

Vegard Hansen and Trond Håkon Trondsen

# An application of approximate dynamic programming/ reinforcement learning to salmon production scheduling

Master's thesis in Industrial economics and technology management

Supervisor: Stein-Erik Fleten

Co-supervisor: Andreas Kleiven

June 2021



Vegard Hansen and Trond Håkon Trondsen

# **An application of approximate dynamic programming/reinforcement learning to salmon production scheduling**

Master's thesis in Industrial economics and technology management  
Supervisor: Stein-Erik Fleten  
Co-supervisor: Andreas Kleiven  
June 2021

Norwegian University of Science and Technology  
Faculty of Economics and Management  
Dept. of Industrial Economics and Technology Management







Kunnskap for en bedre verden

INDUSTRIAL ECONOMICS AND TECHNOLOGY  
MANAGEMENT

TIØ4905/4900 - MASTER THESIS, MANAGERIAL ECONOMICS  
AND OPERATIONS RESEARCH/FINANCIAL ENGINEERING

---

**An application of approximate dynamic  
programming/reinforcement learning to  
salmon production scheduling**

---

*Authors:*

Trond Håkon Trondsen and Vegard Hansen

June, 2021

# Preface

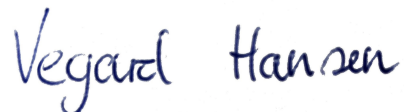
This master thesis is the conclusion of our Master of Science in Industrial Economics and Technology Management at the Norwegian University of Science and Technology. The thesis is written in the intersection between the areas of Managerial Economics and Operations Research and Financial Engineering. The thesis is motivated by our mutual interest in salmon production and by the growing number of successful applications of the interface between advanced optimization and machine learning. The thesis is also motivated by the operations of innovative salmon producers such as Andfjord Salmon.

We have received much appreciated help and guidance in completing this thesis, both from the faculties at NTNU and the industry. We would especially like to thank our supervisor, professor Stein-Erik Fleten and PhD Research fellow Andreas Kleiven for interesting discussions, guidance and valuable feedback and help. We would also like to thank professor Keith Downing for his hands-on feedback on our reinforcement learning model and professor Qing Li from the Hong Kong University of Science and Technology for interesting discussions. Finally we would like to thank Christian Torgersen and his colleagues at Andfjord Salmon for discussions and valuable input on relevant research topics, realistic production and parameter values.

Trondheim, June 24, 2021



Trond Håkon Trondsen



Vegard Hansen

# Summary

Global demand for Atlantic salmon as a healthy, resource-efficient and climate friendly protein source is only expected to rise. The compounded annual growth rate (CAGR) of salmon harvest volumes has been 5% in the period 2001-2020 (Mowi, 2021). At the same time the industry is facing environmental challenges and governmental restrictions on traditional salmon production in fjords and bays. Land based salmon production is a growing part of the industry and a response to the environmental challenges of traditional production, currently attracting large amounts of investor capital (Berge (2021)). Land based production has several advantages, including more controllable production and less biological pollution. However, producing salmon on land while also facing stricter requirements on e.g. feed quality is causing a rise in production costs (Bjørndal and Tusvik, 2019). Smart production scheduling is therefore becoming increasingly more important to the salmon producer, both land based and sea based. With the salmon price being a particularly volatile and varying commodity price, taking this uncertainty into account when scheduling production has been deemed highly interesting by players in the industry. However, the salmon production scheduling optimization problem remains a challenging and computationally demanding problem, with a need for more advanced optimization techniques. An inherent key issue in the problem is determining the true value of the standing biomass. Machine learning methods like reinforcement learning are on the rise, having shown impressive results to complex problems, and are inclined to deal with such issues exactly.

This master thesis represents the first attempt, to the authors' knowledge, to model the salmon production scheduling problem as a Markov Decision Process and to solve this model by approximate dynamic programming or reinforcement learning. Warren B. Powell and David Silver are two front figures in this field and their works are key sources of knowledge and inspiration for this thesis. The ADP/Deep RL model developed is based on a value function approximation policy. This is in turn based on n-step Temporal Difference (TD(n)) learning, testing both custom functions and deep neural networks as value function approximators and using a count-based, problem specific exploration strategy resembling the Upper Confidence Bound method. To be able to solve the stochastic problem version with respect to salmon price uncertainty, a novel semi-parametric structural price model based on forward curve data has been developed. This model generates random salmon price development samples.

The ADP/RL model has been benchmarked against a mixed integer programming model and a rolling horizon model for the deterministic and stochastic problem, respectively. The best schedules produced by the ADP/RL model in its current state earns  $\sim 80\%$  of the profits earned by the MIP model for small problem instances, which is not deemed good enough for use in real world production. However, ADP/RL is in many ways as much an art as a science, requiring great amounts of algorithm tuning and creative problem solving, and the authors are convinced that there is a large potential in applying ADP/RL to not only salmon production scheduling, but also other biological herd management problems. This thesis therefore merely opens the door to an entire field of research the authors believe will be fruitful. The thesis also investigates the value of stochastic solution. The results suggest that the value of stochastic solution is small compared to scheduling with respect to deterministic price seasonality, but large compared to scheduling with respect to a constant price.

# Sammendrag

Global etterspørsel etter laks som en sunn, ressurs-effektiv og klimavennlig proteinkilde er forventet å øke i årene som kommer. Sammensatt årlig vekstrate for slaktevolumer av laks har vært 5% i perioden 2001-2020 (Mowi (2021)). Samtidig møter tradisjonell oppdrett i fjorder og viker både miljømessige utfordringer og restriksjoner fra myndighetene. Landbasert oppdrett er en voksende del av industrien og en respons på de miljømessige utfordringene ved tradisjonell oppdrett som for tiden tiltrekker seg store mengder kapital (Berge (2021)). Landbasert oppdrett har flere fordeler, deriblant mer kontrollert produksjon og mindre biologisk forurensning. På den andre siden fører det å produsere laks på land samtidig som det stadig stilles strengere krav til blant annet forkvalitet til høyere produksjonskostnader (Bjørndal and Tusvik (2019)). Smart produksjonsplanlegging blir derfor stadig viktigere for en lakseoppdretter, både på land og i sjø. Videre er lakseprisen en spesielt volatil og varierende råvarepris, og det å ta dette i betraktning i produksjonsplanleggingen blir ansett som svært interessant av spillere i industrien. Planlegging av lakseproduksjon er et utfordrende og beregningsmessig krevende optimeringsproblem som krever mer avanserte optimeringsmetoder. Et viktig spørsmål i dette problemet er hvordan bestemme verdien av den stående biomassen i anlegget. Maskinlæringsmetoder som forsterkende læring er på fremmarsj, med imponerende resultater å vise til for komplekse problemer, og de er naturlig anlagt for å løse slike utfordringer.

Denne masteroppgaven representerer det første forsøket på å modellere produksjonsplanleggingsproblemet i lakseoppdrett som en Markov beslutningsprosess og å løse denne modellen med approksimert dynamisk programmering eller forsterkende læring. Warren B. Powell og David Silver er to frontfigurer for dette feltet og deres arbeider er viktige kilder til kunnskap og inspirasjon for denne masteroppgaven. ADP/dyp RL modellen tar beslutninger basert på verdifunksjonsapproksimasjon. Modellen er videre basert på n-steps Tidsmessig Forskjell læring, tester både problemtilpassede funksjoner og dype nevralt nettverk som verdifunksjonsapproksimasjoner og bruker en tellebasert, problem-spesifikk utforskningsstrategi som etterligner Øvre Konfidensgrense-metoden. For å kunne løse problemet stokastisk med hensyn på prisusikkerhet har vi utviklet en semi-parametrisk strukturell prismodell basert på forwardkurvedata. Denne modellen genererer tilfeldige prisutviklinger og tilhørende prisforventninger/forwardkurver.

ADP/RL modellen har blitt målt mot en blandet heltallsprogrammeringsmodell og en rullende horisontmodell for henholdsvis det deterministiske og stokastiske problemet. De beste produksjonsplanene generert av ADP/RL modellen i sin nåværende tilstand tjener ca 80% av sammenligningsmodellene. Dette regnes ikke som bra nok for kommersiell bruk i virkelig produksjon. ADP/RL er på mange måter like mye en kunst som en vitenskap og er en metode som krever store mengder algoritmetuning og kreativ problemløsning. Forfatterne er derfor likevel overbevist om at ADP/RL har stort potensial for å løse ikke bare produksjonsplanleggingsproblemet for laks, men også for andre dyrearter. Denne masteroppgaven åpner derfor kun døren til et stort forskningsfelt vi tror vil bli verdifullt. Oppgaven undersøker også verdien av stokastisk løsning av produksjonsplanleggingsproblemet for laks med hensyn på lakseprisen. Resultatene antyder at verdien av stokastisk løsning er liten sammenlignet med deterministisk løsning med hensyn på en forventet prisutvikling inkludert sesongvariasjon, men stor sammenlignet med deterministisk løsning med hensyn på en konstant pris.



# Table of Contents

<b>Preface</b>	<b>i</b>
<b>Summary</b>	<b>ii</b>
<b>Sammendrag</b>	<b>iii</b>
<b>Table of Contents</b>	<b>iv</b>
<b>1 Introduction</b>	<b>1</b>
<b>2 The salmon farming industry</b>	<b>4</b>
2.1 Global scale . . . . .	4
2.2 How salmon farming works . . . . .	4
2.3 Future of the industry - land based or sea based . . . . .	8
2.4 Salmon production scheduling . . . . .	9
2.4.1 The facility . . . . .	9
2.4.2 The optimization problem . . . . .	10
2.5 Financial aspects . . . . .	11
2.5.1 Salmon spot price . . . . .	11
2.5.2 Salmon forward contracts . . . . .	15
2.5.3 Optimal harvest, discount rate and finance structure . . . . .	17
<b>3 The theoretical background</b>	<b>20</b>
3.1 Machine learning . . . . .	20
3.1.1 Artificial neural networks . . . . .	21
3.1.2 Implementation - software packages . . . . .	24
3.2 Markov decision processes . . . . .	24
3.2.1 Profit maximization / problem formulation . . . . .	25
3.2.2 The Bellman equation - the dynamic programming principle of optimality .	25
3.2.3 Solution strategies . . . . .	26

---

3.3	Reinforcement learning - Approximate dynamic programming . . . . .	27
3.3.1	Value function . . . . .	29
3.3.2	Exploration versus exploitation . . . . .	30
3.3.3	On-policy vs off-policy learning . . . . .	31
3.3.4	Model-free vs model-based RL . . . . .	31
3.3.5	Temporal difference learning . . . . .	32
3.3.6	Algorithm tuning - hyper parameters . . . . .	34
<b>4</b>	<b>Relevant literature</b>	<b>35</b>
4.1	The price of Atlantic salmon . . . . .	35
4.2	Relevant ADP/RL applications . . . . .	37
4.3	Salmon production optimization . . . . .	38
4.3.1	Relation to other operations research literature . . . . .	38
4.3.2	Mathematical programming applied to biological systems . . . . .	38
4.3.3	Salmon production specific literature . . . . .	40
4.4	Summary and our contribution . . . . .	42
<b>5</b>	<b>Model and solution approach</b>	<b>45</b>
5.1	Salmon production scheduling as a Markov Decision Process . . . . .	45
5.1.1	Extensive mathematical formulation . . . . .	45
5.1.2	State . . . . .	47
5.1.3	Decision . . . . .	47
5.1.4	Exogenous information . . . . .	48
5.1.5	Transition function . . . . .	48
5.1.6	Direct profit/reward function . . . . .	48
5.1.7	Overall objective . . . . .	49
5.2	Semi-parametric modelling of prices . . . . .	50
5.3	Solution algorithm . . . . .	53
5.3.1	Variants . . . . .	54
5.3.2	Decomposition heuristic . . . . .	54
5.3.3	Key algorithmic elements . . . . .	55
<b>6</b>	<b>Model evaluation</b>	<b>60</b>
6.1	Qualitative assessment . . . . .	60
6.1.1	Model learning development . . . . .	62
6.1.2	Model scalability - runtime . . . . .	63

---

---

6.2	Benchmarking - deterministic optimization . . . . .	64
6.3	Benchmarking - stochastic optimization . . . . .	64
6.4	Value of stochastic solution . . . . .	67
<b>7</b>	<b>How ADP/RL works for salmon production scheduling</b>	<b>69</b>
7.1	Vast action space - Computational time . . . . .	69
7.1.1	Reducing the number of decisions . . . . .	69
7.1.2	Smarter selection of decisions . . . . .	70
7.1.3	Speed-up of model . . . . .	71
7.2	Algorithm convergence - overall performance . . . . .	71
7.2.1	Hyperparameter tuning . . . . .	72
7.2.2	Exploration vs exploitation strategy . . . . .	74
7.2.3	Value function shape . . . . .	74
7.2.4	Value function training algorithm . . . . .	76
7.2.5	Constraint management challenges . . . . .	77
7.3	What characterizes the optimal solution and how ADP/RL can find it . . . . .	78
7.3.1	The number of smolts released . . . . .	78
7.3.2	Harvesting . . . . .	79
7.3.3	The coordination of different tanks - opening up for other methods? . . . . .	80
7.4	Value of stochastic solution . . . . .	81
7.5	Critical reflection . . . . .	82
7.5.1	Results validity . . . . .	82
7.5.2	General reflections . . . . .	83
<b>8</b>	<b>Conclusion</b>	<b>85</b>
	<b>Bibliography</b>	<b>88</b>
<b>A</b>	<b>Appendix</b>	<b>91</b>
A.1	Skretting growth table . . . . .	91
A.2	Numerical illustrative examples to optimal harvesting. . . . .	92
A.2.1	Expected price variations - seasonality and fish size . . . . .	95
A.3	Seasonal component . . . . .	97

# Chapter 1

## Introduction

Global harvest of atlantic salmon amounted to nearly 2,500 Guttet Weight equivalent Tonnes (GWT) in 2019, with Norway producing more than 50% of that (Mowi (2020)). The global production of salmonids (of which the majority is atlantic salmon) still only amounted to 4.4% of global seafood supply, which in turn amounted to 7% of global protein consumption. Both because of a growing world population and how healthy and environmentally friendly salmon is compared to other protein sources, the demand is only expected to grow. In fact, both global and Norwegian salmon production has grown by a compounded annual growth rate of 5% ever since 2000 (Mowi (ibid.)).

Part of the reason for Norway's dominance in the salmon industry is the natural environment of our fjords which is very favorable for salmon. Traditional salmon farming in the fjords of Norway is however facing a number of challenges related to environment and sustainability. Environmental issues include the use of wrasse to remove sea lice, plastic waste pollution, ocean seabed pollution and negative effects on other wildlife in the area (Bjørndal and Tusvik (2019)).

The industry has to adapt to these challenges to be able to meet the increasing demand. One possible and promising solution is to produce salmon in land based facilities. Land based production has several advantages both in terms of sustainability, amongst others because production can be located closer to the markets, and in terms of a more controllable environment and reduced production risks (Bjørndal and Tusvik (ibid.)). Companies that specialize in the segment of land based production are therefore attractive as never before, and large amounts of investor capital are going in that direction (Berge (2021), EY (2020)). The following research confirms this trend.

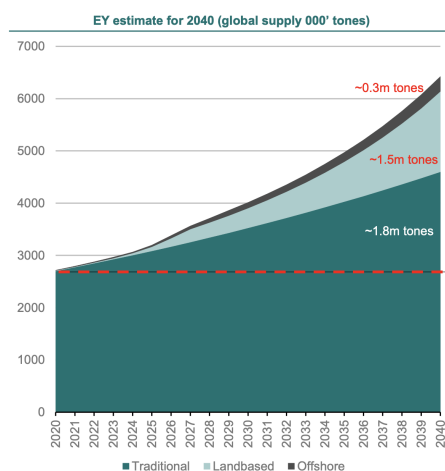


Figure 1.1: Estimated development of production. Source: *Land-based and offshore farming, what will be the impact from new farming methods*, DNB Markets (2021)

---

Land based salmon producers do however face higher production costs, for example from electricity and water management. Increasing costs and competition puts pressure on margins that can give producers an edge over the others. What's more, the salmon price faced by the producer is known as a particularly volatile commodity price. The salmon price is found to vary quite significantly with season, primarily because of biological factors in production (Mowi (2021)). Furthermore, the price per kg of biomass is not the same for all salmon, instead it is generally higher for larger salmon. Salmon are therefore typically categorized in different sales classes, e.g. 3-4 kg and 4-5 kg. An interesting question in planning salmon production therefore is the trade-off between merely producing as much biomass as possible and producing so that harvesting is done when the price is at it's highest. The optimal production schedule is characterized by as high as possible utilization of both the density constraints and the MAB constraint, which is an essential feature of the problem. This requires a delicate balancing of the number of fish released and the harvesting schedule, as well as a detailed biological growth model. Generally speaking, salmon production scheduling represents a very interesting optimization problem which in turn can have significant effects on a salmon producer's bottom line.

In the project thesis leading up to this master thesis, we therefore investigated optimizing salmon production scheduling with respect to price variations. The experimental analysis suggested that there is indeed added value to gain from planning production with respect to price variations. One of the goals for this thesis is however to do what we were not able to do in the project thesis, namely to explore the so-called value of stochastic solution with respect to price uncertainty, i.e. investigating how much there is to gain for the salmon producer in actively and continually taking the price variations into account when planning production.

Our work in the project thesis also revealed how computationally expensive and complex full-scale, detailed versions of the salmon production scheduling problem are when solved with traditional optimization methods such as mixed integer programming (i.e. branch and bound). Straight forward, a state-of-the-art solver like Gurobi showed exploding computational times already for a 3- or 4-tank facility problem. The salmon producer we cooperated most closely with were planning a 10-tank facility and the need for more advanced optimization methods which are more scalable became apparent. This represents the second goal of this thesis: designing a more scalable optimization method for optimizing production schedules.

EY (2020) also write in their report that the rise of land based and offshore salmon farming largely take away Norway's natural competitive edge and that a potential success factor for Norwegian aquaculture to keep on leading the way is "the institutionalization of salmon industry competency and knowledge through big data sets and machine learning (all production technologies)". They further write that a threat to Norwegian leadership in the aquaculture domain is that "the Norwegian industry does not build a sufficient pipeline of talents to transfer the industry to being AI- and knowledge-based due to existing "super profits"".

In Kennedy (2012), "Dynamic programming applications to agriculture and natural resources", the author argues that many biological herd management problems naturally lend themselves to dynamic programming based methods. We agree with this, and argue that three key factors suggest that such methods are a good fit for the salmon production scheduling problem. First, the problem has a repetitive, cyclic nature where decisions are made over time and hence can be modeled as a Markov Decision Process, which will be discussed in more detail later. Second, there exists good sources of research such as the Skretting growth table which enables accurate modeling of salmon growth. Finally, an essential, governing concept to managing biomass systems, whether it is in the shape of salmon, cattle or even forests, is determining and understanding the true value of the standing biomass at any point. This final idea also came from the fact that an inherent challenge with modeling salmon production as a linear programming problem is the so-called end-of-horizon problem, which basically is about finding the true value of the remaining fish in the fish tanks. Dynamic programming methods such as Approximate Dynamic Programming/Reinforcement Learning, as presented by W. Powell (2011), and particularly the subtopic of value iteration methods are at their very core about this exact concept, iteratively estimating the true value of the state of the system and using this value to make the optimal decision. What's more, such methods are flexible, adaptable, and, not least, naturally inclined to dealing with stochasticity, which is a requirement for the problem we are researching.

---

When combining the above insights with the recent rise of machine learning in general and reinforcement learning in particular, investigating the application of the advanced method of approximate dynamic programming(ADP)/reinforcement learning(RL) to salmon production scheduling seemed promising and intriguing. Deep reinforcement learning (the word deep coming from the application of advanced machine learning concepts such as multi-layered artificial neural networks) has in recent years shown impressive results when applied to highly complex problems/challenges, many of them focused on video games, in the computer science community. Perhaps the most famous example is DeepMind’s AlphaZero. AlphaZero is a highly advanced reinforcement learning model which learned to play the games of chess, shogi and go and actually in a matter of days of trial-and-error learning achieved superhuman performance and beat existing world-champion programs (Silver et al. (2017)). Such impressive applications, along with other real-world applications of the same principles, suggest several new applications of such methods might be promising and valuable.

This master thesis aims at thoroughly investigating the application of ADP/RL to the salmon production scheduling problem with a focus on price stochasticity by building such a model. We are the first to do so. Most artificial intelligence applications in the salmon industry have revolved around analysing operational elements such as ”smart fish feeding” and not around production schedule optimization (not including traditional optimization methods), even though our project thesis research revealed significant profit gains can be achieved. Most optimization efforts focused on the salmon production scheduling problem have to the authors’ knowledge applied methods such as traditional mixed integer programming, column generation or general stochastic programming methods, and none have so far gone down the path of dynamic programming. We therefore find our research interesting both from an operations research and machine learning community perspective but also from the salmon farming industry perspective.

The work presented in this thesis had three primary goals. First, we wanted to develop a more scalable optimization model for producing high-quality, profit-maximizing production schedules which can take into account price uncertainty. Second, we wanted to investigate whether ADP/RL can serve this purpose and work also for salmon production scheduling, and if so, then how well it can work and whether or not it has the potential for future industrial application. In this respect, we aim to not only build a successful model on our own, but also to gain experiences and lessons for future research in the same, novel intersection between ADP/RL and salmon production scheduling. Finally, we want to tie up a loose end from our project thesis which is determining the value of stochastic solution of the salmon production scheduling problem with respect to price uncertainty. By reading this thesis, we hope the reader will get significant, holistic insight into these questions. We discuss both the salmon farming industry background and sequential decision making theory and in the end devote a chapter to thoroughly presenting and discussing our learned insights into both the optimal solution of the optimization problem and the nature of ADP/RL approaches to finding this solution. The reader of this thesis will hopefully achieve in-depth, qualitative understanding of the problem at hand.

The thesis is structured as follows. Chapter 2 is meant to give the reader an introduction to the salmon farming industry. This chapter also discusses the dynamics of the salmon price and describes the salmon production scheduling problem in more detail. In chapter 3 we present some theoretical background on machine learning in general and reinforcement learning/approximate dynamic programming in particular. Chapter 4 present relevant literature in order to place our work into a larger context, both from an operations research perspective, from a salmon specific perspective and from an ADP/RL perspective. Our models, both the ADP/RL model and the price simulation model, are presented in chapter 5. An evaluation of the model is presented in chapter 6. Chapter 7 is a thorough discussion of how ADP/RL works for salmon production scheduling, i.e. what poses important challenges and which aspects are promising. Finally we draw conclusions and make suggestions for future research in chapter 8.

## Chapter 2

# The salmon farming industry

The Mowi Salmon Farming Industry Handbook 2021 (Mowi (2021)) gives a comprehensive overview of the salmon farming industry. The reader is referred there for data and parameters not mentioned here. Information not found there has been collected through the authors' various discussions with industry professionals over several years of personal interest in the industry.

### 2.1 Global scale

Farmed Atlantic Salmon harvested worldwide in 2019 amounted to almost 2,500 million GWT (gutted weight equivalent tonnes), with wild caught salmonids at around one third of that. Norway, Chile, Scotland and Canada supply the vast majority of this fish, with Norway harvesting around 1,200 million tonnes GWT in 2019 (Mowi (2020)).

The world demand for seafood based protein grows as the world populations grows and a larger part of the world starts consuming seafood. Atlantic salmon is a particularly interesting species for meeting this demand, because of its low production risks and its high potential for industrialization. Other advantages include a low feed conversion ratio, a low emission gas footprint compared to other protein sources and health benefits in terms of Omega-3 and vitamins. Salmon has also historically been a relatively expensive product on the shelves compared to other protein sources, which is one of the reasons for its high profitability. Furthermore, the supply from wild catching is stagnating in most geographical areas. Farmed Atlantic Salmon, along with other seafoods, is increasingly being seen as both a promising protein source for the future and an exciting industry in terms of business potential (Mowi (2021)).

The industrialization and large upscaling of Atlantic Salmon farming in recent years is increasing in intensity, and the resources put into it invites interest in more advanced research in the business aspects of the industry.

### 2.2 How salmon farming works

Salmon, in this case referring to Atlantic Salmon, is an anadromous fish species. This means they hatch in fresh water and live the first period of their life in fresh water, before they go through a smoltification process while migrating into the saltwater ocean and live there. For this reason, the industry is often segmented in a freshwater part producing salmon smolt weighing typically around 50-500 grams and a saltwater part growing the smolt further to salmon for food, typically harvested at weights between 3 and 7 kilograms. The total salmon farming life cycle is around 3 years, of which the first year is spent in fresh water. The growth time in salt water is 12-24 months, and heavily dependent on water temperature. Atlantic Salmon needs temperatures between 0 and 20 degrees Celsius, but the growth is by far highest at around 14-16 degrees Celsius. It is also

worth mentioning that the risks of diseases are higher at very high temperatures, so that there is a trade-off between disease risks and growth. This is also the reason that optimal conditions of sea based production is found in for an example Norway. When the fish have reached harvesting weight, they are typically collected by a well boat and transported to a processing plant where they are slaughtered and gutted (Mowi (2021)).

