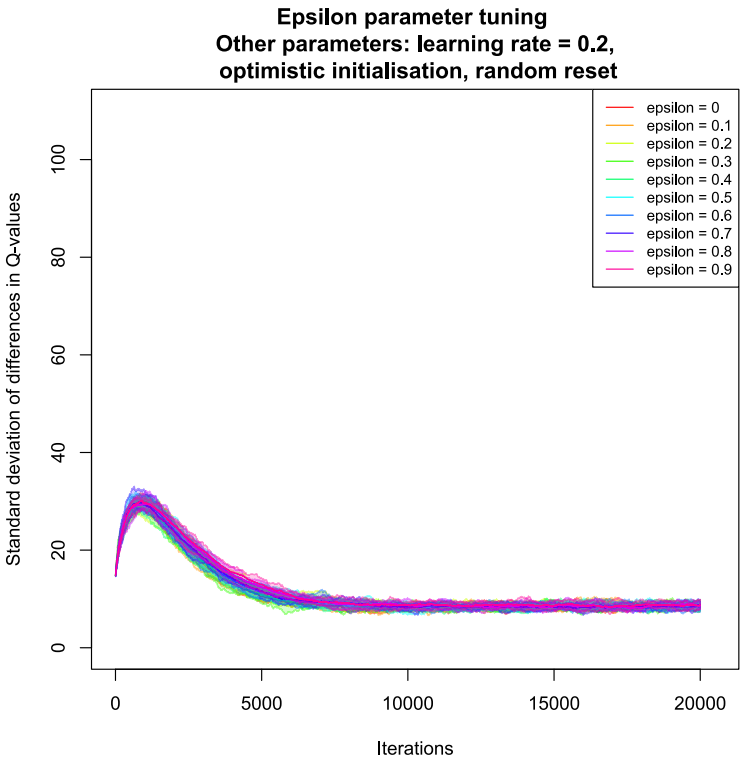
Figure C.2: Initialisation parameter tuning: performance metrics throughout the learning process with parameters: epsilon = 0.1, learning rate = 0.2, random reset.



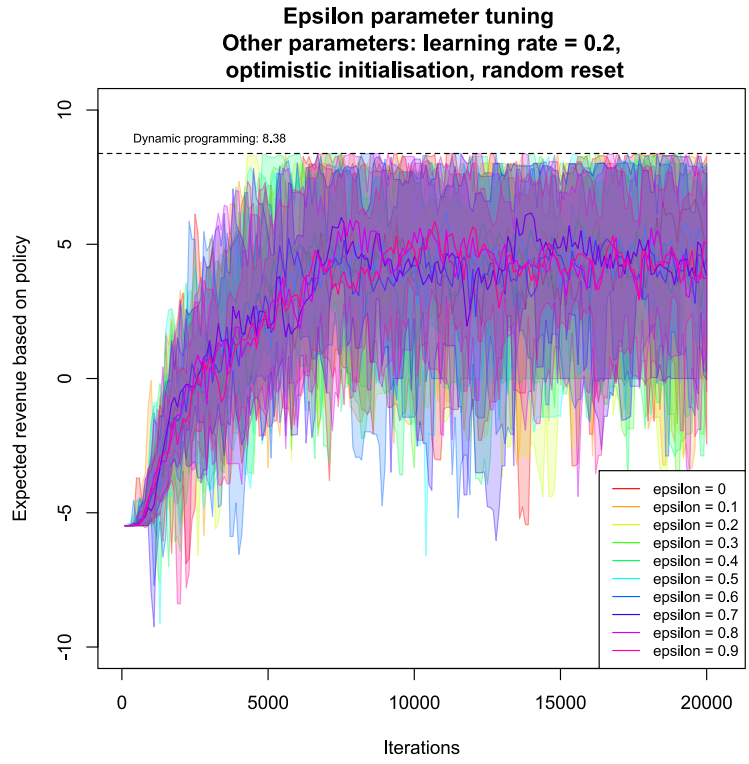
(a) Percentage of correct decisions across all states



(b) Mean of the differences between the true Q-values and the approximated Q-values

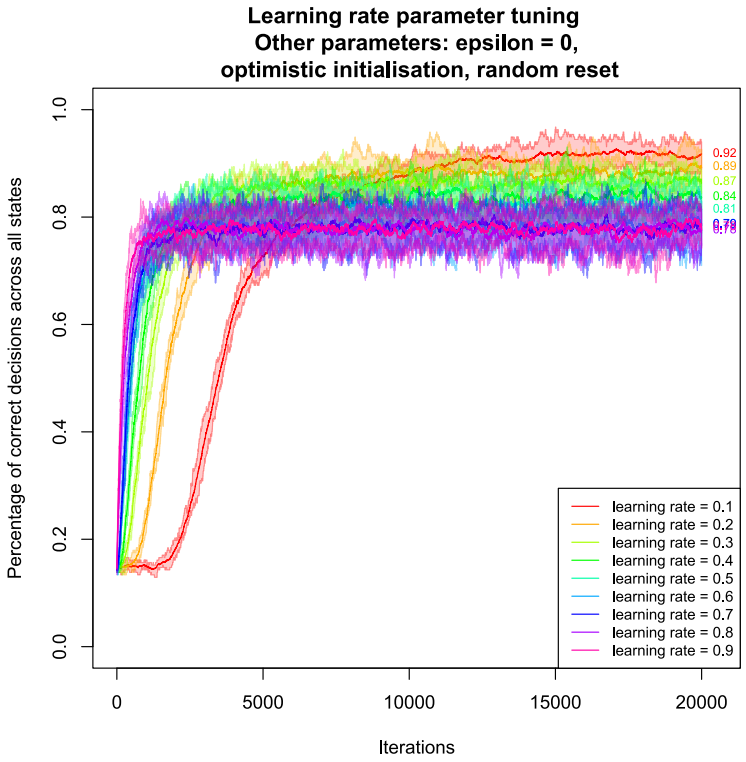


(c) The standard deviation of the differences between the true Q-values and the approximated Q-values

