ERASMUS UNIVERSITEIT ROTTERDAM

Erasmus School of Economics

Master Thesis Econometrics - Quantitative Marketing and Business Analytics

Personalizing Stool-based Screening of Colorectal Cancer Using Multi-Objective Reinforcement Learning

Michael Simon van der Zwan

434569

Supervisor: Prof.Dr.ir. R. Dekker

Second assessor: Dr. P.C. Bouman

Supervisor Erasmus MC: L.A. van Duuren, MSc

December 14, 2021

Abstract

Colorectal cancer (CRC) is one of the most diagnosed cancer types and ranks second in terms of mortality. Studies have shown that a person's hemoglobin concentration indicates the risk of CRC, which can be measured by a Faecal Immunochemical Test (FIT). Therefore, different screening programs using this test are used worldwide to reduce mortality. The Dutch screening program is currently population-based, while it can be improved by personalizing the strategies.

This thesis evaluates two different multi-objective reinforcement learning algorithms to optimize personalized screening strategies based on someone's measured hemoglobin (Hb) concentrations. The algorithms are implemented in the MISCAN-Colon micro-simulation model to learn the optimal strategies and evaluate the related costs and Quality Adjusted Life Years gained.

In the first algorithm, we found two issues that the authors did not take into account. Even after solving these issues, the algorithm could not solve our problem. The second algorithm did find strategies that dominate the current Dutch screening strategy, and these strategies are not dominated by the strategies advised in a recent report for the United States Preventive Services Task Force. However, the algorithm does not converge to the optimal strategies but mostly finds non-optimal strategies.

Contents

1 Introduction

In 2020, over 1.9 million new colorectal cancer (CRC) cases and 936,000 deaths were estimated worldwide, meaning that CRC ranks third in most commonly diagnosed cancer types and is even the second cancer type in terms of mortality [Sung et al., 2021]. Usually, colorectal cancers are preceded by irregularities in the colon, called adenomas. Studies have shown that CRC can be prevented by screening the population to detect and remove these related adenomas [Levin et al., 2008], [Pignone et al., 2002], [Hewitson et al., 2008]. Adenomas can be removed relatively easily, preventing them from developing into harmful cancer [Lansdorp-Vogelaar et al., 2011], [Schreuders et al., 2015]. Hence, screening for having adenomas is an effective strategy.

Currently, endoscopic tests such as colonoscopy are the golden standard of colorectal cancer tests. During this test, the colon and rectum are visualized by a surgeon and found adenomas are removed directly. However, this test is also costly and invasive. Therefore, another type of test is often first used to determine the risk of having adenomas. The most used test is the Fecal Immunochemical Test (FIT), which is a stool-based test and has proven itself to be one of the least invasive and least expensive CRC screening tests [Robertson et al., 2017] [Day et al., 2013]. It measures the concentration of hemoglobin (Hb) in a patient's stool sample. An increased Hb concentration is associated with an increased risk of having adenomas and developing CRC in the future [Grobbee et al., 2017].

Many countries use the FIT qualitatively [Schreuders et al., 2015]. Currently, the applied screening strategy in The Netherlands is a biennial FIT. If a person's concentration of Hb is above a certain threshold (cutoff), he is referred to a hospital for a colonoscopy. Otherwise, the person is scheduled to get a new FIT after two years. The 2-year screening interval is fixed for all screening participants, while studies have shown that it may be cost-efficient to personalize this interval to a patient's FIT concentrations measured in the past [Grobbee et al., 2017]. Therefore, screening can be optimized by personalizing the strategies, since Grobbee et al. [2017] did not determine what is the optimal strategy. In this research, we will answer the following question:

To what extent can the Dutch colorectal cancer screening program be improved by optimizing personalized screening strategies?

A suitable screening interval is determined based on a participant's history of FIT measurements (captured in one "risk estimator") and age. This research aims to find optimal personalized screening strategies based on the benefits and harms of the screening strategy.

The benefits and harms of screening strategies are measured in Quality Adjusted Life Years gained (QALYs). One QALY represents the value of living one year without complications due to CRC. Besides the benefits, screening also comes with harms, the burdens of a patient to undergo a test (like the uncertainty of the outcomes or the need to go to the hospital), which can be represented by a reduction in QALYs. Furthermore, screening also has financial costs (costs of the tests and treatment itself). We aim to maximize the effects and minimize the costs. As these quantities are conflicting, a trade-off has to be found. We have to optimize two objectives, resulting in the Pareto optimal frontier, i.e., all incomparable optimal solutions that maximize the QALYs for given costs. From this frontier, the National Institute for Public Health and the Environment, in Dutch called the 'RIVM', can eventually choose its preferred strategy itself.

An optimal strategy specifies an optimal screening action for all possible states of an individual. Each screening action will result in a particular reward (benefits and costs), depending on the individual's state and might be estimated. Based on these (estimated) rewards, the best action can be chosen. This type of optimization can typically be translated into a (multi-objective) reinforcement learning optimization. Due to the promising results of reinforcement learning in previous studies, we will use this machine learning technique and answer the following question:

What is the performance of multi-objective reinforcement learning techniques when optimizing personalized screening strategies?

In this thesis, two multi-policy multi-objective reinforcement learning methods are examined. The first method, named MPQ-learning (Multi-Pareto Q-Learning) [Ruiz-Montiel et al., 2017], adapted the original Q-learning algorithm [Watkins and Dayan, 1992] to deal with multi-objectives, and it was shown for some cases that it is able to approximate the complete Pareto front. The second method, envelope Q-learning [Yang et al., 2019] includes deep learning by using a neural network to represent value functions over the entire space of preferences. Both methods are implemented in the MISCAN-Colon micro-simulation model [Habbema et al., 1985] to learn and evaluate optimal screening strategies.

The following section starts with relevant background knowledge about CRC development and the screening tests used in this thesis. Furthermore, the MISCAN model [Habbema et al., 1985] is introduced, which is used for the simulation study. Section 3 describes the current literature on reinforcement learning. The models that are used and the implementation of the simulation study are explained in Section 4 and 5. Next, in Section 6 we show the results of our simulation study, whereafter a discussion of the results if followed, including the limitations. Finally, in Section 8 we end with the conclusion of this thesis.

2 Background Knowledge

2.1 Colorectal cancer and colorectal cancer screening

Colorectal cancers are usually preceded by irregularities in the colon called adenomas. An adenoma is a type of polyp, a small cluster of cells that forms on the colon lining. Having adenomas is not unusual; approximately 40% of the Western population will develop at least one adenoma [Leslie et al., 2002]. However, not all adenomas are malignant. Around 3% will eventually develop cancer. Furthermore, the CRC incidence increases with age, meaning that a person at a higher age will have a higher risk of CRC [Brenner et al., 2007].

Adenomas grow into cancer through three stages based on their size. Small adenomas are smaller or equal to 5 mm. When their size becomes larger than 6 mm, they are called non-advanced adenomas, and when they exceed a size of 10 mm, they are called advanced adenomas [Brenner and Werner, 2017]. If the adenomas are not treated, they might develop into cancer, usually taking about ten years [Holme et al., 2013].

When an adenoma eventually is formed into cancer but still not detected, the cancer is called *preclinical.* Once the cancer is detected, it is called *clinical.* The cancer is divided into four stages based on the development of the cancer. The higher the stage, the more the CRC is developed. A person then might get symptoms, whereafter he will be treated. Screening can be defined as a strategy to detect an unrecognized disease among individuals without symptoms [Gini, 2020].

Since not all adenomas develop into cancer, we can distinguish between two types of adenomas. The ones that develop into cancer are called progressive. Otherwise, they are called non-progressive. Unfortunately, the differences between these two cannot be seen, meaning that screening techniques cannot take this difference into account.

Generally, there are two types of screening techniques used in CRC screening programs.

The most effective but most invasive technique is an endoscopic technique, called colonoscopy [Kooyker et al., 2020]. The colon and rectum are visualized and observed by a surgeon, and any adenoma (lesion) is removed immediately. This is very effective but comes with high costs. Due to the relatively high costs and burdens, another technique is often used in advance. This is a stool-based test, called a Fecal Immunochemical Test (FIT), which is used to indicate the risk of having any adenoma. According to Robertson et al. [2017] and Day et al. [2013], this is the least invasive and least expensive. The test measures the Hemoglobin (Hb) concentration in a person's stool sample since it is shown that this concentration increases when having an (advanced) adenoma [Digby et al., 2013], [Van Doorn et al., 2015]. If the measured blood concentration is above a certain threshold, the person is redirected to the hospital to undergo an endoscopy test. Performing this FIT in advance can reduce the number of unnecessary colonoscopies.

Currently, the Dutch program implies a biennially FIT from the age of 55 until 75. The FIT is used qualitatively, meaning that it is evaluated as positive when the measured Hb concentration exceeds a certain threshold (currently 47 μg per gram feces) and negative otherwise. If the test is positive, the person is redirected to the hospital to undergo a colonoscopy. Since the FIT is not 100% accurate, the colonoscopy might also be negative, meaning no adenomas were found. In this case, the person gets a pause for ten years of screening, whereafter again, the biennially FIT will be applied. In case of a positive colonoscopy, the person will be removed from the screening program and will follow a surveillance program since it is known that people having adenomas removed have a 30-35% chance of growing new adenomas within four years [Leslie et al., 2002].

2.2 MISCAN-Colon microsimulation model

To evaluate the performance of a specific screening strategy, one might apply it to a population and monitor the results. This can be compared to a population without screening to see the effects of the program. However, such randomized controlled trials are time-consuming, and when many different policies have to be evaluated, this is not feasible in practice. To tackle this problem, Habbema et al. [1985] developed a microsimulation model that simulates a population of which a fraction develops a specific disease. By simulating this population under different screening strategies, the effects of these different strategies can be compared.

The structure of the input of the MISCAN model consists of three layers. The first layer concerns population-based parameters such as birth and life tables, representing mortality due to causes other than the simulated disease. The second layer defines a specific disease with parameters like the risk of developing the disease and symptoms, speed of developing the disease, and possibly death as a consequence. The last layer concerns the properties of the screening program, including the properties of different tests.

MISCAN-Colon is the version of the model adapted to specifically evaluate colorectal cancer screening [Loeve et al., 1999]. A simulated individual may develop one or multiple adenomas. These adenomas grow independently, and some develop into cancer, depending on the individual's age and a risk factor, and the adenoma location. The risk factor is a random number drawn at the birth of the individual. Each adenoma is modeled as an independent Semi-Markov process, meaning that a person is a collection of multiple stochastic processes. Their state space is shown in Figure 2.1. The disease state of an individual is characterized by the state of the individual's most advanced adenoma or cancer.

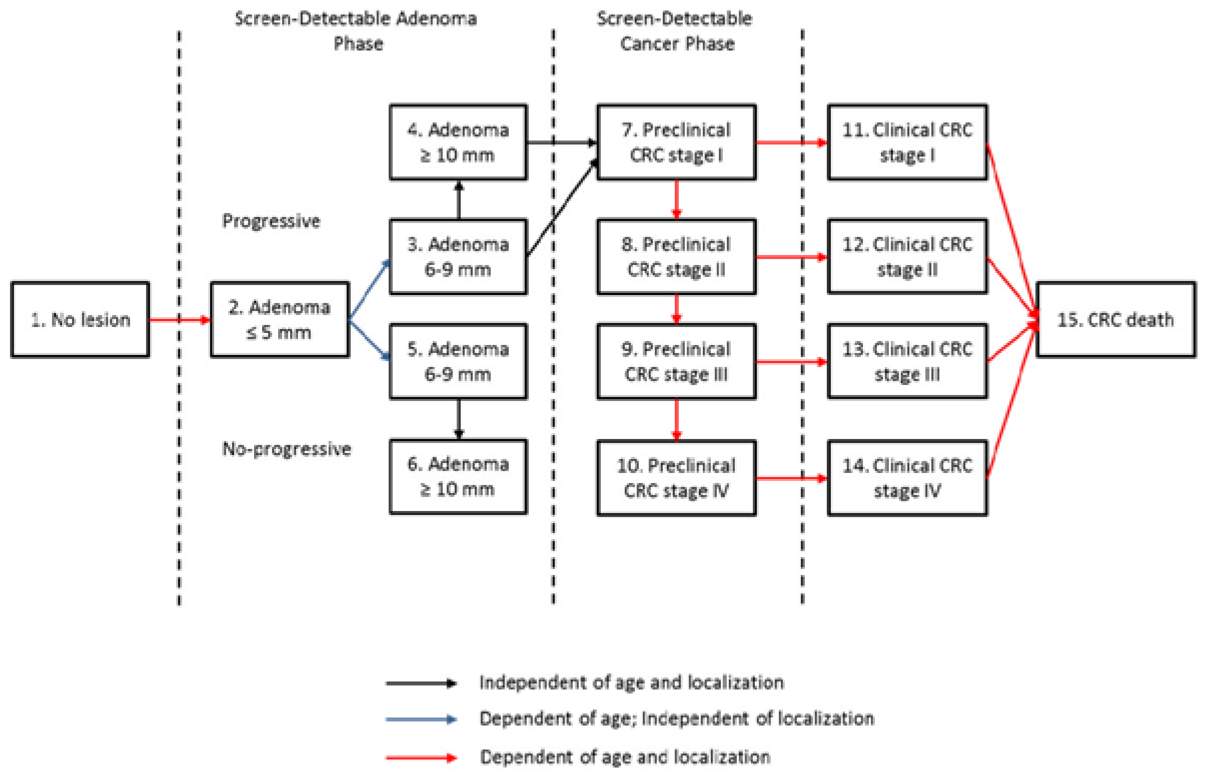

Screen-Detectable Adenoma Phase
Screen-Detectable Cancer Phase
Progressive
No-progressive
1. No lesion
2. Adenoma ≤ 5 mm
3. Adenoma 6-9 mm
4. Adenoma ≥ 10 mm
5. Adenoma 6-9 mm
6. Adenoma ≥ 10 mm
7. Preclinical CRC stage I
8. Preclinical CRC stage II
9. Preclinical CRC stage III
10. Preclinical CRC stage IV
11. Clinical CRC stage I
12. Clinical CRC stage II
13. Clinical CRC stage III
14. Clinical CRC stage IV
15. CRC death
Independent of age and localization
Dependent of age; Independent of localization
Dependent of age and localization

Figure 2.1: Model structure used in MISCAN-Colon with adenoma-carcinoma sequence for progressive adenomas and non-progressive adenoma sequence
Adapted source: Gini [2020]

For this study, MISCAN-Colon was extended with a preliminary module to simulate individual FIT concentrations based on the outcomes of two rounds of the Dutch national screening program [Toes-Zoutendijk et al., 2017], which is described in Van Duuren [2021]. The details of this distribution can be found in Appendix A. In short, the concentrations are drawn from a zero-inflated negative binomial distribution. Suppose the individual is in a healthy stage. In that case, the probability of measuring a zero concentration is 0.840, which decreases to 0.644 or 0.032 when in an adenoma or preclinical stage, respectively. Otherwise, a value is drawn from a negative binomial distribution. The mean of this distribution increases with age and cancer stage and depends on gender and an individual's risk factor, representing whether someone has consistently high or low blood concentrations in their stool. Hence, this factor is responsible for the correlation between the blood concentrations of an individual himself.

3 Literature Review

In this section, a short review of the relevant literature is given, starting with previous studies of personalized cancer screening. Next, the use of reinforcement learning in health care is reviewed, and finally, studies based on multi-objective reinforcement learning are discussed.

3.1 Personalizing cancer screening

Dunnewind [2020] already evaluated multiple different multi-objective genetic algorithms (NSGA-II, SPEA2, PESA-II, and IBEA) to find the cost-effective frontier for colorectal cancer screening strategies, based on the financial costs and life-years gained. However, the evaluated strategies were still population-based. Van Duuren [2021] also used a genetic algorithm to find individual strategies based on age and, at most, the last two FIT results. Three case studies are demonstrated, where the most prominent problem allows for an interval between two FITs of one, two, or three years and eleven different possible fixed cutoffs for the FIT. However, including two or three FITs instead of one to determine the risk of CRC did not affect the performance, which might contradict the literature [Grobbee et al., 2017]. The authors hypothesize that this is caused by their preliminary model used to simulate FIT-values rather than the algorithm. Furthermore, only the results of FITs were incorporated for the decision on a screening interval, whereas a negative colonoscopy result may be very influential on choice for an optimal interval.

Tomer et al. [2019] use joint models for time-to-event and longitudinal data to get personalized schedules for surveillance of low-risk prostate cancer. They focus on risk prediction, while this will not be the focus of our research. Besides, the loss function has one dimension: finding cancer as soon as possible, once it exists, while the costs of screening are not considered.

3.2 Reinforcement learning in health care

In the last couple of years, reinforcement learning also has found its way into health care. While this machine learning technique is often used for computer games or robotics, Zhao et al. [2009] developed a reinforcement learning method (Q-learning) for discovering effective therapeutic regimens in clinical trial design. Based on simulated data, they have found that reinforcement learning can identify individualized optimal regimens in clinical trials that consist of multiple courses of treatment. Furthermore, in Liu et al. [2019] and Liu et al. [2020], potential (deep) reinforcement learning techniques, like deep Q-networks, are suggested for lung-cancer detection and treatment. Ou et al. [2021] use reinforcement learning for multi-round active screening for recurrent diseases. Lu et al. [2020] perform a sensitivity analysis on a state-of-the-art RL algorithm (Dueling Double Deep Q-Networks) applied to hemodynamic stabilization treatment strategies for septic patients in the ICU. However, varying the settings of their model resulted in significantly different policies, suggesting a need for caution when interpreting RL agent output and especially following a found policy. Another field in medical treatment that often uses RL is problems related to maximizing the effectiveness of the treatment while minimizing the side effects [Lizotte et al., 2010], [Laber et al., 2014], [Jalalimanesh et al., 2017].