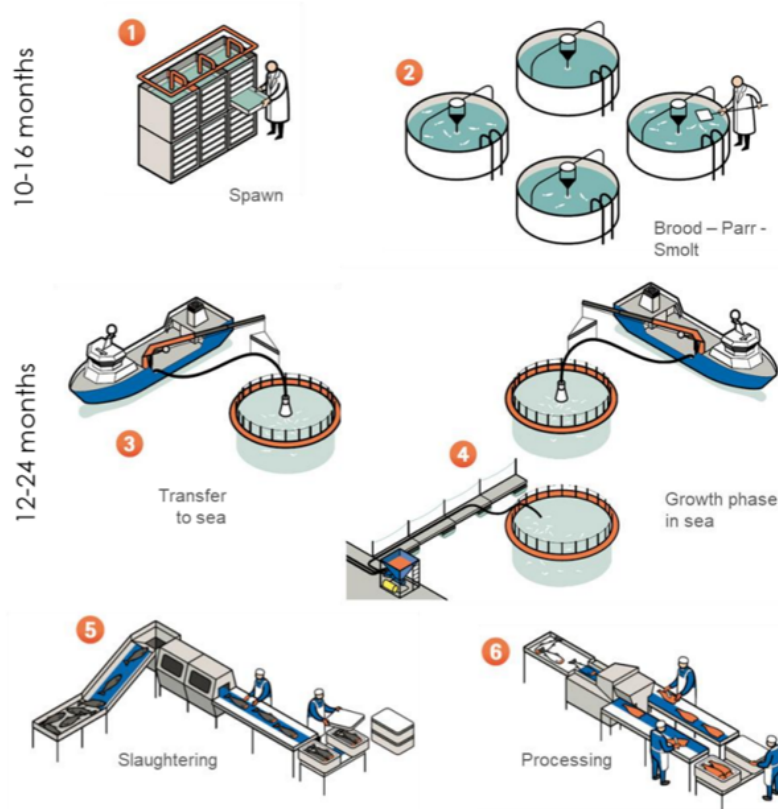


Figure 2.1: The salmon production cycle. Source: Salmon Farming Industry Handbook 2020, by MOWI ASA

Traditional salmon farming has to a large extent taken place in cages in protected fjords or bays. Smolt are produced in land based facilities where salmon fry are hatched in small fresh water tanks and grown to a certain size, typically around 100 grams. They are then released into cages in the fjords during spring or autumn, and held there for 12-24 months. Harvesting is done all throughout the year, with the majority harvested in the last half of the year, when the water temperature is higher and growth is better.

Salmon growth is typically a non-linear function of water temperature, feed quality, feeding saturation, water quality, light conditions and the weight of the salmon itself. In traditional salmon farming however, one tries to keep the water quality and light conditions at the optimal level at all times. Feed quality is a constant area of research and feed amount is usually set to saturation level to ensure maximum growth. asche2011economics argues that this is the optimal course of action for salmon producers. Given this, one can, as we do in this thesis, assume that the fish growth is only dependent on the water temperature and the fish size, as given for example by the Standard Growth Rate (SGR) tables provided by the feed producer Skretting (ref. figure A.1). It is then possible to create a biological growth model. In the case studied in this thesis, the water temperature in production is given at each time, and the growth for each weight class of fish at each time is then given by the SGR table and the time. One way of modelling growth that has been found preferable by researchers before us, is then to build a multi-dimensional matrix of transition coefficients between a set of weight classes at given temperatures.



Genetic differences in fish cause different growth in between the fish in a single batch, even if they all start at the same weight. This means the weight of the fish in a batch is typically a distribution around an average weight. In reality, this distribution is an approximate normal distribution with a coefficient of variation of up to 0.1. When the difference between the smallest and largest fish in a batch gets above a certain level, evolutionary effects divide the batch into what is called "winner fish and loser fish", where the largest fish take most of the food and the difference in growth is increased. For this reason, salmon producer generally do not want too large spread in weight between the fish in a batch.

The main costs in salmon farming are feed costs, smolt costs, labour costs, harvesting costs and other general costs. Of these, feed costs amount to almost half of the costs, with typical costs around 15 NOK/kg of feed and a Feed Conversion Ratio, FCR, of around 1.12 kg feed pr kg salmon growth. Smolt costs of around 12 NOK per smolt for 100 gram smolts and an additional 0.04 NOK per gram for smolts larger than that, as well as 1 MNOK fixed harvesting costs are common parameters reported to us by industry professionals which are used in the calculations in this thesis. Labour costs, interest costs and miscellaneous operating costs are also significant cost components, although these are typically left out of the salmon production scheduling problem.

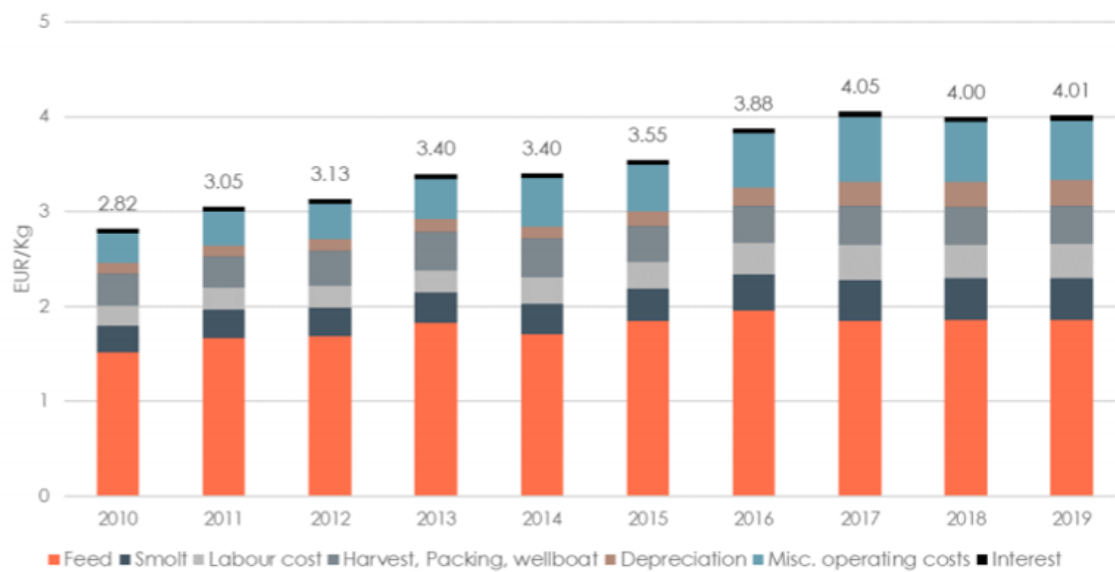


Figure 2.2: Breakdown of major costs in Norwegian salmon production over the last years. Source: Salmon Farming Industry Handbook 2021, by MOWI ASA

The salmon production part of the value chain is done by the most publicly known companies such as MOWI, Lerøy Seafood and SalMar, which are publicly traded at Oslo Børs. The specialized tasks of the value chain such as producing feed, smolt and harvesting is often done by specialized companies, i.e. feed producers, smolt producers and harvesting and processing facilities and wellboat owners. These are often wholly or partially owned by the salmon production companies. In that sense, some of the largest salmon production companies are involved in most parts of the value chain until harvesting. Nevertheless, it is possible for companies to operate in only one part of the value chain, for an example only producing salmon in the saltwater phase. In such cases, the producer typically enters a (long-term) deal with e.g. a feed producer to deliver feed at a certain quality and quantity to a given price. The largest salmon feed producers globally are Skretting, Ewos and BioMar. Increased demand and higher prices for farmed salmon has been the main driver of the revenue growth for aquaculture companies the last years, and this has had spill-over effects to other companies in the value chain. Various companies providing solutions for the aquaculture industry has experienced growth along with the salmon producers. This group of companies have been focusing on improving aquaculture technologies to provide solutions of biological challenges and the general increase in production cost. Such companies include software and technology providers like ScaleAQ. The largest companies among producers of technical solutions and services for the aquaculture industry such as barges, wellboats, feeding systems, cages, mooring systems, sea lice treatments and software are Steinsvik AS, Akva Group ASA, Aas Mek

---

Verksted AS, Optimar AS and Egersund Net AS (Olafsdottir et al. (2019)).

After harvesting, the fish are ready for sale, typically in either fresh or frozen form. The fish first go through primary processing (gutting) into HOG (Head-on-Gutted) fish, which is done in the same facility as the slaughtering. Then (possibly, not always) the fish are sent to secondary processing (smoking, filleting etc). The below figure attempts to illustrate the downstream distribution of Norwegian harvested salmon. While most of Norwegian salmon is sold fresh to the EU, Chilean salmon are mostly sold frozen (Olafsdottir et al. (ibid.)).

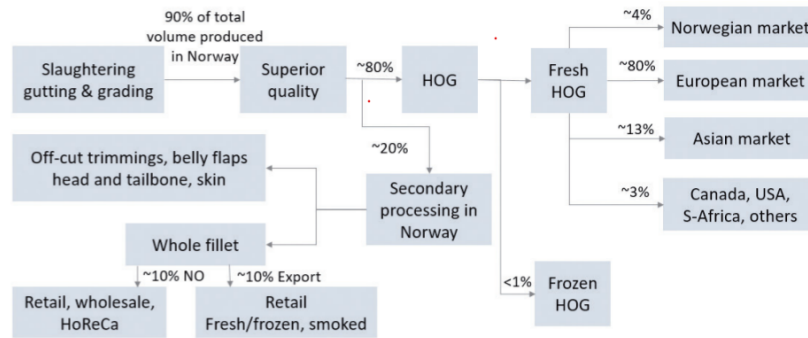


Figure 2.3: Typical downstream distribution of Norwegian harvested salmon. Source: Olafsdottir et al. (2019)

In the case of fresh salmon, a relatively high price differential is required to justify transport over long distances because it requires the cost of airfreight. Such trades vary over time, depending on arbitrage opportunities due to short-term shortages and excess volumes from the various producing regions. Drivers of the salmon price will be further discussed in section 2.5.1 on the salmon spot price.

Salmon are typically sold either through spot price contracts, futures contracts or long-term B2B contracts/agreements. The salmon market is still considered relatively immature, and sales mechanisms and the global market are characterized by this (Denstad, Ulsund, and Lillevand (2015)). It is estimated that approx. 60% is sold on such spot market conditions where contracts are signed on a weekly basis (i.e. delivery within a week), to the highest bidder. The salmon spot price is not regulated. About 40% of the volume is contracted, mostly from the largest producing companies. However, the ratio between spot and contract sales varies (Olafsdottir et al. (2019)). There is an emergence of long-term supplier-customer relationships through delivery contracts. These direct B2B agreements are typically agreements between large producers and large supermarket retailers, secondary processors or other wholesalers with a duration of 3-12 months. Many types of contracts exist, with both fixed and adjustable prices and fixed and adjustable quantities. These serve amongst others to smooth out price variations and to stabilize outlooks for both producers and customers and allow for better utilization of capacities (Denstad, Ulsund, and Lillevand (2015)). Financial forward contracts traded at Fish Pool involve a fixed price for a future delivery of salmon and are purely financial instruments used to hedge against price risks. These will be discussed further in 2.5.2.

As salmon is mostly marketed as a fresh product, each production region has historically focused on developing the nearby regions. Time and cost of transportation has driven this trend. The global trade flow of Atlantic Salmon in 2020 can be seen below (Mowi (2021)).

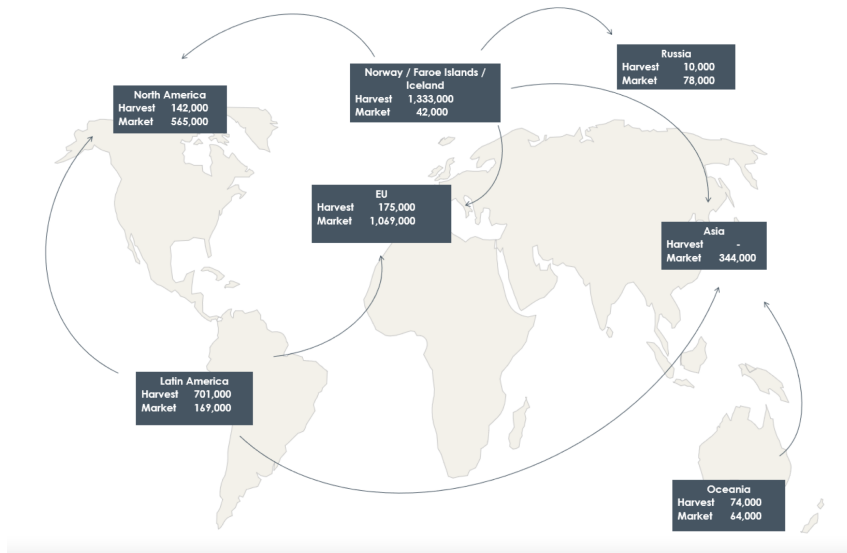


Figure 2.4: Global trade flow of Atlantic Salmon. Source: Salmon Farming Industry Handbook 2021, by MOWI ASA

### 2.3 Future of the industry - land based or sea based

The industry as a whole is facing both governmental and environmental restrictions setting limits on the increased supply of traditionally farmed salmon. Also, traditional salmon farming has large challenges in e.g. the sea lice problem. A company nowadays needs a license from the government to produce salmon. Traditional salmon farming in the fjords has received much criticism for environmental pollution, and there is a shift towards more sustainability in the industry coming up to meet the increasing global demand (Bjørndal and Tusvik (2019)). As polluting the fjords will not be accepted to a larger extent than already done, the industry has two options for increased production, simply put: going out into the deep ocean or going onto land. Large amounts of capital have the last few years been spent on research in both areas (Berge (2021), Hersoug, Mikkelsen, and Osmundsen (2021)). Producing salmon in the open sea is more challenging in terms of cage design, transport costs and wave motions. Producing in tanks in land based facilities has one main challenge in increased costs, for example from water flow management, salting of the water and increased electricity costs. Most likely, salmon farming in the future will be a combination of both land based, sea based and fjord based farming, with land based and sea based increasing in proportion.

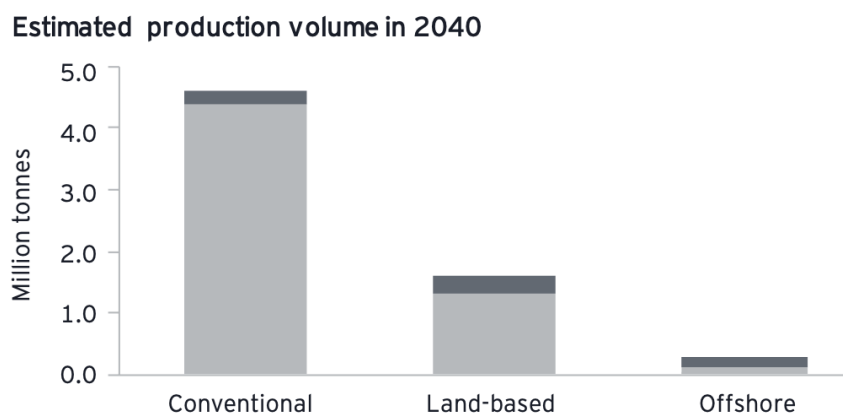


Figure 2.5: Estimated production distribution in 2040. Source: Norwegian Aquaculture Analysis 2020, by EY

---

As we can see from figure 2.5, it is expected that land-based production will be a significant portion of the global salmon supply in 2040. With increasing focus on Environmental, Social, and Corporate Governance (ESG) factors, one advantage of land-based farming is that there is less need for transport because the facilities can be put almost anywhere. This is, however, also a threat to Norwegian producers (EY (2020)) and as pointed out in the introduction, the Norwegian industry should build a sufficient pipeline of talents to transfer the industry to being AI and knowledge-based to maintain their position leading the way in the aquaculture industry.

Bjørndal and Tusvik (2019) conduct a thorough economic analysis of land based salmon farming and its competitiveness. They conclude that while the actually realized production costs are still uncertain, land based farming has several potential advantages in for example taking advantage of the seasonality in the rest of the industry. If successful, land based farming will profoundly change the industry and potentially shift the geographical center of production. The reader is referred there for further in-depth understanding.

## 2.4 Salmon production scheduling

### 2.4.1 The facility

In the sort of land based facilities that are kept in mind for this thesis, production works in many ways like in the traditional fjord based open cage facilities. Information stems from conversations we have had with different Norwegian land based salmon farmers. Each facility usually buys smolt from an external producer, or at least produces smolt in a separate facility or part of the facility and can then release batches of smolt in each tank for further growth until sales ready weight. Larger smolt, so-called postsmolt, are typically more expensive than smaller smolt. In land based facilities, the production environment is typically more controllable. In facilities where temperature can be controlled, smolts can be released throughout the year, however in facilities where sea water is used directly and the water temperature is given, smolts are typically not released during winter because the low temperature and bad growth conditions is not good for the fish in the first phase of life.

One of the most important differences from production in cages at sea to land based production, is water management. There are mainly two different technologies used in land based aquaculture. Recirculating aquaculture system (RAS) has been in use for the longest period of time and is the most common. This technology is basically the same that we find in aquariums, where water is recirculated. Most land based producers add salt to fresh water and heat it up. The recirculating process includes 1) bio-filtration where ammonia (toxic waste) is converted into nitrate, 2) solids removal, 3) oxygenation 4) pH control and 5) temperature control. The other main technology is a flow-through system where water from an outside source flows through the facility before going back into the source. This is typically cheaper than RAS. Disadvantages with flow-through compared to RAS include limited possibility to control temperature. It is also possible to combine these two technologies. With flow-through systems the temperature can be assumed to be given instead of being a variable. That yields less flexibility, but also simpler computational models.

A set of equal or different size water tanks, ranging typically from 5-20 tanks with volumes of thousands of cubic metres of water, are combined in what we term a facility. The facility kept in mind for this thesis has tanks with a volume of 30,000 cubic metres, and plans to have 10 of these. This corresponds to the planned facility of the salmon producer we have cooperated most closely with. Both smolts and feed are bought externally, and a contract is signed with a salmon processing operator which sends a well boat each time the producer wants to harvest fish from one or several tanks. New technology in the shape of sorting nets that are lowered into the fish tanks are used to sort the fish (roughly) by size. This way, the salmon producer can harvest only the largest fish in a batch and let the smaller fish grow additionally.

In many land based facilities the salmon farmer has the opportunity to move and split fish batches between different tanks during growth. This is highly interesting from an optimization perspective because it might improve the possible utilization of the facility capacity, but it also introduces a

---

whole range of mixed integer restrictions which increase the problem complexity. However, for the land based farmer we have cooperated most closely with during this work, this is not an option. This is also the case for traditional sea based farming, and hence serves to make this thesis more relevant for that as well. Therefore this is disregarded in this thesis.

## 2.4.2 The optimization problem

The salmon producer wishes primarily to maximise profits when planning/scheduling production. The only relevant source of revenue is the harvesting and selling of salmon at a size attractive in the market, typical in the range 3-6 kg. The main production costs to take into account in production planning, as described above, are feed costs, smolt costs and harvesting costs. Labour costs and other costs are typically considered fixed with respect to the production schedule, and are hence not relevant.

A set of very important restrictions characterize the salmon production in general and land based production in particular. First, different batches of fish cannot share tanks, meaning a tank must be empty and cleaned for a minimum of around three weeks before new fish/smolt are released in a tank. Second, fish welfare regulations set a limit on the density of fish in each tank, typically around 40 kilograms per cubic metre. Third, each facility is awarded a Maximum Allowable Biomass (MAB) quota from the government of the maximum amount of tons of fish allowed to be in the facility at any time.

The salmon production scheduling problem means to decide on a production plan that maximizes profit from growing and selling salmon for the salmon producer, while satisfying the constraints mentioned above. A production plan is a plan for the management of fish in each tank in the facility for a given period into the future, typically 3-5 years. The specific decisions are when to release smolts in each tank, how many smolts to release, what size of smolts to release and when to harvest from each tank, how many fish to harvest and at what size. The plan is inherently complicated in terms of how all decisions affect each other, and the managers in the industry typically spend weeks discussing and developing the production plan manually, based on years of experience and expertise on salmon growth.

We will now illustrate a few of the trade-offs faced; one wishes to harvest as much biomass throughout the year, but preferably when the price is at its highest. However, because of the MAB restrictions, there can never be more than a certain number of tons in the facility, so if all the tanks are on the same production cycle this greatly limits the output from the facility, as the facility will then be far from the MAB quota at the start of the cycle, and greatly limited by it at the end. As a consequence, one needs to coordinate the different tanks such that the biomass output is maximized without ever exceeding the MAB quota. Having different growth rates due to differing temperatures throughout the year further complicates the issue, because the release time in the year will affect the production time quite significantly. Another question is how many smolts to release. More fish means more biomass output per tank but also meeting the density restriction in each tank at a sooner point (lower weight). Hence, the producer will have to harvest more small fish, typically giving significantly lower price/kg than larger fish (above 4-5 kg). Thus, the combination of the MAB quota and density restrictions should indicate an optimal number of fish released exists, a number that makes the fish reach the optimal harvesting weight exactly when density restrictions and/or MAB restrictions are met (it might not be so simple due to the coordination with other tanks, but the principle remains). The size of the smolt released is also a highly relevant issue, because larger smolt are typically more expensive. However, releasing smolt of size 500 grams versus smolt of size 100 grams can greatly reduce the total production time in the salt water phase, and thus the time until harvesting can be done and a new batch can be released after that.

For each single tank, once the smolt have been released and are approaching sales ready weight, one gets what is called the single rotation problem, namely to find the optimal harvesting time of a batch. This is approached in a theoretical manner later in this chapter. In this situation, harvesting the fish at any given time will give a sales revenue minus a harvesting cost. However, both because the price received for the fish typically rise with growing fish size and because the

---

salmon price is volatile and seasonal, one can often get a better value of the batch of the fish by delaying harvest. Delaying will at the same time accumulate other costs such as extra feed costs, a discounting of the revenue received and also an extra risk taken on, because the production is inherently risky. Also, when salmon reach sexual maturity, typically at around 5-7 kg, growth stagnates (Denstad, Ulsund, and Lillevand (2015)). By delaying harvesting, the producer also has an alternative cost of delaying release of smolt. There is thus a trade-off between the possibly added value from waiting and the cost of waiting at each point in time. To complicate even further, the new sorting lattices, mentioned in the previous subsection, can now be used to harvest only some of the fish in a batch, typically the largest fish. That means extracting the highest-value part of the batch right now, and letting the smaller fish grow up in a better, more low-density environment. Both academic research and industry experience has shown that this can greatly increase the value extracted from a batch of fish, see for example Forsberg (1999). The question then becomes when to harvest which part of a batch, where to set the weight limit for harvesting and when to harvest the remaining fish, and one quickly realizes a myriad of possibilities exist. Experienced salmon producers deem the problem of scheduling their production a constant head-scratcher, in the sense that one can seldom be certain that a solution is the absolute optimal solution and a lot of time goes into careful planning.

The overall problem has a special, decomposed structure which is interesting from an optimization perspective. The structure arises from the fact that the only thing connecting the individual fish tanks is the MAB constraint. Without the MAB constraint, the optimal solution would merely be the optimal solution for a single tank applied to every tank, meaning that all tanks would be on exactly the same production cycle, i.e. they would be synchronous. Because of the MAB constraint however, which to repeat is a restriction on the total biomass allowed in the facility *at any time*, the optimal facility design in terms of number of tanks and tank sizes which allows the best utilization of the MAB quota is such that the sum of the tank biomass capacities (given implicitly by the density restrictions and the volume) exceeds the MAB quota, which in turn makes the optimal production schedule one where the tanks are asynchronous. In other words, the single tank schedules must be combined in a "master problem" such that the fish in the tanks reach harvest weight at different times. The "subproblems" can thus be said to be about determining the specific release of smolt and harvesting configuration for each individual tank (which, principally, are identical to the others). Utilizing this structure will be discussed later and will likely be a key feature in approaching this problem. For example, as will also be discussed in the literature review, most advanced optimization efforts on this problem have been about directly exploiting this structure by using Dantzig-Wolfe decomposition.

## 2.5 Financial aspects

All historical salmon price data are found at [www.fishpool.eu](http://www.fishpool.eu).