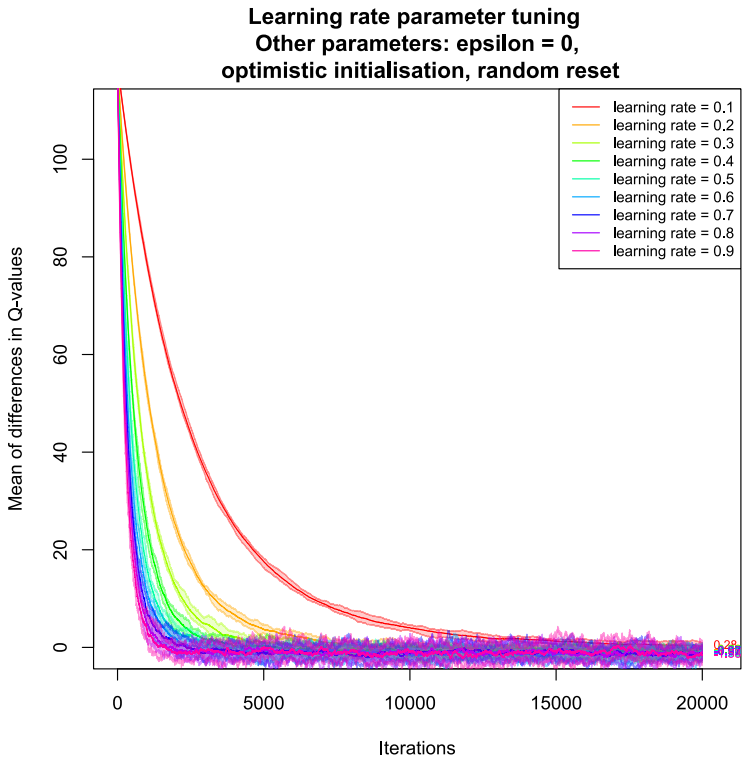


(d) Expected revenue based on the policy derived from Q-learning

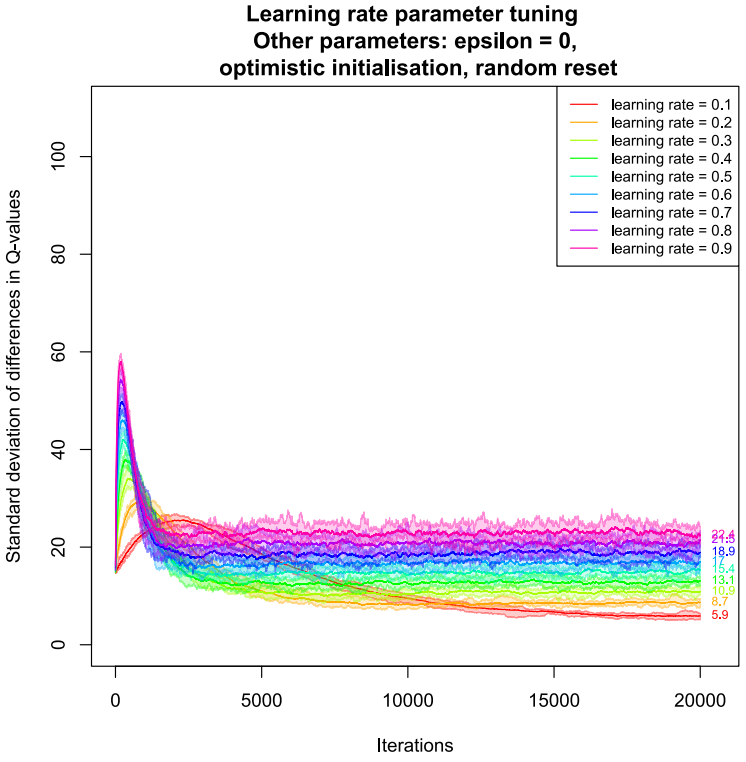
Figure C.3: Epsilon parameter tuning: performance metrics throughout the learning process with parameters: learning rate = 0.2, optimistic initialisation, random reset.