3.3 Multi-objective reinforcement learning

Since RL has shown promising results, the literature is growing fast. The idea of learning from a reward is known for a long time, like training a dog by giving a reward when he responds appropriately to orders. Watkins [1989] started to use this idea in artificial intelligence. Next, he also proposed the algorithm Q-learning [Watkins and Dayan, 1992], which became one of the most popular RL algorithms. However, this algorithm had some limitations, such as the limitation of having one objective. In single-objective RL problems, we have one optimal value, and it might be that multiple different (optimal) policies all will have this value. The goal is to learn one of these optimal policies. Parallel to single-objective RL, there is multi-objective RL (MORL). In MORL, we aim to optimize more than one objective. Usually, these objectives are conflicting, and an improvement in one often results in a loss in another objective. Without any additional information on the user's utility, there can be many possible optimal value vectors. Therefore, we now have to work with sets of optimal value vectors with corresponding policies. If the user's utility is known or some assumptions can be made, the simplest solution is to extend single-objective methods like Q-learning. Then, the method should be adjusted such that the Q-values can be stored as vectors instead of scalars. A scalarization function should be incorporated that matches the user's utility to identify the greedy actions in each state. This is often a linear scalarization function [Perez et al., 2009], [Guo et al., 2009], [Shabani, 2009]. However, without any assumption of the user's utility, this might not be suitable. Then multi-policy approaches can be considered.

Roijers and Whiteson [2017] describe two classes of multi-policy approaches: outer loop and inner loop methods. An outer loop method operates on series of single-objective problems. in contrast, inner loop methods adapt the inner workings of a single-objective method to work with multi-objective solution concepts.

The simplest outer loop method iterates over a series of different user's utility function parameters and reruns a single-policy MORL algorithm for each setting. In Parisi et al. [2014], two different Multi-Objective Reinforcement-Learning (MORL) approaches are presented, called radial and Pareto following, that, starting from an initial policy, perform gradient-based policy-search procedures aimed at finding a set of non-dominated policies.

Pareto-Q-Learning [Van Moffaert and Nowé, 2014] and MPQ-Learning [Ruiz-Montiel et al., 2017] adjusted the inner Q-learning method, making it applicable to multiple objectives. It learns multiple different Pareto optimal policies at the same time. Pareto-Q-Learning is model-based because it stores the expected immediate reward and transition probabilities of all visited transitions during learning and needs to store sets of optimal values from state s to s' using action a at time step t. Ruiz-Montiel et al. [2017] instead only store the sets of values from state s to state s', without explicitly keeping track of the transition probabilities. Since their algorithm is model-free, it does need less storage.

Another limitation of the original Q-learning algorithm is the size of the state-space. Q-learning uses a table with a cell for each state-action pair, and this table is potentially huge. Mnih et al. [2015] came up with the idea of combining Deep Learning with reinforcement learning and use a Neural Net, termed a deep Q-network, to replace the Q-table. As a result, very large state spaces could still be solved. However, this idea was based on the original Q-learning technique, meaning that it only uses one objective [Sewak, 2019].

In Reymond and Nowé [2019], PDQN is proposed, a combination of PQL [Van Moffaert and Nowé, 2014] and a deep RL method to solve multi-objective problems. This is the first time an inner-loop method was devised for a deep reinforcement learning setting. This method was able to approximate the Pareto front of an often-used benchmark problem, *Deep Sea Treasure*. They also created a more complex state-space by creating a traffic intersection, where the traffic lights were considered as the agent. However, solving this problem did not provide a believable estimate. Due to the convergence of the network and the plausible waiting time, they still believe that the method could be further improved for solving high-dimensional state-space problems.

In Mossalam et al. [2016], Deep Optimistic Linear Support Learning (DOL) is introduced, which combines linear scalarization with deep Q-learning. When the relative importance of the objectives is unknown, they vary the weights for the scalarization whereafter the neural network is optimized, resulting in an optimal frontier. For each policy, they store the weights of the network, and for better efficiency, they propose to reuse the weights of previous policies for preferences that are similar to each other. Abels et al. [2019] extended this to use a single neural network to represent value functions over the entire space of preferences. However, scalarized updates are not sample efficient and lead to sub-optimal MORL policies [Yang et al., 2019]. Yang et al. [2019] introduced a new algorithm envelope Q-learning, which utilizes the convex envelope of the solution frontier to update parameters of the policy network. This is still a single-policy algorithm, but it leads to faster convergence when compared to scalarized updates for a given user preference.

Suppose the optimization problem, in particular the state-action space, is not too large. In that case, Deep Reinforcement Learning techniques might be too advanced and probably even computationally more expensive since of the need to optimize the neural network. Then, algorithms like Pareto-Q-Learning [Van Moffaert and Nowé, 2014] and MPQ-Learning [Ruiz-Montiel et al., 2017] might be better. Since it is unknown to what size of the state-action space these algorithms will be successful, we will experiment using one of these algorithms. Furthermore, since Ruiz-Montiel et al. [2017] claim to need less storage, and is unnecessary to explicitly learn the transition probabilities of all visited transitions, done by Van Moffaert and Nowé [2014], the MPQ-Learning algorithm is evaluated. Additionally, we will use a deep learning method called envelope Q-learning [Yang et al., 2019] to evaluate if a more complex method will improve the performance. Envelope Q-learning is one of the latest multi-objective deep reinforcement learning algorithm and they claim it has a relatively fast convergence.

4 Methodology

To understand MPQ-Learning (Multi-Pareto Q-learning) and Envelope Q-learning, it is of essence to know the main principles of reinforcement learning (RL). After the introduction of RL, a detailed explanation of the used methodology is given.

4.1 Reinforcement learning

The main idea behind Reinforcement Learning is that an agent learns by taking actions in an environment. A Reinforcement Learning environment can typically be described by means of a Markov Decision Process (MDP), which can be described as follows. Let $S = \{s_1, ..., s_N\}$ be the state space containing all states the agent can be in and $A = \{a_1, ..., a_r\}$ be the set of actions the agent can take. In each state $s \in S$, action $a \in A$ brings the agent to a new state s' with probability $\mathcal{P}(s'|s, a)$.

Depending on the transition from state s to s' due to action a, the agent will get a reward. After this transition, the agent has to select the next action, which also result in a reward and so forth. Therefore, we can divide the reward into two types: the immediate reward and the future reward. The immediate reward, denoted as $R_t(s, a)$, is the reward the agent gets directly after performing the action, while the future reward $R_{t+j}(s', a)$ for $j = \{1, 2, ..., n\}$ is based on the rewards the agent obtains from the future actions after his transitions to the new states. The agent does not know which action is best, but instead, he has to discover which actions will yield the highest reward by trying them [Sutton and Barto, 2018]. He will learn from his actions, and by repetition of a situation, he will decide upon his action based on the previous learnings. A visualization of the process is shown in Figure 4.1.

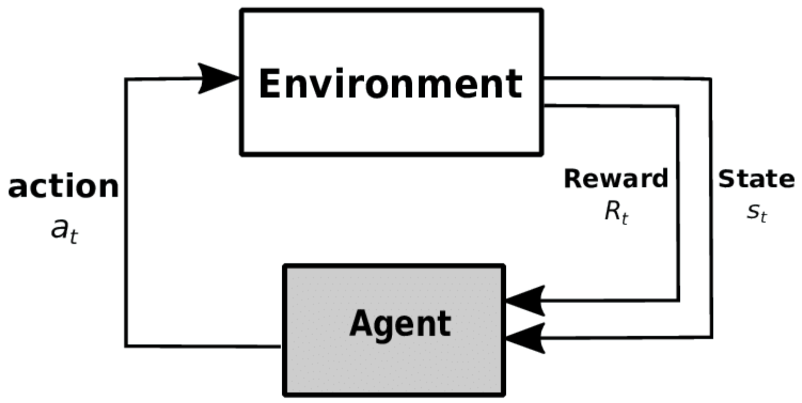


Figure 4.1: Visualisation of the process of Reinforcement Learning

The goal is to learn a deterministic stationary policy π, which maps each state to an 'optimal' action, such that the expected return received from the current time step t and onwards, is maximized. Here deterministic means that each state is mapped to a specific action, which is not stochastic. Furthermore, a stationary policy means that the policy does not change over time. The total expected (discounted) return from state s depends on the policy π and is called the value function $V^{\pi}(s)$. The formal definition of the state-dependent value function is defined as:

$$V^{\pi}(s) = E_{\pi}\Big\{\sum_{k=0}^{\infty}\gamma^{k} r_{t+k+1}|s_t = s\Big\}, \tag{4.1}$$

where $\gamma \in [0,1]$ is the discount factor and r_t the reward at time step t. When there is more than one objective, the rewards are vectors instead of scalars. This is usually called a Multi-Objective Markov Decision Process (MOMDP).

An essential part of this machine learning technique is the way the agent learns. Watkins [1989] proposed Q-Learning, a method that learns the value of each action for each state, which will converge to the optimum action-value as long as all actions are repeatedly sampled in all states and the action-values are represented discretely [Watkins and Dayan, 1992]. This method stores all current estimates of the state-action values in a table (Q-table), where the value of action $a \in A$ at state $s \in S$ is denoted as

$$Q(s,a) := r(s,a) + \gamma \sum_{s' \in S} \mathcal{P}(s'|s,a) \cdot V^{\pi}(s'). \tag{4.2}$$

This standard Q-learning algorithm then utilizes the Bellman optimality operator T:

$$(TQ)(s,a) := r(s,a) + \gamma E_{s' \sim \mathcal{P}(\cdot|s,a)}(HQ)(s'), \tag{4.3}$$

where $(HQ)(s') := \sup_{a' \in A} Q(s',a')$ is an optimality filter over the Q-values for the next state s', i.e., the highest (expected) reward by means of selecting the best action in the next state s'. After an action has been performed and the new state has been visited by the algorithm, the corresponding cell of the Q-table is updated as follows:

$$\hat{Q}(s,a) \leftarrow (1-\alpha_t)\hat{Q}(s,a) + \alpha_t(r(s,a) + \gamma(H\hat{Q})(s')), \tag{4.4}$$

where α_t is a learning rate that can vary over time. This parameter determines how much the new experienced reward influences the previous experienced rewards. If α_t is large, the new rewards have high impact. However, the duration of this impact is short. There is no general optimal value for α_t, such that the algorithm learns as fast as possible. Different options should be considered to determine a suitable value.

During learning, the agent needs to choose between performing the current best action (exploitation) and another action (exploration). To select between these two options, the agent uses an action selection strategy, such as ϵ-greedy, where a random action is selected with probability ϵ and the greedy action with probability $(1 - \epsilon)$. Again, there is no general optimal value for ϵ and different values should be considered.

4.2 MPQ-learning

MPQ-learning [Ruiz-Montiel et al., 2017] is a multi-objective RL algorithm. It aims to find all deterministic Pareto-optimal policies for a MOMDP simultaneously. The reward r of equation (4.4) is not a scalar anymore but becomes a vector $\vec{R} : S \times A \times S \rightarrow \mathbb{R}^2$. The two components of a reward vector $\vec{r} = \vec{R}(s, a, s')$ both stand for one of the scalar reward received after performing action a in state s and reaching state s'. The rewards in our problem are (1) the financial costs and (2) the QALYs.

Since there is more than one optimal combination of the objectives in multi-objective optimization, we keep track of multiple non-dominated vectors simultaneously. Parallel to the Q-table in the single-objective case, we keep track of a $\mathbb{Q}$-table. For each state-action pair (s,a), $\mathbb{Q}$(s,a) contains all non-dominated reward vectors found so far. Elements in $\mathbb{Q}(s, a)$ are pairs $(\vec{q}, P)$, where $\vec{q}$ is one of the non-dominated reward vectors. P is a set of indices that tracks how the vector $\vec{q}$ has been obtained. Recall that each vector $\vec{q}$ is obtained by a past transition (s, a, s') by the agent. Therefore each element $p \in P$ is a pair (s', i) that indicates the state s' that was transitioned to when $\vec{q}$ was learned. i Stands for the element in $\mathbb{V}(s')$ that was used to calculate $\vec{q}$. Parallel to V(s) in the single-objective case, we define the set $\mathbb{V}(s)$ as the set having all non-dominated expected discounted returns in state s, which is as follows:

$$\mathbb{V}(s) = ND \bigcup_{a \in A} \Big\{ \vec{q} \mid (\vec{q}, P) \in \mathbb{Q}(s, a) \Big\}, \tag{4.5}$$

where ND stands for non-dominated, meaning that the set only contains the vectors that are not dominated by any other vector in the set. A vector $\vec{a}$ is dominated by a vector $\vec{b}$ if for all elements $a(i)$ it holds that $b(i) \geq a(i)$, and that for at least one element $a(i)$ it holds that $b(i) > a(i)$.

$\mathbb{Q}(s, a)$ are initialized as $\mathbb{Q}_0(s, a) = \{\vec{0}, \varnothing\}$, $\forall a \in A, \forall s \in S$. During the learning process, $\mathbb{Q}(s, a)$ is updated as follows:

$$\mathbb{Q}_n(s,a) = \begin{cases} \mathbb{N}_{n-1}(s,a) \cup \mathbb{U}_{n-1}(s,a) \cup \mathbb{E}_{n-1}(s,a) & \text{if } s = s_n \wedge a = a_n \\ \mathbb{Q}_{n-1}(s,a) & \text{otherwise,} \end{cases} \tag{4.6}$$

where we introduce three different sets: $\mathbb{N}$ (New), $\mathbb{U}$ (Updated), and $\mathbb{E}$ (Extra). These sets contain new reward vectors that are included in $\mathbb{Q}_n(s,a)$. In MPQ-learning, there are two different updating procedures, which depend on the transition state, and therefore determine the composition of the sets $\mathbb{N}$, $\mathbb{U}$, and $\mathbb{E}$. The updating procedures are:

1. Creating new vector estimates every time a state s' is reached for the first time from s after performing action a. In this case, only the set $\mathbb{N}$ contains vector estimates, while $\mathbb{U}$ and $\mathbb{E}$ are empty.
2. Updating the vector estimates in $\mathbb{Q}(s,a)$ when transition state s' is already explored from s after action a. This includes updating, creating, and deleting particular vector estimates. During this procedure, $\mathbb{N}$ is empty, while $\mathbb{U}$ and $\mathbb{E}$ may have vector estimates.

We first discuss the first case, which means that state s' has not been reached yet from state s after performing action a. In other words, the transition (s,a,s') is new in the current time step. The sets $\mathbb{U}$ and $\mathbb{E}$ remain empty, and only the set $\mathbb{N}$ is filled, according to the following definition:

$$\mathbb{N}_{n-1}(s,a) = \Big\{((1-\alpha_n)\vec{q} + \alpha_n\Big[\vec{r}_n + \gamma\vec{v}_j\Big], P \cup \Big\{(s',j)\Big\}) \;\Big|\; (\vec{q},P) \in \mathbb{Q}_{n-1}(s,a) \wedge \vec{v}_j \in \mathbb{V}_{n-1}(s') \wedge s' \not\sqsubset \mathbb{Q}_{n-1}(s,a)\Big\}. \tag{4.7}$$

First, consider the conditional part on the right side of the definition. The most right condition indicates that state s' has not been reached from the state action pair (s,a). The other two conditions imply an iteration over all currently non-dominated reward vectors in $\mathbb{Q}_{n-1}(s,a)$ and all non-dominated vectors in $\mathbb{V}_{n-1}(s')$. In each iteration, we update a current reward vector $\vec{q}$ with the direct reward $\vec{r}_n$ obtained by the action and a non-dominated reward that could be obtained from state s', represented by vector $\vec{v}_j$. Simultaneously, we add state (s',j) to the set P to indicate that state s' can be reached with (s,a) and that the calculated rewards were obtained with vector $\vec{v}_j$. Since we loop over all elements in $\mathbb{Q}(s,a)$ and $\mathbb{V}(s')$, $\mathbb{N}$ contains $|\mathbb{V}_{n-1}(s')| \cdot |\mathbb{Q}_{n-1}(s,a)|$ elements.

In the second case, state s' has already been reached from state s after performing action

a. In other words, the transition (s, a, s') is known in the current time step. In this case, $\mathbb{N}$ is empty, and the sets $\mathbb{U}$ and $\mathbb{E}$ are filled. Since it is not the first time the agent has done this transition, at least one vector of $\mathbb{Q}_{n-1}(s, a)$ can be linked to an expected discounted reward vector $\vec{v}_j \in \mathbb{V}_{n-1}(s')$. Therefore, the vectors in $\mathbb{Q}_{n-1}(s, a)$ that are linked should be updated with their linked $\vec{v}_j$, which is captured in the set $\mathbb{U}$ as follows:

$$\mathbb{U}_{n-1}(s,a) = \Big\{((1-\alpha_n)\vec{q} + \alpha_n \Big[\vec{r}_n + \gamma\vec{v}_j\Big], P) \;\Big|\; (\vec{q}, P) \in \mathbb{Q}_{n-1}(s,a) \wedge (s', j) \in P \wedge \vec{v}_j \in \mathbb{V}_{n-1}(s')\Big\}. \quad (4.8)$$

Again, if we look at the conditional part on the right side of the definition, the first and the last condition imply an iteration over all currently non-dominated reward vectors in $\mathbb{Q}_{n-1}(s, a)$ and all non-dominated vectors in $\mathbb{V}_{n-1}(s')$. The middle condition makes sure that only the vectors of $\mathbb{Q}_{n-1}(s, a)$ are used if linked to the transition to s', and that they are related to the corresponding discounted reward vector $\vec{v}_j$. Then the update is done by the weighted sum of the already learned expected reward $\vec{q}$ and the new immediate reward plus the discounted expected future reward $(\vec{r}_n + \gamma\vec{v}_j)$.