### 2.5.1 Salmon spot price

The salmon spot price market is where salmon are traded bilaterally for delivery within one week from a salmon producer to retail or to external processing or wholesale.

Head-on-gutted (HOG) salmon prices are normally reported in terms of specific reference prices. The three reference prices most widely used worldwide are: (1) the NOS/FHL FCA Oslo price, (2) the Urner Barry FOB Seattle price, (3) and the Urner Barry FCA Miami price. These prices are based on prices for superior quality fish, and not reflecting freight. Furthermore, each reference price refers to one specific product. For example, The FCA Oslo price refers to the price per kilogram HOG salmon between 4-5 kg packed fresh in a standard box and delivered in Oslo. The FCA Oslo price represents about two thirds of the global quantities for Atlantic salmon, and is thus the most important reference price used. NASDAQ have their own reference price index published for weekly spot prices for different sized HOG salmon which is based on FCA Oslo. This index is calculated based on reports of achieved prices and volumes made by a representative

---

panel of Norwegian salmon exporters and producers (Denstad, Ulsund, and Lillevand (2015)). As mentioned, roughly 60% of salmon is sold in the spot market. Thus, a salmon producer faces significant price risks, as the salmon price is found to be particularly volatile. Factors affecting the market price of salmon include:

- Supply (absolute and seasonal variation)
- Demand (absolute and seasonal variation)
- Globalisation of the market
- Presence of sales contracts reducing supply available in the spot market
- Flexibility of market channels
- Quantity
- Disease outbreaks
- Food scares

Basically, all price dynamics has it main reason in supply and demand dynamics. Supply and demand are in turn determined by other factors, such as the ones above. The price of distributing salmon, which in turn affects the salmon price, is determined by geographical locations of production and consumption. Salmon in Japan is typically more expensive than in the EU due to the longer travel from the producing countries (almost all salmon being produced in Norway, Chile or Scotland). Furthermore, there are different trade deals in different regions and a lack of price regulation in the salmon spot market, which is cause for imperfections in the market (Olafsdottir et al. (2019)). Also, salmon has been found to have positive cross-price elasticity with other protein sources, which might serve as substitutes, and the price of other protein sources such as meat affect the salmon spot price (Lodhi (2015)). There exists different value-added products of salmon, often considered more high-end products, which might be priced considerably higher than and independent from the reported spot price and Norway has a standardized distinction between product quality, the grades being Superior, Ordinary or Production quality. The introduction of standards like the Aquaculture Stewardship Council (ASC) Salmon standard has also become an important product differentiator. The introduction of the the many contract types and bilateral relationships, as well as the increasing degree of vertical integration in the business further complexify the market.

Because of limited possibility to store salmon, most of the explanation behind changes in salmon spot price can be related to production risks that affect harvest volumes. We will now have a look at historical salmon spot price development, and discuss different factors affecting both supply and demand. A larger share of production being land based (with more controllable environments) in the future might lead to fewer abrupt supply changes and hence reduce price volatility. However, salmon prices have been shown to be highly fluctuating, and will probably stay so in the upcoming years while land based farming develops further.

---

### Salmon spot price 4-5 kg August 2010 - August 2020

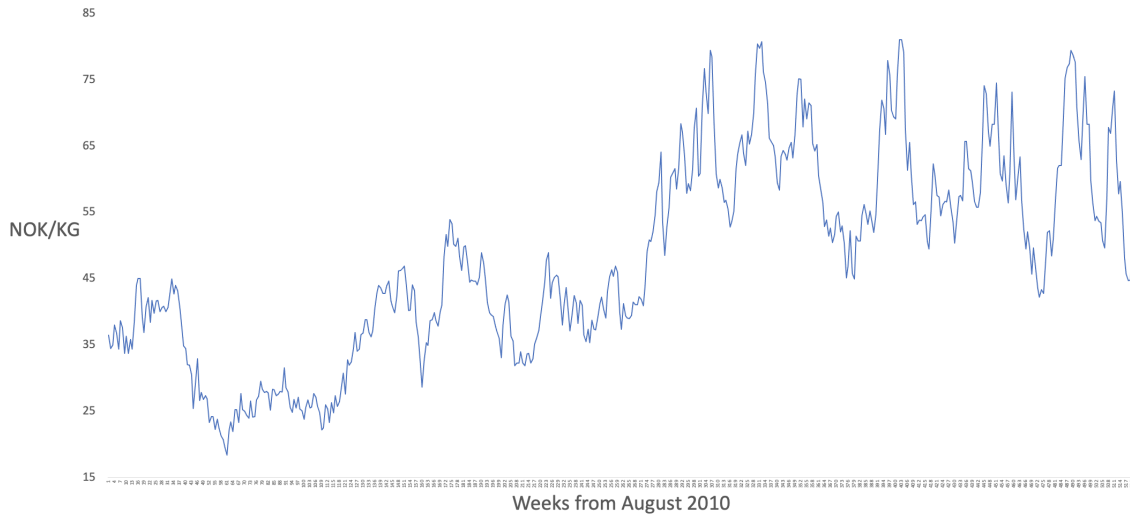


Figure 2.6: Spot prices August 2010 - August 2020. Source: fishpool.eu

From figure 2.6, we can see that there is quite a bit of noise in salmon spot prices. Annualized volatility were estimated to be approximately 40% in 2016, which was significantly higher than the average of commodities at approximately 25% (Asche, Misund, and Øglend (2016)). Highly volatile prices has its main reason in inelastic supply in the short run as a consequence of biological factors in the production, as well as a long production cycle. Salmon farmers generally spend significant efforts on forecasting harvest volumes in the industry, as this is the most important indicator for price movements. They do this by reviewing reported feed consumption data, smolt market sales, seawater temperatures and vaccination rates (Mowi (2021)). As salmon production is a biological process, realised production does not always concur with planned production, which leads to periods of under- and oversupply, which again lead to fluctuating prices. That means, when supply or demand changes, it is not possible for producers to turn on some switch to get more salmon in the short run. Examples include shortage in supply because of mass deaths due to algae blossoms, or change in demand because of holidays. The latter is to some degree possible to schedule. However, some risks in production are not possible to predict and hard to plan for. As land based producers do not face biological risks to the same degree, they might benefit from periods of biological problems at sea based farms. One might even imagine situations where, if one has information about production levels of large salmon producers or particular large biological accidents, one can to some extent forecast certain price fluctuations.

In figure 2.6, we can see that there is a significantly higher average price after 2015. Possible explanations include higher demand compared to supply and/or higher production costs. Although the introduction of land based farming gives higher competition, these producers will also have higher production costs and land based farming is still in a very early stage and small compared to traditional farming. That suggests prices in the future will be partly determined by a trade-off between increased competition and higher average production costs.

The seasonality in the salmon price can largely be explained by biological factors in production. Approx. 89% of Norwegian salmon is sold as fresh products and therefore have limited storage possibility (Olafsdottir et al. (2019)). In traditional sea based salmon farming, salmon smolts are not released all year round but rather in the times of the year where the water is warmer and more healthy for the fish in its first stages of life. Harvesting on the other hand, can be done throughout the year, but is also mostly done during the summer half of the year. The reader is referred to Mowi (2021) for detailed graphs showing this seasonality. These factors, combined with a typical demand peak around christmas times and the inelastic supply discussed earlier, together create a natural seasonality in salmon prices.

Historical seasonality can be found by adjusting an oscillating function to historical data by regres-



sion. It is possible to use different functions, and a model will of course always be a model that does not reflect reality perfect. Nevertheless, we have used a function on the form  $S = \alpha \cos(\frac{2*\pi*(t)}{52} + \phi)$  in a regression to find an estimate for the seasonality, where t is the week of the year. Salmon spot prices and the fitted regression line is plotted in figure 2.7.

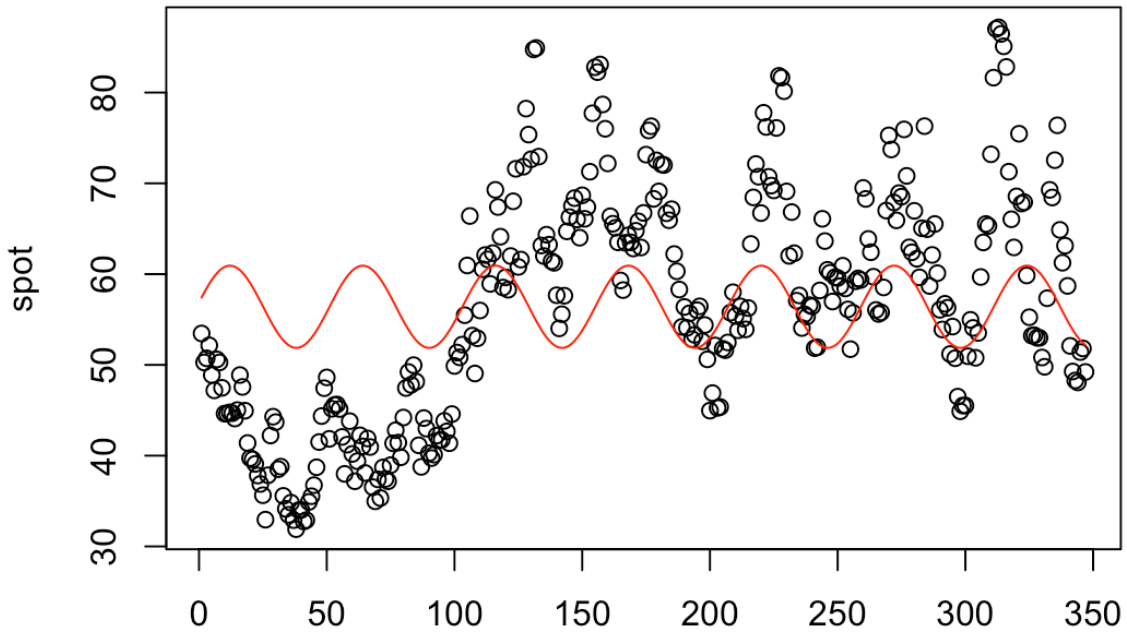


Figure 2.7: Plot of historical salmon spot prices and the regressed cosine-function

The results show 8% deviation from the average to the top/bottom. There are of course a lot of fluctuations around this estimate, but it might be used to say that, given this function and historical prices, the expected price on top of the cycle is 8% higher than the average price. The expected price is 8% lower at the bottom of the cycle. We will later use an amplitude of 5-10% of the average price as a reference point for the seasonality in the price. The industry expects a price top ranging from Christmas and until the end of Q1, and the phase of the regressed cosine function is in line with this. Yet, the real seasonality might not best be represented by a smooth sine/cosine function, but rather by a function with a wider top, stretching over Q1. Furthermore, the amplitude and timing of seasonality peaks can vary from year to year in reality. The seasonality amplitude is also typically relatively small compared to the random fluctuations in the salmon price, which is one of the reasons why the seasonality might be hard to see when looking at a graph of historical prices.

Salmon spot prices are given for different for different weight classes. There are (usually) different prices for these classes, which are for example 3-4 kg, 4-5 kg and 5-6 kg. The majority of salmon is sold in these weight ranges. In general, larger fish are worth more in NOK/kg. The price development of these three sales classes are plotted from August 2019 to August 2020 in 2.8. We only present these over one year, so that we can see the price dynamics better.



Figure 2.8: Spot prices August 2019 - August 2020. To see the details in the weight class price differences, only one year is presented. Source: fishpool.eu

We see that these graphs have a high degree of correlation, and that there usually is a difference in price between the weight classes. The reasons for the (varying) price difference between weight classes are that they can to some extent be seen as different products combined with supply and demand mechanisms (Olafsdottir et al. (2019)). Bigger fish contain a slightly higher percentage of meat, and a lower percentage of bones etc. Because the fish is bigger, it is also possible to make bigger fillets which might be attractive. Hence, they are not necessarily perfect substitutes. The price of a heavier fish is therefore usually higher. At some points however, these graphs might cross each other so that, for an example, a fish of 5-6 kg is actually lower paid than 4-5 kg in NOK/kg like in April 2020. Around this point in time, the price gaps between sales classes were very low. At other points, the difference is relatively large. During the presented period the maximum price difference between two weight classes were almost 15%. Larger differences between sales classes imply a higher optimal harvesting weight for salmon farmers. Over five years starting in April 2015, the price of the mentioned weight classes were as follows:

Sales class	3 - 4 kg	4 - 5 kg	5 - 6 kg	FPI
Average price (NOK/kg)	56.96	58.86	60.52	58.79
% deviation from FPI	-3.105	0.120	2.944	0

Table 2.1: Average price difference between weight classes. For reference, the basis of salmon forward contracts (Fish Pool Index (FPI)) is included. This is calculated as  $0.3P(3 - 4kg) + 0.4P(4 - 5kg) + 0.3P(5 - 6kg)$ . Salmon forward contracts is further described in the next subchapter. Source: Fishpool.eu and NASDAQ Salmon Index

It is worth noticing that producers get paid for the weight of salmon after some weight loss due to slaughtering at approximately 18%, that is the difference between live weight and gutted weight. The latter is usually referred to as gutted weight equivalent tonnes, GWT. This difference does not really matter for the study we are going to do, but this weight difference is nice to keep in mind when considering profits based on weight. Finally, salmon prices are always referenced in price/kg.

## 2.5.2 Salmon forward contracts

A forward contract is an agreement between a buyer and seller to trade an asset or commodity at a specified price at a future date. The price of the asset is set when the contract is drawn up. For the case of salmon, the underlying basis is the *Fish Pool Index*<sup>™</sup> (FPI), which is a synthetic spot price reflecting the current market price of 1 kg fresh Atlantic Salmon. This price is computed as  $FPI = 0.3P(3 - 4kg) + 0.4P(4 - 5kg) + 0.3P(5 - 6kg)$ , superior quality head-on-gutted salmon. Thus, forward prices does not have weight classes, such as the spot price. The FPI price is reported on weekly basis, while the underlying of forward prices are computed as the monthly average of

---

weekly FPI prices. The final clearing services are operated by NASDAQ OMX, so that counterparty risk is removed. The forward contracts traded on Fish Pool are purely financial instruments used for price risk hedging.

The organizer behind the FPI is Fish Pool ASA. Fish Pool ASA facilitates an international market place for financial salmon contracts. They are licensed by the Norwegian Ministry of Finance, and have been providing risk management tools for market participants subject to salmon price risk since 2005. Financial contracts are primarily used by salmon farmers, exporters, importers, processors and retailer in the value chain of salmon to hedge their salmon price risk (FishPool (2020)). Fish Pool ASA provides daily forward prices based on the latest market trades. For a forward market to be efficient, sufficient numbers of long-term and short-term positions should be taken. With less than 10% of the total production of salmon trading at Fish Pool, the volume is relatively low, compared to other, more mature forward markets. The volume has, however increased from recent years, and Fish Pool expect the volume to increase as they will get access to the Euronext infrastructure (FishPool (ibid.)). Out of the contract lengths, Andersen (2019) found half-year and quarterly, as well as the front-month, 1 and 2 months to expiration contracts to be superior in terms of trading volume.

Forwards are today's value on future delivery of fish. Forwards of longer maturities contain less noise than spot prices. The forward curve as of August 2020 is plotted in figure 2.9. The *expected* seasonality is well reflected in the forward curve, which can be constructed by plotting forward prices of different maturities. Later, the forward curve will be used to estimate the seasonal factor for our price model.

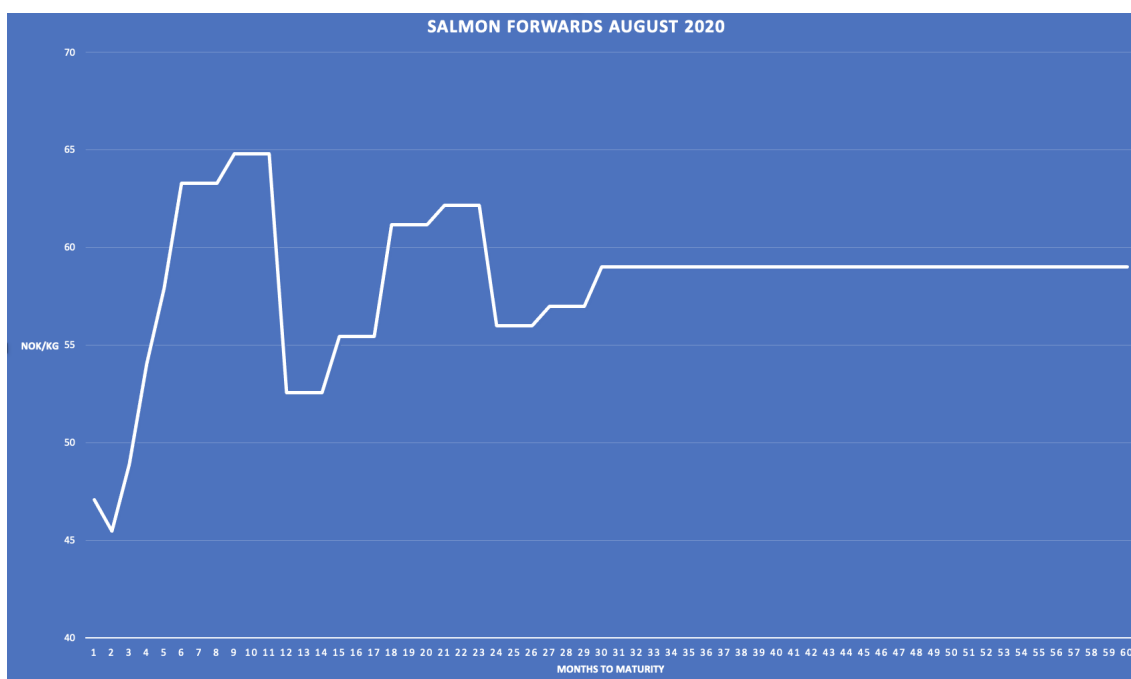


Figure 2.9: Forward curve August 2020. Source: fishpool.eu

We see that the expected seasonality in the forward curve confirms the estimated peak in Q1 based on historical prices. However, for this particular point in time, there is an 18% deviation to the maximum and minimum from the mean price. The expected prices in this forward curve take into account that there will be higher demand around Christmas. Prices were expected to rise from Summer until October, and then stay high and almost flat until March before they drop. This corresponds very well to the expectations of the industry, where they are talking about a "Christmas rush" and a price peak in Q1. We see that the forward price is flat beyond 2.5 years due to low trading volumes and high uncertainty in very long contracts.

---

### 2.5.3 Optimal harvest, discount rate and finance structure

In this section we present the single rotation harvesting problem, aiming to give the reader introductory insight into the qualitative aspects of harvest timing. Based on this, we discuss the importance of the discount rate applied by the salmon producer. Finally, we discuss how this discount rate in turn is dependent on the financial structure of the salmon producer.

The so-called single batch rotation problem is a (theoretical) problem considering a *one-time* investment in fish and when to harvest these fish. Studying this isolated subproblem of the salmon production scheduling problem can give important insights. Because of the time value of money, the discounting rate is an important factor in this deciding the optimal harvesting schedule of a single batch. The elements going into the calculations are the revenue from the harvested biomass, the feeding costs accumulated throughout the production, fixed and/or variable harvesting costs and (possibly) insurance costs. Smolt release costs are irrelevant to this problem. In reality, however, a salmon producer as considered in this thesis has a set of tanks or cages, and will release and grow and harvest fish in these batches multiple times into the future. In the multiple tank, multiple batch rotation problem this presents, once the batch is harvested, space is available for a new batch and new smolts are released. The multiple batch rotation harvesting problem therefore includes, in addition to the abovementioned elements, an additional opportunity cost for delaying all future batch harvests. This is because delaying harvest of a batch a certain amount of time then also delays all future cash flows from that tank by the same time.

Let us first consider the theory behind the single batch rotation optimal harvesting problem. Consider a case where we release a single batch of  $N$  fish in a tank. Ignoring mortality, the number of fish in the batch will remain unchanged. Say that all the fish are of equal weight at each point in time,  $w(t)$ .  $w(t)$  can be calculated as a Markovian process from  $w(0)$  (the smolt size), because growth is assumed to only be a function of temperature and size. Then the biomass in the tank at time  $t$  can be calculated as

$$B(t) = Nw(t) \quad (2.1)$$

The value of a batch of fish today which is harvested at time  $t$  and continuously discounted is then, assuming the whole batch is harvested at once,

$$\pi(t) = e^{-rt}B(t)P(w, t) - e^{-rt}C^{hf} - e^{-rt}C^{hv}N - \int_0^t e^{-ru}C^f Nw'(u)du \quad (2.2)$$

where  $P(w, t)$  is the salmon price as a continuous function of fish size and time,  $C^f$  is the feed cost per kg of fish growth (feed cost multiplied by FCR),  $C^{hf}$  is a fixed cost on harvesting,  $C^{hv}$  is a variable cost of harvesting per fish and  $w'(t)$  is the growth at time  $t$ . The last term represents the feeding costs which have accumulated throughout the time until harvest.

The optimal harvesting time is thus found by finding

$$\max_{0 \leq t \leq T} \pi(t) \quad (2.3)$$

In the following, we will omit the harvesting costs from our discussions because they have relatively limited effect on optimal harvesting time (see Bjørndal (1988)), and we will also ignore insurance costs. We have stated that the price is a function of both fish size  $w$  and time  $t$ , which is true in general for the salmon price, but in our case of a single batch rotation problem, the weight  $w$  is simply a function of time  $t$ , so  $P(w, t) = P(w(t), t) = P(t)$ . Furthermore, we can define  $V(t) = B(t)P(t)$ . Finally, we can name  $C^f(t) = NC^f w'(t)$  the feeding cost at time  $t$  (which is a function of the growth, or weight change, at time  $t$ ). To find the optimum, we take the derivative with respect to time and set equal to zero, which gives

$$\pi'(t) = V'(t)e^{-rt} - rV(t)e^{-rt} - e^{-rt}C^f(t) = 0 \quad (2.4)$$

By dividing by  $e^{-rt}$  and rearranging, this becomes the criterion for the optimal harvesting time  $t^*$ ,

$$V'(t^*) = rV(t^*) + C^f(t^*) \quad (2.5)$$

---

which can be understood as the marginal increase in value/revenue having to equal the opportunity cost plus the feed cost at time  $t$ . As long as  $V'(t)$ , the marginal increase in revenue at time  $t$ , is larger than the opportunity cost and the marginal feed cost at time  $t$ , it is advantageous to delay harvesting. Or, in even simpler terms, if the value increase from delaying harvest is larger than the added cost of delaying harvest, then the salmon producer should delay harvesting until the point where these are equal. The potential profit today needs to be weighted against the option to harvest at a later point and pay the operating and financial costs in the meantime. The reason the revenue from harvesting at a later point will be higher is primarily because the fish grow so there will be more biomass. What makes it interesting in terms of optimization, however, is that fish growth diminishes with increasing weight (stagnating when sexual maturity is reached at around 5-7 kg (Denstad, Ulsund, and Lillevand (2015))) and that there also is a mortality to consider. Because of this, it is not optimal to wait forever, *even* in the case of a non-changing price and no density restriction in the tank/cage. When one also adds a change in price, and then adds real production constraints such as density restrictions, this problem gets rather intricate. Bjørndal (1988) goes even further than the theory presented here by also including harvesting costs and insurance costs and by providing practical examples of numerical calculations given an empirical weight function  $w(t)$ . He does not, however, consider density restrictions in the salmon cages. We will also leave this (rather important) restriction for this part, although it is an essential part of the problem we tackle in the rest of this thesis. The interesting takeaways from this analysis are as follows (see Bjørndal (ibid.)).

- Both interest rates, feed costs, harvesting costs and insurance costs have an effect on the optimal harvesting time and should be considered. However, the effect is relatively small compared with the solution in the no-cost example where harvesting is done when biomass is maximised (this necessitates an included mortality rate) (or when the density capacity is reached).
- Increasing interest rate, feed costs and insurance costs push the optimal harvesting time closer in time, i.e. the fish should be harvested sooner, whereas increasing harvesting costs (either fixed or variable) push the optimal harvesting time further into the future, i.e. later harvesting. That is because harvest costs occurring at a later point in time is also discounted by the discount rate. Harvesting costs have a relatively smaller effect than feed and insurance costs.
- In the case where growth rates differ with weight, as is the case in real production, selective/partial harvesting can be profitable and depends on how the price varies with fish size.
- Considering a real salmon production where production will go on forever and harvesting the first batch makes room for releasing a new batch (the multiple rotation problem) makes the problem more complex, but essentially only works to push the optimal harvesting time closer in time, i.e. to harvest sooner, because all future cash flows are also delayed by delaying harvest and the opportunity cost of the fish farm has to be included on the right hand side of equation 2.2.