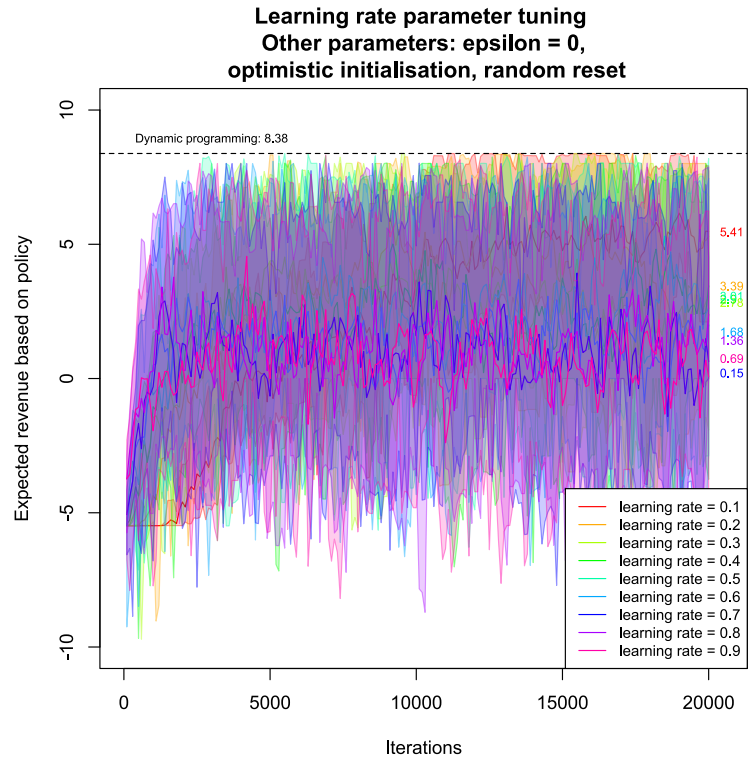
(a) Percentage of correct decisions across all states



(b) Mean of the differences between the true Q-values and the approximated Q-values

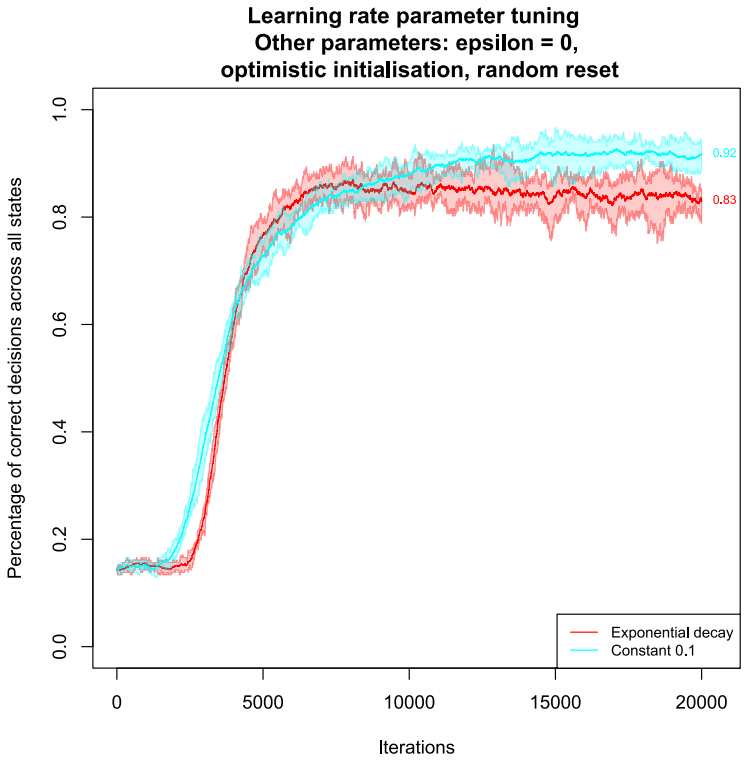


(c) The standard deviation of the differences between the true Q-values and the approximated Q-values