It might be that after the last time that the transition (s, a, s') took place, new reward vectors $\vec{v}_j \in \mathbb{V}(s')$ are learned. This implies that new non-dominated expected rewards can also be achieved from state s after performing action a. In this case, we have extra discounted reward estimates $\vec{v}_j \in \mathbb{V}(s')$, which are not yet linked to any expected reward $\vec{q} \in \mathbb{Q}(s, a)$. We have to start from the initial value $\vec{0}$ to determine the value in the new pair that is being inserted in $\mathbb{Q}_n(s, a)$. However, we can use the already established pairs of indices from the estimates we already had for the new vector $\vec{v}_j$. Therefore, given a new vector $\vec{v}_j$, we insert a new vector in $\mathbb{Q}_n(s, a)$ for each vector in $\mathbb{Q}_{n-1}(s, a)$, using the same set of indices without the indices (s', k) (whatever k is) and add the pair (s', j). These cases are included in $\mathbb{E}$ as follows:

$$\mathbb{E}_{n-1}(s,a) = \Big\{(\alpha_n \Big[\vec{r}_n + \gamma\vec{v}_j\Big], (P \setminus s') \cup \Big\{(s', j)\Big\}) \;\Big|\; \vec{v}_j \in \mathbb{V}_{n-1}(s') \wedge s' \sqsubset \mathbb{Q}_{n-1}(s,a) \wedge (s', j) \not\sqsubset \mathbb{Q}_{n-1}(s,a) \wedge \exists \vec{q} \;\; (\vec{q}, P) \in \mathbb{Q}_{n-1}(s,a)\Big\}. \quad (4.9)$$

The first condition in the conditional part of the definition implies an iteration over all non-dominated vectors in $\mathbb{V}_{n-1}(s')$. The second condition makes sure that the transition (s, a, s') is not new, while the third condition makes sure that only the vectors $\vec{v}_j$ are selected that are not linked yet to any $\vec{q} \in \mathbb{Q}_{n-1}(s,a)$. Finally, the last condition iterates over all P, so for each vector estimate that already was in $\mathbb{Q}_{n-1}(s,a)$ a new vector is added. However, the links to s' in P are replaced with the new (s',j).

The union of these three sets will eventually be the new $\mathbb{Q}_n(s,a)$. A practical example of the MPQ-learning updates based on these three sets can be found in Ruiz-Montiel et al. [2017] Section 2.4. Since the sets $V(s')$ only take the non-dominated vector estimates (as described in equation 4.5), a dominated vector estimate in $\mathbb{Q}_{n-1}(s,a)$ will not be found in $\mathbb{Q}_n(s,a)$ anymore. Hence, MPQ-learning keeps a vector in $\mathbb{Q}(s,a)$ for each possible combination of non-dominated vectors from the state-vector sets of reachable states. However, they do not limit the number of these vectors, which depending on the problem might become extremely large.

In Ruiz-Montiel et al. [2017] the action selection during the learning process is based on a probability that is proportional to the number of vector estimates of $\mathbb{Q}(s,a)$ that are also inside $\mathbb{V}(s)$. This implies that an action having more non-dominated vectors has a higher probability of being selected. The probabilities are defined as:

$$Pr\Big\{a_n = a | s_n = s\Big\} = \frac{\Big|\Big\{\vec{q} : \vec{q} \in \mathbb{V}_{n-1}(s) \wedge \exists P : (\vec{q}, P) \in \mathbb{Q}_{n-1}(s,a)\Big\}\Big|}{|\mathbb{V}n-1(s)|}. \tag{4.10}$$

We will use an ϵ-greedy mechanism, meaning that a random selection is chosen with probability ϵ, and an action is chosen according to Expression 4.10 with probability $(1-\epsilon)$.

4.3 Envelope Q-learning

The second algorithm that is used is called envelope Q-learning [Yang et al., 2019], which is also developed for MORL[1]. In contrast to the previous algorithm, this can be seen as an outer-loop method. The goal of this algorithm is to learn all optimal policies over the entire preference space by varying linear relative preferences $\boldsymbol{w}$ to scalarize the objectives.

Since the Belmann optimality operator T of equation (4.3) can only be used for the single-objective case, we first define a multi-objective evaluation operator $\mathcal{T}_\pi$. Given a policy π and sampled transitions τ, this operator is defined as:

$$(\mathcal{T}_\pi \boldsymbol{Q})(s,a,\boldsymbol{w}) := \boldsymbol{r}(s,a) + \gamma E_{\tau\sim(\mathcal{P},\pi)} \boldsymbol{Q}(s',a',\boldsymbol{w}), \tag{4.11}$$

[1]The code of Yang et al. [2019] can be found at https://github.com/RunzheYang/MORL

where $\boldsymbol{Q}(s, a, \boldsymbol{w})$ are the Multi-Objective Q-estimates (MOQ) of expected total rewards under m-dimensional preference vectors $\boldsymbol{w}$. We use $\tau \sim (\mathcal{P}, \pi)$, since the transitions given an action can be stochastic, following a probability distribution $\mathcal{P}$. Furthermore, the selected (future) actions depend on the policy π.

We then define an optimality filter $\mathcal{H}$ for the MOQ function as

$$(\mathcal{H}\boldsymbol{Q})(s, \boldsymbol{w}) := \arg_Q \sup_{a \in \mathcal{A}, \boldsymbol{w}' \in \Omega} \boldsymbol{w}^\intercal \boldsymbol{Q}(s, a, \boldsymbol{w}'),$$

where the $\arg_Q$ takes the multi-objective value corresponding to the supremum (i.e., $\boldsymbol{Q}(s, a, \boldsymbol{w}')$ such that $(a, \boldsymbol{w}') \in \arg\sup_{a \in \mathcal{A}, \boldsymbol{w}' \in \Omega} \boldsymbol{w}^\intercal \boldsymbol{Q}(s, a, \boldsymbol{w}')$). In other words, $(\mathcal{H}\boldsymbol{Q})(s, \boldsymbol{w})$ is the optimal expected multi-objective reward in state s for preference $\boldsymbol{w}$. Note that the return of $\arg_Q$ depends on the chosen preference $\boldsymbol{w}$ for scalarization. Now, we can define a *multi-objective optimality operator* $\mathcal{T}$ as:

$$(\mathcal{T}\boldsymbol{Q})(s, a, \boldsymbol{w}) := \boldsymbol{r}(s, a) + \gamma E_{s' \sim \mathcal{P}(\cdot|s,a)} (\mathcal{H}\boldsymbol{Q})(s', \boldsymbol{w}). \tag{4.12}$$

For proof of the feasibility of using this optimality operator, I refer to Yang et al. [2019].

Instead of using a Q-table for all state-action pairs, we use a deep neural network with the state s and preference $\boldsymbol{w}$ as input and $|\mathcal{A}| \times m$ Q-values as output. Using double Q learning with target Q networks following Mnih et al. [2015] we minimize the following loss function at each step k:

$$L_k^A(\theta) = E_{s,a,\boldsymbol{w}}\Big[\|\boldsymbol{y}_k - \boldsymbol{Q}(s, a, \boldsymbol{w}; \theta)\|_2^2\Big], \tag{4.13}$$

where $\boldsymbol{y} = E_{s'}[\boldsymbol{r} + \gamma \arg_Q \max_{a,\boldsymbol{w}'} \boldsymbol{w}^\intercal \boldsymbol{Q}(s', a, \boldsymbol{w}'; \theta_k)]$, which empirically can be estimated by sampling transitions (s, a, s', r) from a replay buffer, whitch is a batch of stored transitions that have occurred in the past. In other words, the loss function uses the previous optimization result as an initial guess in each step.

However, the optimal frontier contains many discrete solutions, which makes the landscape of the loss function substantially non-smooth. Therefore, optimizing L_k^A directly is challenging in practice, so we make use of the method known as *homotopy optimization* [Watson and Haftka, 1989] by using an auxiliary loss function L_k^B:

$$L_k^B(\theta) = E_{s,a,\boldsymbol{w}}[|\boldsymbol{w}^\intercal \boldsymbol{y}_k - \boldsymbol{w}^\intercal \boldsymbol{Q}(s, a, \boldsymbol{w}; \theta)|]. \tag{4.14}$$

First, L_k^A ensures that the prediction of $\boldsymbol{Q}$ is close to any real (perhaps not optimal) expected total reward, while later L_k^B can provide an auxiliary force to pull the current guess along the direction with better utility, since we include the preference $\boldsymbol{w}$. In L_k^A the objectives have the same 'weight'. However, in L_k^B we optimize the scalarization of the objectives, which depends on the preference.

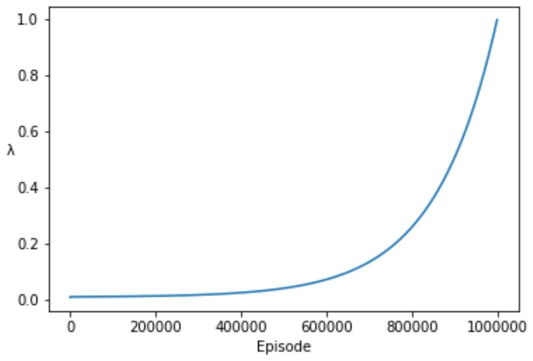


Figure 4.2: Visualization of the shift of L_k^A to L_k^B by increasing λ from 0 to 1 using 1,000,000 Episodes.

Therefore, the final loss function is a combination $L_k(\theta) = (1-\lambda) \cdot L_k^A(\theta) + \lambda \cdot L_k^B(\theta)$, where λ is a weight to trade-off between losses L^A and L^B. By slowly increasing the value of λ from 0 to 1, we shift our loss function from L_k^A to L_k^B. A visualization of the shift used in this research is shown in Figure 4.2, but other shifts might also be suitable.

The loss functions in (4.13) and (4.14) have an expectation over $\boldsymbol{w}$. Note that $\boldsymbol{w}$ influences the agent's preferred actions but does not influence the transitions and the rewards. This means that the same transitions can be used using different preferences ($\boldsymbol{w}$) for learning. Therefore, we can increase sample efficiency by using a scheme similar to Hindsight Experience Replay (HER) [Andrychowicz et al., 2017]. In this paper, an agent performs in each episode actions according to different random goals. When updating, they use the previous transitions for the current update with multiple different goals in parallel. Although the context of their and our application are completely different, a similar idea of this scheme can be used to update our multi-objective Q-network.

This is done as follows: in each episode, we sample a random preference $\boldsymbol{w}$ from a distribution $\mathcal{D}_{\boldsymbol{w}}$, which is used for the standard learning phase. The transitions $(s_t, a_t, \boldsymbol{r}_t, s_{t+1})$ are stored in a replay buffer $\mathcal{D}_\tau$. When updating the multi-objective Q-network, we select a mini-batch of size N_τ of transitions randomly from $\mathcal{D}_\tau$ and each transition is associated with $N_{\boldsymbol{w}}$ different preferences $\{\boldsymbol{w}_1, ..., \boldsymbol{w}_{N_{\boldsymbol{w}}}\}$, which are also randomly sampled from $\mathcal{D}_{\boldsymbol{w}}$. Finally, the Q-network will then be updated using a total batch of $N_\tau \times N_{\boldsymbol{w}}$ samples. The skeleton of the algorithm is shown in Algorithm 1.

Since Envelope Q-learning includes the preference ($\boldsymbol{w}$), the multi-objective rewards of actions can easily be transformed into a single objective. Therefore, the action selection mechanism is similar to the ϵ-greedy strategy used in single-objective Q-learning. Each episode, a random $\boldsymbol{w}$ is selected, and based on this preference and the current state, the best action is determined using the current multi-objective Q-network. With probability 1-ϵ, the agent chooses this best action. However, with probability ϵ, the agent performs a random action, which still can be the best option.

Algorithm 1: Envelope MOQ-Learning
Adapted source: Yang et al. [2019]

```
Input : a preference sampling distribution $\mathcal{D}_w$, path
        $p_\lambda$ for the balance weight $\lambda$ increasing from
        0 to 1.
Initialize replay buffer $\mathcal{D}_\tau$, network $\boldsymbol{Q}_\theta$, and $\lambda = 0$.
for episode = 1,...,M do
    Sample a linear preference $\boldsymbol{w} \sim \mathcal{D}_w$
    for $t = 0, ..., N$ do
        Observe state $s_t$.
        Sample an action $\epsilon$-greedily:
        $a_t = \begin{cases} \text{random action in } \mathcal{A}, & \text{w.p. } \epsilon; \\ \max_{a \in \mathcal{A}} \boldsymbol{w}^\intercal \boldsymbol{Q}(s, a, \boldsymbol{w}; \theta), & \text{w.p. } 1-\epsilon. \end{cases}$
        Receive a vectorized reward $\boldsymbol{r}_t$ and observe
          $s_{t+1}$.
        Store transition $(s_t, a_t, \boldsymbol{r}_t, s_{t+1})$ in $\mathcal{D}_\tau$
        if update then
            Sample $N_\tau$ transitions
              $(s_t, a_t, \boldsymbol{r}_t, s_{t+1}) \sim \mathcal{D}_\tau$.
            Sample $N_w$ preferences $W = \{\boldsymbol{w}_i \sim \mathcal{D}_w\}$.
            Compute $y_{ij} = (\mathcal{T}\boldsymbol{Q})_{ij} =$
            $\begin{cases} \boldsymbol{r}_j, & \text{for terminal } s_{j+1}; \\ \boldsymbol{r_j} + \gamma \arg_Q \max\limits_{a \in \mathcal{A}, \boldsymbol{w}' \in W} \boldsymbol{w}_i^\intercal \boldsymbol{Q}(s_{j+1}, a, \boldsymbol{w}'; \theta), & \text{o.w.} \end{cases}$
            for all $1 \leq i \leq N_w$ and $1 \leq j \leq N_\tau$.
            Update $Q_\theta$ by descending its stochastic
              gradient according to equations 4.13 and
              4.14:
            $\nabla_\theta L(\theta) = (1-\lambda) \cdot \nabla_\theta L^A(\theta) + \lambda \cdot \nabla_\theta L^B(\theta)$.
            Increase $\lambda$ along the path $p_\lambda$.
```

4.4 State-action space

The (discrete) state-space is based on the characteristics of the patients. The characteristics used are the age and the screening history, which are the results of the screening tests that a person already has done. We will experiment using different state spaces. In the initial case, we will use the same state-space used in Van Duuren [2021]. The screening will be done in the age between 40 and 85 years, and we split the range into cohorts of 5 years, which result in $\mathcal{T} := \{40, 45, ..., 80\}$. Furthermore, we use a risk estimator of having CRC. We assume that a concentration above 100μg/g implies a risk of 1. Therefore the risk estimator is defined as:

$$R^k := \min\left[\frac{1}{100k}\sum_{i=0}^{k} S_{n-i}, 1\right],$$

where S_j is the participant's j^{th} quantitative FIT-result, i.e., the measured Hb concentration. We use $k = 1$, since variation of k did not improve the quality of the policies in Van Duuren [2021] and averaging the Hb concentrations would also decrease the extreme Hb values. The risk is discretized as $\mathcal{R} := \{0, 0.125, 0.25, ..., 1\}$.

Therefore, this primary case will have two dimensions $\mathcal{T} \times \mathcal{R}$ and a total of $9 \cdot 9 = 81$ individual states. Since the MISCAN-Colon micro-simulation does not distinguish yet between males and females when simulating CRC, we do not differentiate in gender and select the male version of the simulation of Hb concentrations, which is already prepared for differentiation in gender in the MISCAN simulation.

Many variations in state spaces can be evaluated. The cohort sizes of R and T can be adjusted. A larger history of quantitative FIT results, including the second and third last FIT results, can be added to the state-space, and the result of a colonoscopy could be incorporated. During the research, the use of specific variations in state spaces has to be decided.

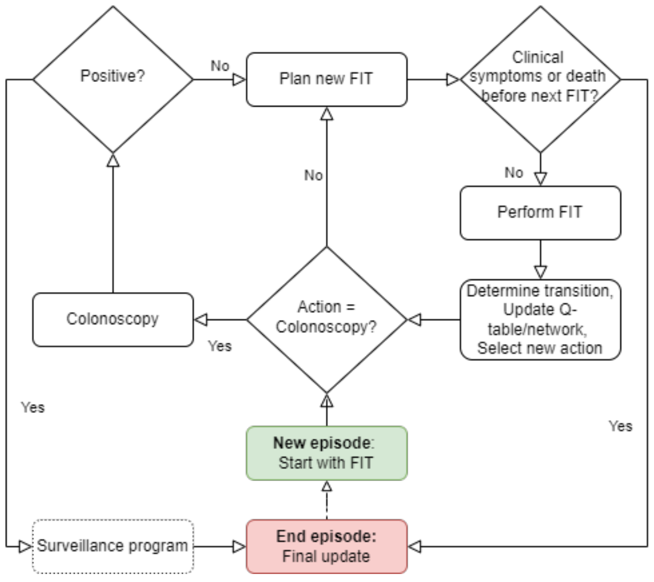
The actions represent the screening strategy. An action determines the next screening decision, having three options: Stop screening, FIT after x years or undergo a colonoscopy. Depending on the different options of x, which will be between one and ten years, the agent has at most 12 different actions. For the primary situation, we use $x = \{1, 2, 3\}$, and exclude the option 'Stop screening', resulting in a total of four different actions: $\mathcal{A} := \{COL, FIT_1, FIT_2, FIT_3\}$. The entire state-action space is $\mathcal{T} \times \mathcal{R} \times \mathcal{A}$, having 324 unique state-action pairs.

The (negative) rewards related to the actions depend on the actual costs of screening and the surveillance or treatment costs as a consequence of the the selected screening action, which is explained in the next section.

5 Implementation in MISCAN

Our algorithm is designed to optimize FIT-based screening programmes. In such programmes, individuals take a FIT test with a certain interval. Once the blood concentration measured by the FIT is above a prespecified cutoff, the participant is redirected to a hospital for a follow-up colonoscopy. Our algorithm aims to personalize the cutoff and screening interval.

To learn and evaluate the long- and short-term harms and benefits of the algorithm's screening policies, the algorithm was built into MISCAN-Colon. Figure 5.1 shows the structure of the combination.