The reader is referred to Bjørndal (ibid.) for further reading and more thorough derivations. We are here primarily concerned with the qualitative considerations regarding optimal harvesting time.

In appendix A.2 the reader will find a set of illustrative numerical examples. These show that the discount rate might play an important role for the optimal harvesting schedule. Sometimes, a different rate to discount future cash flows will make one salmon producer harvest today, while another might delay harvesting until the next time period. In fact, when planning production and harvesting times in particular, we believe that a salmon producer should keep in mind all the elements such as feed costs, delay of release of the next batch, as well as harvesting costs, smolt release costs and other relevant costs. We show that the expected price changes are particularly relevant to these decisions. In addition, the time value of money and its relation on the applied discounting factor is a key concept, which will be discussed further here.

In equation 2.2, 2.4 and 2.5, the discount rate  $r$  can be set equal to the risk free rate. That is the case if we operate in forward contracts with reasonably long maturity, or equivalently risk adjusted

---

future prices. However, the fact that unpredictable events may lead to jumps in the price over a short period has implications on how we discount future cash flows. The probability that such events occur is difficult to predict. Therefore, it is also hard to estimate the correct discount rate. If we estimate the probability of such events higher, for example, the discount rate also needs to be higher. Therefore, expected (but uncertain) prices should be discounted with an appropriate risk adjusted rate. It is natural to assume that the main source of income at salmon companies come from selling salmon. Then the company risk will be approximately equal to the project or activity risk. Therefore, it is natural to assume that the weighted average cost of capital (WACC) is very well suited for discounting cash flows.

We then have that today's value of one unit (kilogram) salmon delivered at time T is equal to

$$\frac{\mathbb{E}[S_T]}{(1 + WACC)^T} \quad (2.6)$$

Where  $S_T$  is the spot price at time T and the WACC is given by

$$WACC = \frac{E}{E + D}r_e + \frac{D}{E + D}r_d * (1 - t) \quad (2.7)$$

Where  $t$  is the corporate tax rate,  $E$  is the value of equity and  $D$  is value of debt with discount rates  $r_e$  and  $r_d$ , respectively. As the expected value is not risk adjusted, it is discounted with a factor that accounts for risks, given by the WACC. Often, capital is more expensive for young companies, because they do not earn money yet and are associated with larger risks. The required rate of return for investors will thus be higher because of the higher risk. For land-based salmon producers, of which there are still only few and relatively young and small companies, financing is almost solely made by equity in the early phases. Investments are very large and banks assume risk to be high which make financing by debt difficult. Solheim and Trovatn (2019) assume that a land-based facility is financed with 89% equity and 11% debt. This will, however, vary with the development of the company. For example, Atlantic Sapphire, one of the leading land based salmon producers, announced that they secured a USD210m senior secured credit facility with DNB in April 2020. DNB commented in a panel debate that although they are cautious in financing land-based projects, Atlantic Sapphire has a proven track record of industrial experience and closeness to customers, which significantly reduces transportation cost. These are all factors that reduce the credit risk to an acceptable level for the banks. More mature companies will typically have more debt financing, but also lower risk. For example, Mowi, the world's largest salmon producer, reports an equity ratio of 50.9% in Q1 2021. Ultimately, this leads to a lower WACC.

That the discount rate does have an impact on optimal harvesting is confirmed by Mistiaen and Strand (1998). Consequently, companies discounting their future cash flows with a higher discount rate will in general have an incentive to harvest sooner than those discounting their cash flow with a lower discount rate.

## Chapter 3

# Theoretical Background

This chapter will provide some theoretical background on a set of algorithms stemming from the interception between dynamic programming theory and machine learning called approximate dynamic programming or reinforcement learning. We first introduce the general concept of machine learning. Then we discuss a type of problems which are called Markov decision processes and explain why salmon production scheduling can be seen as such a problem. Then we discuss reinforcement learning (RL) and approximate dynamic programming (ADP) in particular, and various concepts thereunder. We also explain why ADP and RL, though stemming from two different communities, are two sides of the same coin. Most of the material on reinforcement learning and approximate dynamic programming has been distilled from books, papers and articles found on the internet. The two most important sources of inspiration are worth mentioning and they are "Approximate dynamic programming: Solving the curse of dimensionality" by Warren Powell (W. Powell (2011)) and the UCL course on reinforcement learning by David Silver (Silver (2015)). The authors have gained most of their basic understanding of ADP and RL from these two renowned experts on the topic, and their various successful applications of the methods inspired the question of whether such methods could work well for salmon production scheduling.

### 3.1 Machine learning

Machine learning is an important subtopic of artificial intelligence that focuses on how a computer can learn by applying different statistical models on data.

Machine learning can roughly be divided into three different categories:

- **Supervised learning:** In supervised learning, an agent (action-taker) is given a set of input (independent variables) and output (dependent variables) data, which in turn is used to learn the function that has produced the output from the input. Thus, supervised learning tries to generalize a pattern by looking at examples. Examples of supervised learning include regression and classification.
- **Unsupervised learning:** Unsupervised learning is equal to supervised learning except that the dependent variables are not given and need to be discovered by the model. Thus, unsupervised learning tries to find patterns in the independent variables themselves. Clustering is an example of unsupervised learning.
- **Reinforcement learning:** In reinforcement learning, an agent is placed in an environment and learns how to take actions in order to maximize the notion of cumulative reward. This is learned through a series of reinforcements where the model experiences what kind of rewards different actions lead to.

---

### 3.1.1 Artificial neural networks

Artificial neural networks (ANNs) are a general class of functions which take an input and produce an output, just like any other function. These "functions" have grown to be of vast importance in almost all types of machine learning because of their desirable qualities, such as their general applicability and universal approximation capabilities.

Neural networks were originally invented as an attempt to artificially mimic how human brains work. They are named after the *neurons* of the brain, and such neurons are combined like nodes in a network. Each node takes a number of inputs, multiply each of them with a corresponding *weight*, sums these weighted inputs and finally adds a *bias* to this sum, before sending the final number output on to the nodes in the next layer of the network. Mathematically, the output of every node  $j$  is the following:

$$\hat{a}_j = \sum_{i=0}^n w_{ij} a_i + b \quad (3.1)$$

where  $i$  is the index of all nodes in the previous layer and  $b$  is the bias. This is the primary idea of a neuron, i.e. a linear combination of inputs. However, in order to further expand the capabilities of neural networks, amongst others to be able to approximate non-linear functions, the notion of activation functions were made a standard part of most neural networks applied today. An activation function is a (simple) function applied to the output of each node before the it is sent on to the next layer in the network. ReLU, tanh and sigmoid are widely used activation functions. We will not be going deeper into their specifics here. This makes the total mathematical operation of each neuron the following:

$$a_j = g(\hat{a}) = g\left(\sum_{i=0}^n w_{ij} a_i + b\right) \quad (3.2)$$

where  $g$  is the activation function.

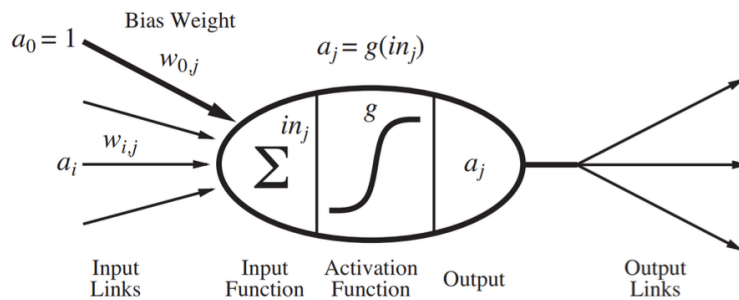


Figure 3.1: Simple neuron model from Russell and Norvig (2002)

ANNs are organized such that you might have different layers of nodes. When a network has more than two layers (the input and output layers), the layers in between are called hidden layers. When a network has hidden layers, it is called a deep network. The term deep learning, which applies to machine learning in general, and *deep reinforcement learning* which applies to RL in particular, refers to the use of deep neural networks. If inputs are sent chronologically through the network, we call it a *feedforward* network. The network can also be *recurrent*, which means that there are loops where information is sent in loops between layers. Other types of networks exist as well. In the case of one or none hidden layers, we are dealing with a shallow network. Shallow neural networks usually require more *features* in order to perform well. Features are characteristics found in training data. However, this depends on the nature of the problem. One of the advantages of deep neural networks is that we can feed them with raw data instead of a human deciding on what



to feed the network with. Then the DNN will decide itself what features are important, and then develop features itself. Each layer can create abstractions which are then fed to the next layer and so on to create more and more complex features. The drawback of deeper networks is on the other hand that they typically require more training. The picture below illustrates a deep feedforward neural network of fully connected layers.

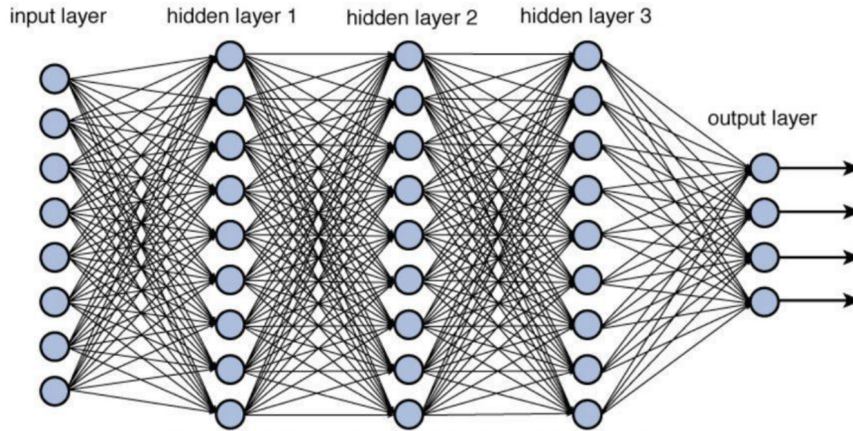


Figure 3.2: Deep neural network example

Each node performs a linear combination of inputs, but the whole network can be non-linear. In fact, ANNs are universal function approximators, which means that they can, in theory, approximate any kind of function. The universal approximation theorem was first proven in a basic version by Hornik, Stinchcombe, and White (1989), later extended in many ways and recently given a thorough description by Kratsios (2021). The approximation and the training of the network is done by changing the different weights of the linear combination performed in each node as well as the node bias for each node in the network. The set of weights and biases of all nodes are called the parameters of the network, often denoted  $\theta$  in parametric functions. The typical use of neural networks is to either approximate unknown functions, such as in RL, or to recognize features in input data. In modern, complex applications of neural networks, the networks can contain a large number of hidden layers and many thousands of nodes. As an example, while not relevant to this thesis, we mention how neural networks are used for image recognition by actually feeding the color value of each picture pixel as a single input to the network and sending these values through many layers in order to have the output nodes recognize what the picture is showing.

### Gradient descent

Training a neural networks means optimizing the parameters (i.e. node weights and biases) in order to make the network produce the desired output. The desired output depends on the task at hand. In a reinforcement learning problem, the task is to approximate some unknown function. The most common way of doing this is a numerical optimization algorithm known as *gradient descent*. One calculates some sort of *loss*, which is a function of the error between the output calculated by the network for a given input and the *target* output, which is (our estimate of) what the output should have been. In order to minimize this error/loss, one takes the derivative of the loss with respect to each of the function parameters and then adjusts each parameter a small step in the direction of the negative gradient. This is then repeated iteratively. Mathematically, given a loss function  $\Theta$ , the parameters are updated as follows

$$\Theta \leftarrow \Theta - \alpha \nabla \text{Loss}(\Theta) \quad (3.3)$$

Here,  $\alpha$  is the learning rate, which can be tuned in order to tell the network how fast it should converge. This might have big impact on the final performance of the network. The learning rate

---

might be constant, or decrease over time. Similar to the equation above, the node weight updates can be mathematically written as

$$w_i = w_i - \alpha \frac{\partial}{\partial w_i} Loss(\Theta) \quad (3.4)$$

Many variants of the gradient descent algorithm exists. One way to perform gradient descent is to randomly select training samples and update after each sample. The name of this method is stochastic gradient descent (SGD). Several other, modern algorithms have been developed alongside the development of neural networks in order to improve training. Adam, AdaGrad and RMSprop are three widely used optimization algorithms today. Such variants make various extensions to the algorithm by adding features such as momentum and decay rates, keeping track of per-parameter learning rates and so on, but they are all based on the same principle, which is taking a step in the direction of the negative gradient of the loss function.

## Backpropagation

With the introduction of multiple layers in the networks, the output becomes a nested function of the input and calculating the gradient becomes not straight forward. The solution is based on the *chain rule* for derivatives. Basically, to first update the parameters in the output layer, the derivative of the loss function (i.e. some function of difference between target and prediction) is taken with respect to the output(s) of the node(s) in the output layer which is then multiplied by the derivative of these node outputs with respect to the individual node parameters and then this is used to update the parameters for these node(s) according to 3.4. Then, to update the parameters in the layer before the output layer, the derivative of the loss function is taken with respect to the output layer node outputs and multiplied by the derivative of these outputs with respect to the outputs in the previous layer and then again multiplied by the derivative of these outputs with respect to the respective node parameters, which is then updated. So it goes on, for each layer and each node of the network. In other words, the chain rule is used systematically in many turns while *propagating backwards* through the network, updating all network parameters. This is called backpropagation and it is a cornerstone of all neural network training.

To illustrate, the rule for updating weight  $j$  for node  $k$  in the output layer  $N$  is given as,

$$w_{j,k} \leftarrow w_{j,k} - \alpha \times \frac{\partial a_k^N}{\partial w_{j,k}} \times \Delta_k \quad (3.5)$$

where

$$\Delta_k = \frac{\partial Loss}{\partial a_k^N} \quad (3.6)$$

Now we want to propagate the modified error  $\Delta_k$  backwards to all nodes with connection to node  $k$ . The intuition is that each node in connection with node  $k$  is responsible for some of the loss, proportional to the weight. Thus, the error with respect to node  $j$  in layer  $N-1$  is:

$$\Delta_j = \sum_k \frac{\partial a_k^N}{\partial a_j^{N-1}} \Delta_k \quad (3.7)$$

This makes the update of weight  $i$  for the last hidden layer equal to the output layer updates,

$$w_{i,j} \leftarrow w_{i,j} - \alpha \times \frac{\partial a_j^{N-1}}{\partial w_{i,j}} \Delta_j \quad (3.8)$$

We will not go further details of the mathematical explanation and derivation of backpropagation here.

---

### 3.1.2 Implementation - software packages

Following the accelerating growth and popularity of machine learning and in particular the use of neural networks, various software packages have been developed to streamline the implementation of neural networks. TensorFlow and PyTorch are the two most widely recognized, maintained and practically used such tools. We have used PyTorch for the work developing our machine learning model.

#### PyTorch

PyTorch is an open source machine learning library based on the Torch library and provides a wide range of algorithm tools for deep learning. It is primarily developed by Facebook's AI Research lab (FAIR) and is a free and open-source software. In the implementation of our machine learning model, Python is used as interface and programming language.

Pytorch provides several high-level features which profoundly ease the implementation of neural networks. First, Pytorch has a Tensor class to store and operate on homogenous multidimensional rectangular arrays of numbers. Tensors can be seen as NumPy arrays, but they can also be operated on GPUs which gives them a computational advantage. Second, PyTorch provides a framework (syntax and fast, underlying code) for defining what they call computational graphs, which in general means any function (neural network or other) which produces an output from an input tensor. With a focus on neural network, the implementation has been streamlined for deciding on e.g. the number of nodes, layers and types of layers in the network. Third, Pytorch provides a method of automatic differentiation which they call Autograd. When applying the computational graph (the function), a recorder records what operations have been performed on each tensor element, and then it applies backpropagation (described above) to calculate the gradient of the output with respect to the parameters of the function. Finally, PyTorch has streamlined the training of neural networks by providing readily available, optimized implementations of standard algorithm building blocks such as loss functions, parameter optimizers and activation functions.

## 3.2 Markov decision processes

A Markov Decision Process (MDP) is a discrete-time stochastic control process. This is a mathematical framework to model decision making where outcomes are partly random and partly controlled by the decision maker. MDPs consist of the following:

- All possible states of the environment  $S$ , called the state space
- All possible actions in each state  $A(s)$ , called the action space
- A transition model  $P(s' | s, a)$  that provide the probability of reaching state  $s'$  when performing action  $a$  from state  $s$
- The direct reward (or cost) received by the agent in state  $s$  by doing action  $a$ ,  $R(s,a)$

In a MDP, an agent "moves through" the environment, observing the state of the environment, by making a decision at each discrete time point, which in turn sends the agent to a new state at the next time point.

Transition between states is assumed to obey the *Markov property*, which means that transition to the next state is only dependent on the current state. Thus, state transition is independent of how we got to the current state.

The state space, which is the environment in the case of RL, can be partly or fully observable to the decision maker (agent) depending on how much of the state space the agent can observe.

---

Both state spaces and action spaces can be made up of continuous or discrete variables, or both. They can also be one-dimensional or multi-dimensional. The problem can also be both infinite and finite (i.e. episodic).

### 3.2.1 Profit maximization / problem formulation

The *policy* of an agent is the agent's "guide" to making decisions. Thus, the policy is basically a mapping from a state in the state space  $S_t$  to an action in the action space. The policy might be deterministic, usually denoted  $\mu_{s_t}$  and mapping each state to a specific action, or stochastic, usually denoted as  $\pi(a_t|s_t)$  and giving a probability for making a specific action. After making an action  $a$ , the agent receives an immediate reward  $r_t$  and is "sent" to a next state  $s_{t+1}$ .

As the agent "plays the game", it receives rewards in each state, accumulating a total reward over time, which is often discounted by a discount factor:

$$G(\tau) = \sum_{t=0}^{\infty} \gamma^t R(s_t, a_t) \quad (3.9)$$

where  $\tau$  is a trajectory, i.e. a sequence of states and actions

$$\tau = (s_0, a_0, s_1, a_1, \dots) \quad (3.10)$$

Over time the agent's goal is to find the policy which maximizes the cumulative reward it receives given a starting state. In other words, the objective function of the overall problem (infinite horizon) is

$$\max_{a_t \in A(s_t)} \mathbb{E} \left\{ \sum_{t=0}^{\infty} \gamma^t R(s_t, a_t) \right\} \quad (3.11)$$

### 3.2.2 The Bellman equation - the dynamic programming principle of optimality

The problem of maximizing the cumulative reward and finding the optimal action in each state and time step is often computationally intractable and highly complex. Richard Bellman, known as the father of dynamic programming, therefore came up with the idea of breaking the problem down into one subproblem for each time step in order to find the overall optimal policy. The reason this is possible is because of the Bellman principle of optimality. This states that the optimal path has the property that whatever the previous states and decisions taken for the initial period, the decisions chosen for the remaining period must be optimal with respect to the remaining problem. Directly following from this is that given a certain state and time step we are in, if we assume we know the optimal *remaining* path (and hence optimal value, i.e. cumulative discounted reward) given each state we might end up in in the next time step, we must simply now make the decision which gives the highest value of immediate reward now *plus* the (discounted) value of the next state. This led to the famous Bellman equation:

$$V(s) = \max_{a \in A(s)} R(s, a) + \gamma \sum_{s'} P(s' | s, a) V(s') \quad (3.12)$$

where  $V(s)$  is the optimal (/true) value of being in state  $s$ .

The name dynamic programming comes from this iterative nature of the problem breakdown, which is such that by finding the optimal value of a state, we can use the Bellman equation to find the optimal value in the state preceding that one.

---

### 3.2.3 Solution strategies

Dynamic programming and reinforcement learning is in general concerned with two types of problems/mechanisms: prediction and control. Prediction is about determining the true value of a given policy, and control is about improving a policy. Both are key to dynamic programming methods, which is about using value functions to organize and structure the search for good policies. Based on these two mechanisms, there are two main solution strategies to solving MDPs which both start from the Bellman equation and iteratively improve the policy of the agent. These are called value iteration and policy iteration, and they partly follow the same logic. The methods as explained here are simplified solution strategies only applicable to small state spaces and not directly what is used in most advanced reinforcement learning models, but they illustrate principles which substantiate the whole ADP/RL field. (There is also a third strategy, the linear programming approach to dynamic programming, where the value of a state is found by solving a linear program, but this is not relevant for this paper and is generally disregarded for large-scale problems).

#### Policy iteration

Policy iteration is about choosing a policy  $\pi$ , estimating the true value  $V^\pi$  of this policy by performing a (sufficient) number of iterations of policy evaluation (prediction), then using the policy improvement theorem to find a new, improved policy based on the estimated value and then repeating this until convergence. In other words, repeatedly switching from policy evaluation to policy improvement. The policy evaluation involves looping over each state  $s$  several times (again, we are here talking classical dynamic programming for small state space problems for illustrating the principles) and estimating the value of the policy  $\pi$  using the following update rule:

$$\begin{aligned} v_{k+1}^\pi(s) &= \mathbb{E}_\pi[R_{t+1} + \gamma v_k(S_{t+1}) \mid S_t = s] \\ &= \sum_a \pi(a \mid s) \sum_{s'} p(s' \mid s, a) [R(s, a) + \gamma v_k^\pi(s')] \end{aligned} \quad (3.13)$$

Then, after this has converged (close enough) to the true value, i.e.  $|v_k^\pi - v_{k+1}^\pi| < \Delta$ , we go over to the policy improvement step by choosing a new policy  $\pi'$ , such that

$$\pi' = \arg \max_a \sum_{s'} p(s' \mid s, a) [R(s, a) + \gamma v_k^\pi(s')] \quad (3.14)$$

The key to policy iteration is in the policy improvement theorem which states the "power of the max operator" in that any policy which is better than the current value for even a single state while giving the same value for all other states, is a better policy than the previous policy. Policy improvement is in other words about making a new, greedy policy with respect to the newly calculated value of the current policy. While basic policy iteration offers convergence guarantees, it has the drawback of each iteration involving a policy evaluation, which in itself may require many sweeps of the state space which may only in the limit converge to the true value  $V^\pi$ . Hence, policy iteration is often slow to converge.

#### Value iteration

Value iteration is actually a special case of policy iteration where one truncates the policy evaluation already after one time step (hence does not wait for convergence toward  $V^\pi$ ) and uses Bellman's optimality equation and the max operator directly in each step. In other words, both a policy evaluation and a policy improvement is done simultaneously in each step. One then loops over each state a sufficient number of times until the update is satisfyingly small using the following update rule:

$$\begin{aligned} v_{k+1}^* &= \max_a \mathbb{E}[R_{t+1}(s, a) + \gamma v_k(S_{t+1}) \mid S_t = s] \\ &= \max_a \sum_{s'} p(s' \mid s, a) [R(s, a) + \gamma v_k^*(s')] \end{aligned} \quad (3.15)$$

---

Often more used than value iteration is the term generalized policy iteration (GPI), which refers to the more general idea of letting policy evaluation and policy improvement processes interact, often in large state space settings where approximate algorithms are used, but we will not go deeper into this term here as it is not strictly important what we call it.

Of these two, value iteration remains the most widely used method in both ADP and RL, both because it is the simplest to implement and because it often is the most natural way of solving many problems. Most of the work in ADP and RL has therefore focused on this, and so will the remainder of this thesis do. The value iteration algorithm forms one of the most basic elements of most ADP/RL methods.

### 3.3 Reinforcement learning - Approximate dynamic programming

Solving large scale stochastic problems by exact dynamic programming, i.e. recursively solving the Bellman equation backwards in time, is usually intractable because of the well-known "curse of dimensionality" (W. Powell (2011)). The curse of dimensionality is three-fold. First, one has to loop through each state of the state space, which for a multidimensional state space quickly becomes computationally intractable. Second, one has to calculate the expectation of the values of the next states, which is time-consuming when the number of random variables grows. Finally, the action space may be so large that deciding on a specific action means evaluating virtually infinitely many actions.

The solution to resolving the curse of dimensionality in order to be able to apply dynamic programming principles to large scale MDPs is, basically, in *approximating* the value function instead of computing it exactly. By 1) stepping forward in time, instead of backwards, and by 2) using Monte Carlo sampling to generate random information and 3) using so-called post-decision state variables, one omits the calculation of the expectation in each step. Further, by choosing some sort of a value function representation which is more *compact* than simply a look-up table value for each single state in the state space, one is able to generalize learnings of the value of one state to other states as well. This latter element is key, because by doing this one no longer has to loop through every state in the state space. Instead, one can iteratively try to improve the approximation of the optimal value function going through the MDP, and hence iteratively make better and better decisions.