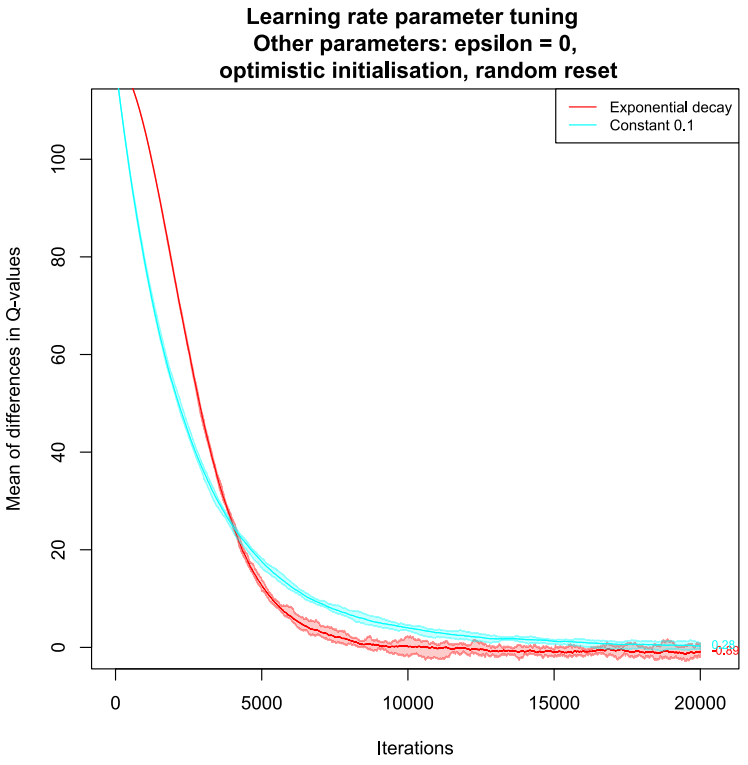


(d) Expected revenue based on the policy derived from Q-learning

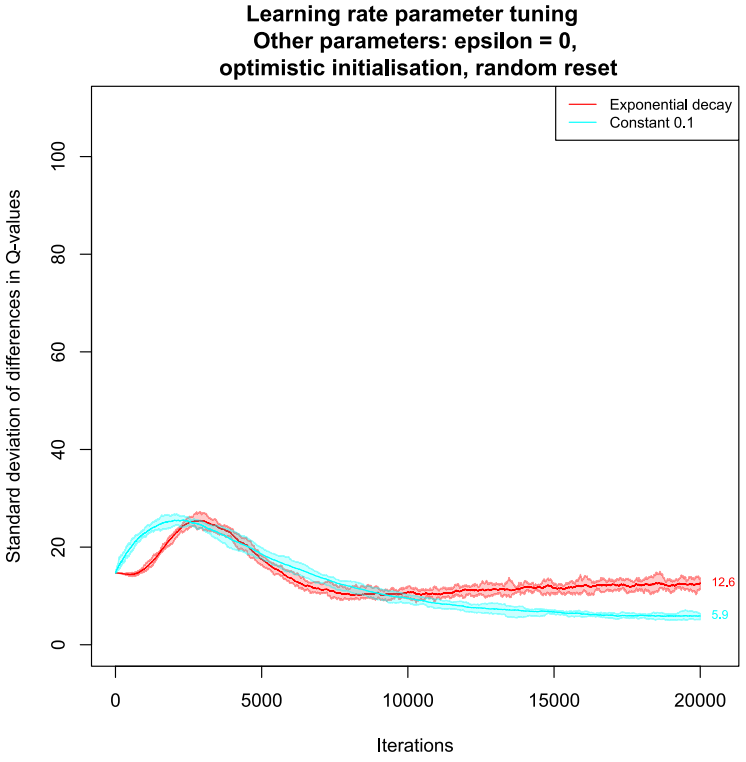
Figure C.4: Learning rate parameter tuning: performance metrics throughout the learning process with parameters: epsilon = 0, optimistic initialisation, random reset.



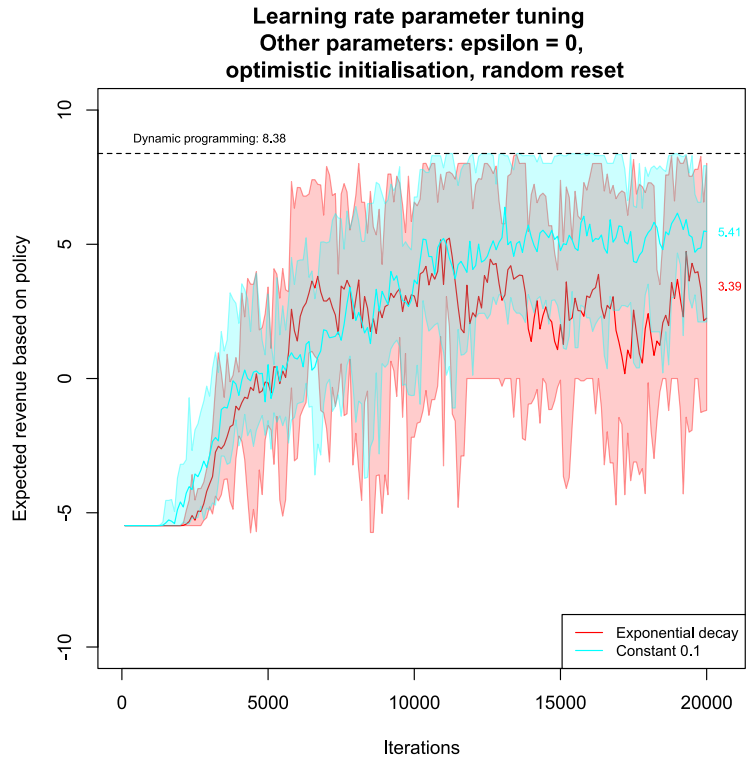
(a) Percentage of correct decisions across all states



(b) Mean of the differences between the true Q-values and the approximated Q-values



(c) The standard deviation of the differences between the true Q-values and the approximated Q-values



(d) Expected revenue based on the policy derived from Q-learning

Figure C.5: Learning rate parameter tuning (exponential decay vs constant): performance metrics throughout the learning process with parameters: epsilon = 0, optimistic initialisation, random reset.

D EXTENDED BAYESIAN NETWORK

Functional node	**Notation**	**Function**
Probability of success	ProbSucc_i	Trap_i · Source_i · Reservoir_i
Volume	Volume_i	$\frac{\text{Area_i} \cdot \text{Thickness_i} \cdot \text{Geometry_factor} \cdot \text{Porosity_i} \cdot \text{NG_i} \cdot \text{SHC_i}}{\text{B_i}}$
Original oil in place	OOIP_i	Volume_i · ProbSucc_i
Reserves	Reserves_i	OOIP_i · RF_i
Revenue	Revenue_i	Reserves_i · OilPricePerUnit - Costs_i