Positive?
No
Plan new FIT
Clinical symptoms or death before next FIT?
No
Perform FIT
No
Determine transition, Update Q-table/network, Select new action
Colonoscopy
Action = Colonoscopy?
Yes
Yes
Yes
New episode: Start with FIT
Surveillance program
End episode: Final update

Figure 5.1: Flowchart of the learning process implemented in MISCAN

Participants undergo their first FIT once they reach the first screen-eligible age (the first element in $\mathcal{T}$). Combined with the measured blood concentration, we obtain the participant's state $s \in \{\mathcal{T} \times \mathcal{R}\}$, from which the first action $a \in \mathcal{A}$ can be determined.

If the selected action a is a FIT after x years, the MISCAN simulation continues in combination with the RL algorithm, and after x years, the new FIT is performed, whereafter we know the new state $s' \in \{\mathcal{T} \times \mathcal{R}\}$. Now we know the transition $(s, a, \boldsymbol{r}, s')$, where $\boldsymbol{r}$ is reward vector, containing long and short term estimates of the financial costs and QALYs of this action. Using this transition, we can update the Q-table and the multi-objective Q-network in the MPQ-learning and Envelope Q-learning algorithms respectively.

If the chosen action is a *colonoscopy*, the patient is directly referred to a hospital for a colonoscopy. After a positive colonoscopy, the person will go to a surveillance program. This program is fixed because it is not part of the optimization problem to which the algorithm is applied. The Q-table/network is updated with the costs and QALYs associated with the surveillance when the person dies. If the colonoscopy is negative, a new FIT after a fixed period is planned, whereafter the new state of the person can be decided. Then we can perform an update, where $\boldsymbol{r}$ consists of both the short- and long-term costs and QALYs of the colonoscopy and the FIT. This period is defined before the simulation starts. The Dutch screenings strategy uses a period of ten years, while Van Duuren [2021] used five years. To compare our results with Van Duuren [2021], we will also use a period of five years.

Note that if a person dies before the screen-eligible age, the person does not begin the RL process, since the agent has not chosen any action yet. This means that during the life of this individual nothing is learned and we do not count this as an episode. Furthermore, when a person dies after the screen-eligible age, this is denoted as the final state. The final reward is calculated, which depends on the caused death and possible surveillance or treatment costs. The QALYs in the final update are based on the life expectancy of the individual without CRC, his other cause (OC) death. If a person dies, the difference between the OC death and the CRC death is added to the QALYs as a penalty. Details of the specific rewards are included in Appendix B.

The MISCAN-Colon micro-simulation is programmed in Python, where we incorporated the RL algorithms. Therefor, the screening process is adjusted to allow dynamic strategies, instead of static strategies. Originally, a strategy is completely defined before the simulation starts, meaning that all FITS, having a fixed cutoff, are scheduled beforehand. Now, each next FIT is scheduled after the last FIT, while the cutoffs varies over age and changes during training. Furthermore, the costs and effects are normally calculated afterwards based on all events and durations that are stored. During training, we also calculate the costs and effects per individual, which is needed for the updates.

Common seeds are used for reproducability and to compare the results of different strategies. For testing we used the same (MISCAN-Colon) simulation without updating the MPQ-table or Q-network, but only selecting the optimal action for each state. For speeding up the process of testing the envelope Q-learning algorithm, we created a table with the optimal action for each state based on the output of the neural network, whereafter the neural network is not needed in the simulation.

6 Results

In this section we start with the results of MPQ-learning, whereafter Envelope Q-learning is evaluated. We evaluate the strategies that we have found and compare the best strategies with the current Dutch screening strategy and the strategies advised by Knudsen et al. [2021] to the United States Preventive Services Task Force (USPSTF) in October 2020.

6.1 MPQ-learning

Currently, there is no package of MPQ-learning and the authors did not provide any codes. Therefore, we implemented the algorithm ourselves. For validation of a correct implementation of the algorithm, we started by replicating the example in Section 2.4 of Ruiz-Montiel et al. [2017]. We were able to successfully reproduce each step. Furthermore, we were able to continue the learning process and converge to the optimal Q-table.

However, when implementing this method in MISCAN, it appeared that two situations were not accounted for. The first problem was that it is possible that completely identical reward vectors are created in the same state-action pair, resulting in unnecessary extra storage and calculations. We explain this using a made-up example shown in Figure 6.1. Suppose after a few transitions we have found two reward vectors in $\mathbb{V}(s_1)$ and both are related to $\mathbb{V}(s_2)$ and $\mathbb{V}(s_3)$. Next, After some more transitions, where s_2 is reached from other states than s_1, we have obtained a new non-dominated vector v_3. Note that $\mathbb{V}(s2)$ does not contain a vector v_1 nor v_2. Apparently they were both dominated by v_3 and therefore removed from $\mathbb{V}(s_2)$. Consequently, the two vectors in $\mathbb{V}(s_1)$ refer to non-existent vectors in $\mathbb{V}(s_2)$. Figure 6.1 shows the situation at this point.

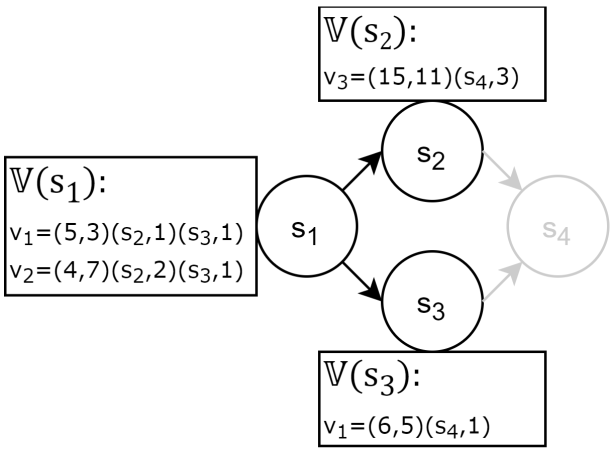


Figure 6.1: Example of problem 1 of MPQ-learning

When a new episode starts and we again get the transition (s_1, a_1, s_2), the transition is not new, so $\mathbb{N} = \emptyset$ and because v_1 and v_2 are not related to an existing vector, it also means that $\mathbb{U} = \emptyset$. As a consequence, v_3 in $\mathbb{V}(s_2)$ is an 'extra' vector, that will create two new vectors in the set $\mathbb{E}$ as described in Section 4.2. However, based on equation (4.9) the two new vectors will become identical: the reward vector is $\alpha_n\Big[\vec{r}_n + \gamma\vec{v}_j\Big]$. Furthermore, $P \leftarrow (P \setminus s') \cup \Big\{(s', j)\Big\}$. This means that for v_1 in $\mathbb{V}(s_1)$ we create a new vector by replacing $(s_2, 1)$ with $(s_2, 3)$ and we keep $(s_3, 1)$, which together results in $v_3 = \Big(\alpha_n\Big[\vec{r}_n + \gamma\vec{v}_3\Big]\Big)(s_2, 3)(s_3, 1)$ in $\mathbb{V}(s_1)$. For v_2 we replace $(s_2, 2)$ by $(s_2, 3)$ and keep $(s_3, 1)$, which together results in $v_4 = \Big(\alpha_n\Big[\vec{r}_n + \gamma\vec{v}_3\Big]\Big)(s_2, 3)(s_3, 1)$ in $\mathbb{V}(s_1)$. Even though the reward values of v_1 and v_2 were different, the created vectors in $\mathbb{E}$

are identical and they will always be updated the same way and create the same vectors. If this happens often, the sets grow exponentially fast. Since, identical vectors do not add any information, we decided to always drop vectors if they were identical.

The second problem is related to the vector (v_j) identification numbers j. In this algorithm it is possible to create two different vectors with the same number j for the same state-action pair. Again, we evaluate a made-up example, which is shown in Figure 6.2. In this example we ignore the reward values, since these are not relevant to the problem.

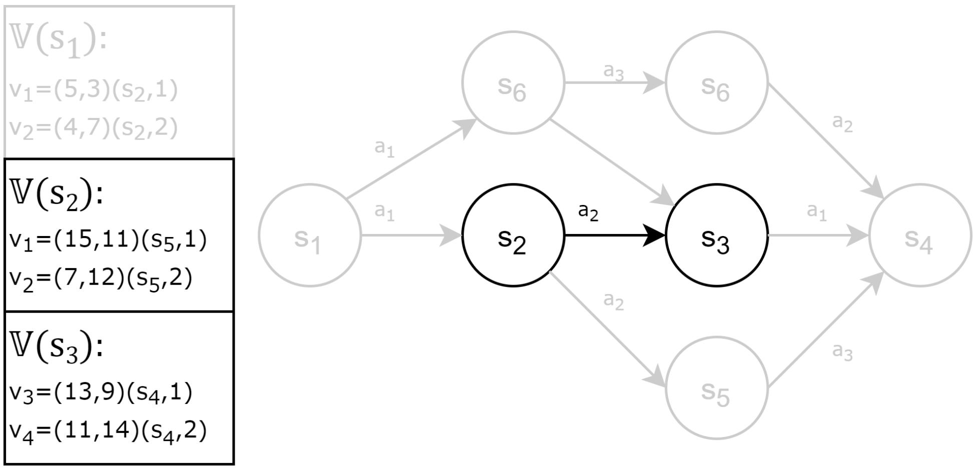


Figure 6.2: Example for problem 2 of MPQ-learning

In this example we have learned some vectors in $\mathbb{V}(s_2)$ and $\mathbb{V}(s_3)$ and now it is the first time that we get the transition (s_2, a_2, s_3). Therefore, we know that $\mathbb{U} = \mathbb{E} = \emptyset$ and we only have to look at the set $\mathbb{N}$, which is formed according to equation (4.7). As described in section 4.2, $\mathbb{N}$ contains $|\mathbb{V}_{n-1}(s')| \cdot |\mathbb{Q}_{n-1}(s,a)|$ elements, so $\mathbb{N}$ consists of four new vectors: all combinations of one element of $\mathbb{V}(s_2)$ and one element of $\mathbb{V}(s_3)$. The number j of the vectors in $\mathbb{V}(s_2)$ are kept the same, since these values are used to link the vectors in $\mathbb{V}(s_1)$ to those in $\mathbb{V}(s_2)$. This link should still be held, which can also be seen in the example of the original paper [Ruiz-Montiel et al., 2017]. However, we cannot give two vectors in one state the same number j. On the other hand, if we give the vectors a new number j, the link with s_1 does not exist anymore, so we throw away some information. Ideally, we also create two new vectors in s_1 per situation, but this results in inefficiencies in the backtracking procedure. Therefore, we decided to let one vector keep its original number j and to give the other vectors a new number. Consequently, these are not linked to vectors in other $\mathbb{V}(s_i)$ sets. Therefore, when state s_2 is visited next, these vectors end up in the $\mathbb{E}$ set.

The solution to the second problem, led to a third problem: storage errors. Since the MIS-

CAN simulation is very stochastic, the algorithm often encountered 'extra' vectors in $\mathbb{V}(s_j)$ that were used to create the $\mathbb{E}$-set according to Equation (4.9). However, per 'extra' vector $|\mathbb{Q}_{n-1}(s,a)|$ new vectors are created. This grows exponentially fast, resulting in a storage error. Even if only the non-dominated vectors are kept, infinitely many non-dominated rewards can still be found during learning.

We emailed the authors asking about the difference in storage space of the algorithms Pareto Q-learning [Van Moffaert and Nowé, 2014] and MPQ-learning [Ruiz-Montiel et al., 2017], one of the authors of MPQ-learning mentioned the following (The complete reply can be found in Appendix C):

> About 40 years ago, White analyzed the problem of multi-objective dynamic programming and proposed an exact algorithm. However, I am not aware of any practical applications of that algorithm. Why? Probably because even very simple MORL problems can have an exponential (or even infinite) number of solutions, which makes it impractical to apply exact algorithms. Both PQ and MPQ address this same problem (in the case where transition probabilities are unknown), and so are likely bound to have similar limitations in practice. (L. Mandow, personal communication, July 13, 2021)

In the literature, there is not much experienced yet with MPQ-learning. However, Hasan et al. [2019] mentioned that MPQ-learning requires high convergence time. Mandow and Pérez-de-la Cruz [2018] came up with MPQ2, which is an adjustment of MPQ-learning and should need less training, but this method will take more space than the original method. Besides, they tested the methods on the well-known MORL benchmark Deep See Treasure (DST) [Vamplew et al., 2011], in which an agent controls a submarine searching for treasures in a 10×11-grid world, while trading off time-costs and treasure-values. This is a much simpler problem compaired to our personalized screening optimization due to the stochastic nature of the MISCAN simulation.

Based on the reply of L. Mandow and our findings, we concluded that MPQ-learning is not suitable for solving our problem and decided to switch to the second method, Envelope Q-learning, which is not precisely the scalarization method that L. Mandow suggested. However, it incorporates the linear weights, where scalarization is often based on, to find optimal strategies over the whole preference space.

6.2 Envelope Q-learning

The multi-objective Q-networks were implemented using the same hyperparameters as used in Yang et al. [2019] for all cases except the 'Multi-Objective SuperMario Game': we used four fully connected hidden layers with $[16, 32, 64, 32] \times (\dim(S) + m)$ hidden nodes respectively, where $\dim(S)$ is the dimensionality of the state space (Age and Risk) and m the number of objectives (costs and QALYs), both equal 2. This means that in our tests the number of hidden nodes were $[64, 128, 256, 128]$ in total respectively. The input of the network is a concatenation of state representation (i.e., the age and the risk of the individual) and (linear) preference weights. The output layer is of size $m \times |\mathcal{A}|$, which represents the multi-objective expected total reward for each action $a \in \mathcal{A}$ given the input state $s \in \mathcal{S}$ for each preference weights $\boldsymbol{w} \sim \mathcal{D}_{\boldsymbol{w}}$.

We started with multiple runs using the state-action space as described in Section 4.4. However, due to a mis-specification[2] of the complete preference space, the results were not useful and are excluded in this thesis. In addition, we learned that the RL algorithm slows down MISCAN significantly with a factor 700-7000. Since a minimum of 1,000,000 episodes is preferred such that the outcomes fully represent a population, this resulted in more than 10 days of training per case. Based on the first runs, we therefore decided to reduce the size of the state-action space. This increases the speed of the simulation in two ways. First in each episode less actions and updates are performed, second the convergence requirements denoted by Watkins and Dayan [1992] are met more quickly because a reduction of the number of state-action pairs implies that they are sampled more often.

The results shown in this thesis are based on this reduced state-space. The risk is discretized as $\mathcal{R} := \{0, 0.125, 0.250, 0.375, 0.5\}$. If an individual's risk is 0.5 or above, a colonoscopy is always performed. Furthermore, the screen-eligible ages are reduced to 55 to 75 years, i.e. $\mathcal{T} := \{55, 60, 65, 70\}$. In total the state-space was reduced from 81 to 16 unique states. We also stopped the surveillance at an age of 76. Since Van Duuren [2021] evaluated all feasible strategies for this state-action space by enumeration[3], we planned to validate our results to this benchmark. We are aware that the used costs, shown in Appendix B slightly differ from Van Duuren [2021], which means that our results will always deviate from his results.

[2]We initially used a preference space based on a willingness-to-pay for a QALY between 10,000 and 100,000 dollars. However, the weights should have been based on the costs of each extra QALY compared to a reference strategy.

[3]He only calculated the strategies following the *impact ordering assumption* (Assumption 5.2 in the respective thesis): Given two individuals with the same age, the one with a higher risk of CRC must get an action with equal or higher impact. A colonoscopy has the highest impact and an increasing FIT interval corresponds with a decreasing impact.

The used preference-space $\mathcal{D}_{\boldsymbol{w}}$ is also based on the results of Van Duuren [2021][4]. We selected the Pareto optimal strategies with the lowest and highest costs and calculated the extra costs per extra QALY gained for both strategies compared to the reference strategy, also called the Absolute Cost-Effectiveness Ratio (ACER). In this thesis, we used a universe without screening as a reference. This resulted in a minimum ($ACER$) of 57.830 and a maximum of 117.694. During training the $ACER$ is randomly sampled from a uniform distribution $ACER \sim U(57.830, 117.694)$. Next, the weights of the objectives ([QALY, Costs]) are calculated as $\left[\frac{ACER}{1+ACER}, \frac{1}{1+ACER}\right]$ respectively.

We started with four tests using different parameters, shown in Table 1. Due to extremely slow performance of the algorithm in combination with MISCAN and the need of many episodes for learning, we were unable to perform a decent test to find optimal parameter values[5]. Therefore, we based our parameter values on Yang et al. [2019] and deviate a little to look for possible improvements.

Although we expected to deviate from Van Duuren [2021], the results differed much more than expected. After evaluating the outcomes thoroughly, we found a difference in the simulation after performing a colonoscopy. Originally, the screening strategy is implemented statically in MISCAN, meaning that the strategy is completely defined before the simulation starts. Therefore, we had to adjust this structure to a dynamic strategy, i.e., after each action a new action will be determined depending on the state. However, after a colonoscopy the next step is still fixed: start with surveillance program after a positive colonoscopy or plan a new FIT over five years after a negative colonoscopy. It appeared that the latter was not set correctly, meaning that the next FIT is not planned after a negative colonoscopy. Due to the (new) dynamic nature no more future FITs are scheduled anymore. This can happen quite often depending on how often a colonoscopy is prescribed in the policy. Nevertheless, using $\epsilon = 0.5$ for the ϵ-greedy action selection, means that each time an action is selected, there is at least 25% change of selecting a colonoscopy, whereafter screening stops.

Table 1 summarizes the parameter values used in the four different tests. These parameters were described extensively in Section 4.3. Both the value of ϵ and the option to *decay*

[4]All feasible strategies for the used state-action space are shown in Figure 6.2 of Van Duuren [2021].

[5]We used an Intel(R) Core(TM) i5-10500T CPU (6 cores) @ 2.30GHz, and 8GB memory without using a GPU, while the authors of envelope Q-learning had a GeForce GTX TITAN X GPU and 32GB memory and for the SuperMario experiments even a cluster with twenty 2080 RTX GPUs and 200GB memory.