Both the operations research and stochastic programming communities and the machine learning community have developed these ideas, partly together, partly separately. While the operations research and stochastic programming communities named the area *approximate dynamic programming* (ADP) and the machine learning community named it *reinforcement learning* (RL), it is important to understand how these two names refer to the same types of solution strategies to the same types of problems (at least principally) (W. Powell (ibid.)). (Neuro-dynamic programming is another name used by some communities). The difference between ADP and RL is therefore not in the algorithms as such, but in 1) the terminology and notations used, 2) the "focus area" of the algorithmic research, and 3) the typical applications of the methods. The operations research community often uses ADP as an alternative solution strategy to large-scale industrial mathematical programming problems such as inventory routing problems, production scheduling problems, energy trading problems or even stock trading problems. The control theory community also uses ADP for typical process control problems such as energy flow problems. The most famous applications of RL in the RL community (as a subdivision of the larger computer science community), however, are in playing video games. The AlphaGo model and the RL model playing the game of Dota are extremely impressive such examples, but the RL community has also, like the ADP community, focused on real-world problems like robot control or business management. The RL community otherwise tend to focus more on the extensive use of neural networks to e.g. do visual recognition of states. The ADP community tends to focus more on finding high quality heuristic solutions to optimization problems, while RL practitioners often work on automating complex tasks. Overall, however, there is no principal difference between ADP and RL. This is important

---

to stress, because we will use both names interchangeably throughout this thesis. While the authors definitely come from the operations research community and have a background primarily in mathematical programming, we experienced how, when deepdiving into ADP, one quickly moves into "artificial intelligence land" and realizes that the merger of neural networks with ADP (i.e. RL) is extremely valuable. Throughout the work developing our ADP/RL model we have therefore found ourselves researching advanced machine learning concepts related to neural networks while also maintaining an operations research perspective on our problem. We will therefore deliberately use both names in conjunction (ADP/RL). Sometimes we also use the term *deep* RL, which refers specifically to RL models which apply deep neural networks. We will now move on to discuss important ADP/RL concepts as background theory for the rest of the thesis.

As already mentioned, approximate dynamic programming and reinforcement learning focuses on how agents can learn to maximize some cumulative reward. By "learning" in this respect, we mean "playing the game" by being out in the environment and then learning better approximations of the value of being in a state by observing the immediate rewards the agent receives. The agent is often initialized without any information. Each decision is made by considering the Bellman optimality criterion, which means that it is the estimate of the value of the next states that indirectly determines what actions the agent will take. The optimal policy and the optimal value function are therefore by definition linked together. Over time, the estimates of the value function (hopefully) improve so that they approximate the optimal value function, which in turn gives the optimal policy. The direct use of the Bellman equation for each decision, as well as the fact that the policy (i.e. decisionmaking) is implicitly given by the current value function approximation, corresponds to the general class of policies called *value function approximation policies* by W. B. Powell (2016) in his overarching framework for sequential decisionmaking problems. We specify that this policy class is what we will typically be referring to when we use the term ADP/RL in this paper, as this is the by far most common type of ADP/RL. RL can be divided into Prediction and Control. Prediction is when RL is used to learn a value function for a given policy. On the other hand, Control is where RL is used to learn a policy that maximizes the reward which may include prediction. As the latter is mostly relevant for our thesis, reinforcement learning will refer to the concept of Control from now on. This is also most common in other literature (Silver (2015)).

The RL agent's task is to decide what actions to take in the different states it can enter. This is done by observing how the environment responds to different actions in terms of state change and received reward. As the name *reinforcement* learning tells, the agent tries to improve its performance through iterations where each reward signal the agent receives acts as a reinforcement.

In RL, reward is the measurement of how good or bad a state or action is. However, the agent can also take into account future rewards so that it chooses the best actions to receive highest possible cumulative reward. An important note to rewards is that the reward might be intermittent and delayed. Take a board game as an example. The agent takes several actions throughout the game, and receives some reward for winning the game in the end. However, it is difficult for the agent to learn which actions was actually important to take the win. This is known as the credit assignment problem.

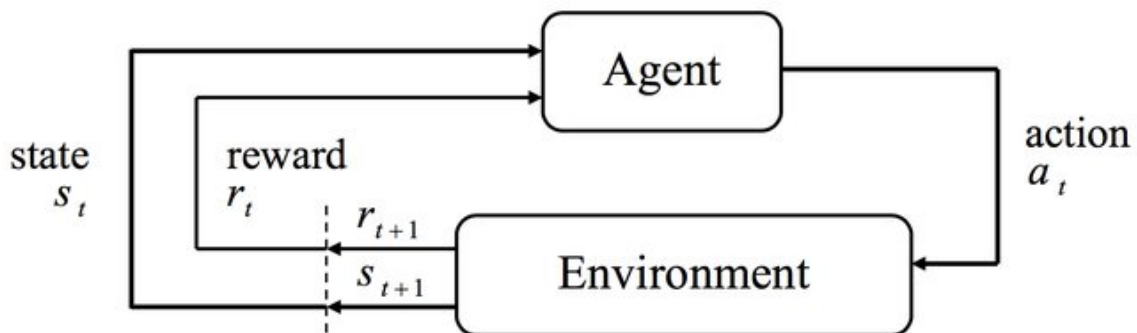


Figure 3.3: Illustration of interaction between agent and environment

---

The ADP/RL algorithm (for an infinite horizon problem) logic goes as follows: Subject to the chosen exploration vs exploitation strategy, the agent is out in the environment, "playing the game", i.e. performing as good as it knows how to by choosing decisions which maximize the Bellman equation. In each iteration, it maintains a value function (given by the function approximation parameters) which gives a value estimate for each state, which in turn determines the actions. After a certain amount of steps, (each single step in TD(0)-learning), the agent can now learn from its experiences, because now the rewards the agent has received since it last visited a state gives new information regarding the true optimal value of that state. A new, hopefully more correct target for the respective state can then be calculated, where the target is typically based on both the rewards experienced and a bootstrapping of the current value function estimate. If the agent has behaved *better* (i.e. closer to optimally) than it did in the previous iteration, the rewards collected after the visit of the respective state should better approximate the true value of that state. A loss function is then applied on the difference between the current value estimate of that respective state and the newly calculated target for that state. The basic idea is then to differentiate this loss function with respect to the function parameters and then adjust each parameter a small step in the direction of the negative gradient in order to over time minimize the loss. This updates the valuefunction for next iteration, hopefully making the agent make even better choices going forward.

### 3.3.1 Value function

#### Types of value functions

As discussed, the value of following a policy  $\pi$  is given as

$$V^\pi(s) = \mathbf{E}[G \mid s_0 = s, a_t = \pi(s_t)] \quad (3.16)$$

where  $G$  is defined as in 3.2.1. The optimal value of a state, regardless of policy, is given by equation 3.12, which is the Bellman equation. That means the value function estimates how good a given state is. We can also estimate how good a given state-action pair is, i.e. the value of being in a given state and taking a given action and subsequently either following a given policy or making the optimal action. This type of reinforcement learning is called Q-learning, where the name comes from the Q-function:

$$Q^\pi(s, a) = \mathbf{E}[G \mid s_0 = s, a_0 = a, a_t = \pi(s_t), t \in [1, \infty]] \quad (3.17)$$

The corresponding optimal Q-value is then defined as

$$Q^*(s, a) = R(s, a) + \gamma \sum_{s'} P(s' \mid s, a) V(s') \quad (3.18)$$

Q-learning and Q-functions have been considered less relevant for our problem and will therefore not be discussed further in this thesis, but they are a large subtopic in RL.

There are a number of different ways to calculate, estimate and save the value of different states, depending on the art and classification of the problem.

The traditional way to save state values is in the form of a look-up table, which stores a single scalar value either for each state or for each aggregated state. In exact dynamic programming, one maintains one value for each state, the number of which quickly explodes in large-scale problems. Even using aggregated state spaces or feature spaces, which are powerful techniques in ADP-context, multiple dimensions quickly makes the number of scalar values to save intractable.

The idea has therefore been to look for more *compact* representations of the state values. Such representations can be divided into parametric and non-parametric functions.

Non-parametric functions include for example K-nearest neighbours and are functions of a state which typically try to value states by looking at the relationship between states. This is generally highly problem-specific and very little used in practice and will therefore not be discussed further here.



---

The most common in terms of RL learning is parametric functions, i.e. some sort of a classical mathematical function, smooth or otherwise, which has certain tuneable parameters. This can for example be a neural network, or a custom value function like a polynomial. Such functions take the state as input, either directly in the form of the state variables or in the form of certain state features, and calculates the value of the state by performing mathematical operations on the parameters and the input variables. The advantages of parametric functions lie both in their wide applicability, their very compact representation of the state space values (reducing the amount of values to keep track of from a virtually infinite number of states to a handful of function parameters) and in the extensively developed theories and practices on regressions, which aims to adjust the function parameters in order to approximate some target function.

### 3.3.2 Exploration versus exploitation

One important dilemma in RL is the tradeoff between exploration and exploitation. Exploration refers to the exploration of the state space by visiting unseen states, while exploitation refers to following the current best policy to receive as much reward as possible. The latter choice is called *greedy*. As an example, with a low degree of exploration, the agent might get stuck in a path without knowing if other paths is *better* even though the agent thinks it has a good policy. We therefore want to make sure the agent explores other states as well, to not get stuck in a lock minimum. That means the agent has to give up some short-term reward in order to find the best long term policy (Silver (2015)).

An ADP/RL model therefore needs to be set up with some sort of exploration strategy. There has been developed several advanced exploration strategies, all of which requiring considerable application specific tuning. Exploration vs exploitation remains one of the most fundamental and interesting challenges to ADP and RL (alongside e.g. value function design and training method) and is therefore an area of considerable research.

The most natural and perhaps most intuitive way to explore a state space is by introducing randomness, i.e. by making random decisions. The most common exploration strategy is therefore the  $\epsilon$ -*greedy* strategy (W. Powell (2011)).  $\epsilon$  is chosen as a random number between zero and one. The ADP/RL agent then chooses a random action with probability  $\epsilon$  and a greedy action with probability  $(1 - \epsilon)$ . This strategy is seen in almost any textbook introduction to ADP or RL, primarily because it is by far the simplest to implement.

Silver (2015) discusses several principles which might be used for exploration. Another easy to implement strategy mentioned both there and in W. Powell (2011) is *optimistic initialization*. This basically means initializing the value function approximation with a (very) high value for all states. This way, the agent will itself want to explore all of these high values. While  $\epsilon$ -*greedy* and optimistic initialization are often effective and part of many successful RL applications, they might not be the best alternatives, and most advanced applications require a cleverer strategy. One more advanced strategy is curiosity driven exploration.

#### Curiosity driven exploration

Curiosity-driven exploration is a collective term for a set of more advanced exploration strategies often encountered in complex RL applications. Silver (2015) calls this "optimism in the face of uncertainty". The basic idea is to add an intrinsic element of "motivation" to the reward function which rewards the agent's curiosity regarding less frequently visited parts of the state space.

While there has been many research efforts on the topic of curiosity-driven exploration, Pathak et al. (2017) has been credited with advancing the field significantly. Their approach is one of the most advanced types of curiosity driven exploration which (simplified) employs a separate model which trains to predict the outcome of the agent's actions and which then gives the agent a reward for choosing actions which have a high estimation error, the logic being that the agent is bad at predicting the dynamics of states it has visited few times. This goes under the subtopic of prediction-based exploration.

---

Another subtopic of curiosity driven exploration considered more relevant for the problem in this thesis, is count-based exploration. Here the basic (simplified) idea is to count how many times a part of the state space has been visited and give the agent an intrinsic reward for visiting novel states. Several advanced variations exist. For example, "counting" can be done both by a density model or via a hashing function. One of the more popular approaches is called Upper Confidence Bound (UCB). A variant of this is used by Google DeepMind's AlphaGo model (Silver et al. (2017)), Google DeepMind being one of the world leading communities in the development of RL. UCB favors actions which have a high potential for being optimal, by being *optimistic about uncertainty*. The agent then maximises  $Q_t(s, a) + U_t(a)$ , where Q is value of the current estimate of the Bellman equation and U is an additional function which is reversely proportional to the number of times action a has been chosen (Silver (2015)). The idea is to design U so that the true optimal value is below (Q+U). Memory-based exploration builds upon the abovementioned strategies, especially the count-based exploration strategies, by saving both a lifetime and an episodic memory (in a way which keeps the memory space required acceptably low). These methods are both applicable and promising for our problem and will therefore be discussed more in-depth later.

There are also other exploration strategies (also outside the curiosity driven umbrella) that will not be discussed further here, such as Boltzmann exploration and Thompson sampling, both using chance and randomness to ensure exploration.

### 3.3.3 On-policy vs off-policy learning

As described, the RL policy refers to a function (in the most general sense) mapping from a state to an action. However, in a reinforcement learning algorithm, we can actually distinguish between two policies. One is the policy used to generate the behaviour and the data which forms the basis for training, called the behaviour policy or the sampling policy (we prefer sampling policy, which is often used in stochastic problems). The other is the policy we are attempting to evaluate and improve, called the target policy or the learning policy. The concepts of on-policy vs off-policy learning are closely related to that of exploration vs exploitation, because they represent two different strategies to choose when deciding on an exploration strategy.

In on-policy learning, these policies coincide. This means that the updates to the value estimates are dependent on the generated trajectory and hence the sampling policy. To ensure both exploration and convergence towards optimality in on-policy learning, it is often required that  $\epsilon$  decreases with time (in the case of  $\epsilon$ -greedy policies, which is the most typical use-case).

In off-policy learning, the sampling policy and the learning policy are two distinct policies. On-policy learning is hence a special case of off-policy learning. Off-policy learning often uses a concept called importance sampling, which effectively enables the updating of the learning policy with respect to the optimal value while still generating behaviour with a sampling policy. Off-policy learning is often more noisy, slower to converge and more unstable than on-policy learning. Still, off-policy learning has been devoted more research in the later years because of its greater potential and applicability to a wider array of problems with the use of more advanced and more problem-specific exploration strategies and hence sampling policies.

We point out a common misunderstanding. On-policy and off-policy learning is not the same as online and offline learning. Online vs offline learning refers to whether value function estimate updates are performed as the RL agent is playing along and gathering data, so that it is receiving continuous updates, or whether it is done after a given amount of data has been gathered.

### 3.3.4 Model-free vs model-based RL

RL algorithms can be either model-based or model-free. The distinction between these two are whether the agent has access to a complete model of the environment or not. If the agent has not access to a model of the environment, it has to learn the model through interaction, which is the model-free approach. In this setting, the model of the environment refers to the model telling the agent explicitly what will be the next state and what will be the reward given a state and a

---

decision, so that the agent can use this when making a decision. Dynamic programming with use of the Bellman equation is therefore model-based. AlphaZero is another example of model-based RL, using models of the rules of the board games chess, shogi and go to tell the agent what will be the immediate(!) consequences of it's actions (Silver et al. (2017)). Q-learning and SARSA are examples of model-free algorithms where the model is not explicitly told what will be the reward and next state given a state and decision, but rather has to learn this on its own as part of the learning.

One major advantage of model based learning is that the agent can look ahead to consider different outcomes. This yields a higher sample efficiency than model-free algorithms (Stray (2019)). As model-free algorithms often do not have the same sample efficiency, they can be easier to implement and tune. At this time, model-free methods are more popular and have been more tested than model-based methods (Achiam (n.d.)). Whether or not the algorithm is model-free or model-based often depends entirely on the specific problem being approached and on what is the goal of the algorithm. The focus of this thesis is a model-based RL algorithm.

### 3.3.5 Temporal difference learning

RL methods are categorized into subclasses. One such subclass is Monte Carlo-methods, which typically refers to RL agents learning in finite environments where each episode has a definitive ending (such as winning or losing a video game) and where a key element often is that the RL agent's *target* for a given state (i.e. the current iteration's value estimate for that particular state) is a sum of the episode return, given by all the remaining rewards earned in the episode, subsequent of the state. We will not go further into Monte Carlo methods here, as they are not strictly relevant because they only apply to episodic problems (salmon farming is in theory an infinite horizon problem, although we often use the term episode in this paper to denote the time the RL agent gathers data before each training/value estimate update). We do however mention it because the notion that the target for the estimate should depend on all downstream states and rewards is important.

Temporal Difference (TD) learning is another subclass which is often mentioned as the counterpart to Monte Carlo-methods. This method can be model-free and hence learn directly from experiences, and also employs statistical bootstrapping, which means that estimates are updated using the already learned estimate of the value function, without knowing the actual outcome (Stray (2019)).

---

**Algorithm 1:** Tabular TD(0) for estimating  $V^\pi$

---

Initialize  $V(s)$  arbitrarily,  $\pi$  to the policy to be evaluated;

**for** *for each episode* **do**

    Initialize  $s$ ;

**for** *each step of episode* **do**

$a \leftarrow$  action given by  $\pi$  for  $s$ ;

        Take action  $a$ ; observe reward,  $R$ , and next state,  $s'$ ;

$V(s) \leftarrow V(s) + \alpha(R + \gamma V(s') - V(s))$ ;

$s \leftarrow s'$ ;

**end**

**end**

---

Here,  $\alpha$  is the learning rate that adjust the speed of convergence, and  $\gamma$  is the discount factor.  $R + \gamma V(s')$  is the TD target where the last term is the bootstrapping term, while  $\delta_t = R_t + \gamma V(s'_{t+1}) - V(s_t)$  is known as the temporal difference error for time step  $t$ .

Important to this thesis is the concept of  $n$ -step bootstrapping, often called TD( $n$ ) learning. The idea here is to not let the update depend only on the next state but on several of the subsequent states. Then the temporal difference target, which we call the  $n$ -step return  $G_t^{(n)}$  is extended to,



---

An important insight is that Monte Carlo-methods, which use the full return  $G_t$  as target have a low bias but high variance as an estimate of the true value of the state. The high variance comes from the fact that the target depends on very many downstream actions and transitions. TD(0), on the other end of the scale, has a high bias, because of the statistical bootstrapping using a (not yet correct) estimate of the true value, but also a much lower variance because it only depends on one downstream action and transition.  $TD(\lambda)$  and  $TD(n)$  therefore act as adjustable middle grounds in the bias-variance trade-off in RL (Silver (2015)). Monte-Carlo methods typically have good convergence properties and are intuitive. TD learning convergence is often more sensitive, but learning is in return also often more efficient. This trade-off can have large effects on a RL algorithm.

### 3.3.6 Algorithm tuning - hyper parameters

RL is famous for being a notoriously challenging algorithm in terms of convergence towards optimality. This is amongst others due to the estimation of moving targets and the approximation of a function we do not know the shape of and only get noisy signals from. This was discussed with NTNU professor and experienced practitioner of RL Keith Downing on a meeting on 9th April, 2021.

RL implementations for complex real-world problems such as salmon production scheduling often requires very much tuning of algorithmic hyperparameters. Typical hyperparameters in an ADP/RL algorithm include, but are not limited to, the following:

- Learning rate: This is related to the valuefunction optimizer/training algorithm, and decides how large change is applied to the valuefunction parameters/weights in each iteration. This is one of the first places to start tuning when tuning an ADP/RL algorithm.
- Policy-related parameters such as " $\epsilon$ " percentage share of total actions which are random, exploring actions or magnitude of exploration bonuses in case of a bonus-based exploration strategy.
- Function training algorithm, meaning how often and with what targets to adjust the parameters of the valuefunction.
- Batch size, meaning how many experiences to select for training on in each iteration.
- Discount rate, used in calculating both the targets in the training algorithm and for use in Bellman's optimality criterion when choosing decisions in each time step. Note that this is not necessarily the same discount rate as the financial discount rate discussed in chapter 2.

When practically implementing an ADP/RL algorithm, all of the above parameters, plus several other, more application specific parameters will have to be tuned to ensure satisfying performance/convergence of the algorithm. As ADP/RL typically do not offer general convergence guarantees, convergence is strongly dependent on these parameters. In an ADP/RL setting, theoretical "soundness" can therefore be very unrelated to actual practical performance of a model. It is also very hard or even impossible to "imagine" or reason what the hyperparameters should be in large scale problems, mainly because the whole concept of the gradients of the loss function between the valuefunction predictions and a (moving!) target with respect to the (many!) parameters in the valuefunction and how these change and how these affect the valuefunction updates and the valuefunction shape is immensely complex.

# Chapter 4

## Relevant literature

This chapter is a review of literature and research relevant to this thesis. The structure of the chapter reflects the thesis' conjoining of several fields. First, we review literature on commodity price modelling in general and on salmon price modelling in particular. Second, we review a few ADP/RL applications which might be deemed relevant to this thesis. Third, we review literature relevant to salmon production scheduling optimization. Finally, this thesis is placed in the context of the reviewed literature and the contribution of this thesis is presented.

The search for literature has been done using Google Scholar and the NTNU university library at [www.oria.no](http://www.oria.no).

### 4.1 The price of Atlantic salmon

In this section we first review important papers which contribute to the understanding of the salmon price. We then move on to discuss different researchers' efforts to model and forecast the salmon price.

Commodities are widely traded, relatively homogeneous raw materials such as sugar, grains, iron, crude oil and salmon. Commodities need to be stored, which typically has a storage cost. In salmon production, products may be stored in fresh condition for a limited time, but considerably longer in frozen or smoked condition. However, storage is still limited compared to many other commodities. Pindyck (1990) found that inventories may serve to smooth production during periods of low or normal prices, but during periods of temporarily high prices inventories play a more important role in production facilitation, delivery scheduling and avoiding stock-outs. Deaton and Laroque (1992) also show that the prices behave differently if there is a possibility to store the commodity. Thus, the possibility to store the salmon may smooth prices somewhat.

Although possibility in theory would smooth the price development, salmon prices are very volatile where supply and demand are the strongest drivers, as explained in chapter 2.5.1. Bjørndal, Knapp, and Lem (2003) study the global supply and demand for salmon, which contribute to an understanding of the drivers behind salmon prices. In later years, the long term price has been growing. This implies that growth in market demands might have been stronger than the growth in supply, and/or that production costs have been increasing. While Bjørndal, Knapp, and Lem (*ibid.*) studies the global distribution, Asheim et al. (2011) investigated the historical development of salmon supply to show that the price of farmed salmon has limited effect on the quantity of supply, because of a highly inelastic supply in short-run. Because of a production time of approximately three years (including smolt production etc.), producers struggle to meet short term changes in demand. They found that farms' total biomass and seasonal factors are the main determinants of shifts in salmon supply in the short term.

Oglen (2013) found an increasing trend in price volatility. This effect is explained by tighter

---

supply/demand conditions and lower short-run supply elasticity, and he points out possible explanations with basis in 1) strong demand for salmon from Norway, 2) increased capacity utilization as a response to favorable demand conditions and the resulting effects of occasionally binding MAB restrictions (introduced in 2005), 3) the increasing use of bilateral contracts over spot trading and 4) the overall strong prices for relevant commodities globally increasing production costs and contributing to strong demand for salmon. Consequently, the forward-market is pointed out as one possible explanation to higher price volatility because they tie up some of the supply in the short run. Thus, the short run supply becomes even more inelastic with the possibility to trade in the forward-market. Bloznelis (2016) identified two periods of different price volatility regimes, before and after 2006. Both volatility and conditional correlations increased from 1996–2005 to 2007–2013, and return dynamics became more homogeneous across weight classes. This substantiates the findings of Oglend (2013).

Instead of exploring general price trends or volatility, Asche and Guttormsen (2001) show that price patterns between different weight classes exists, using prices from 1992-1998. Producing salmon is a biological production process. Hence, productional constraints imply that different fish farmers are likely to have a similar distribution of different sizes of fish over time. If there are no perfect substitutes for the different sizes of fish in the short run, the production cycle can cause different relative prices between the different sizes over the year.

The salmon price has a seasonal pattern, that Asche and Bjorndal (2011) discuss. They find that the most significant influence on seasonality is from water temperature. Temperature has big implications for salmon growth, where lower temperature yields lower growth. Both setting out smolt and harvesting at cold water is unfavorable due to growth conditions. However, higher temperature also yields higher risk of diseases and algae blooms. Sudden mass deaths might have the biggest influence on large fluctuations in prices when they occur.

Fish pool opened in 2006, which made spot and forward prices on Atlantic salmon publicly available. This also made studies on risk related to prices easier. Asche, Misund, and Øglend (2016) investigate to what extent forward prices of salmon can provide unbiased estimators of spot prices. They found that forwards provide a good estimate of future spot price. However, they also found that the "discover prices effect" that often exist in mature forward markets is limited due to low volumes of trading. That means we cannot fully say that the forward price is the markets expectation of the spot price at maturity of the forward. On the contrary, they found that actually the spot price plays a "leadership role" for the forward price. Ankamah-Yeboah, M. Nielsen, and R. Nielsen (2017), using a more recent data set, found a higher degree of maturity of the salmon futures market, and that this trend was likely increasing. This goes against Asche, Misund, and Øglend (2016) and suggests that analyses using forward prices as estimates of future spot prices are indeed worthwhile, at least for shorter term contracts.