Table D.1: The functional node functions in extended Bayesian network

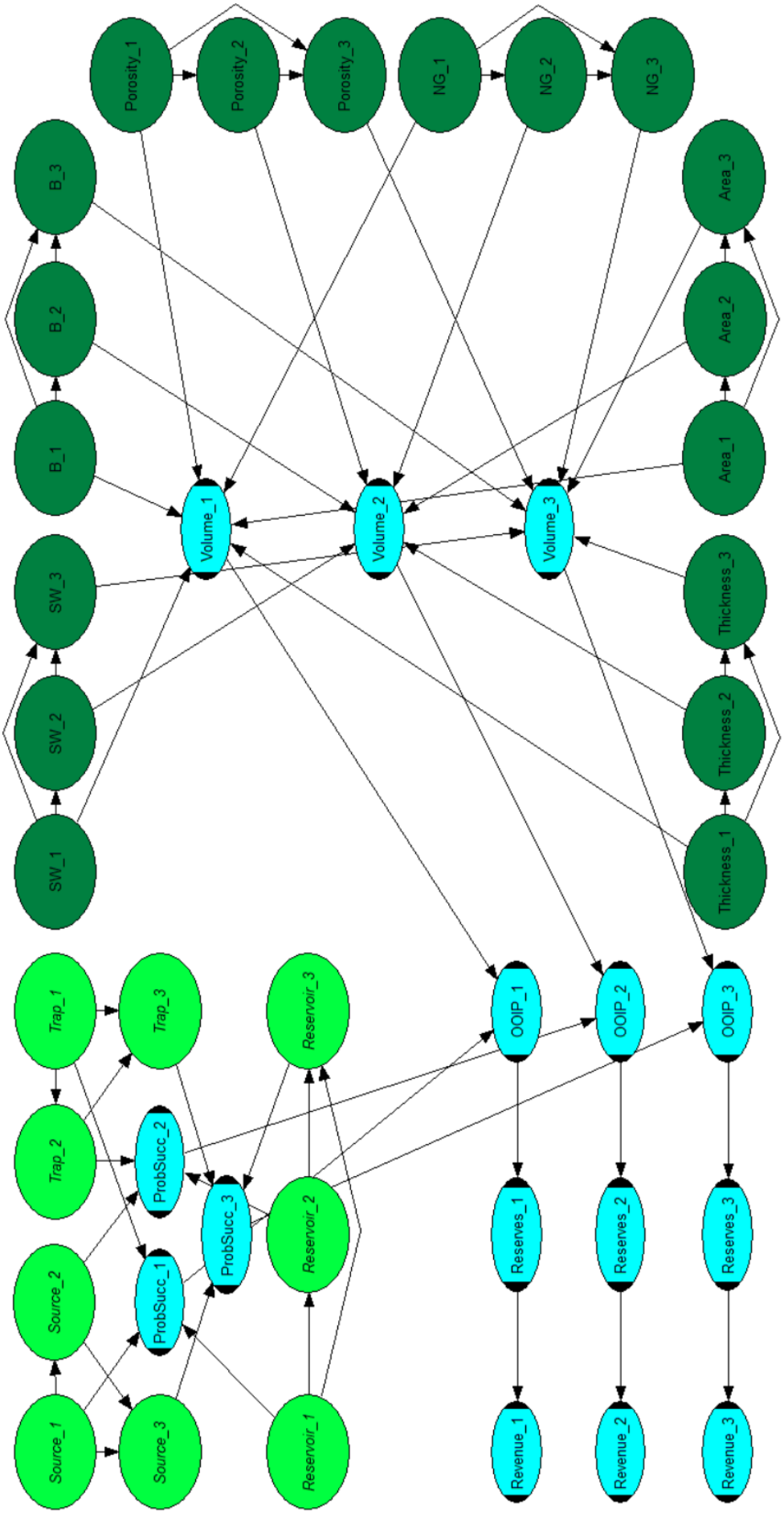


Figure D.1: Bayesian network for the extended model (full page view)