Table 1: Used parameters for the four different runs

Parameter	Case 1	Case 2	Case 3	Case 4
γ	0.99	0.97	0.99	0.97
Memory size	1000	2000	4000	2000
N_τ (Batch size)	64	128	256	128
$N_{\boldsymbol{w}}$ (Weight numbers)	8	16	32	16
ϵ	0.5	0.5	0.5	0.5
ϵ decay	FALSE	FALSE	FALSE	FALSE
Update frequency	100	100	100	1000
Optimizer	Adam	Adam	Adam	Adam
Learning rate	1.00E-03	5.00E-04	1.00E-05	0.05
Homotopy	TRUE	TRUE	TRUE	TRUE
Episodes	1.00E+07	1.00E+06	1.00E+07	1.00E+07

(ϵ-greedy selection) are not adjusted, since the environment is highly stochastic and we need to keep exploring. Furthermore, we always used the *Adam* optimizer [Kingma and Ba, 2017] for optimizing the neural networks and we always applied the *homotopy optimization.* The number of episodes is adjusted in one case only. However, due to the excessive running time, all tests were stopped before reaching this number, meaning that *Episodes* is not the actual number of episodes. Consequently, this number is used for increasing λ only, as described in Section 4.3.

Since an annual discount factor of 3% is commonly used in cost-effective analyses of health care, we used a discount factor of 0.97 per update for γ. However, in the original paper [Yang et al., 2019], $\gamma = 0.99$ was used by default, and since many costs are included in the last update of an episode, we also tested the effect of using this value. Furthermore, we varied the memory and batch size and the number of sampled weights. Increasing these parameters slows down the algorithm significantly, but might result in better learning. Finally, we used different values for the *Learning rate* used for updating the Q-network, which might have much impact in the performance of our training. In the original paper a learning rate of 1e-3 was used by default, but this does not mean that this is suitable for our problem. Therefore, we tried four different values.

Case 1

The first case was stopped after 3,240,000 episodes. Every 10,000 episodes, we calculated the QALYs and costs of the strategy learnt up to then to monitor the algorithm's progress. Using MISCAN, we evaluated the strategy for a preference of 86.265 with a population of 1,000,000 individuals. Each point in Figure 6.3 represents one of these strategies. Its colour shows in what episode it was obtained. Since we are maximizing the QALYs and minimizing the costs, we expect the lightest dots to be in the left upper corner of the plot. However, this is not the case. The strategies in the left upper corner were found early in the learning process. The algorithm appears to end up in three different other areas, having higher costs but lower QALYs, meaning that they are dominated for every possible preference weights.

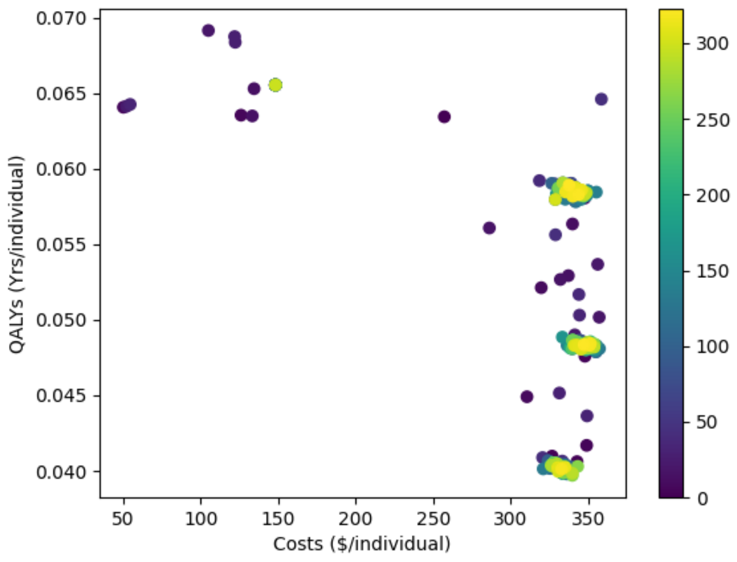
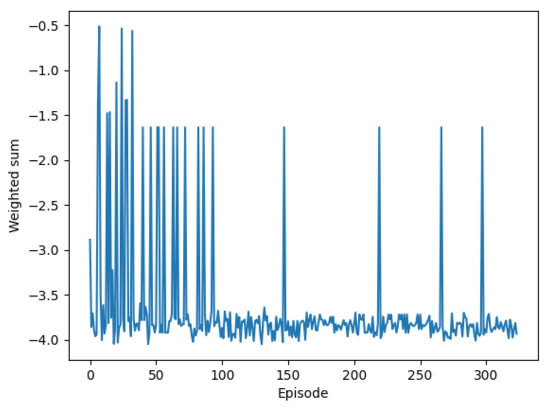


Figure 6.3: Scatter plot of the QALYs and costs of the strategies with preference weight of 86.265, obtained after each 10,000 episodes of training. The colour indicates at what point in the learning process the strategies were obtained, e.g., the colour at number 50 is the strategy after 50 × 10,000 episodes of training. Costs and QALYs were estimated using a MISCAN simulation with 1,000,000 individuals.

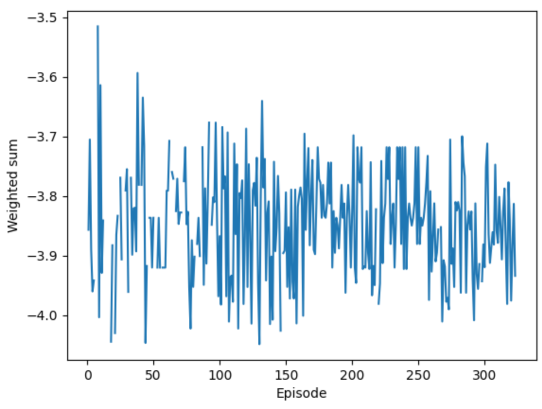
To take a closer look at the progress of the objectives, we plotted the weighted sum of the objectives each 10,000 episodes in Figure 6.4a. The weighted sum is calculated as

$$\frac{86.265}{87.265} \cdot \; QALYs + \frac{1}{87.265} \cdot \; -costs.$$

Due to some 'extreme' values, it is not clearly visible if the algorithm still improves before it was stopped. However, Figure 6.4b shows the observations without values larger than -3.4. Still no trend can be seen. Therefore, we conclude that using the parameters shown in Table 1 the algorithm is not able to converge to the optimal policies.



(a) All values



(b) The vertical axis is limited to -3.4

Figure 6.4: For each policy after 10,000 episodes of training, the weighted sum of the costs and QALYs are shown.

Case 2

In the second test, we decreased the total number of episodes, affecting the homotopy optimization as described in Section 4.3. Furthermore, we increased the memory and batch size, slowing down the method, but potentially speeding up the learning process. We also adjusted the learning rate of the neural network. The simulation is stopped after 910,000 episodes and we evaluate the results up to this point.

Figure 6.5 shows a scatter plot of the QALYs and costs of the optimal policy for a preference weight of 86.265, each 10,000 episodes of training. It is similar to the figure obtained in the first test: the costs and QALYs are comparable and the algorithm converges to non-optimal strategies. Despite the weighted sum of the objectives shows a slight increase over the iterations, as shown in Figure 6.6b, this increase is negligible compared to the peaks in Figure 6.6a.

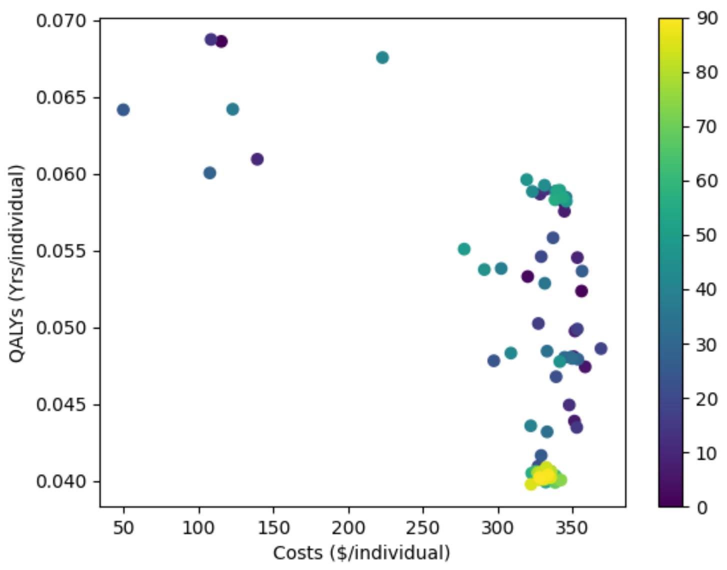
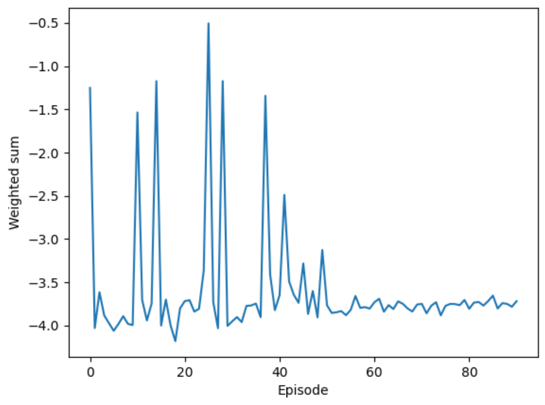
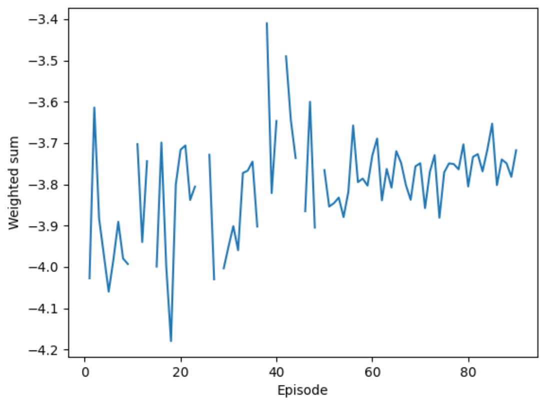


Figure 6.5: Scatter plot of the QALYs and costs of the strategies with preference weight of 86.265, obtained after each 10,000 episodes of training. The colour indicates at what point in the learning process the strategies were obtained, e.g., the colour at number 50 is the strategy after 50 × 10,000 episodes of training. Costs and QALYs were estimated using a MISCAN simulation with 1,000,000 individuals.



(a) All values



(b) The vertical axis is limited to -3.4

Figure 6.6: For each policy after 10,000 episodes of training, the weighted sum of the costs and QALYs are shown.

Case 3

The parameters of the third test are most similar to the parameters used in Yang et al. [2019], only the learning rate of the neural network was adjusted. This test had the worst computational performance. The scatter plot in Figure 6.7 shows that the model has not converged yet contrasting tests one and two. Figure 6.8 does not show an increasing trend, merely large, random variation.

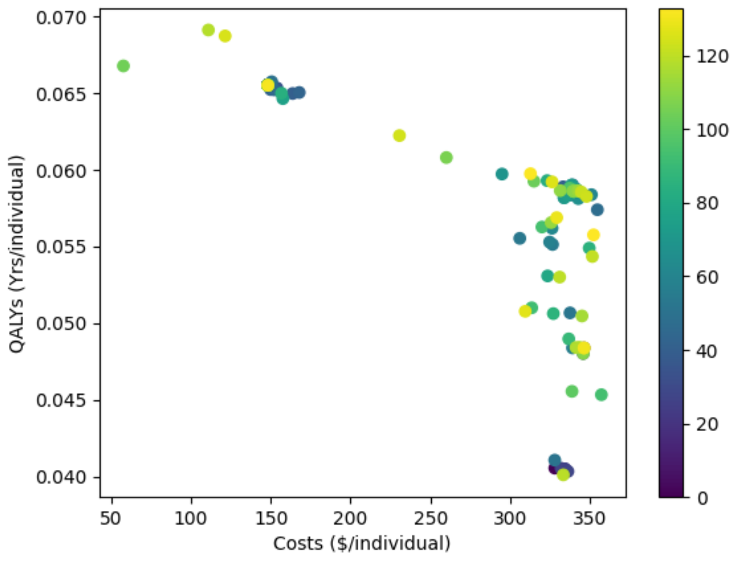
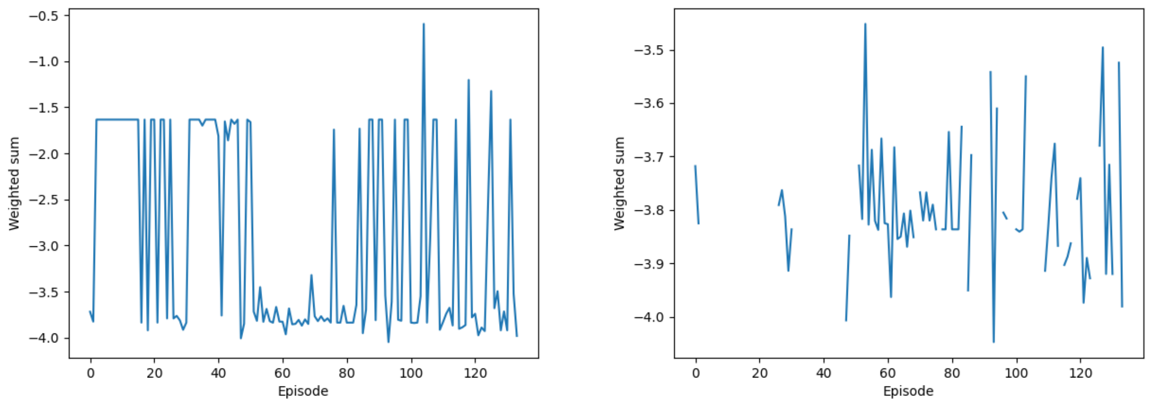


Figure 6.7: Scatter plot of the QALYs and costs of the strategies with preference weight of 86.265, obtained after each 10,000 episodes of training. The colour indicates at what point in the learning process the strategies were obtained, e.g., the colour at number 50 is the strategy after 50 × 10,000 episodes of training. Costs and QALYs were estimated using a MISCAN simulation with 1,000,000 individuals.



(a) All values *(b) The vertical axis is limited to -3.4*

Figure 6.8: For each policy after 10,000 episodes of training, the weighted sum of the costs and QALYs are shown.

Case 4

Case four uses a significantly larger *learning rate* and an increased *update frequency* for Double Q-network. Figure 6.9 shows less random variation compared to the other cases. Moreover, the algorithm converges to the same local optima as found in test one. Figure 6.10 does not show a clear increasing trend. Note that the peaks in this graph are lower compared to those cases one, two and three.

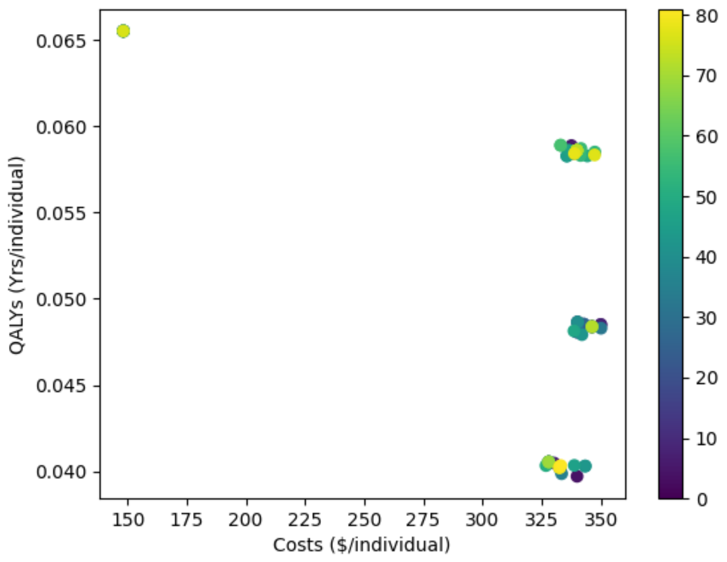
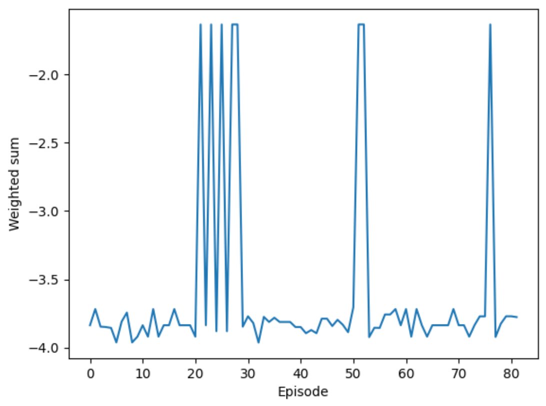
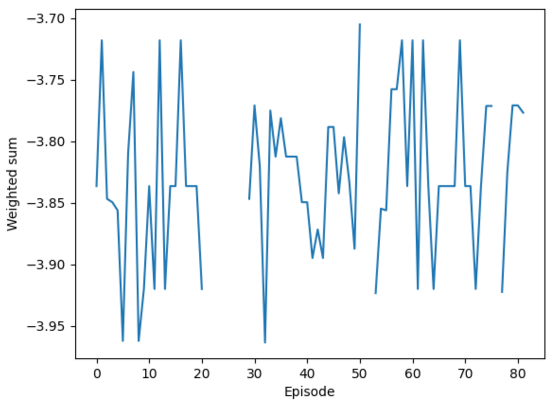


Figure 6.9: Scatter plot of the QALYs and costs of the strategies with preference weight of 86.265, obtained after each 10,000 episodes of training. The colour indicates at what point in the learning process the strategies were obtained, e.g., the colour at number 50 is the strategy after 50 × 10,000 episodes of training. Costs and QALYs were estimated using a MISCAN simulation with 1,000,000 individuals.



(a) All values



(b) The vertical axis is limited to -3.4

Figure 6.10: For each policy after 10,000 episodes of training, the weighted sum of the costs and QALYs are shown.