Producers who combine price forecasts with production planning will have a potential advantage compared to other farmers. Guttormsen (1999) test the methods of Classical Additive Decomposition (CAD), Holt Winters Exponential Smoothing (HW), Auto Regressive Moving Average (ARMA), Vector Auto Regression (VAR) as well as two different naive models for salmon price modelling. The results do not provide any obvious winner, but several methods provide good predictions of price changes. Thus, he did show that participants in the salmon industry can produce forecasts that are of good quality relatively easily.

Another approach to price series modelling is based on the framework of Schwartz (1997). They showed that knowledge about the forward price curve are essential for making optimal decisions. This work was based on oil, but has later been applied to other commodities. The work of Schwartz (1997) was further extended by Schwartz and Smith (2000), where dynamics of the spot price is modelled as the sum of a short term factor and a long term factor. The application of the latter model to salmon prices is confirmed by Ewald and R. Ouyang (2017) that built a model based on the Schwartz-Smith two factor model and seasonality represented by a truncated Fourier series. In another paper Ewald, R. Ouyang, and Siu (2017) again use the same dynamics as Schwartz-Smith with a real option approach to find optimal harvesting. Their approach can also be used to determine the value of leasing or owning a fish farm.

It is of interest to model forward prices as they provide price expectations for any given time, despite

---

the findings of Asche, Misund, and Øglend (2016), but in line with the findings of Ankamah-Yeboah, M. Nielsen, and R. Nielsen (2017). Monteiro (2020) suggest a novel semi-parametric structural model to compute continuous forward curves in the electricity market. The price of electricity show some of the same patterns as the salmon price in terms of volatility, although electricity has even more limited options for storage. Nevertheless, the basis of the model is a data set of historical forward prices. PCA analysis is used to explain a high percentage of the variance of forward prices at different maturities with a few factors. Then an AR(1)-GARCH(1,1) is used for modelling forward prices. Thus, a time series model is used to predict the spot development, while a parametric approach is used to model the forward curve for each time step in the spot price. Because such an approach has never been applied to price modelling of salmon, but still seems to be an increasingly popular method for other commodities, we adopt this framework of price modelling.

## 4.2 Relevant ADP/RL applications

This section reviews a few ADP/RL applications which both give examples of typical ADP/RL models and which might serve to give some inspiration for using the same methods on salmon production scheduling.

A long-standing goal of artificial intelligence has been to build an algorithm that learns, *tabula rasa*, superhuman proficiency in challenging domains. There are several very impressive RL applications where model performance becomes superior to humans. This is demonstrated by for an example Mnih et al. (2013). They first made a deep learning model that could learn control policies directly from high-dimensional sensory input using RL. The model is a variant of Q-learning trained neural network, whose input is raw pixels and output is a value function estimating future rewards. By applying the method to seven Atari 2600 games, with no adjustment of the architecture or learning algorithm they found the model to outperform all previous approaches on six of the games as well as beating a human expert on three of them. Another, perhaps the most famous, result is from when Silver et al. (2017) made the model called AlphaGo, which became the first program to defeat a world champion in the game of Go. The program was later developed into AlphaZero, which also masters both chess and shogi at superhuman levels. The tree search in AlphaGo evaluated positions and selected moves using deep neural networks. These neural networks were trained by supervised learning from human expert moves, and by reinforcement learning from self-play. The algorithm were based solely on reinforcement learning, without human data, guidance or domain knowledge beyond game rules.

An essential part of the usability of RL/ADP to optimization is to what degree it performs as good as traditional optimization approaches on small scale problems. Successful applications of ADP include transportation, finance, healthcare, energy, and supply chain management (e.g., Fang et al. (2013), Lei and Y. Ouyang (2017)). For example, Rivera and Mes (2017) considered the planning problem faced by Logistic Service Providers, i.e. transporting freights periodically, and they used a "basis function neural network" approach and the non-stationary least squares method. That means using a least squares regression with basis functions to approximate the value functions which can remarkably decrease the computational time/curse of dimensionality. Yin et al. (2016) considered a metro train rescheduling problem with uncertain time-variant passenger demands, and they utilized linear and separable basis functions as value function approximations.

Salas and W. B. Powell (2013) benchmark an approximate dynamic programming algorithm and find that it is capable of designing near-optimal control policies for time-dependent, finite-horizon energy storage problems, where wind supply, demand and electricity prices may evolve stochastically. The algorithm was able to design storage policies that are within 0.08% of optimality in deterministic comparisons and within 1.34% in stochastic ones on a single storage with different data input. They emphasize that the algorithm easily scales to a lot of devices since the size of the decision problem grows linearly with the number of devices. On the matter of solution time, Papageorgiou et al. (2015) study the application of ADP to a deterministic maritime inventory routing problem with a long planning horizon. They used a value function approximation that is a separable piecewise linear continuous function and introduced a multi-period look-ahead strategy.



---

On average, after 30 minutes of CPU time, the quality of the ADP solution was 92% compared to the best known solution, while local search with considering many periods simultaneously was at 84% and Gurobi at 72%. Thus, their work show that ADP solution reach good performance quicker than MIP solutions. Both of these studies show promising results in terms of producing good solutions faster than traditional optimization methods.

As earlier mentioned, neural networks can perform as powerful and universal approximators for nonlinear mapping (Khosravi et al. (2011)). Because of the wide applicability, neural networks have been extensively used in the ADP and reinforcement learning fields as a powerful and adaptable class of nonlinear forms of value function approximations (W. Powell (2011)). The literature review suggests that neural networks can play an important role when it is difficult to find suitable (linear) basis functions and there is no reasonable prediction of the nonlinear structures of the value function. However, these networks might need a lot of iterations to converge, and require careful tuning of hyper parameters to converge correctly.

## 4.3 Salmon production optimization

### 4.3.1 Relation to other operations research literature

The salmon production scheduling problem that is the focus of this master thesis has parallels to several other problem classes within operations research. First and foremost, it has similarities with a range of problems in other biological systems, which will be described in the next subsection. However, our problem can also be seen in relation to more general problem classes studied in the operations research community.

The salmon production scheduling problem can be seen as a type of manufacturing problem in the operations research literature. However, there is typically no demand to be met in salmon problems. Machine scheduling and jobshop scheduling problems as described by Graves (1981) and Fuchigami and Rangel (2018), with multiple jobs (fish batches to grow) and multiple, identical machines (fish tanks), can be relevant reference points. These types of combinatorial optimization problems are known to be hard, and solution methods prevalent in the research literature are dominated by heuristical methods. One important distinction away from these problems, is the biological growth element as well as the MAB restriction which is a sort of resource restriction that works "cross-machine".

Our problem also has parallels to certain inventory-related problems. The so-called warehousing problem, with buying and selling of products in each time-period, is another relevant problem to relate to, where dynamic programming is often suggested as a solution method. The distinction from this would be that in salmon production scheduling each "product" has to "wait a certain time in the warehouse" for the growth process before it can be sold.

Both hydropower and water reservoir management problems and gas storage problems have similar features to the salmon production scheduling problem and to the warehousing problem more generally. Both problem types also lend themselves naturally to sequential decision making algorithms, and involve the issue of determining the value of the resource, e.g. what is the value of a batch of fish or the water in the reservoir or the gas stored. Price dynamics are hence often essential in such problems. However, again, the biological growth is one of the additional complications of the salmon production scheduling problem relative to these other problem types.

### 4.3.2 Mathematical programming applied to biological systems

The most similar problems to the salmon production scheduling problem are found in other biological systems. The application of mathematical programming to biological systems is a very large and valuable field of research. When excluding fields such as biochemical systems and applications in medicine and only considering fields related to some sort of harvesting-for-profit systems, the search narrows somewhat. Large areas of research in this category are, amongst others, agricul-

---

ture/farm design or operations optimization, crop and livestock production optimization, animal feed optimization and plant or forest management optimization. Some of the difficulties typically encountered in modelling biological systems include the often great number of factors affecting growth (of an animal or plant), as well as non-linearities and in some cases the solution of differential equations. For example, growth is actually weight-dependent, which means one gets equations of the form  $newWeight = growth(oldWeight) * oldWeight$ , which are nonlinear, and, furthermore, growth is a result of a multitude of environmental factors affecting an entire population as well as genetic factors affecting each individual. Simplifications and linearizations therefore typically have to be made, at least in mathematical programming methods. In the following we will mention a few research efforts from other biological systems that have parallels to salmon production optimization.

One interesting area of research has been poultry production planning and scheduling. Both Taubenetto (1996) and Brevik et al. (2020) are examples of this. Brevik et al. (ibid.) address this on a high-level supply chain, deterministically, with rather simplified growth modelling of chicken and call it the Chicken Flock Sizing, Allocation and Scheduling problem. Their problem of allocating chicken eggs from broiler breeders to broiler farms and subsequently coordinating logistics for slaughtering from each farm after the poultry has grown as close as possible to a target weight and minimizing total costs and penalties doing so is solved by rolling horizon heuristics. This problem has several parallels to supply chain optimization for large traditional salmon producers, with smolt production facilities representing egg breeders, salmon cage locations representing farms and well boats representing slaughtering trucks. Taubenetto (1996) goes more specifically into the day-by-day production scheduling for each chicken plant, involving decisions such as selecting "grandparent chicken", deciding the flow of eggs to growers ("housing" the chicken), scheduling slaughtering processes based on live weight distributions of each flock and deciding on the number of products (wings, chest, thighs etc.) to produce. Here, a large decision-support tool has been developed, and a more advanced growth model has been applied in order to account for both day-to-day growth and weight distribution within each class. This is similar to the production scheduling of a single salmon production location with several cages, except for the added complexity of one chicken possibly becoming several different products (wings, thighs etc.), which is not the case in salmon farming. Another similar feature of the scheduling nature of both poultry production and salmon production (as well as other live animal breeding business) is the welfare-related restriction of different flocks of chicken (or batches of salmon) not being allowed to share a house (or cage/tank for salmon). In other words, there is an "all in, all out"-policy with washing of the facilities in between each cycle to prevent diseases spreading. One typical element considered in livestock breeding problems is the diminishing feed conversion ratio (FCR) as the animals grow larger (lower growth per feed input), which directly affects the harvesting scheduling problem. This is for the most part simplified in salmon research literature, where FCR is considered a constant. To our knowledge, this is because the FCR generally is less variable for atlantic salmon than for livestock.

Another interesting application area is forest management, which also has parallels to salmon production. Forest management optimization is a large research field, where the decisions typically are which areas of the forest to harvest at what time and in what order, with considerable setup costs and complicating constraints on harvesting. In this field, the focus has mostly been on harvesting scheduling and not multi-cycle planting and harvesting, which is natural due to the long life cycle of trees. One similarity to salmon production is the increased value gained from each tree from delaying harvest because of a larger diameter on each tree, such as the higher price for larger sales classes of fish. Furthermore, in the stochastic versions of the forest management problem, uncertainties are typically modelled in both, prices, tree growth and in physical damages (fires, pests etc.). This would be the equivalent to uncertainty in the salmon price, salmon growth and salmon mortality, which are the typical uncertainties in the salmon production problem. Lohmander (2007) writes about several mathematical programming solution methods for these stochastic forest management problems, mentioning both stochastic dynamic programming and various other multi-stage stochastic programming algorithms.

In the book "Dynamic Programming applications to agriculture and natural resources", Kennedy (2012) argues that both agriculturerelated problems of farm design, crop management and livestock breeding and natural resource management problems such as land management, forest management and, last but not least, fisheries management, are natural candidates for solution by dynamic

---

programming algorithms. The author bases this on amongst others the repetitive, cyclical nature of decisions over time and the elementary question of deciding the value of an amount of a certain biological resource. Note that the author here by fishery management primarily addresses catching of wild fish in open sea and not fish farming, but we will argue that many of the same points stand also for fish farming. In all of these cases, there are typically both production risks in growth, mortality and price risk. Furthermore, there is typically a long time horizon and the main question of when to harvest and/or when to start a new production cycle given a limited number of areas/facilities. This suggests, as per the author, that dynamic programming algorithms such as approximate dynamic programming or stochastic dual dynamic programming will work well for these problems.

### 4.3.3 Salmon production specific literature

Salmon farming specific literature contains several subcategories. There is much literature on farming of other fish species as well, but this will for the most part be kept outside of this thesis. Research on the biological aspects of salmon and salmon growth, e.g. on sea lice and on feed mixture optimization, are large fields, but will also be excluded from this project as they are not what we deal with. There is also a substantial amount of literature on e.g. investments in salmon farming facilities, production cost increase breakdown and analysis and more, but these sort of topics will also be kept outside of this project. We will focus on research on the operations aspect of salmon production scheduling, typically involving decisions about the release of either fry (eggs) or smolt and harvesting, with the objective of either minimising costs, maximising biomass output or maximising profits. Salmon face a number of challenges in the ocean, like varying growth conditions, potential disease outbreaks, escape from cages etc. All these elements make the production process uncertain. When the salmon is ready to be sold, producers face uncertain and volatile salmon price. Furthermore, as described in chapter 2, there are several complicating constraints on production. Thus, there is a need for support to make profitable decisions regarding harvest and forward sales. Our literature has revealed that salmon production scheduling optimization is a relatively young and small field of research compared to other operations research areas. This could be because the industry only in quite recent years have gotten industrialized and up-scaled and because not many regions in the world focus on salmon farming. Much of the literature on salmon production originates in Norway or another of the large salmon producing countries, which is natural. The importance of production planning for salmon is however reflected in the growing amount of literature on the area. We will in the following go through relevant literature with respect to optimizing salmon production scheduling in particular.

In the very beginning, research papers focused mainly on optimal harvesting decisions in the single rotation problem (harvesting of a single batch of fish) with an analytical, microeconomic approach. Lillestøl (1986) develops analytical derivations of optimal feeding schedules and harvesting times for a single batch of fish, the so-called single rotation problem, where the full batch is harvested at once. He argued that the full, global problem description would be too comprehensive to be solved exactly, and argued in favour of simplified, discrete growth modelling in return for better solution estimates. He suggested dynamic programming approaches for the problem. Bjørndal (1988) also took an analytical approach to finding the optimal harvest time of a single batch of fish, and he includes smolt release costs and fish insurance costs as well. He does however use a simplified, continuous growth model, only modelling average weight of the batch. He makes simple analytical remarks regarding the possibility (and value) of selective harvesting of a batch and regarding the multiple rotation problem. Neither of these two early papers seem to be written with mathematical programming in mind, however, and they are theoretical rather than practical in approaching the problem in the eyes of a salmon farmer. Bjørndal (ibid.) mentioned that fish farming actually has more in common with forestry and agriculture than with ocean fishing. He also later co-wrote the book Asche and Bjørndal (2011) which addresses most aspects of salmon production and salmon production facility investments from an economical perspective.

Arnason (1992) and Mistiaen and Strand (1998) follow the abovementioned papers in their analytical, microeconomic approach. The former proves that the optimal feeding schedule and the optimal harvest time are inevitably interdependent. The latter is the first to include what we deem

---

the more interesting salmon price variations, with a piecewise constant price for different weight classes of fish. Mistiaen and Strand (1998) found that small changes in discount rate have big implications for the optimal time of harvesting. These results are of interest because firms tend to have different cost of capital depending on size, maturity and cash flow.

As both mathematical programming as a developed field and the global salmon market grew larger, research combining these emerged. Sparre et al. (1976) and Forsberg (1996) were among the first in doing so. Forsberg (ibid.) used discrete weight classes and weight grading of fish (yielding a distribution of the weight of the fish in a batch) and formulated a mixed-integer, multi-period deterministic programming model for harvesting different fish cohorts to maximise profit. The growth model takes feed type, fish size and water temperature as input. Other input parameters are the salmon price, feeding costs, transportation and slaughtering. More realistic, complicating harvesting restrictions are implemented. This model does not consider production restrictions as limits for Maximum Allowable Biomass (MAB), which was introduced later, in 2005. It also misses possibilities for other important restrictions as well boats, fallowing etc. In Forsberg (1999), he uses a similar model to investigate the value of graded/selective harvesting vs having to harvest the full batch at once. The results suggest that by sorting fish before slaughtering, the salmon producer can increase profit by 10% compared to slaughtering the whole cohort at the same time.

Pascoe, Wattage, and Naik (2002) stated that there so far was a gap between practice and theory, in the sense that research on optimal harvest timing did not take into account the great risks in production as well as other essential real-world production constraints. Forsberg and Guttormsen (2006) stated that fish farmers traditionally have focused either on production planning or price forecasts, and conclude that these aspects should be combined in order to make good harvesting decisions. He therefore goes more in depth on the value of information in fish farming. The large price variations must be taken into account when making production plans, which is an important idea this thesis in part is based on.

More recent research has attempted to use more advanced mathematical programming on more realistic problem versions, aiming at providing practical decision support for salmon farmers. As mentioned, MAB restrictions were introduced in Norway in 2005, and Stikholmen (2010) could not in 2010 conclude in whether the aquaculture industry had become more or less efficient in the previous decade. Langan and Toftøy (2011) were of the earliest in this respect. They developed an optimization tool for salmon production scheduling when maximizing MAB utilization of the saltwater facilities. During development, they discussed different ways of linearising and discretising the growth model, but ended up with a rather simple model where all the fish in a batch have the same average weight at each point, and hence with no weight distribution within the batch.

Their two-stage stochastic model considered uncertainty in growth and mortality of the salmon but not in the salmon price, and showed that minimizing deviation from MAB quota is not the same as increasing the biomass output. Hæreid (2011) instead created a one-year horizon stochastic model with uncertainty in prices, focusing on which sales contracts the salmon producer should enter. Rynning-Tønnesen and Overaas (2012) built on these previous papers and developed a tactical planning model that makes smolt delivery and deployment as well as harvesting plans. The model considers uncertainty in growth and mortality, but leaves out price uncertainty. Hæreid, Schütz, and Tomasgard (2013) takes this further and combined the best of the above by creating a multi-stage stochastic model with a time horizon of 3-5 years and with uncertainty in both prices, growth and mortality. This was however not implemented and solved for actual cases, from what we can understand. Both Rynning-Tønnesen and Overaas (2012) and Hæreid, Schütz, and Tomasgard (2013) have a more advanced growth model with multiple weight classes and a distribution of fish between several weight classes in each time step, giving a weight distribution within each batch. Kure and Frøystein (2013) is in turn one of the first papers to apply optimization to smolt production (the freshwater phase) in particular. They minimize total costs in smolt production given that smolt orders from the saltwater locations should be delivered. An essential difference in their model is that temperature is a variable in a land based facility, and they modelled this using SOS2 sets, with uncertainty in the water intake temperature.

Looking at research papers who included more parts of the value chain in the problem, we have Denstad, Ulsund, and Lillevand (2015) and Bravo et al. (2013). Also inspired by Hæreid (2011),

---

Denstad, Ulsund, and Lillevand (2015) took salmon production optimization one step further by introducing several downstream parts of the value chain, such as sales allocation to different products, different contract types as well as inventory management and global logistics for bringing salmon to the market. They implemented both a deterministic and a stochastic model, the latter with uncertainty in prices and in water temperature. This is a multi-national logistics problem, and thus written with a large global salmon producer in mind, and hence the pure salmon production scheduling problem of both deploying smolt and harvesting salmon in several cycles is deliberately modelled with less detail in this paper. Bravo et al. (2013) made two models, one for the freshwater phase and one for the saltwater phase, and used these two in conjunction.

The models above do not apply any particular advanced optimization algorithms such as for example decomposition, and there has not been many attempts at doing so, neither exact nor heuristic, despite the fact that several of the research efforts conclude that multi-cage, multi-cycle problem versions including uncertainty with a detailed growth modelling are practically unsolvable with MIP and branch-and-cut commercial solvers alone. We are aware that there has been attempts at solving freshwater phase problems by column generation. Cobo et al. (2019) uses particle swarm optimization to solve a similar problem for other fish species production in Spain. Yu, Leung, and Bienfang (2006) solved a multi-cycle, multi-pond problem for shrimp farming by using a network flow formulation such as the average weight formulation presented in this project, but this has a limited growth model in terms of weight distributions. Solving large scale problems in salmon production scheduling still seems to be an area for future research.

As induced by Kennedy (2012) (see the previous subsection), other researchers have also figured that dynamic programming methods are appropriate for aquaculture production scheduling optimization. Although most of these efforts date several years back, are based on other species than salmon and deal with highly simplified problem versions compared to this thesis, they are important mentions because they show that others have seen the logical modelling of the problem as a Markov Decision Process and solved it by dynamic programming methods. Both Karp, Sadeh, and Griffin (1986) and Leung and Shang (1989) model shrimp production as a Markov Decision Process and solve it by exact, backward dynamic programming. They also mention that because they use exact methods, the models are limited to very small problem versions with small state spaces. The state space of the salmon production scheduling problem as it is presented in this thesis would be computationally intractable by such exact methods.

## 4.4 Summary and our contribution

As described in chapter 2, traditional salmon farming has taken place in cages in fjords or bays, where smolt ("youth" fish) have been grown to sales ready weights. Before this, smolts are produced from eggs in freshwater facilities on land, in tanks. Different papers have focused in different parts of this value chain. In the table below are the most relevant and inspiring research efforts with respect to this thesis. We limit the table to papers focusing on the "pure" production scheduling problem with emphasis on release and harvest scheduling. This gives an overview of the focus area and the solution approaches of the literature on the salmon production scheduling problem.

Paper	Deterministic/ stochastic	Objective function	Growth model	Remarks	Solution method
<b>Saltwater phase: salmon production in sea</b>					
Forsberg (1996)	Deterministic	Profits	Discrete, with distribution	No MAB restrictions, only one tank (?)	N/A, only formulation
Langan and Toftøy (2011)	Two-stage, uncertain growth and mortality	Profits	Discrete, no distribution		MIP, with Xpress
Rynning-Tønnesen and Overaas (2012)	Two- and multistage, uncertain growth and mortality	Profits	Discrete, with distribution	Focus on smolt ordering and deployment	MIP, with Xpress
Hæreid, Schütz, and Tomasgard (2013)	Multi-stage, uncertain growth, mortality and prices	Profits	Discrete, with distribution	No actual model imple- mentation, only formulation	N/A
<b>Freshwater phase: smolt production on land</b>					
Kure and Frøystein (2013)	Stochastic, uncertain water intake temp	Costs	Discrete, with distribution	Temperature as variable	MIP, with Xpress
<b>Other aquaculture species, selected papers</b>					
Cobo et al. (2019) (Seabream)	Deterministic	Profits	Simple, without distribution	Full batch harvesting	Particle Swarm Op- timization (heuristic)
Karp, Sadeh, and Griffin (1986) (Shrimps)	Deterministic and stochastic	Profits	Simple, without distribution	Full batch harvesting	Exact, backward dynamic program- ming
<b>Our contribution: Land based salmon farming</b>					
<b>This master thesis</b>	Deterministic and stochastic wrt. price	Profits	Discrete, with distribution	Focus on land based salmon farming and price uncertainty	ADP/RL

Table 4.1: Summary of relevant literature regarding salmon production scheduling

The authors are also aware of at least two current research efforts which apply Dantzig-Wolfe decomposition and column generation with Branch-and-price to the salmon production scheduling problem. This is an exact solution method, unlike ADP/RL which is a heuristic method. These efforts, including our own, reflect the increasing interest in this optimization problem and in the computational complexity of the problem. These works are not yet published and/or finished and we are therefore not able to discuss them closer, however relevant. In addition, there are papers such as Denstad, Ulsund, and Lillevand (2015) and Bravo et al. (2013) with a larger value chain focus, shifting the problem more towards a logistics and operations problem, which thus is not strictly relevant.

As explained earlier, this paper is written with a land based salmon producer in mind, partly

---

because the salmon producer we have been cooperating most closely with is land based and partly because this is an important part of the future of the salmon industry. In land based salmon farming, growth rates and mortality risks are greatly reduced due to more a controllable environment. Thus, the single greatest remaining uncertainty factor is the salmon price, with it's large variations both throughout the year, between different sales classes and from year to year. The problem is however, for most practical matters, similar to that of a traditional salmon producer in cages in the fjords. The difference in focus is mostly conceptual and primarily comes to show in that growth rates and mortality is considered less uncertain, and that there is a larger focus on detailed growth modelling reflecting the more controllable environment in tanks on land.