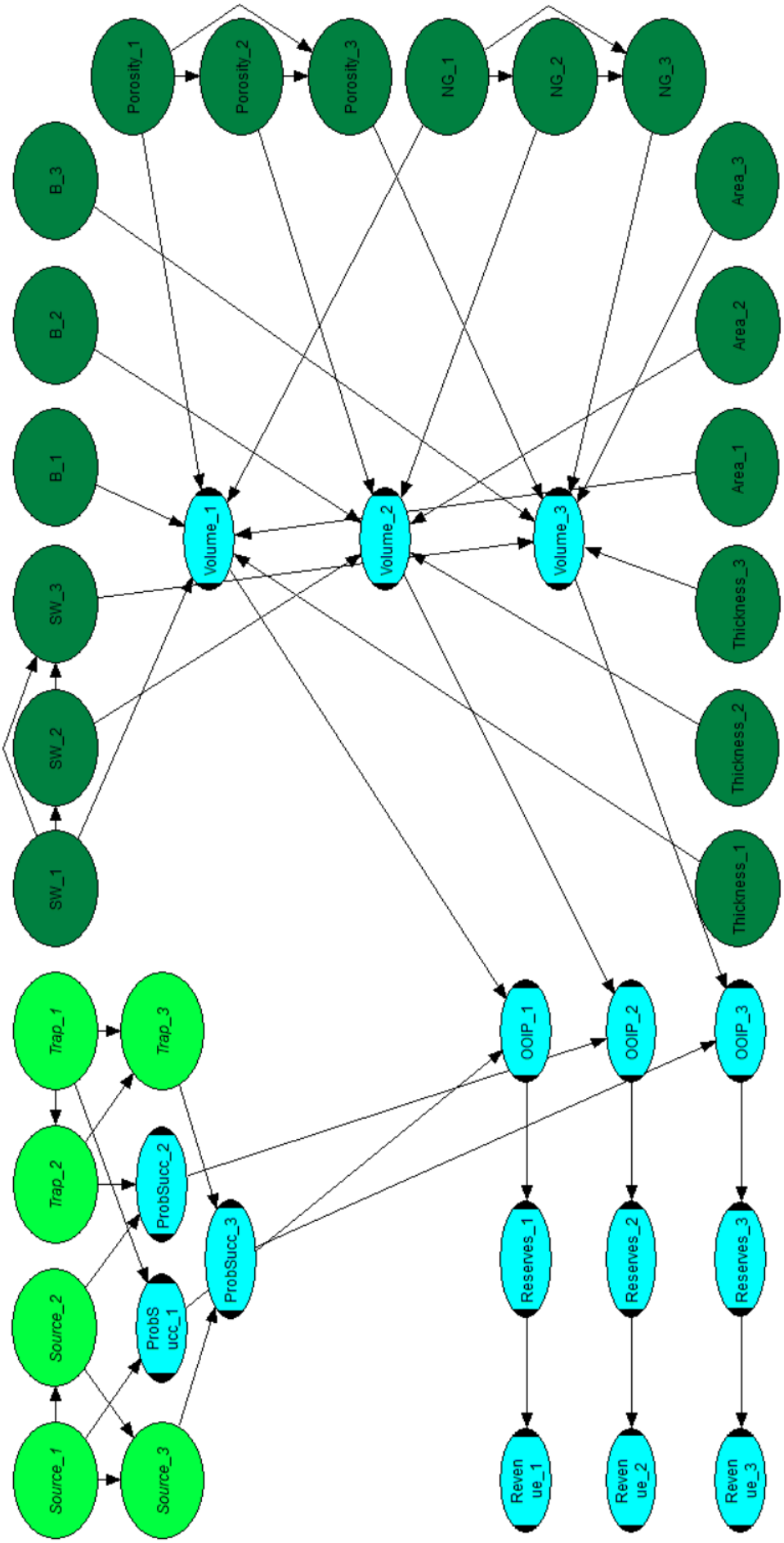


Figure D.2: Bayesian network structure for the extended model for the real data case (full page view).

E | DATA

Prospect	Probability of success of:					
	Trap geometry	Trap seal	Source presence	Source migration	Reservoir presence	Producability
Albert	1	0,791210949	1	1	1	1
Martin	1	0,897530864	1	0,84532872	1	1
Vetle	1	0,897530864	1	0,744	1	1
Harald	0,99	0,992631579	1	0,75	1	1
Stig	0,99	0,966666667	1	0,85125	1	1
Bernt	0,99	0,966666667	1	0,876470588	1	1

Table E.1: Data for the success probabilities of the geological factors.

Prospect	Parameter	Unit	p1	p10	p20	p30	p40	p50
Albert	Area	km2	1,007947	1,770312	2,173161	2,616944	2,895591	3,047911
	Gross reservoir thickness	m	96,22831	101,1084	103,2678	104,4445	105,9937	107,2383
	Hydrocarbon saturation	frac	0,727863	0,753239	0,765631	0,772628	0,778381	0,78422
	Net/Gross ratio	frac	0,752962	0,799304	0,824985	0,836765	0,847123	0,85545
	Porosity	frac	0,15731	0,164772	0,167296	0,169111	0,170214	0,171429
	Hydrocarbon Pore Volume	Mm2	2,148605	4,856443	6,592206	8,221015	10,06505	12,01652
	Inplace volume of oil	MBOE	5,682014	15,9796	25,34271	32,45542	40,23199	49,45366
Martin	Area	km2	2,268786	2,95815	2,965409	3,284247	3,355971	3,499895
	Gross reservoir thickness	m	99,95763	105,5912	107,6727	109,5184	110,7756	112,3905
	Hydrocarbon saturation	frac	0,728931	0,752969	0,76249	0,769731	0,777293	0,783673
	Net/Gross ratio	frac	0,826128	0,850311	0,863107	0,870198	0,875353	0,881884
	Porosity	frac	0,172679	0,176239	0,178326	0,179567	0,18106	0,182064
	Hydrocarbon Pore Volume	Mm2	3,232726	7,237718	9,539738	12,18812	14,08101	16,33998
	Inplace volume of oil	MBOE	11,53798	29,29421	38,32549	49,33762	57,2752	65,273
Vetle	Area	km2	2,005458	2,913065	3,267807	3,968241	4,397641	4,627567
	Gross reservoir thickness	m	98,28983	102,1229	104,1986	105,6628	107,0401	108,3666
	Hydrocarbon saturation	frac	0,726499	0,754093	0,762779	0,772786	0,778717	0,785248
	Net/Gross ratio	frac	0,837844	0,854957	0,863554	0,870047	0,875723	0,881623
	Porosity	frac	0,178351	0,184524	0,187061	0,18875	0,190213	0,191639
	Hydrocarbon Pore Volume	Mm2	2,425008	7,519531	11,50464	14,96609	18,45483	22,05148
	Inplace volume of oil	MBOE	9,336734	27,02698	45,64555	64,38889	77,31565	91,45497
Harald	Area	km2	3,261719	4,537934	4,967864	5,202656	5,284062	5,522745
	Gross reservoir thickness	m	27,64902	34,57339	36,96244	39,12254	41,91317	48,35099
	Hydrocarbon saturation	frac	0,667417	0,710472	0,736039	0,746836	0,75893	0,770718
	Net/Gross ratio	frac	0,568742	0,620968	0,643771	0,664434	0,685678	0,727733
	Porosity	frac	0,16025	0,169134	0,173511	0,175976	0,177887	0,179551
	Hydrocarbon Pore Volume	Mm2	1,513987	4,738232	5,657981	7,020702	8,759666	9,935102
	Inplace volume of oil	MBOE	4,29862	13,56871	17,54949	20,25894	25,90426	30,78019
Stig	Area	km2	6,826105	9,334332	9,97836	10,38543	10,54134	10,83556
	Gross reservoir thickness	m	27,36049	33,88346	36,48091	39,77349	41,84666	53,08098
	Hydrocarbon saturation	frac	0,667449	0,710472	0,731106	0,746137	0,758931	0,770775
	Net/Gross ratio	frac	0,565154	0,635242	0,656632	0,677236	0,695295	0,734388
	Porosity	frac	0,157242	0,169075	0,173038	0,176087	0,177828	0,179838
	Hydrocarbon Pore Volume	Mm2	3,621537	8,461952	12,75706	15,5182	18,51117	20,98102
	Inplace volume of oil	MBOE	10,78855	27,48429	36,68138	49,10581	63,82484	75,35074
Bernt	Area	km2	3,998015	5,407399	5,834958	6,182939	6,500657	6,732759
	Gross reservoir thickness	m	27,71472	33,96065	36,71502	39,01253	42,03935	47,87441
	Hydrocarbon saturation	frac	0,667449	0,710472	0,731106	0,746137	0,75893	0,770761
	Net/Gross ratio	frac	0,573792	0,623434	0,647086	0,664566	0,683813	0,725547
	Porosity	frac	0,160999	0,170539	0,173915	0,176199	0,177935	0,180554
	Hydrocarbon Pore Volume	Mm2	1,306792	4,049815	5,385778	7,990555	11,06616	12,59671
	Inplace volume of oil	MBOE	5,305511	13,43609	18,67363	25,13166	31,07181	36,85154