Case 5

None of these four test cases showed an increasing trend in objective throughout the learning process. Therefore we set up a fifth case, doubling the ACER, meaning that

$ACER \sim U(57.830, 235.388)$. We expected that this may result in another type of strategies. Furthermore, we added an extra hidden layer to the neural network. However, these adjustments did not result in different (more expensive) strategies. At that time, we were not aware that in our implementation of MISCAN, screening was stopped after a negative colonoscopy. This resulted in strategies with much lower costs than expected. Therefore, increasing the maximum preference weight had no effect. The results of case five are included in Appendix D.

6.3 Obtained screening strategies

In this section, we show the actual policies obtained by the first four cases. Figure 6.11(a)-(d) show the best strategy of each case based on a preference weight of 86.265. Figure 6.11(e) shows the strategy with highest QALYs out of all four cases. Figure 6.11(f)-(h) show one strategy from the three local optima obtained in cases one, two and four.

Strategies are used as follows. Directly after an individual is tested with a FIT, their risk score is calculated (see Section 5). Based on the individual's risk score (vertical axis) and age (horizontal axis), an action is prescribed. The actions FIT_x prescribe another FIT after x years, the action COL prescribes a colonoscopy. Note that, due to our choice of parameters, a colonoscopy is prescribed for risk values greater than 0.5.

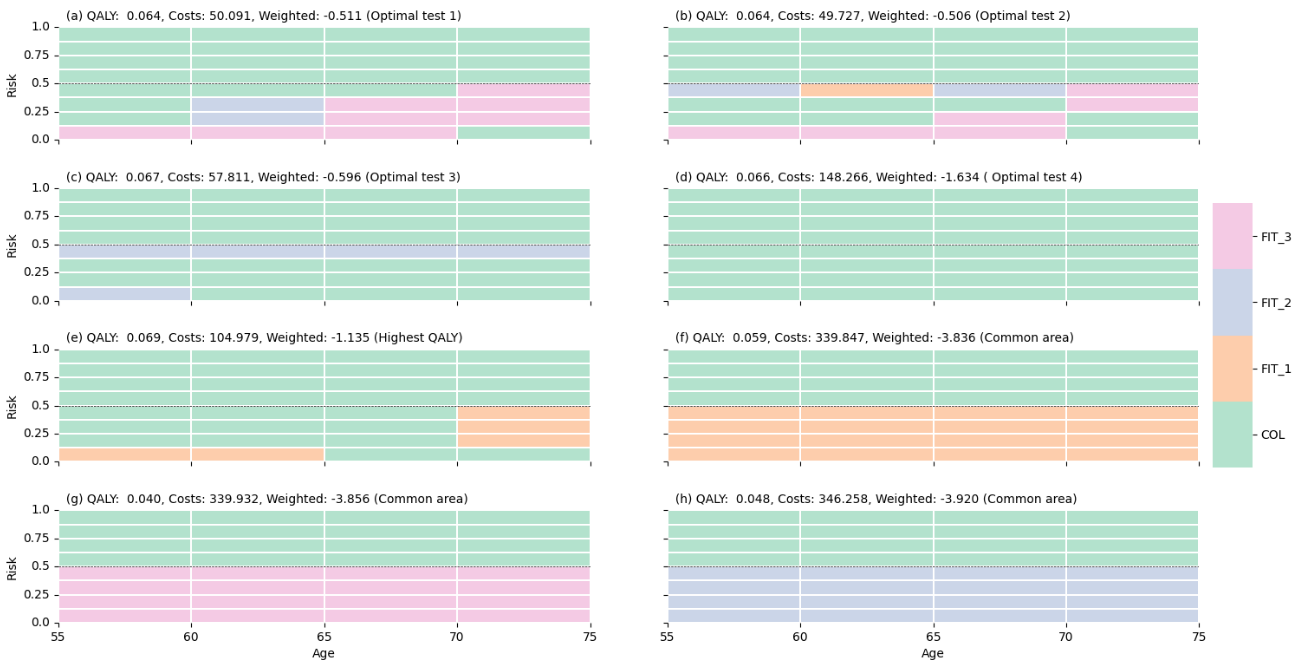


Figure 6.11: Eight different strategies obtained in cases one to four using preference weight 86.265. Directly after a FIT is performed, someone's risk value is calculated. Based on the person's risk value (vertical axis) and age (horizontal axis), an action is chosen. With actions FIT_x, the individual is invited for another FIT after x years. With action COL, the individual is directly referred to a hospital for a colonoscopy.

Starting with the strategies shown in figures (a) to (d), we see that a colonoscopy is prescribed very often. With strategy (d), all participants get a colonoscopy at the age of 50, and consequently quit screening or participate in surveillance. This strategy is much more expensive compared to strategy (c) (56.5%), while the QALYs do not increase much. It is unclear why this strategy maximizes the QALYs gained. It prescribes many colonoscopies, but at some points performs a FIT, resulting in a small increase of 0.003 QALYs (4.5%). The difference in costs is larger (29.2%). Comparing (c), (e) and (d) shows that performing a FIT instead of a colonoscopy clearly reduces costs, but does not necessarily change the QALYs. This difference of the costs can be explained by the fact that a colonoscopy is significantly more expensive compared to a FIT (see Appendix B). However, the FIT is less accurate meaning that an increase in QALYs is less reasonable.

Note that the action FIT_1 for the age of 70-75 at a risk between 0.125-0.5 in Figure (e) is irrelevant. The individuals that are still in the screening program at an age of 65 all get a colonoscopy, whereafter screening stops. Therefore, the difference between (d) and (e) is only the FIT_1 at the age of 55-65 for a risk between 0-0.125. Intuitively, these strategies prescribe a colonoscopy at the age of 65, but once an individual has a risk above 0.125, the colonoscopy is planned earlier in strategy (d). Various explanations of the difference in costs between (d) and (e) can be devised. It can be that more individuals had a positive colonoscopy, which caused a decrease in surveillance or treatment costs compared to what they would have been if they had stopped screening after a negative colonoscopy earlier. But it is also possible that the surveillance programme has higher costs than when it starts at a later age (without a significant difference in QALYs).

The last three strategies shown (f)-(h) are selected from the three non-optimal areas of case one, two and four, and often found by the RL algorithm. These strategies always prescribe the same screening interval, despite the FIT result. Note that these strategies were dominated by others. It is likely that these strategies often occur because of a large *learningrate* of the Q-network. The *learningrate* is the largest in case one and four, and the three areas are most evident. We assume that the rewards of a specific action have such a high effect on the Q-network, resulting in (almost) always the same action after updating.

Another interesting point in the strategies is the *order of impact* assumption introduced by Van Duuren [2021]. Normally (after learning), we expect that more invasive screening actions are prescribed for increasing risk. For example, the strategy shown in Figure 6.11(a) adheres to this assumption for the ages 60-64: a person with a risk of 0-0.124 gets a FIT_3 planned, while for a higher risk between 0.125-0.374 it becomes a FIT_2 and eventually for a risk above 0.375 a direct colonoscopy is selected. Van Duuren [2021] assumed that strategies are infeasible if they do not adhere to this assumption. However, since screening of a person stops after a negative colonoscopy, it is more difficult to determine the invasiveness of the colonoscopy. It is the most invasive action of the four, but it also can result in no future screening and no surveillance costs, which makes it in total much less invasive. Since we are not sure the order of impact assumption still holds, we adjust the strategies in Figure 6.11, where the order of impact assumption is violated and calculate its costs and effects. This applies to strategies (a), (b), (c) and (e). The strategies can be adjusted in two ways: (1) we fix the action at risk zero and adjust the actions for increasing risks to actions with equal or higher impact, or (2) fixing the action at a risk of 0.5 and adjusting the actions for lower risk values. These adjustments result in (1) an increase of colonoscopies or (2) a decrease in colonoscopies respectively. The costs and QALYs of the new strategies are shown in Table 2.

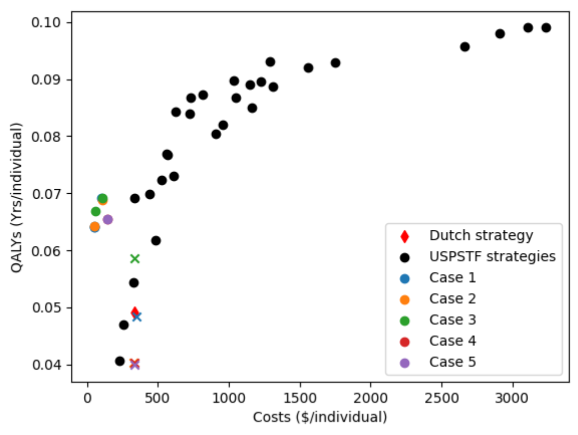
Table 2: Results of adjusting the strategies (a)-(c) and e of Figure 6.11 by ensuring the order of impact, where 'Original' is the learned strategy as shown in Figure 6.11. The results are estimated using a population of 10,000,000 individuals.

	Original			Order by impact increasing[a]			Order by impact decreasing[b]		
	QALYs	Costs	Weighted	QALYs	Costs	Weighted	QALYs	Costs	Weighted
a	0.05903	33.09	-0.3208	0.05929	30.00	**-0.2851**	0.05887	32.73	-0.3169
b	0.05921	28.24	**-0.2651**	0.05921	28.37	-0.2665	0.04085	319.50	-3.6209
c	0.06161	31.08	-0.2953	0.06161	31.08	-0.2953	0.04341	323.58	-3.6651
e	0.06397	93.56	**-1.0089**	0.06405	93.88	-1.0125	0.06398	97.61	-1.0553

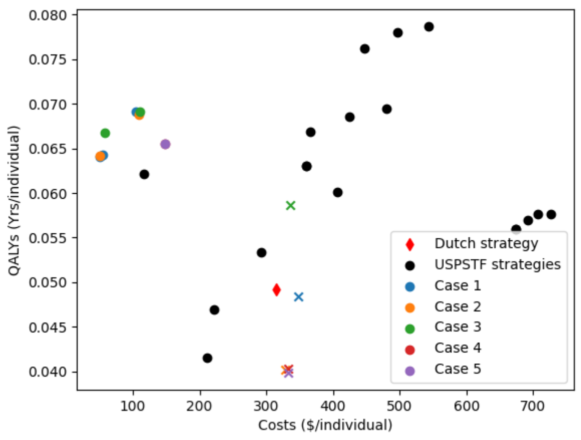
[a] Starting from a risk of 0.0, increasing the risk should result in an equal or more intense screening action
[b] Starting from a risk of 0.5, decreasing the risk should result in an equal or less intense screening action

As can be seen in the table, adjusting the strategies to ensure the order of impact does not ncecessarily result in a better strategy. In one case, strategy (a), it improved the strategy, but for all other strategies, this is not the case. This is in line with the idea that a colonoscopy in our research has a more complicated effect.

Although our settings differ from Van Duuren [2021] and the algorithms we used do not converge to an optimal point, we can still evaluate the performance of the strategies we found. We do this by comparing the strategies with the current Dutch screening strategy [Toes-Zoutendijk et al., 2017], and the strategies advised by Knudsen et al. [2021] to the United States Preventive Services Task Force (USPSTF) in October 2020. These strategies are population-based, meaning that the action only depends on age, and the FIT is always positive above 20μg/g. A scatter plot is shown in Figure 6.12a, where we only show the non-dominated and the last strategy found for each case (including the fifth case). First of all, we have discovered new strategies not dominated by either the Dutch or the USPSTF strategies. However, for all cases, we end up in a worse strategy with about the same costs as the Dutch strategy, but either more QALYs or less. It is remarkable that we only found strategies having cost less than \$400 per individual, while Knudsen et al. [2021] also advised strategies resulting in much more costs and also more QALYs. This likely is because, in our strategies, we stop screening after a colonoscopy. To indicate that this has much impact, we applied the same rule to the Dutch and USPSTF strategies and calculated again the costs and QALYs belonging to each strategy, which is shown in Figure 6.12b. It can be seen that indeed the costs are significantly reduced, and they still do not dominate our best-found strategies. However, some USPSTF strategies still have twice as many costs, and a few strategies also have more QALYs. If we look at the three strategies having QALYs above 0.075 per individual, we see that these consist of a yearly FIT varying the screen eligible age: all start at 45, while screening stops at 75, 80, or 85 years. In our tests we only screen between 55 and 75 years, meaning that our algorithm could not find these USPSTF strategies. This does not mean that using our settings, there are no more expensive strategies possible, and further experiments are needed to investigate this.



(a) Normal strategies



(b) No screening after negative colonoscopy

Figure 6.12: Scatter of the QALYs and Costs of the dominating strategies and the last strategy found in each case compared to the Dutch strategy and the strategies of the USPSTF. $\times$ denotes the last strategy found in that case.

On the other hand, we do have found strategies that are still non-dominated by these USPSTF strategies and they seem to be quite good. This implies that personalizing indeed improves the QALYs for given costs. Furthermore, the option to stop screening from a certain point also has a high impact, which could have caused that these strategies perform well. However, to make strong conclusions, also further investigation is needed.

7 Discussion

In this section, we discuss and compare the performance of the two algorithms introduced in this thesis, called MPQ-learning [Ruiz-Montiel et al., 2017] and envelope Q-learning [Yang et al., 2019].

First, both algorithms are not published with a code package, for example for python. In fact, the authors of MPQ-learning do not even provide example code. Consequently, a lot of time was spent on programming and debugging the algorithms, making these methods less recommended from a practical point of view. Besides, both algorithms are relatively new, so there is no literature that validates these models.

MPQ-learning did not get a result as we encountered two problems. The first issue, memory issues due to duplicate reward vectors being created, could be solved easily. The second issue, the exponential growth of reward vectors, was more complicated to tackle and partly causes the model to fail. It appears that the highly stochastic nature of our environment is the reason that this algorithm fails: the number of intermediate non-dominated reward vectors exceeds its limits. Since we cannot control the number of vectors, this results in a storage error. Our solution to overcome the second problem is not optimal, and the algorithm needs further investigation. However, one of the authors indicated that this model is likely bound to have limitations in practice as it may find an exponential or even infinite number of solutions. Hence, this algorithm is unsuitable for our problem.

Most challenging about implementing envelope Q-learning was to understand the code of the authors, to extract the relevant parts, and to implement those in MISCAN. The authors use the Python package PyTorch [Paszke et al., 2019] which activates the GPU, known to do fast calculations with neural networks.. However, PyTorch is based on CUDA [NVIDIA et al., 2020], meaning that the GPU can only be used with a decent, suitable NVIDIA graphics card, which was unavailable within our resources. Therefore, we could not use this advantage, making envelope Q-learning more than 500 times slower than MPQ-learning (depending on the used parameters). We do not know to what extent this would improve the computational performance. Also, once both algorithms had been trained, selecting the best action out of the multi-objective

Q-network instead of choosing an action out of a table increases the computing time about 35 times. Nevertheless, the algorithm did give a result in contrast to MPQ-learning.

Due to the slow performance of the algorithm in combination with the large number of parameters of the algorithm itself and the neural network, it was not possible to perform a decent parameter optimization. We tested the algorithm with five different parameter sets with execution times varying from a few days to more than two weeks. Each case resulted in comparable strategies. This indicates that the choice of parameters does not greatly influence the algorithm's performance. However, it might be worth testing different parameter values than those considered in this thesis. One important parameter to highlight is the preference space we used. Since we expected our strategies to be in the same QALY and costs range as those found by Van Duuren [2021], we based our preference space on their results. However, by accident, the structure of our strategies was different from those in Van Duuren [2021]. In the strategies found by our algorithm, participants quit screening after a negative colonoscopy. This difference resulted in a whole different range of qualifying preference weights. If we consider the two best incomparable strategies of the four cases in this thesis, i.e. the one with the highest QALYs and the one with the lowest costs, the range of the preference weight is between 58.531 and 59.500, whereas our algorithm searched in the range between 57.830 and 117.694. Even though the smaller range is included in the search range, it can be assumed that using a smaller range would have resulted in better learning.

Unfortunately, we must conclude that envelope Q-learning using our settings does not converge to an optimal strategy. However, we cannot claim that reinforcement learning is not suitable for solving our problem, and further research should be done. Since the natural history of colorectal cancer is highly stochastic between individuals, we look for the average best actions for the entire population while learning per individual. However, an individual's optimal strategy may not be optimal for the entire population. Furthermore, the most relevant rewards (costs) are at the end of an episode, which is also more challenging for learning. On the other hand, it should be able to learn the average rewards, when having a sufficient learning rate. The methods in this research are complex and multi-objective based. Besides, we used a neural network since the state-action space can be significantly more extensive than when using a Q-table. Due to our limited sources, we could not use a large state-action space, meaning that we had to deal with the burdens of deep learning without using the benefits.

One advantage of the MISCAN-Colon simulation model is that is is designed to work in parallel, but we could not use this advantage in combination with the RL algorithms. We assume that envelope Q-learning is an algorithm that can also be used in parallel, since it uses hindsight experience replay in combination with double Q-learning. However, the current architecture of the algorithm implemented in Python is not suitable yet for parallel computation, meaning that much more programming is needed. Therefore, we could not test this during this thesis, but this might be investigated in future research.

We suggest that further research takes a step back, starting to examine if reinforcement learning is possible in this environment. The multi-objective nature makes it much more complex, but scalarization is a suitable solution since it is common to represent a QALY by a monetary value in health. If there is proof that a simple RL algorithm can optimize a strategy (for a given preference), then if needed, a step further can be retaken to find an optimal Pareto front, taking into account the computational performance of RL.