Research on optimization of salmon production scheduling is, as shown, quite limited compared to other areas of operations research. For solving salmon production scheduling optimization problems on a real-production scale, it has become clear to us, other current researchers on the field and through the literature reviewed, that more advanced optimization techniques are needed. Most of the few such efforts conducted so far have applied methods such as Branch-and-Price, which focus on deterministic optimization, or stochastic programming methods such as 2-stage or 3-stage models without the application of more advanced decomposition methods. For the problem we consider however, advanced multistage stochastic programming decomposition approaches like stochastic dual dynamic programming and approximate dynamic programming, both also falling under the sequential decision making umbrella, are relevant. They do however need some clever adaptations for application to salmon production scheduling, mainly because of the complicated value function the problem presents.

Most machine learning applied to salmon production has not been about mathematical optimization but about operational analytics such as e.g. monitoring fish feeding using cameras to discover efficiency gains in feeding.

To our knowledge, ADP/RL has not been applied to salmon production in particular and our paper will thus be the first effort towards applying ADP/RL to salmon production scheduling. This requires modelling the problem as a Markov decision process. The complexity of the full-scale salmon production scheduling problem has revealed that smarter optimization methods are needed, and our contribution will be to investigate whether or not ADP/RL could serve this purpose also for salmon farming. The goal is twofold: both designing a more scalable solution method in terms of computational time and finding high-quality solutions close to optimality. The many impressive and highly complex applications of ADP/RL and the rise of machine learning suggests possible application to many operations research areas, and we will take the, to our knowledge, first step in applying it to salmon production scheduling. Because of the novelty of the application, our focus will to a large extent be on discussing what further research should focus on and on concluding on whether ADP/RL has potential to work for our problem.

# Chapter 5

## Model and solution approach

In this chapter we start by formulating and modelling the salmon production scheduling problem as a Markov Decision Process. We then present the salmon price model we have developed. Finally, we present our algorithm and solution approach for solving this MDP by ADP/RL.

### 5.1 Salmon production scheduling as a Markov Decision Process

#### 5.1.1 Extensive mathematical formulation

Below is the extensive mathematical formulation of the salmon production scheduling problem that is being attacked in this thesis.

##### Sets and indices

$T$	Set of all time periods
$U$	Set of all tanks in the facility
$W$	Set of all weight classes (smallest is 50 grams and largest is
$W^g$	Subset of all weight classes where the fish have not yet reaches sales ready weight
$W^s$	Subset of all weight classes above the first sales ready weight class
$J$	Set of all sales classes
$W_j$	Set of all weight classes corresponding to sales class $j$
$S$	Set of different smolt types (smolt weights)



---

## Parameters

$\pi_{jt}$	Price of salmon in sales class $j$ at time $t$
$\pi^{avr}$	Average price of all sales classes throughout production period
$C_s^s$	Cost of smolt type $s$
$C^f$	Cost of 1 kg of feed
$C^h$	Fixed cost on harvesting
$\bar{W}_w$	Average weight of weight class $w$
$\bar{W}_s^s$	Average weight of smolt of smolt type $s$
$\bar{G}_{wt}$	Growth in grams of one fish in weight class $w$ during time period $t$
$incBio_w$	Amount of fish in weight class $w$ from start of planning period
$\rho_{\tilde{w}wt}$	Share of fish in weight class $\tilde{w}$ growing into weight class $w$ from time $t$ to time $t+1$
$FCR$	Feed conversion ratio
$SR$	Survival rate of fish during one time period
$D_{sw}$	Share of number of smolts of smolt type $s$ released that goes into weight class $w$
$\beta$	Adjustment factor for valuation of fish in tanks at end of horizon
$\gamma_t$	Discounting factor for time $t$
$\gamma^h$	Discounting of value of fish in tanks at end of horizon
$F^{max}$	Maximum number of fish in a tank (big M factor)
$m$	Minimum number of fish in a tank
$V$	Volume of each tank
$Dens^{max}$	Maximum density restriction in each tank
$MAB$	Maximum Allowable Biomass in facility
$minHarv$	Minimum harvested biomass at each harvesting

## Variables

$f_{uwt}$	Amount of fish in weight class $w$ at end of timeperiod $t$ in tank/unit $u$
$x_{uwt}$	Amount of fish slaughtered from weight class $w$ at end of timeperiod $t$ in tank/unit $u$
$y_{ust}$	Amount of smolts of smolt type $s$ released in tank $u$ at time $t$
$\delta_{ut}$	Binary helper variable, 1 if tank $u$ is occupied at time $t$ , 0 otherwise
$\alpha_t$	Binary harvesting variable, 1 if harvesting from facility in time $t$ , 0 otherwise

Maximize

$$\begin{aligned} \sum_{u \in U} \sum_{t \in T} \sum_{j \in J} \sum_{w \in W_j} x_{uwt} \pi_{jt} \gamma_t \bar{W}_w - \gamma_t C^h \sum_{t \in T} \alpha_t - \sum_{t \in T} \sum_{w \in W} C^f FCR \bar{G}_{wt} \gamma_t f_{uwt} - \sum_{u \in U} \sum_{t \in T} \sum_{s \in S} C_s^s \gamma_t \bar{W}_s^s y_{ust} \\ + \sum_{u \in U} \sum_{w \in W} \gamma^h \beta \pi^{avr} \bar{W}_w f_{uw|T} \end{aligned} \quad (5.1)$$

subject to

Growth and mass balance constraints:

$$f_{uw0} = \sum_{s \in S} D_{sw} y_{us0} + incBio_{uw} \quad w \in W, u \in U \quad (5.2)$$

$$f_{uw(t+1)} = \sum_{\tilde{w} \in [0, w]} SR \rho_{\tilde{w}wt} f_{u\tilde{w}t} + \sum_{s \in S} D_{sw} y_{us(t+1)} \quad w \in W^g, t \in T/|T|, u \in U \quad (5.3)$$

$$f_{uw(t+1)} = \sum_{\tilde{w} \in [0, w]} SR \rho_{\tilde{w}wt} f_{u\tilde{w}t} - x_{uw(t+1)} \quad w \in W^s, t \in T/|T|, u \in U \quad (5.4)$$

---

Production constraints:

$$\sum_{w \in W} f_{uwt} \geq m\delta_{ut} \quad t \in T, u \in U \quad (5.5)$$

$$x_{uwt} \leq SR\rho_{\bar{w}w(t-1)}f_{u\bar{w}(t-1)} \quad w \in W^s, t \in [1, |T|] \quad (5.6)$$

$$x_{uw0} \leq incBio_{uw} \quad w \in W, u \in U \quad (5.7)$$

$$\sum_{w \in W} \sum_{u \in U} x_{uwt} \leq F^{max} \alpha_t \quad (5.8)$$

$$\sum_{u \in U} \sum_{w \in W} x_{uwt} \bar{W}_w \geq minHarv * \alpha_t \quad t \in T \quad (5.9)$$

$$\sum_{s \in S} y_{ust} \leq F^{max} (1 - \delta_{u(t-1)}) \quad t \in [1, |T|], u \in U \quad (5.10)$$

$$\sum_{w \in W} \bar{W}_w f_{uwt} \leq V Dens^{max} \delta_{ut} \quad t \in T, u \in U \quad (5.11)$$

$$\sum_{w \in W} \sum_{u \in U} f_{uwt} \bar{W}_w \leq MAB \quad t \in T \quad (5.12)$$

### 5.1.2 State

In this and the following subsections follows the formulation of our problem, i.e. the problem expressed in mathematical terms above, as a MDP.

For the full facility (i.e. multitank) problem, a system state includes the following information:

- The current salmon (spot) prices (one for each sales class)
- The current time and time of year (month), in case of stochastic solution
- The amount of fish in each tank in each weight class

The salmon spot price and the time represents the part of the state exogenous to the salmon farmer, and the information about the fish in the tanks of the facility represent the internal state.

### 5.1.3 Decision

One system decision contains the following information:

- Whether or not harvesting is done from each tank
- Whether or not smolt of type s are being released in each tank
- The amount of fish harvested from each weight class from each tank
- The amount of smolt of smolt type s released in each tank

---

### 5.1.4 Exogenous information

As we model both the fish growth (and water temperature) and fish mortality deterministically, the only random process is the development of the salmon prices. In each new time step, a new salmon price is realized and becomes part of the exogenous part of the state.

### 5.1.5 Transition function

Given a current state and a decision made, the transition to the next state is deterministic except for the price development.

For the exogenous information, the transition function increments the time by one unit and samples a new price realization from the price model.

A decision contains a decision regarding either harvesting or smolt release for each individual tank. The transition function therefore calculates the transition for each tank independently:

- If the tank is empty, and smolts are released into the tank, the next state will contain the amount of smolt released distributed with a normal distribution with coefficient of variation 10% around the average weight of the smolt type  $s$ . If smolts are not released, the tank remains empty.
- If the tank has fish and some of these fish are harvested, we first remove the harvested fish and then let the remaining fish grow according to the growth model. We thus subtract the harvested amount of fish from each weight class from the current amount in the respective weight classes and let the remaining amount of fish in each weight class grow for one period, calculating how they transition into new weight classes.
- If the tank has fish but no fish are harvested, the fish will only grow, meaning they transition into new, higher weight classes according to the growth model.

A key part of the transition function is the salmon growth model we have developed. The growth model is based on the Skretting growth rate table which can be found in the appendix. Skretting is a world leading salmon feed producer and their data sources for salmon growth are a widely acknowledged standard used in the industry. We will not go deeply into the specifics of the growth model here. The model assumes that the fish are fed to saturation and that water quality is maintained at an optimal level, both of which are standard practice in the industry (Asche and Bjørndal (2011)). Then, the Skretting table provides daily growth in percentage of fish weight per fish for a given temperature and a given fish weight. The water temperature is given as a temperature profile with a specific temperature for each month of the year, because the salmon producer we have cooperated most closely with uses seawater in their facility. As we have pre-defined a set of 50 weight classes ranging from 50 grams to around 7 kilograms, the growth model then uses the temperature and the weight to calculate a three-dimensional transition matrix which calculates what share of the fish in each weight class will grow into a specific other weight class at a certain time period. This transition matrix is the key part of how the fish are transitioned into a new state. This matrix also serves to linearize the growth model in a way that makes it usable in linear programming (as in the MIP formulation above), and it is a central part in the Markov state transition probabilities.

### 5.1.6 Direct profit/reward function

The common term in RL setting is reward function, but we prefer the term direct profit function as this is more relatable to our problem.

The direct/immediate profit  $R(s_t, a_t)$  made from being in a particular state (at a particular time) and making a particular decision is calculated as the sum of the following terms:

- 
- The harvesting revenue:  $\sum_{u \in U} \sum_{j \in J} \sum_{w \in W_j} x_{uw} \pi_j \bar{W}_w$
  - The feed cost:  $\sum_{w \in W} C^f FCR \bar{G}_{wt} f_{uw}$
  - The smolt cost:  $\sum_{u \in U} \sum_{s \in S} C_s^s \bar{W}_s^s y_{ust}$
  - The fixed harvesting cost:  $C^h \alpha_t$
  - A penalty for harvesting fish before they are sales ready (and penalty for breaking the density or MAB constraints in case of soft modelling of constraints)

Where the notation of the parameters and variables are the same as defined above in the extensive mathematical formulation.

### 5.1.7 Overall objective

The overall objective of a MDP is to find the policy which maximizes the expected cumulative discounted reward/profit earned by the agent.

In other words,

$$\max \mathbb{E} \left[ \sum_{t=0}^{\infty} \gamma^t R(s_t, a_t) \right]$$

where R is defined in the previous subsection.

#### Assumptions and clarifications

The following are some assumptions and clarifications made in this master thesis regarding the salmon production modelling specifics. These are primarily assumptions reflecting the type of production of the salmon farmer we have cooperating most closely with, but also assumptions made to ease computation because not all model factors are equally relevant for determining the suitability of ADP/RL as a solution method for the problem in general. The general assumptions for the project are discussed here.

First, this project is meant to take the perspective of a land based salmon producer, which is a new and growing part of the industry, and which our main industry contact represents. We assume that fish growth and mortality, as well as any other production risks, are sufficiently controllable that they can safely be left out of this project without loss of relevance of our results.

Second, different feed types and the choice between them, as well as the feeding schedule, is left out. Earlier researchers such as Asche and Bjorndal (2011) have found that the optimal choice for a salmon producer generally is to always provide oxygen and feed to saturation. This also makes growth modelling easier, because growth then can be assumed to only be a function of weight and temperature and can be linearised as a Markovian process.

We also make the simplifying assumption that all harvested salmon will be sold immediately and in the spot market. The entering of futures contracts and other forms of price hedging is beyond the scope of this project. If there is a fixed delay between harvesting time and selling time of a fish, we assume that our results are not affected because the salmon price applied in the model can simply be shifted accordingly. We also do not take into account the weight loss between so-called live weight and harvested weight (typically around 18% (Mowi (2021))). This does not have any effect on the optimization problem, as it effectively only serves to raise the weight limit for all sales classes.

We do not take smolt production and smolt ordering uncertainty into account, meaning, we assume that the salmon producer can order smolt with any average weight he/she desires and get delivery at any time. In reality, there can be deviations from the order in terms of both size, number and time of delivery, but this is kept out of this thesis. Furthermore, for most of the calculations, we

---

will assume only smolts of average weight 100 grams are available, which is the most common type of smolts.

Water temperature was initially meant to be treated as a decision variable in this thesis, as is the case in many land based facilities with RAS technology. However, the industry contact we are cooperating with is applying an innovative, cost-efficient solution where seawater is pumped into the facility and therefore the temperature is treated as an exogenous input parameter following a yearly profile provided to us. This eases modelling efforts. It also serves to make our project more similar to traditional salmon production in fjords.

A rather big problem feature left out of this thesis is the option to move and split fish batches between different tanks during growth. The reason for this is explained in 2.4.1. While this feature increases the optimization problem complexity, we argue that it is not necessary for qualitatively determining whether or not ADP/RL will perform well for the salmon production scheduling problem. The same argument is made for another, smaller problem feature, which is giving the farmer the choice between different sizes of smolts to release (e.g. choosing between 100, 200 or 300 gram smolts). While this is built into the model (and the ADP/RL model is perfectly able of handling it), it will be kept out of computations and evaluations.

## 5.2 Semi-parametric modelling of prices

As discussed, forward prices are today's value of fish delivered at some point in the future. Simulation of future expectations is essential to make our rolling horizon model perform well. Also, we need to simulate spot price dynamics in order to generate price samples and be able to solve the stochastic problem. As the salmon price behaves similarly to electricity prices in many ways, our stochastic price model is inspired by Monteiro (2020). She presents a novel semi-parametric structural model for forward curves. The basis of the approach is a data set of historical forward prices of salmon ranging from 2015-2020. All price data are collected from fishpool.eu. The reason only data after 2015 are used is that there seems to be a regime shift at around this point in time, as discussed in 2.5.1. These prices are then log-, mean and seasonal adjusted to find residuals  $\epsilon_{t,j}$  before performing a principal component analysis (PCA). Here,  $t$  is the time with weekly resolution, while  $j$  is the time to maturity. The basis for the seasonal component estimation can be seen in Appendix A.3.

The procedure is as follows:

- Prepare data set by log-, mean and seasonal adjustment
- Perform PCA analysis on residuals
- Decide how many components are needed to include in model in order to explain a sufficient part of the variance
- Save component weights
- Fit AR(1)-GARCH(1,1) for components.
- Generate  $S$  scenarios  $N$  steps ahead for residuals
- Generate  $S$  scenarios  $N$  steps ahead for the components
- Generate  $S$  price scenarios  $N$  steps ahead

In the context of this study, PCA decomposes the matrix of residuals in two separate sets: one of time-indexed series, called components, and another of their factor loadings, which change with maturity and represent the weight of each component on  $\epsilon_{t,j}$ . The approach is analysed with respect to how many components are necessary to explain a high percentage of the variance of the residuals. As PCA's objective is dimensionality reduction, its use is only reasonable if the number

of used components is significantly lower than the number of maturities. The component series are uncorrelated, which means univariate models can fit each of them separately.

An issue regarding PCA performance on the residuals is how many forward maturities should be included in the data set for the analysis. Due to liquidity issues in long forward contracts, and requirement of at least 26 months for our rolling horizon model (that is how long horizon we found appropriate for the sub problem) we chose to do the analysis on 26 maturities. Summarizing, the data input were contracts from 0-26 months from 2015 and 2020. The results of the PCA showed that five components were needed to explain approximately 86% of the variance in the data set. We find this level of explanation satisfactory and do not include more components to maintain a high degree of dimensionality reduction. The weights for component  $i$  and maturity  $j$   $w_{i,j}$  were saved for use in prediction of forwards with different maturities. By fitting autoregressive(1) - generalized autoregressive conditional heteroskedasticity(1,1) (AR(1)-GARCH(1,1)) model to the residuals we were able to simulate new residuals  $\epsilon$  for  $S$  scenarios  $N$  time steps ahead for each component. Finally, we were able to predict future prices scenarios for  $S$  scenarios,  $N$  time steps ahead and  $j$  maturities. The procedure can be summarized as follows

AR(1) model:

$$c_{t,k} = \alpha + \phi c_{t-i,k} + \xi_{t,k} \quad (5.13)$$

With  $\xi_{t,k} \sim GARCH(1,1)$  :

$$\xi_{t,k} = \mu + \sigma_{t,k} Z_{t,k} \quad (5.14)$$

$$\sigma_{t,k}^2 = \omega + \alpha(\xi_{t-1,k} - \mu)^2 + \beta\sigma_{t-1,k}^2 \quad (5.15)$$

Where  $Z_{t,k} \sim N(0,1)$

In the AR model,  $c$  is component,  $\phi$  is a constant and  $\xi$  is residual. The GARCH model is used for simulation of new residuals as input in the AR model.  $\mu$ ,  $\alpha$  and  $\beta$  are constants,  $\sigma$  is variance and  $Z$  is a white noise process.

Five random samples of the simulation are shown in the plot below.

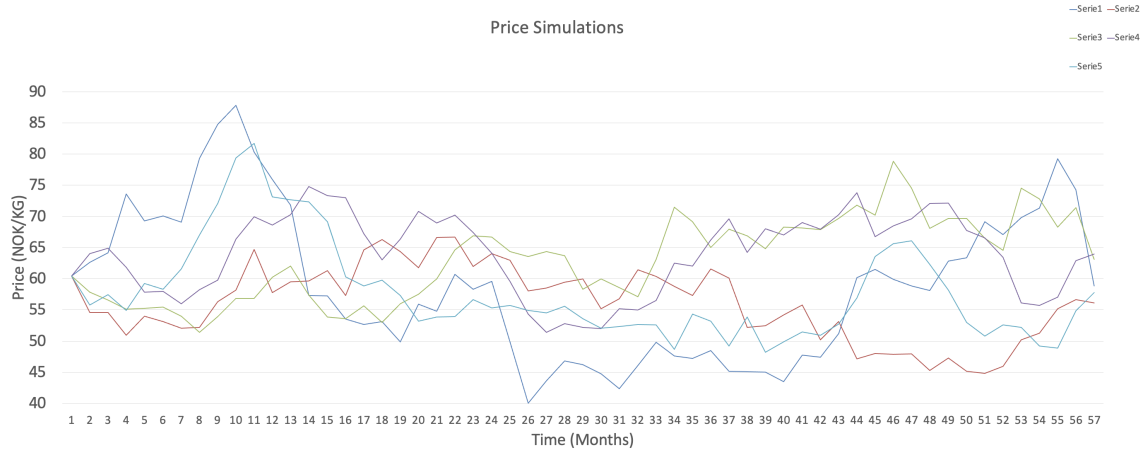


Figure 5.1: Five examples of price simulations

From the plot of price simulations we can see that the predictions are in range approximately 40-90 NOK/kg. These numbers seem realistic in light of historical variations and seem qualified to be used as input in the optimization models. It is, however worth noting that there might be deviations from this range over more price samples.

---

To give the reader more insight to how the price model works, we have also plotted one simulation with corresponding price expectations below.



Figure 5.2: Simulated price development (purple) with corresponding 26 months price expectations starting at  $t = 0$ ,  $t = 21$  and  $t = 42$  (blue, red and green, respectively)

Here, the expectations (or forward price curve) at a given point in time, for example  $t = 0$ , is calculated as the sum  $w_{i,j} * c_{t,i}$  for each component  $i$  and maturity  $j$ . These can be provided for any  $t$ , but only three examples are shown here for illustrative purposes. Figure 5.2 is based on one random price realization. How well the expectation corresponds with the realized simulation will vary over time, but they seem to imitate the historical forward curves reasonably well. For the purpose of our thesis this price model seems to give satisfactory results.

---

## 5.3 Solution algorithm

The following algorithm, here written in pseudocode, represents the ADP/RL model we have developed. The model has been implemented in Python. The biological growth model, the transition function and the direct profit function, as well as other algorithm components, have also been built from scratch and implemented in Python.

---

### Algorithm 2: Stochastic ADP/RL

---

**Result:** Optimal policy + according optimal value of each state  
Initialization of value function approximation and salmon growth model;

```
for  $n$  iterations do
  State  $\leftarrow$  random start state;
  Draw sample price realization;
  for  $T$  timeperiods do
    for each tank do
      | Get relevant decisions for each tankstate;
    end
    Find relevant combinations of relevant single tank decisions and construct candidate overall decisions;
    for each candidate decision do
      | Calculate direct profit of (state, decision);
      | Calculate exploration bonus of (state, decision);
      | Calculate potential penalties of (state, decision);
      | Next state  $\leftarrow$  transition(state, decision);
      | Calculate value of next state = value function(next state);
    end
    Draw random number between 0 and 1;
    if random number  $\leq \epsilon$  then
      | Choose candidate decision with highest (direct profit + exproation bonus - penalties +  $\gamma$  * value of next state);
    else
      | Choose candidate decision with highest (direct profit - penalties +  $\gamma$  * value of next state);
    Next state  $\leftarrow$  transition(state, chosen decision);
    State  $\leftarrow$  next state;
    Update counts for exploration bonus;
    Save experience (state, decision, directprofit + penalties);
    Append experience to episode data;
  end
  Select  $s$  random experiences from episode data;
  for each selected state/experience do
    | Target  $\leftarrow$  Sum of discounted remaining profits until  $T$  + discounted value of state in time  $T$ ;
    | Calculate loss = target - value of state;
    | Take gradient of loss wrt. valuefunction parameters;
    | Update parameters by taking step in direction of negative gradient to minimize loss;
  end
   $\epsilon \leftarrow \epsilon - \delta$  (reduce  $\epsilon$ )
end
return Value function with trained parameters, policy given by value function
```

---



---

### 5.3.1 Variants

#### Hard vs soft constraints

The tank density constraints and the facility MAB constraint are the primary interesting production constraints for this problem, and these can be modelled as either hard or soft in the eyes of the ADP/RL agent.

When modeling the density and MAB constraints as hard constraints, we only present the ADP/RL agent with candidate decisions which respect these constraints. This involves checking each relevant decision generated for whether or not making this decision in the current state will lead to these constraints being violated in the next state. In other words, when modelling as hard constraints the ADP/RL agent never gets the opportunity to violate these constraints.

When modelling these constraints as soft constraints, the agent can perform any relevant actions, but is then faced with a penalty incorporated in the direct profit function when making decisions which leads to states where the constraints are violated. This requires modifying the direct profit function slightly and introduces the said penalties as tunable parameters in the algorithm.

#### Stochastic vs deterministic solution

The stochastic and deterministic models are solving two principally different problems. The deterministic problem assumes either a constant or a known salmon price whereas the stochastic problem involves not knowing how the price will develop in subsequent time steps.

In the deterministic version, the solution algorithm is the same as for the stochastic version except that for each episode, the price is set equal to a constant price of 60 NOK/kg. The algorithm for the stochastic version is described above.

The important difference between the solution of the stochastic problem with ADP/RL and the deterministic problem with ADP/RL is that the stochastic version needs to be able to take into account the development of the salmon price. It also needs to take into account the time of the year of a given state, given that we assume a certain seasonality, which is implemented in the price model. The key point here is that the value function approximation needs to take both the current salmon price and the current time of year as input features, and needs to have a shape which allows these features to adjust the value of the state. In case of a custom value function approximation, this needs to be handled manually by e.g. introducing a factor which is a function of type  $(1 + \sin/\cos)$  of the time of year) in the value function approximation and by also introducing some function of the salmon price as a factor. In case of the neural network, this is as "simple" as adding the salmon price and the time of the year in the given state as input features to the network and letting the network figure out the shape itself. In practice, however, this radically changes how to neural network behaves and converges.

### 5.3.2 Decomposition heuristic

As described in 2.4.2, the salmon production scheduling problem has an inherent decomposed structure.