Table E.2: Data for the volume parameters (Part 1)

Prospect	Parameter	Unit	p60	p70	p80	p90	p99
Albert	Area	km2	3,429426	3,860033	4,136227	4,574367	4,782471
	Gross reservoir thickness	m	108,4507	109,6181	111,2345	113,2701	117,7698
	Hydrocarbon saturation	frac	0,789693	0,795831	0,801058	0,811163	0,828818
	Net/Gross ratio	frac	0,864227	0,872109	0,883325	0,898538	0,923225
	Porosity	frac	0,172661	0,173736	0,175168	0,177201	0,181424
	Hydrocarbon Pore Volume	Mm2	14,48484	16,82475	19,92005	23,71551	35,31333
	Inplace volume of oil	MBOE	58,10945	70,16395	84,43268	104,356	148,7053
Martin	Area	km2	3,731806	3,738286	4,018061	4,222243	4,602283
	Gross reservoir thickness	m	113,3247	115,0735	116,5275	118,5504	123,0985
	Hydrocarbon saturation	frac	0,789603	0,795036	0,800749	0,811577	0,837697
	Net/Gross ratio	frac	0,887843	0,893819	0,901881	0,911702	0,930916
	Porosity	frac	0,183163	0,184203	0,185714	0,187585	0,190957
	Hydrocarbon Pore Volume	Mm2	18,22822	20,66386	22,22545	25,30443	32,15642
	Inplace volume of oil	MBOE	75,56622	85,04058	97,01089	113,7053	153,9129
Vetle	Area	km2	4,726922	4,911069	5,112016	5,338281	5,842897
	Gross reservoir thickness	m	109,3639	110,7269	112,235	114,8878	120,7063
	Hydrocarbon saturation	frac	0,792231	0,797547	0,803611	0,813436	0,837862
	Net/Gross ratio	frac	0,886609	0,892826	0,900411	0,911958	0,931665
	Porosity	frac	0,192859	0,194284	0,195817	0,197825	0,203047
	Hydrocarbon Pore Volume	Mm2	26,62464	30,70769	35,24517	41,28947	54,33226
	Inplace volume of oil	MBOE	108,472	127,5217	146,6609	175,6808	240,2168
Harald	Area	km2	5,691633	5,801128	6,141832	6,264807	6,803183
	Gross reservoir thickness	m	59,05461	61,66179	64,07082	67,70745	72,83282
	Hydrocarbon saturation	frac	0,782357	0,791653	0,80144	0,826316	0,860752
	Net/Gross ratio	frac	0,784722	0,813165	0,835579	0,858102	0,931629
	Porosity	frac	0,181828	0,183635	0,185874	0,18829	0,191533
	Hydrocarbon Pore Volume	Mm2	13,28917	14,87233	18,27737	21,978	30,48333
	Inplace volume of oil	MBOE	39,7606	51,57449	60,09997	74,57973	111,1611
Stig	Area	km2	11,17141	11,343	11,95983	12,52056	13,3474
	Gross reservoir thickness	m	58,40196	61,48091	64,45613	67,17204	73,98321
	Hydrocarbon saturation	frac	0,782415	0,793323	0,801571	0,823943	0,860798
	Net/Gross ratio	frac	0,782738	0,812127	0,83437	0,860078	0,923588
	Porosity	frac	0,181806	0,183436	0,185729	0,188852	0,193853
	Hydrocarbon Pore Volume	Mm2	24,88268	28,99177	38,0899	45,87516	65,29286
	Inplace volume of oil	MBOE	92,09536	122,26	141,5442	171,9553	243,8097
Bernt	Area	km2	7,051039	7,101171	7,533395	7,880646	8,505558
	Gross reservoir thickness	m	58,17154	61,52647	64,02223	67,03935	73,39459
	Hydrocarbon saturation	frac	0,782401	0,793153	0,807409	0,826362	0,860798
	Net/Gross ratio	frac	0,77809	0,808301	0,831515	0,856024	0,919833
	Porosity	frac	0,181961	0,182924	0,185422	0,188092	0,194518
	Hydrocarbon Pore Volume	Mm2	14,92154	17,02954	22,38101	27,68515	38,07511
	Inplace volume of oil	MBOE	46,91637	55,2632	65,57769	97,99164	154,1256