7.1 Limitations

First of all, it is essential to note that the results strongly depend on the model of the simulated blood values. Our model uses the same preliminary, unvalidated model to simulate these blood values as used in Van Duuren [2021], while they stated that the results conflict with Grobbee et al. [2017] and emphasize the need for further research or calibration. The same applies to the risk estimators. A more accurate risk estimator may improve the policies significantly. Since the goal of this research was mainly focused on the learning algorithms and was aimed at comparing our algorithms with the genetic algorithm used in Van Duuren [2021], we used the same unvalidated models. However, we assume that an improved blood simulation model and risk estimator would also improve the learning process of our RL algorithms. Since then, the 'estimated' states have been more accurate, meaning that the rewards related to that state will vary less between different episodes. Eventually, we were not able to compare our results with Van Duuren [2021] since we unintentionally stopped screening after a negative colonoscopy. This highly influenced our results and resulted in unrealistic policies, and therefore we could not validate the performance of our algorithm well. However, some strategies were non-dominated compared to the USPSTF strategies and the Dutch strategy, which suggests that the option to stop screening at specific points might have promising results. This should certainly be further examined.

Since MPQ-learning failed in the simulation, we mainly focus on envelope Q-learning. One of the most significant limitations was the computational performance. Without carefully chosen parameter values, the algorithm is likely to perform non-optimal. However, using our computers, we were not able to decently optimize the parameter values. Furthermore, it is unknown how many episodes are needed to fully learn the environment. Normally, 1,000,000 to 10,0000,000 individuals are needed to correctly simulate a population. Therefore at least this number of episodes would be needed for the agent to learn, but it could also be that a multiplication is needed. Then, computation time becomes an even greater issue.

Another limitation was that we had to define the preference space before the simulation started. Since we were unaware that screening stopped after a negative colonoscopy, our preference space was almost three times larger than eventually the range of the preferences based on our results. A better specification could also have improved the performance.

8 Conclusion

In this thesis, two different multi-objective reinforcement learning algorithms are evaluated to find optimal personalized strategies for stool-based screening programs for colorectal cancer (CRC). We used the MISCAN-Colon simulation model [Loeve et al., 1999] to simulate CRC in a population and to evaluate the effects of selecting a screening interval based on individuals' age and result of their previous screening test. The first algorithm, MPQ-learning [Ruiz-Montiel et al., 2017], stores non-dominated vector sets for each state-action pair. However, it did not succeed in completing the learning process. We found two issues that the authors of the algorithm did not take into account. We gave a solution for both problems, but one of the solutions still does not entirely solve the problem. Finally, the algorithm does not control or limit the total (non-dominated) reward vectors, resulting in a storage error.

The second algorithm, envelope Q-learning, replaces the Q-table with a neural network. By varying the preference weights of the two objectives, the Q-network learns optimal strategies over the entire preference space. We performed five tests using different parameter values with execution times varying from a few days to more than two weeks, but none of the tests converged to an optimum. The obtained screening strategies could not be compared to those obtained by Van Duuren [2021] because we found a significant difference in the structure of our screening strategies, only after performing the tests. However, during training, we found some strategies, which are not dominated by the strategies evaluated by Knudsen et al. [2021] for the United States Preventive Services Task Force (USPSTF) in October 2020 and even dominate the current Dutch screening strategy [Toes-Zoutendijk et al., 2017].

After all, we did not find a complete Pareto front of optimal strategies using these two RL algorithms. Therefore, we cannot give a satisfying answer to our research questions. We proved that the current Dutch strategy can indeed be improved by personalizing the strategies, which is also the conclusion of Van Duuren [2021]. However, the algorithms did not converge to 'optimal' strategies. Therefore, we conclude that using our settings, the algorithms are not suitable to solve the problem. However, we still believe that RL is a powerful algorithm that should be able to learn optimal strategies. Further research is needed to strongly conclude the success or failure of reinforcement learning for this problem.

References

Axel Abels, Diederik Roijers, Tom Lenaerts, Ann Nowé, and Denis Steckelmacher. Dynamic weights in multi-objective deep reinforcement learning. In *International Conference on Machine Learning*, pages 11–20. PMLR, 2019.

Marcin Andrychowicz, Filip Wolski, Alex Ray, Jonas Schneider, Rachel Fong, Peter Welinder, Bob McGrew, Josh Tobin, Pieter Abbeel, and Wojciech Zaremba. Hindsight experience replay. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, pages 5055–5065, 2017.

Hermann Brenner and Simone Werner. Selecting a cut-off for colorectal cancer screening with a fecal immunochemical test. *Clinical and translational gastroenterology*, 8(8):e111, 2017.

Hermann Brenner, Michael Hoffmeister, Christa Stegmaier, Gerhard Brenner, Lutz Altenhofen, and Ulrike Haug. Risk of progression of advanced adenomas to colorectal cancer by age and sex: estimates based on 840 149 screening colonoscopies. *Gut*, 56(11):1585–1589, 2007.

Lukejohn W Day, Taft Bhuket, and James Allison. Fit testing: an overview. *Current gastroenterology reports*, 15(11):357, 2013.

Jayne Digby, Callum G Fraser, Francis A Carey, Paula J McDonald, Judith A Strachan, Robert H Diament, Margaret Balsitis, and Robert JC Steele. Faecal haemoglobin concentration is related to severity of colorectal neoplasia. *Journal of clinical pathology*, 66(5): 415–419, 2013.

N. Dunnewind. Finding cost-effective colorectal cancer screening strategies using multi-objective evolutionary algorithms and the miscan-colon microsimulation model. Master's thesis, February 2020. URL `http://hdl.handle.net/2105/51692`.

Andrea Gini. Microsimulation models to inform colorectal cancer screening decisions: From validated tools to tailoring recommendations. *PhD thesis*, 2020.

Esmée J Grobbee, Eline H Schreuders, Bettina E Hansen, Marco J Bruno, Iris Lansdorp-Vogelaar, Manon CW Spaander, and Ernst J Kuipers. Association between concentrations of hemoglobin determined by fecal immunochemical tests and long-term development of advanced colorectal neoplasia. *Gastroenterology*, 153(5):1251–1259, 2017.

Ying Guo, Astrid Zeman, and Rongxin Li. A reinforcement learning approach to setting multi-objective goals for energy demand management. *International Journal of Agent Technologies and Systems (IJATS)*, 1(2):55–70, 2009.

JDF Habbema, GJ Van Oortmarssen, J Th N Lubbe, and PJ Van der Maas. The miscan simulation program for the evaluation of screening for disease. *Computer methods and programs in biomedicine*, 20(1):79–93, 1985.

Md Mahmudul Hasan, Khin Lwin, Maryam Imani, Antesar Shabut, Luiz Fernando Bittencourt, and M Alamgir Hossain. Dynamic multi-objective optimisation using deep reinforcement learning: benchmark, algorithm and an application to identify vulnerable zones based on water quality. *Engineering Applications of Artificial Intelligence*, 86:107–135, 2019.

Paul Hewitson, Paul Glasziou, Eila Watson, Bernie Towler, and Les Irwig. Cochrane systematic review of colorectal cancer screening using the fecal occult blood test (hemoccult): an update. *American Journal of Gastroenterology*, 103(6):1541–1549, 2008.

Øyvind Holme, Michael Bretthauer, Atle Fretheim, Jan Odgaard-Jensen, and Geir Hoff. Flexible sigmoidoscopy versus faecal occult blood testing for colorectal cancer screening in asymptomatic individuals. *Cochrane Database of Systematic Reviews*, (9), 2013.

Ammar Jalalimanesh, Hamidreza Shahabi Haghighi, Abbas Ahmadi, Hossein Hejazian, and Madjid Soltani. Multi-objective optimization of radiotherapy: distributed q-learning and agent-based simulation. *Journal of ExpErimEntal & thEorEtical artificial intElligEncE*, 29 (5):1071–1086, 2017.

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization, 2017.

Amy B. Knudsen, Carolyn M. Rutter, Elisabeth F. P. Peterse, Anna P. Lietz, Claudia L. Seguin, Reinier G. S. Meester, Leslie A. Perdue, Jennifer S. Lin, Rebecca L. Siegel, V. Paul Doria-Rose, Eric J. Feuer, Ann G. Zauber, Karen M. Kuntz, and Iris Lansdorp-Vogelaar. *Colorectal Cancer Screening: An Updated Decision Analysis for the U.S. Preventive Services Task Force*. U.S. Preventive Services Task Force Evidence Syntheses, formerly Systematic Evidence Reviews. 2021. URL `http://europepmc.org/books/NBK570833`.

Arthur I Kooyker, Esther Toes-Zoutendijk, Annemieke WJ Opstal-van Winden, Manon CW Spaander, Maaike Buskermolen, Hanneke J van Vuuren, Ernst J Kuipers, Folkert J van Kemenade, Chris Ramakers, Maarten GJ Thomeer, et al. The second round of the dutch colorectal cancer screening program: Impact of an increased fecal immunochemical test cut-off level on yield of screening. *International journal of cancer*, 147(4):1098–1106, 2020.

Eric B Laber, Daniel J Lizotte, and Bradley Ferguson. Set-valued dynamic treatment regimes for competing outcomes. *Biometrics*, 70(1):53–61, 2014.

Iris Lansdorp-Vogelaar, Amy B Knudsen, and Hermann Brenner. Cost-effectiveness of colorectal cancer screening. *Epidemiologic reviews*, 33(1):88–100, 2011.

A Leslie, F A Carey, N R Pratt, and R J C Steele. The colorectal adenoma–carcinoma sequence. *British Journal of Surgery*, 89(7):845–860, 11 2002. ISSN 0007-1323. doi: 10.1046/j.1365-2168.2002.02120.x. URL `https://doi.org/10.1046/j.1365-2168.2002.02120.x`.

Bernard Levin, David A Lieberman, Beth McFarland, Kimberly S Andrews, Durado Brooks, John Bond, Chiranjeev Dash, Francis M Giardiello, Seth Glick, David Johnson, et al. Screening and surveillance for the early detection of colorectal cancer and adenomatous polyps, 2008: a joint guideline from the american cancer society, the us multi-society task force on colorectal cancer, and the american college of radiology. *Gastroenterology*, 134(5):1570–1595, 2008.

Zhuo Liu, Chenhui Yao, Hang Yu, and Taihua Wu. Deep reinforcement learning with its application for lung cancer detection in medical internet of things. *Future Generation Computer Systems*, 97:1–9, 2019.

Zhuo Liu, Gerui Zhang, Zhao Jingyuan, Liyan Yu, Junxiu Sheng, Na Zhang, and Hong Yuan. Second-generation sequencing with deep reinforcement learning for lung infection detection. *Journal of healthcare engineering*, 2020, 2020.

Daniel J Lizotte, Michael H Bowling, and Susan A Murphy. Efficient reinforcement learning with multiple reward functions for randomized controlled trial analysis. In *ICML*, 2010.

Franka Loeve, Rob Boer, Gerrit J van Oortmarssen, Marjolein van Ballegooijen, and J Dik F

Habbema. The miscan-colon simulation model for the evaluation of colorectal cancer screening. *Computers and Biomedical Research*, 32(1):13–33, 1999.

MingYu Lu, Zachary Shahn, Daby Sow, Finale Doshi-Velez, and Li-wei H Lehman. Is deep reinforcement learning ready for practical applications in healthcare? a sensitivity analysis of duel-ddqn for sepsis treatment. *arXiv preprint arXiv:2005.04301*, 2020.

Lawrence Mandow and José-Luis Pérez-de-la Cruz. Pruning dominated policies in multiobjective pareto q-learning. In *Conference of the Spanish Association for Artificial Intelligence*, pages 240–250. Springer, 2018.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. Human-level control through deep reinforcement learning. *nature*, 518(7540):529–533, 2015.

Hossam Mossalam, Yannis M Assael, Diederik M Roijers, and Shimon Whiteson. Multi-objective deep reinforcement learning. *arXiv preprint arXiv:1610.02707*, 2016.

NVIDIA, Péter Vingelmann, and Frank H.P. Fitzek. Cuda, release: 10.2.89, 2020. URL `https://developer.nvidia.com/cuda-toolkit`.

Han-Ching Ou, Haipeng Chen, Shahin Jabbari, and Milind Tambe. Active screening for recurrent diseases: A reinforcement learning approach. *arXiv preprint arXiv:2101.02766*, 2021.

Simone Parisi, Matteo Pirotta, Nicola Smacchia, Luca Bascetta, and Marcello Restelli. Policy gradient approaches for multi-objective sequential decision making. In *2014 International Joint Conference on Neural Networks (IJCNN)*, pages 2323–2330. IEEE, 2014.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc., 2019. URL `http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf`.

Julien Perez, Cécile Germain-Renaud, Balázs Kégl, and Charles Loomis. Responsive elastic computing. In *Proceedings of the 6th international conference industry session on Grids meets autonomic computing*, pages 55–64, 2009.

Elisabeth FP Peterse, Reinier GS Meester, Lucie de Jonge, Amir-Houshang Omidvari, Fernando Alarid-Escudero, Amy B Knudsen, Ann G Zauber, and Iris Lansdorp-Vogelaar. Comparing the cost-effectiveness of innovative colorectal cancer screening tests. *JNCI: Journal of the National Cancer Institute*, 113(2):154–161, 2021.

Michael Pignone, Melissa Rich, Steven M Teutsch, Alfred O Berg, and Kathleen N Lohr. Screening for colorectal cancer in adults at average risk: a summary of the evidence for the us preventive services task force. *Annals of internal medicine*, 137(2):132–141, 2002.

Mathieu Reymond and Ann Nowé. Pareto-dqn: Approximating the pareto front in complex multi-objective decision problems. In *Proceedings of the adaptive and learning agents workshop (ALA-19) at AAMAS*, 2019.

Douglas J Robertson, Jeffrey K Lee, C Richard Boland, Jason A Dominitz, Francis M Giardiello, David A Johnson, Tonya Kaltenbach, David Lieberman, Theodore R Levin, and Douglas K Rex. Recommendations on fecal immunochemical testing to screen for colorectal neoplasia: a consensus statement by the us multi-society task force on colorectal cancer. *Gastroenterology*, 152(5):1217–1237, 2017.

Diederik M Roijers and Shimon Whiteson. Multi-objective decision making. *Synthesis Lectures on Artificial Intelligence and Machine Learning*, 11(1):1–129, 2017.

Manuela Ruiz-Montiel, Lawrence Mandow, and José-Luis Pérez-de-la Cruz. A temporal difference method for multi-objective reinforcement learning. *Neurocomputing*, 263:15–25, 2017.

Eline H Schreuders, Arlinda Ruco, Linda Rabeneck, Robert E Schoen, Joseph JY Sung, Graeme P Young, and Ernst J Kuipers. Colorectal cancer screening: a global overview of existing programmes. *Gut*, 64(10):1637–1649, 2015.

Mohit Sewak. *Deep reinforcement learning.* Springer, 2019.

Nazanin Shabani. *Incorporating flood control rule curves of the Columbia River hydroelectric system in a multireservoir reinforcement learning optimization model.* PhD thesis, University of British Columbia, 2009.

Hyuna Sung, Jacques Ferlay, Rebecca L Siegel, Mathieu Laversanne, Isabelle Soerjomataram, Ahmedin Jemal, and Freddie Bray. Global cancer statistics 2020: Globocan estimates of incidence and mortality worldwide for 36 cancers in 185 countries. *CA: a cancer journal for clinicians*, 71(3):209–249, 2021.

Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction.* MIT press, 2018.

Esther Toes-Zoutendijk, Monique E van Leerdam, Evelien Dekker, Frank Van Hees, Corine Penning, Iris Nagtegaal, Miriam P van der Meulen, Anneke J van Vuuren, Ernst J Kuipers, Johannes MG Bonfrer, et al. Real-time monitoring of results during first year of dutch colorectal cancer screening program and optimization by altering fecal immunochemical test cut-off levels. *Gastroenterology*, 152(4):767–775, 2017.

Anirudh Tomer, Daan Nieboer, Monique J Roobol, Ewout W Steyerberg, and Dimitris Rizopoulos. Personalized schedules for surveillance of low-risk prostate cancer patients. *Biometrics*, 75(1):153–162, 2019.

Peter Vamplew, Richard Dazeley, Adam Berry, Rustam Issabekov, and Evan Dekker. Empirical evaluation methods for multiobjective reinforcement learning algorithms. *Machine learning*, 84(1):51–80, 2011.

Sascha C Van Doorn, Inge Stegeman, An K Stroobants, Marco W Mundt, Thomas R De Wijkerslooth, Paul Fockens, Ernst J Kuipers, Patrick M Bossuyt, and Evelien Dekker. Fecal immunochemical testing results and characteristics of colonic lesions. *Endoscopy*, 47(11): 1011–1017, 2015.

L.A. Van Duuren. A genetic algorithm to personalise stool-based colorectal cancer screening. Master's thesis, February 2021. URL `http://hdl.handle.net/2105/55679`.

Kristof Van Moffaert and Ann Nowé. Multi-objective reinforcement learning using sets of pareto dominating policies. *The Journal of Machine Learning Research*, 15(1):3483–3512, 2014.

Christopher JCH Watkins and Peter Dayan. Q-learning. *Machine learning*, 8(3-4):279–292, 1992.

Christopher John Cornish Hellaby Watkins. Learning from delayed rewards. 1989.

Layne T Watson and Raphael T Haftka. Modern homotopy methods in optimization. *Computer Methods in Applied Mechanics and Engineering*, 74(3):289–305, 1989.

Runzhe Yang, Xingyuan Sun, and Karthik Narasimhan. A generalized algorithm for multi-objective reinforcement learning and policy adaptation. *arXiv preprint arXiv:1908.08342*, 2019.

Yufan Zhao, Michael R Kosorok, and Donglin Zeng. Reinforcement learning design for cancer clinical trials. *Statistics in medicine*, 28(26):3294–3315, 2009.

Appendices

A Probability distribution of haemoglobin concentrations

(Adapted from Van Duuren [2021])

As described in Section 2, the FIT test returns the haemoglobin concentration in a patient's stool. To incorporate this in our simulation, we developed a probability distribution for the Hb concentrations. The Hb concentration depends on the a patient's age, gender, and individual risk factor and the cancer stage in $\mathcal{X}^u$ the patient is in. The Hb concentrations were modelled by a mixed-effects zero-inflated negative binomial model (ZINB), expressed by:

$$\mathbb{P}(S_n|\boldsymbol{x}_n) \sim \begin{cases} NB(S_n|\boldsymbol{x}_n), & \text{if } u \geq \varphi(x\_STAGE) \\ 0, & \text{otherwise.} \end{cases} \tag{A.1}$$

Here S_n is the Hb value obtained at the n^{th} test and $\boldsymbol{x}_n = [x\_\text{AGE}, x\_\text{GENDER}, x\_\text{STAGE}]$ is the vector that contains the age, gender and cancer stage of the patient respectively at the moment of the test. The model is called zero-inflated because it contains a lot of zeroes compared to a regular negative binomial distribution. The number of zeroes is determined by the value $\varphi(x) \in [0,1]$ which depends on the patient's current state $x \in \mathcal{X}^u$. A standard uniform random variable u is drawn at every FIT to determine whether the new Hb value is a zero or is to be drawn from the NB distribution.

The pdf of this NB distribution is specified as follows:

$$f(S_n|\boldsymbol{x}_n) = \frac{\Gamma(S_n+\theta)}{S_n!\Gamma(\theta)} * \left(\frac{\theta}{\theta+\mu_n}\right)^{\theta} * \left(\frac{\mu_n}{\theta+\mu_n}\right)^{S_n} \tag{A.2}$$

with θ the dispersion factor, constant throughout the algorithm, and $\Gamma(.)$ the gamma function. The mean μ_n is defined by

$$\mu_n = \mu(\boldsymbol{x}_n) = exp\{b_{\text{INTERCEPT}} + \gamma + b_{\text{AGE}} * x\_\text{AGE} + b_{\text{MALE}} * I(x\_\text{GENDER} = \text{MALE}) + \\ b_A * I(x\_\text{STAGE} = \text{A}) + b_{PC} * I(x\_\text{STAGE} = \text{PC})\}. \tag{A.3}$$

Here b_i stands for the weights of each of the properties in $\boldsymbol{x}_n$ and the intercept, $\mathbf{I}(.)$ represents the indicator function which equals one if the statement between brackets holds true and zero otherwise. A and PC stand for the adenoma and preclinical stages respectively. γ is an individual risk factor, a random value that is drawn for all patients individually and remains constant during their lives. It is responsible for the correlation between the Hb concentrations obtained in the same patients. Its distribution is $\gamma \sim \mathcal{N}(0, \sigma)$.

Table 3 gives an overview of the calibrated values. Figure A.1 displays histograms of the theoretical Hb concentration distribution for each of the three stages for a fixed age and gender. Note that the distribution is highly zero-inflated, especially for the healthy and adenoma stages. Therefore, the histograms are also shown for larger Hb values. In turn, the distribution for the preclinical stage is highly skewed: the three theoretical distributions generate maximum Hb values of 632, 1406 and 4968590 respectively.

Table 3: Overview of the variables in the model for Hb concentrations and their values.

Symbol	Description	Value
$\varphi(H)$	Probability of inflated zero for healthy stage	0.840
$\varphi(A)$	Probability of inflated zero for adenoma stage	0.644
$\varphi(PC)$	Probability of inflated zero for preclinical stage	0.032
$b_{\text{INTERCEPT}}$	Intercept	1.481
b_{AGE}	Weight of age factor	0.0181
b_{MALE}	Weight of gender factor	0.2832
b_{AGE}	Weight of adenoma stage	0.609
b_{PC}	Weight of preclinical stage	8.74
θ	Dispersion factor	0.262
σ	Standard deviation of individual risk factor distribution	0.1120

Figure A.2 shows the Hb value distribution for a given cancer stage as observed in the Dutch screening programme. We observe that the theoretical distribution needs further calibration, especially the preclinical cancer stage, since it is not similar to the observed distribution. This is also confirmed by the maxima of the distributions mentioned above as the observed Hb values merely exceed 300μg/g.

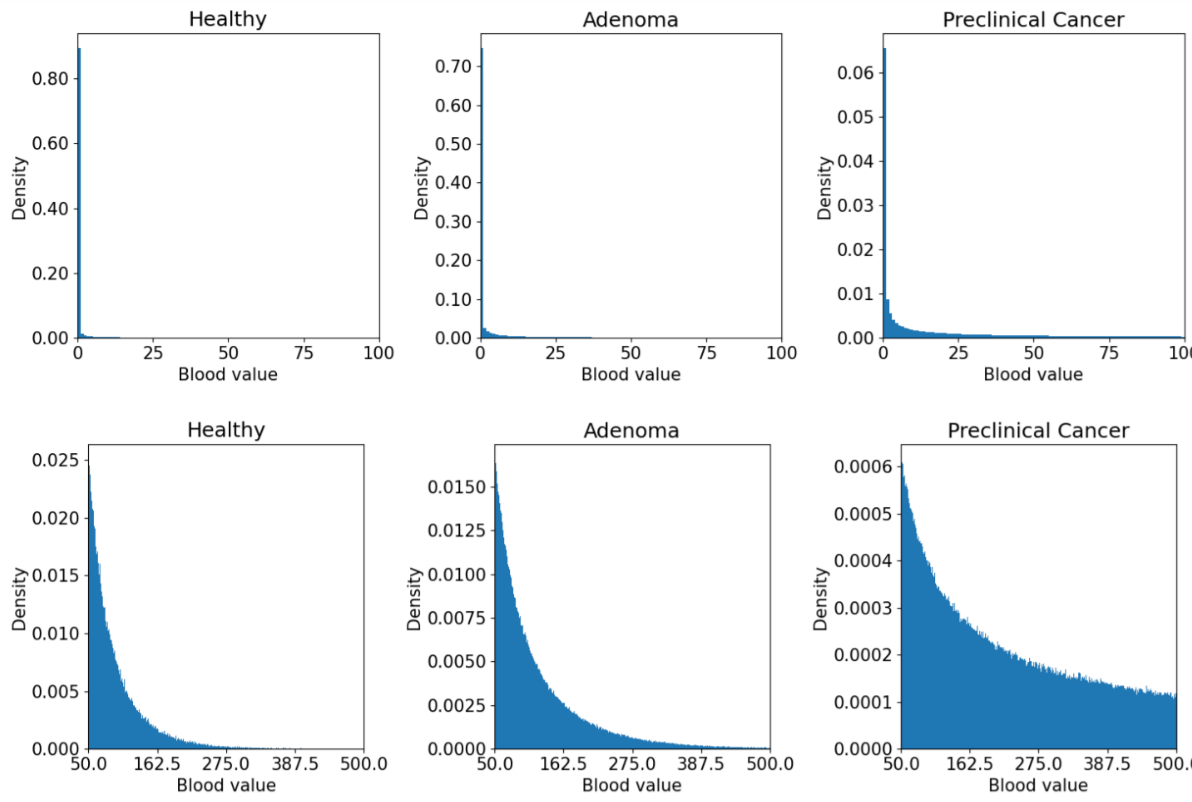


Figure A.1: Histograms of the simulated ZINB-distributions for each of the three cancer stages with the given parameters. The upper plot displays the Hb concentrations between 0 and 100, the bottom plot shows them between 50 and 500. The age is set constant at 55 and gender is fixed at male. Each of the histograms was generated with 1e7 individuals.

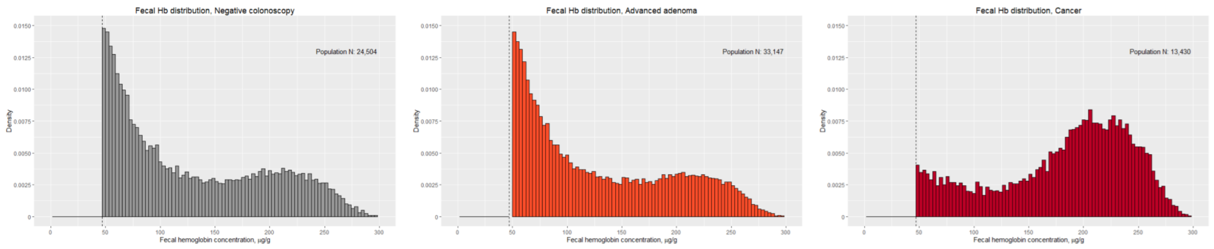


Figure A.2: Histograms of the observed Hb concentration given the result of a consequent colonoscopy in the Dutch screening programme. From left to right, the states healthy, adenoma and preclinical cancer are displayed. The horizontal axis represents the observed Hb value, ranging from 50 to 300μg/g. The vertical axes show the density, ranging from 0 to 0.0150. Only Hb values above 50μg/g are shown: we only observe a patient's cancer stage with a colonoscopy after the patient tested positive, i.e. obtained a concentration above 47μg/g.

B Rewards used in RL algorithm

Our RL algorithm is based on maximizing the reward, where the *Quality Adjusted Life Years (QALYs)* gained are maximized, and the *Costs* are minimized. The screening tests that are used have fixed *QALYs* and *Costs*, which are extracted from Peterse et al. [2021] and can be found in Table 4.

Table 4: Used (negative) rewards per screening test

Test	QALYs	Costs
Negative FIT	-0.000063	$ 40.00
Positive FIT	-0.00133	$ 40.00
Negative Colonoscopy	-0.000496	$ 1,279.00
Positive Colonoscopy	-0.001401	$ 1,656.00

When CRC is detected, we get treatment or surveillance costs, which are also incorporated in the rewards. These costs depend on the state of the person's disease (see Figure 2.1) and the time a person is in these states. The used costs can be found in Table 5, which are also extracted from Peterse et al. [2021].

Table 5: Used (negative) rewards per year that a person is in a specific state

	Initial care [a]		Continuing care [b]	
	QALYs	Costs	QALYs	Costs
Clinical/Screen-detectable 1	-0.12	$ 42,763.00	-0.05	$ 4,301.00
Clinical/Screen-detectable 2	-0.18	$ 58,796.00	-0.05	$ 4,944.00
Clinical/Screen-detectable 3	-0.24	$ 83,247.00	-0.24	$ 7,456.00
Clinical/Screen-detectable 4	-0.7	$ 121,828.00	-0.7	$ 34,017.00
	Terminal care [c]		Terminal care, other cause [d]	
	QALYs	Costs	QALYs	Costs
Clinical/Screen-detectable 1	-0.7	$ 82,519.00	-0.05	$ 25,450.00
Clinical/Screen-detectable 2	-0.7	$ 92,366.00	-0.05	$ 26,997.00
Clinical/Screen-detectable 3	-0.7	$ 96,461.00	-0.24	$ 35,026.00
Clinical/Screen-detectable 4	-0.7	$ 119,979.00	-0.7	$ 77,051.00

[a] The first year after CRC is found
[b] The years between the first year and last year of the person after CRC is found
[c] The last year before the person dies of CRC
[d] The last year before the person dies to an other cause

In training, it is not simply possible to calculate the QALYs gained of a postponed death due to the screening program since we do not know when the person would have died if he did not participate in the screening program. This is because cancer gets an entirely different simulation if screening is incorporated. However, we do know when the person would have died without

CRC since this was determined before the simulation. Therefore, we know if the person died earlier due to CRC, which we see as a loss in QALYs. Assuming that screening would reduce this loss, we take this as a negative reward, which then can be maximized.

C Conversation storage problem MPQ-learning

"Dear Prof. Mandow,

Currently, I am writing my master thesis and I have read your paper "A temporal difference method for multi-objective reinforcement learning". I believe MPQ-learning is a really interesting method and I consider using it in my thesis. However, I initially considered Pareto Q-learning of Van Moffaert and Nowé, 2014.

In your research it is claimed that MPQ-learning need less storage space:

"*However, in [18] memory requirements are likely to be larger, due to: (i) the need to keep track of the expected rewards and transition probabilities of all transitions visited during the learning phase, and (ii) the need to store sets of optimal values NDt(s, a, s) for all transitions from a state s to s through action a at time step t, rather than the sets of optimal values per state s calculated by our Q-learning approach, that are typically a subset of the optimal values of all transitions to s .* " - Ruiz-Montiel et al., 2017

However, there is no concrete comparison between the two methods and I wonder if you could explain to me why MPQ-learning is better. Is the difference mainly storage without loss in performance or is there a trade-off between these two? In both researches I cannot find anything about the size of state-spaces the methods are capable of solving, so it is hard to decide when storage actually becomes an issue.

Perhaps you have experienced more in practice and you could help me by deciding which method to choose. I am eager to receive your reply.

Sincerely,

Michael van der Zwan

(Erasmus University Rotterdam)"

"Dear Michael,

Thank you for reading our paper. Any comments and/or suggestions will be welcome.

Let me try to answer your question from two perspectives: theoretical and practical.

From a theoretical point of view, MPQ and Pareto Q (PQ) are actually different kinds of algorithms. In my humble opinion, the best way to understand PQ is as a multiobjective extension of Value Iteration (rather than Q-learning). I am not sure I recall all the specific details of PQ right now, but I think the paragraph you mention refers to the fact that PQ needs to store more heavy data sets (the ND sets) than MPQ for each visited transition.

About 40 years ago, White analyzed the problem of multi-objective dynamic programming and proposed an exact algorithm. However, I am not aware of any practical applications of that algorithm. Why? Probably because even very simple MORL problems can have an exponential (or even infinite) number of solutions, which makes it impractical to apply exact algorithms. Both PQ and MPQ address this same problem (in the case where transition probabilities are unknown), and so are likely bound to have similar limitations in practice. If you want to figure out how complex it is to approximate the full Pareto set even in simple MORL instances, you may want to take a look at,

`https://arxiv.org/abs/2009.08198`

Therefore, my experience and advice is that, unless your problem instance is really really simple and really really limited, your best chances of success are probably to resort to either,

- A scalarized approach, i.e. reducing the problem to the optimization of some scalar multi-attribute utility function.

- Some kind of approximation (as opposed to exact) algorithm, in case you really need to approximate the problem's Pareto front.

I hope you will find these comments useful. Please, let me know if I can be of further help.

Best,

Lawrence

L. Mandow

Dpto. Lenguajes y Ciencias de la Computación

Universidad de Málaga (Spain)"

D Results case 5

For the fifth case we added an extra hidden layer, resulting in a Q-network of five hidden layers with $\{64, 128, 256, 128, 128\}$ nodes. Furthermore, the used parameter values are shown in Table 6. The simulation is stopped at 80%, which was after 1,510,000 episodes (excluding individuals who died before the screening eligible age). The weighted sum in Figure D.2 is calculated as $\frac{86.265}{87.265} \cdot QALYs + \frac{1}{87.265} \cdot \; - costs$.

Table 6: Used parameters for the fifth run

Parameter	Case 5
γ	0.97
Memory size	3000
N_τ (Batch size)	192
$N_{\boldsymbol{w}}$ (Weight numbers)	24
ϵ	0.5
ϵ decay	FALSE
Update frequency	500
Optimizer	Adam
Learning rate	0.05
Homotopy	TRUE
Episodes	2.00E+06

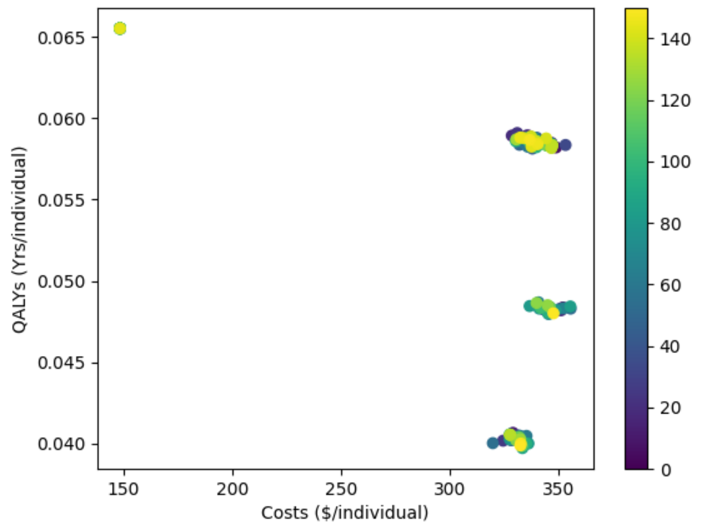
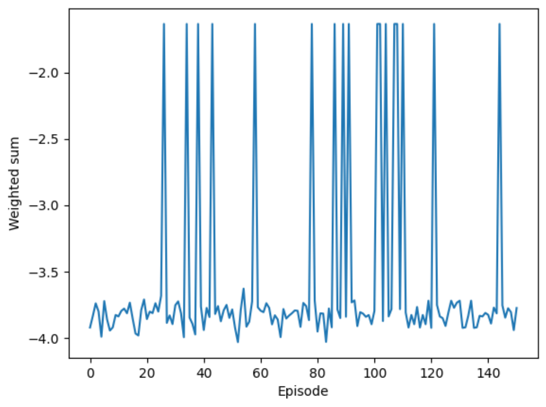
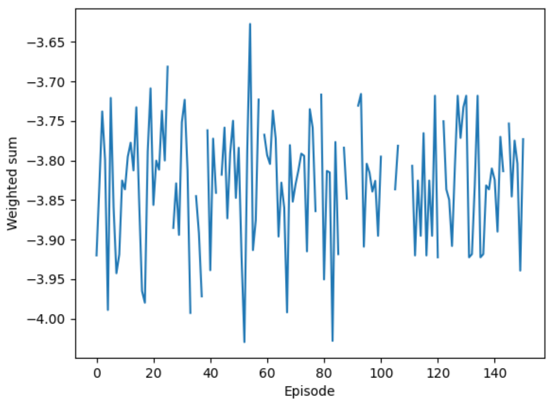


Figure D.1: Scatter plot of the QALYs and costs of the strategies with preference weight of 86.265, obtained after each 10,000 episodes of training. The colour indicates at what point in the learning process the strategies were obtained, e.g., the colour at number 50 is the strategy after 50 × 10,000 episodes of training. Costs and QALYs were estimated using a MISCAN simulation with 1,000,000 individuals.



(a) All values



(b) The vertical axis is limited to -3.4

Figure D.2: For each policy after 10,000 episodes of training, the weighted sum of the costs and QALYs are shown.