Table E.3: Data for the volume parameters (Part 2)

Prospect	Mean of in place volume (MBOE)	Mean of recoverable volume (MBOE)
Albert	54,98	24,47
Martin	67,89	26,45
Vetle	102,61	39,12
Harald	39,80	17,98
Stig	85,77	39,93
Bernt	47,32	21,43

Table E.4: Data for the recovery factor parameter

Variable	**Dependence estimation in words**
Trap	The wells in this region have the same trap style, the same top seal and the same fault seal trapping mechanism
Source	The wells in this region have the same source rock. The migration concept is similar. Local variations are higher due to differences in faults and differences in migration distances.
Reservoir	The wells in this region have the same depositional environment and general reservoir units.
Area	NA
Thickness	NA
Porosity	The wells in this region have the same depositional environment and general reservoir units.
Hydrocarbon saturation	The wells in this region have the same depositional environment and general reservoir units.
N/G ratio	The wells in this region have the same depositional environment and general reservoir units.
B (formation volume factor)	The wells in this region have the same source rock. The migration concept is similar. Local variations are higher due to differences in faults and differences in migration distances.

Table E.5: Data for dependencies between wells (descriptions)

Variable	Dependency in words	Dependency in correlations
Trap	Medium to high	0.7
Source	Medium to low	0.5
Reservoir	Medium for wells far away or high for wells close to each other	0.6 (far away) or 0.8 (close)
Area	NA	NA
Thickness	NA	NA
Porosity	Medium for wells far away or high for wells close to each other	0.6 (far away) or 0.8 (close)
Hydrocarbon saturation	Medium for wells far away or high for wells close to each other	0.6 (far away) or 0.8 (close)
N/G ratio	Medium for wells far away or high for wells close to each other	0.6 (far away) or 0.8 (close)
B (formation volume factor)	Medium to low	0.5

Table E.6: Data for dependencies between wells (correlations)

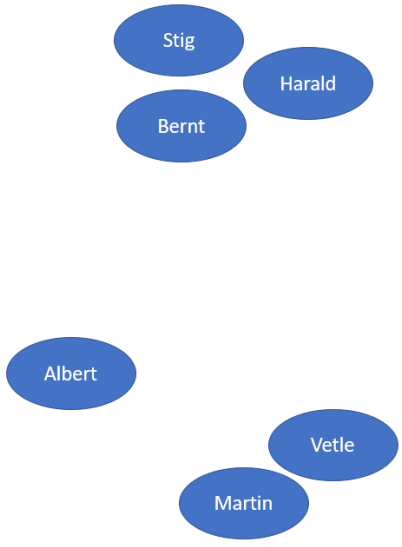


Figure E.1: Geographical relationship between the six wells in the data set

Prospect	Cost of drilling the prospect (million US Dollars)
Albert	9
Martin	8
Vetle	4
Harald	5
Stig	11
Bernt	6

Table E.7: Costs of drilling the six wells in the data set

Prospect	Outcome of the well
Albert	Success
Martin	Success
Vetle	Failure
Harald	Success
Stig	Success
Bernt	Success

Table E.8: Outcome of drilling the six wells in the data set

F SYNTHETIC DATA CASE

Random variable	Marginal distribution
Trap_1	Bernoulli(p=0.956)
Trap_2	Bernoulli(p=0.983)
Trap_3	Bernoulli(p=0.958)
Source_1	Bernoulli(p=0.851)
Source_2	Bernoulli(p=0.75)
Source_3	Bernoulli(p=0.877)
SHC_1	Normal(mean = 0.77, sd = 0.0449)
SHC_2	Normal(mean = 0.77, sd = 0.0457)
SHC_3	Normal(mean = 0.77, sd = 0.043)
B_1	Normal(mean = 0.288, sd = 0.0297)
B_2	Normal(mean = 0.32, sd = 0.0.0397)
B_3	Normal(mean = 0.31, sd = 0.0276)
Porosity_1	Normal(mean = 0.18, sd = 0.0077)
Porosity_2	Normal(mean = 0.18, sd = 0.0081)
Porosity_3	Normal(mean = 0.179, sd = 0.0077)
NG_1	Normal(mean = 0.738, sd = 0.0984)
NG_2	Normal(mean = 0.742, sd = 0.0973)
NG_3	Normal(mean = 0.745, sd = 0.0938)
Area_1	Normal(mean = 10.7, sd = 1.31)
Area_2	Normal(mean = 5.44, sd = 0.769)
Area_3	Normal(mean = 6.69, sd = 0.975)
Thickness_1	Normal(mean = 49.9, sd = 13.6)
Thickness_2	Normal(mean = 50.5, sd = 13.6)
Thickness_3	Normal(mean = 49.9, sd = 13.6)

Table F.1: Marginal distributions for synthetic data case