We have exploited this structure to design an alternative solution approach which trades off some potential loss of optimality with much faster training for the multi-tank problem. The use of this alternative algorithm is thus to speed up solution of the multitank problem. This idea builds on an assumption that in the steady-state optimal production schedule, each tank is on a steady-state production cycle where new fish are released immediately after the tank is empty (and washed) and then is continuously in production. In other words, given this assumption the only difference between the tanks is that they (potentially) are out of synchronization with each other, in order to exploit the MAB restriction. According to our research so far with the MIP model, this assumption does not necessarily always hold exactly, especially with large price variations, but it is still "quite

---

close” to holding, and we expect the loss of optimality by making this assumption to be in the order of  $< 5 - 10\%$  of optimality.

With this assumption as a foundation, one can then further draw the conclusion that the value of the whole system state is nothing more than the sum of the values of the individual tank states. Our alternative solution algorithm involves only training a value function for the single tank problem until satisfying convergence is achieved. This is the training part of the algorithm. Then, for the inference part of the algorithm (inference in RL setting refers to using the model for producing results), the single tank value function is used to produce a schedule for the multi tank facility. The algorithm simply makes decisions for the multitank facility by using the value of the next state as the sum of the values of the individual tank states, which are given by the single tank value function which has been trained in the first part of the algorithm.

The algorithm is explained with high-level pseudocode in the figure below.

---

**Algorithm 3:** Alternative ADP/RL decomposition heuristic algorithm

---

**Result:** Optimal policy + according optimal value of each state

**Part 1: Train single tank value function**

**Train single tank value function as per the main ADP/RL algorithm described above;**

**Part 2: Produce multi tank schedule**

State  $\leftarrow$  empty start state;

Draw price realization scenario to evaluate;

**for**  $t$  in  $T_{horizon}$  **do**

    Schedule += state;

**for each tank do**

        | Get relevant decisions for each tankstate;

**end**

    Find relevant combinations of relevant single tank decisions and construct candidate overall decisions;

**for each candidate decision do**

        | Calculate direct profit of (state, decision);

        | Calculate potential penalties of (state, decision);

        | Next state  $\leftarrow$  transition(state, decision);

        | Calculate value of next state = sum over each tank of value function(next tank states);

**end**

    Choose candidate decision with highest (direct profit - penalties +  $\gamma$  \* value of next state);

    Next state  $\leftarrow$  transition(state, chosen decision);

    State  $\leftarrow$  next state;

**end**

**return** Single tank value function, multi tank production schedule

---

### 5.3.3 Key algorithmic elements

#### Value function

One of the biggest focus areas of our research has been on the implementation of a value function approximation and which such approximation would work best, given our intuition about the optimization problem at hand.

#### Neural Network

We have implemented and tested different deep feedforward neural networks as value function approximations. Our primary focus has been on networks with one or two hidden layers and 20-60 nodes in the hidden layers (depending on the number of tanks) as these seemed to perform best.

---

## Custom value function

As an alternative to value function approximation based on a neural network, we have also tried to design a custom value function based on our knowledge of the problem. Our primary efforts have been testing one function based on second degree polynomials and one piecewise linear function.

Variants of the following function were used, with the idea of using a second degree polynomial to catch the concavity of the value function in the number-of-fish variable: (PS: the function below is only written in order to illustrate the principle and is not necessarily mathematically accurate)

$$V(s) = \sum_{i=0}^k ((1 + a_1 \sin(a_2(\text{month}/12)) + a_3 \cos(a_4(\text{month}/12))) p \sum_{i=0}^n (-a_{i1} w_i f_i^2 + a_{i2} w_i f_i + a_{i3}) - a_5 \exp(\sum_{i=0}^n w_i f_i - \text{densityCap}) - a_6 \exp(\sum_{u \in U} \sum_{i=0}^n w_i f_i - \text{MAB}) + C$$

where the a's are the tunable parameters of the function. Again the notation partly follows the notation defined in the extensive mathematical formulation above. The first factor is the sine/cosine intended to capture seasonality, which is multiplied by the salmon price p and by the sum of a second degree function of the number of fish in each weight class, where each such number is weighted by the weight of the weight class. From this value two penalty terms are subtracted which are designed like exponential functions in order to be differentiable at all points. The penalty terms grow exponentially when the biomass in a tank exceeds the density restriction (which translates to a biomass restriction given the volume of the tank) or when the total biomass exceeds the MAB restriction. The constant term C is added to represent all future batches after the current batch and hence is initiated with a large absolute value. The idea is to split the value function into two parts, the value of the current standing biomass in the tanks plus the value of all future batches. We also know that the function has some degree of concavity in the number of fish, because, as discussed in chapter 2, there exists an optimal amount of fish for each batch.

The second custom value function applied is a piecewise linear functions of the numbers of fish in the different tanks. The idea is to provide a more flexible function shape than a simple second degree function that is still simpler to understand and initialize than a neural network. The function is then split into linear segments based on both average weight of the fish in each tank and the total amount of fish in each tank. In other words, average weight and total amount of fish were the input features when this function was used. We tried this both with and without penalty terms and seasonality adjustments. The drawback with a piecewise function is that more iterations are needed for each segment's tunable parameters to be updated sufficiently (need to visit each segment of the piecewise function sufficiently many times).

## Policy implementation, how decisions are made in each time step

The decision space for a single fishtank in our problem is quite complex compared to typical ADP and RL applications. The main complexity comes from the non-convexity and binarity of the decision space, meaning that if the tank is empty, the decision is about the amount of smolts to release and if the tank is populated, the decision is about harvesting. In typical and comparable ADP/RL applications, decisions are more low-dimensional, such as deciding the flow of a fluid through a system (control theory) or deciding which direction to walk in (RL for games). The size of the decision space in the problem studied here is enormous. When deciding about smoltrelease for an empty tank, one could in theory release every amount of fish from zero to the constant "Maxfish", which is the highest number of fish we consider relevant to consider, typically in  $6 * 10^5$  scale for the facility parameters applied. This gives in theory  $S * \text{Maxfish}$  number of possible decisions to consider for each tank, where S is the number of different smolt types. When deciding about harvesting for a populated tank, one could in theory choose to harvest any number of the fish in the tank, starting from the largest weight class and moving down the weight classes. This gives in theory  $\text{Maxfish}$  number of decisions to consider for each tank with harvest ready fish.

Furthermore, because of the decomposed structure of the problem, the possible decisions for each

---

tank, as described above, needs to be combined with possible decisions for all the other tanks in the facility to make up one overall decision. By this we mean that to find the globally optimal overall decision for a 2-tank facility when, say, one tank is empty and one is populated, we need to consider every combination of relevant decisions for the two tanks. This means for every amount of smolts considered released in the empty tank, one needs to combine this with every relevant amount of fish harvested from the populated tank. The amount of decisions that would need to be evaluated in theory grows exponentially with the number of tanks. When the amount of tanks increases to e.g. ten tanks, the theoretical amount of decisions to consider becomes in the order of  $\approx Maxfish^{10}$ .

In our problem, we do not have a "clean" analytic function for the policy, that is, the decision to be made given a state. In other types of dynamic programming applications one makes the decision, implements the policy, by taking the derivative of the cost-to-go function (a common name for the whole right-hand side of Bellman's equation) with respect to the decision variables. In our case, this is not possible, and we have to do a "manual" evaluation of each relevant decision we want to consider, meaning evaluating Bellman's optimality criterion for each relevant decision. That is, for each state and possible decision to consider, we calculate both the direct/immediate profit and the value of the next state one would end up in if making the actual decision, and then add these together (only multiplying the next state value by the appropriate discount factor) and use this as a basis for comparison with the other relevant decisions we want to evaluate. We then choose the decision with the highest such value.

Because of the sheer amount of decisions to consider for each tank and the exponential nature of the combination of single tank decisions into an overall decision, we have to make heuristic simplifications in our algorithm. This speeds up the algorithm and makes it less computationally demanding, but also potentially goes at the expense of optimality. All of the three measures below are discussed further in section 7.1.

First, we reduce the amount of relevant harvesting decisions to consider by only considering harvesting whole sales classes of fish. This means we have abstracted up from individual fish-level and further up from weightclass-level and to sales class level. Only in the special case of hard constraints where harvesting all the fish in the sales ready sales classes will still not satisfy the density or MAB constraint in the next state, we consider harvesting not sales ready fish, and then relevant decisions are selected on weightclass-level, starting from the largest weight class just before the fish are sales ready.

Second, we reduce the amount of smolt release decisions to consider for each empty tank to certain points in the interval  $[0, Maxfish]$ .

Finally, when combining the relevant decisions between different tanks, we have tried to exclude "irrelevant" combinations of singletank decisions before evaluating the decisions. This has the largest impact on computability, because of the exponential behaviour of the combination of single tank decisions into an overall decision.

## Value function training method

Introductions to ADP/RL algorithms often present temporal difference learning, as described in chapter 3.3.5. This means that during the forward pass of the algorithm, the value function is trained at every time step with target equal to the temporal difference target.

In our problem, which in theory is an infinite horizon problem, we have designed a training algorithm which balances the low variance of temporal difference (0) learning and the low bias of Monte Carlo methods. Our method resembles TD(n)-learning with n-step bootstrapping, where the n varies. In our algorithm, the agent first "plays" through the forward pass, which is an episode of salmon farming of a set length T, say, 60 months, using the current sampling policy as decided by the value function. In each step the agent gathers an "experience", and saves all the experience in a storage called "episodedata". One experience is comprised of a state, a decision, a corresponding immediate profit and the corresponding next state. There is then no training during the forward pass. After the forward pass is done, the algorithm then selects a number (a predefined sample-

---

size) of random "experiences"/time points from the episodedata gathered in the forward pass, i.e a set of random numbers in the interval  $[0, T]$ . For each of the selected "training times"  $t$ , the algorithm takes the corresponding state (the state the agent was in at that time step), and creates a corresponding target for the value function according to the following expression:

$$r_t + \gamma r_{t+1} + \gamma^2 r_{t+2} + \dots + \gamma^{(T-1)-t} r_{(T-1)} + \gamma^{T-t} V(S_T) \quad (5.16)$$

Where  $\gamma$  is the discount rate and  $S_T$  is the state at time  $T$ , the end of the episode.  $r_t$  is here simplified notation for  $R(s_t, a_t)$ , where  $s_t$  and  $a_t$  are the state the agent was in and the action the agent took (according to the sampling policy) at time  $t$  in the respective episode/iteration. The idea is based on TD(n)-learning, only that in our case, the  $n$  parameter is equal to the time remaining until  $T$  for each of the selected training times.

The rationale behind this training method is primarily faster convergence, because the bootstrapping term becomes less significant because of the discounting (lower bias), and more emphasis is put on the more recent reinforcements/direct profits. This also speeds up convergence because, say for a given state in a tank where the fish are not yet harvest ready, the immediate profit will be negative because of the feedcost. Training on a temporal difference target on this state alone would then give a negative reinforcement to the value function. For most states, except the ones where the fish are harvested, reinforcements would be negative, but with a small magnitude. This lays greater requirements on the careful tuning of hyper parameters such as the learning rate and the type of gradient descent algorithm used. With our training, less training is needed and the convergence is more robust because the reinforcement is more "detailed"/specific, meaning the reinforcement takes into account what will happen several steps in the future. We still need bootstrapping in the algorithm though, i.e. we cannot use traditional Monte Carlo learning because we have an infinite horizon problem and Monte Carlo learning only applies to episodic problems. Because we have an infinite horizon problems, the parameters/weights of the value function approximation has no time subscript, and we can therefore train/adjust the same parameters with different training targets.

For the actual regression done to fit the value function approximation to the targets, we use the PyTorch toolkit for this and build the function approximations in a way that we can use the automatic differentiation and adjustment tools that PyTorch offers. What is done in principle is simply for each of the selected training times calculating a loss function, which is the error between the current value function prediction of the value of the relevant state and the calculated target. This loss function is then differentiated with respect to the parameters of the value function to find the gradients with respect to each parameter. Each parameter is then updated by taking a step in the direction of the negative gradient in order to minimize the loss. While most of the algorithms follow this basic principle, PyTorch offers many variants of the "optimizer" (the function actually updating the parameters after the gradients are calculated), such as Adam, RMSprop and SGD. These differ mainly in how much they use momentum and other methods to adjust the step size.

### Exploration strategy - sampling policy

In a salmon production schedule, there are certain "key decision dimensions" which by and large determine the production schedule:

- The number of smolts released in a batch
- The number of different harvests each batch is divided into and the average weight of the harvested fish in each harvest
- The coordination of different tanks, i.e. when smolts are released in the different tanks

Realizing this, and that these are the dimensions along which exploration needs to be done, is the most innovative and problem specific breakthrough we have had in applying ADP/RL to salmon production scheduling. We believe this represents a key insight.

---

The above three are the primary dimensions of the decision (and state) space along which we want the agent to explore. These dimensions are somewhat independent from each other, meaning that in a certain state (e.g. all tanks empty), only the number of smolt and tank coordination dimensions might be relevant, while in another state (e.g. all tanks have sales ready fish), only the harvest weight and tank coordination dimensions are relevant. This dynamic, where exploring is done along certain dimensions which can be (partly) independent from each other, is quite specific to our problem, in an ADP setting.

This has been exploited by implementing a more advanced exploration strategy than the common, simple  $\epsilon$ -greedy strategy which most ADP applications apply. The sampling/behaviour policy of our ADP/RL agent is such that in  $\epsilon$  percent of the time steps, the direct profit function adds an *exploration bonus* (often called intrinsic motivation in RL literature) to the evaluation criterion (the Bellman equation) of each decision. The exploration bonus we apply is count-based, and hence resembles the Upper Confidence Bound (UCB) method (Silver (2015)). Shortly speaking, the algorithm divides the number of fish released, the average weight of each batch and the absolute difference in average weight between the different tanks into three sets of intervals. The ADP/RL agent then keeps track of how many times the agent has been in or taken actions which correspond to each of these intervals in each of the three dimensions. In  $\epsilon$  percent of the time steps, the ADP/RL then includes an exploration bonus in the evaluation of each relevant decision which is inversely proportional to the amount of times the relevant exploration dimensions have been explored, raised to a power between 0 and 0.5 ( $\lambda$ ). The exploration bonus added to the evaluation of each decision is then

$$bonus * \frac{1}{n^\delta} \tag{5.17}$$

Where  $n$  is the number of times, say, between 50,000 and 75,000 smolts have been released. The bonus is initialized as a very large sum with value larger than the largest imaginable harvest revenue. This has the added effect of initializing all states with a high value, another exploratory principle discussed in section 3.3.2.

This exploration strategy resembles the Upper Confidence Bound method in that it favours states which the agent has visited few times, i.e. has "little knowledge and is curious about". Over time, the exploration bonus becomes relatively smaller as the estimate of the true value of the state (hopefully) gets better. Also, after each iteration,  $\epsilon$  is decreased slightly, such that towards the end of the training, i.e. in the last iterations, the sampling policy is greedy/exploitive almost all of the time. The sampling policy then converges towards the target policy (see section 3.3.3).

# Chapter 6

## Model evaluation

In this chapter we will evaluate the performance of the ADP/RL model, both qualitatively and quantitatively, versus a MIP model for the deterministic version and a rolling horizon model based on the MIP model for the stochastic version, respectively. We will also use the rolling horizon model to perform a test to determine the value of stochastic solution of the salmon production scheduling problem with respect to price uncertainty.

The results by the ADP/RL model presented here are the results by the hard constraints variant. The soft constraints variant generally performed slightly worse.

All calculations were run on the Solstorm computational cluster belonging to NTNU. The specific nodes used had the following hardware specifications:

Type	Dell PowerEdge R640
CPU	2x 2.4GHz Intel Xeon Gold 5115 CPU – 10 core
RAM	96Gb

Table 6.1: Hardware specifications

### 6.1 Qualitative assessment

On the next page is a screenshot of a production plan for a 30-month period for 2 fish tanks in a constant-price scenario. It is one of the best-performing schedules we have been able to produce using the ADP/RL model.

The illustration (rotated 90 degrees), shows the time period (month) along the horizontal axis, starting from month 0 on the left, and the average weight of each weight class on the vertical axis, starting from 50 grams and going up to 7 kilograms. The upper half of the plot represents the first tank, the lower the other tank. Each number represents the amount of fish in that weight class in that time period in that tank. The smolts are already from delivery distributed around an average weight of 100 grams with a coefficient of variation equal to 10%, which is common in real production. As is also common, and given by the growth model, the fish grow into new weight classes and get a wider distribution as time goes and genetic differences come into play. The blue marked cells indicate where, when and how many fish are harvested. The reason the growth is not the same all the time, but sometimes steeper and sometimes less steep, is the changing season and that fish grow faster in warmer water during summer time and slower during winter.

The following are some key insights we can draw from qualitative inspection of this production plan regarding the performance of the ADP/RL model.

The ADP/RL model seems to release a reasonable amount of smolts in each batch, and also keeps each tank on a continuous production cycle, meaning that new fish are released as soon as a batch is harvested. This last point means the model has "understood how valuable biomass is".

A reasonable amount of fish means the amount which allows the model to utilize the density and MAB restrictions to a (relatively) high degree. This will be discussed further in chapter 7. In terms of the harvesting of each batch, it is more difficult to say without deeper analysis whether the harvesting is optimal, i.e. whether for example it would have been better to let the fish grow to a larger weight. If that is the case, the model has not in this case been able to learn that delaying harvesting and reaching a state with fish with larger average weight would have been better. However, a relatively low price differentiation between the sales classes of 2% indicates that harvesting early is more optimal, and this being a typical price differentiation seen in the real market, the harvesting done here is qualitatively similar to harvesting schedules in real production.

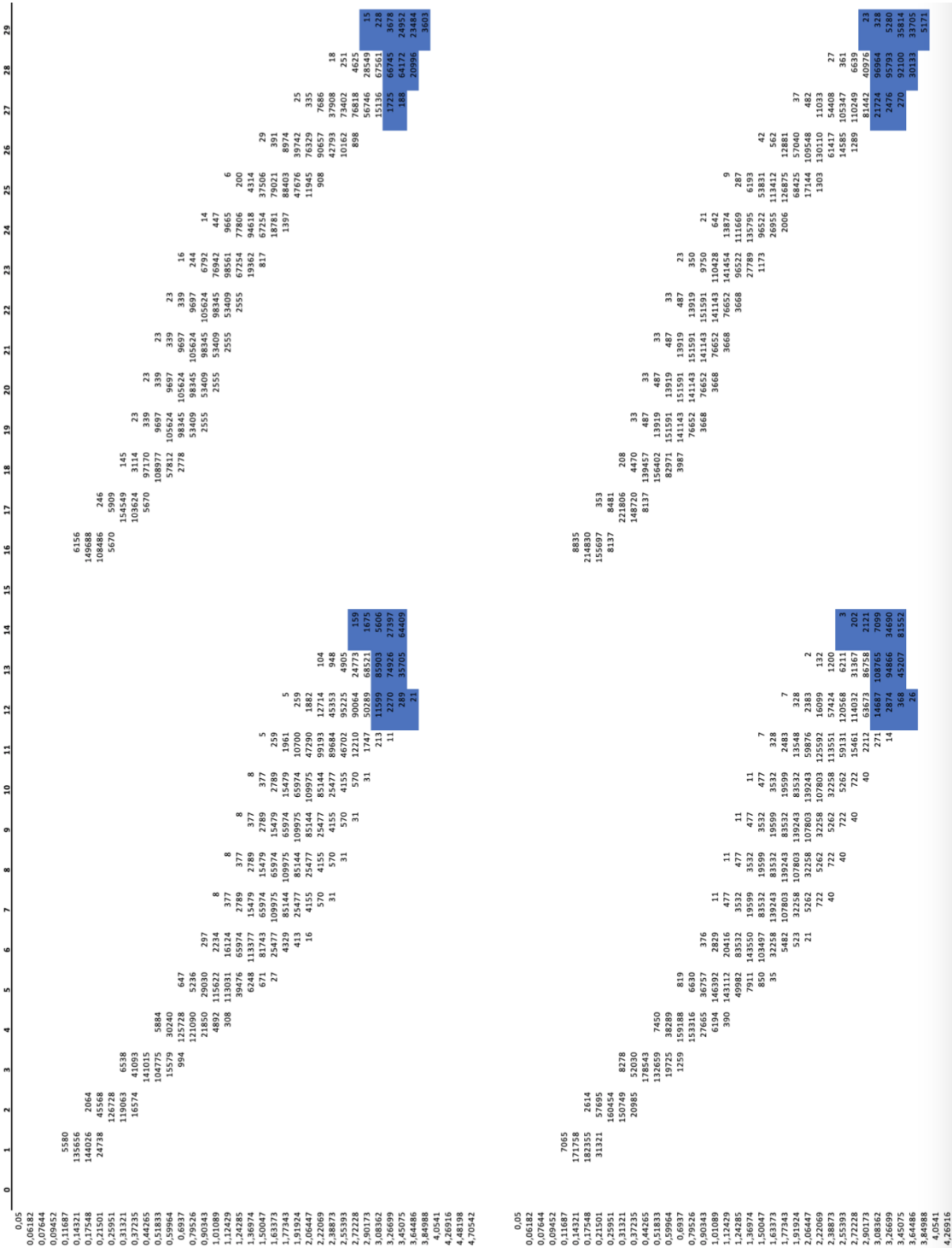


Figure 6.1: Example production schedule by our ADP/RL model



The MIP model solution for the same problem parameters (i.e. the optimal solution, not shown here) has almost the same way of harvesting the batches, but releases slightly more fish and is better able to utilize the density restriction and the MAB capacity for a longer time, meaning it lies closer to these limits for longer periods. One clear difference is that the two tanks in the ADP/RL solution are on a synchronous production cycle, while for the optimal solution. The reason they are is that the ADP/RL model only releases so many fish that even the sum of the tank biomasses never exceeds the MAB. The MAB quota in our problem parameters is, and typically would be for such a facility, set to slightly less than the sum of the tank density capacities. Again, we repeat how the tank density restrictions translates into a biomass restriction given the tank volume. In other words, to be able to fully utilize the capacity of the facility, the optimal solution has to put the two tanks on asynchronous production cycles, meaning they are in opposite phase with each other and fish are released in one tank halfway into the batch cycle for the other tank. This is a challenging realization for the ADP/RL model to make, that in order to fully utilize the capacity of the facility by releasing and harvesting even more fish, the model has to put the two tanks on different cycles. This model version has not been able to learn that yet. This will be discussed further in section 7.3.3.

### 6.1.1 Model learning development

The graph on the right is a plot showing the development of the ADP/RL agent's performance while "out farming salmon and earning profits". The upward trend which can be seen is in fact proof of machine learning. The plot is a scatter plot, plotting the total cumulative profit (not discounted) earned in the episode in NOK on the y-axis and the episode/iteration number on the x-axis. Our ADP/RL agent starts out with a randomly initialized neural network value function approximation and uses this in the policy (i.e. a value function approximation policy (W. Powell (2011))). In other words, the only thing making the agent able to perform better is that the neural network is *learning* and improving its estimate of which states are the most valuable and hence which states it "chooses to go to" (in our case, for example, what number of fish to have in a batch are most valuable). The main reason for the high variance, i.e. the seemingly large spread in earnings (earnings ranging from 200 to 600 MNOK even for a short iteration number range), is that we found the far most effective learning occurs when each episode is initiated from a random start state. A random start state means with a random number of fish at a random size in each tank. Because each episode contains 5 years of salmon farming, each tank only has time for around 3 batches harvested, but each harvesting is so valuable in terms of revenues (almost 100 MNOK) that only having time for one more harvesting, which for example might happen if the episode starts with a batch already at harvesting ready weight, the total profits earned will be much larger than an episode starting from a state of empty tanks, even if the agent's choices (actual performance) are the same. For example, with a given starting state and a given set of choices, at the end of the 5 years the ADL/RL agent might have a fish batch in a tank which is almost ready for harvesting, but not quite. This means the revenues from this last batch will not be counted in the total iteration profit, but the feed and smolt costs related to it will. The variance in earnings per iteration can also be attributed in part to the random exploration in the sampling policy. Each time step, an exploratory action is selected if a random drawn number between 0 and 1 is below a certain threshold (an important hyper parameter), and this can cause some iterations/episodes to contain more random, exploratory actions than others, and exploratory actions are by definition not exploitive (not "optimal").

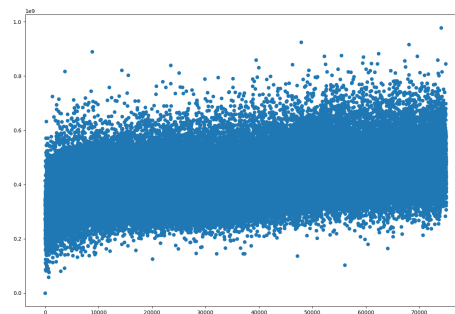


Figure 6.2: ADP/RL model development over 75,000 episodes: One episode equals 4.5 years of salmon farming, on the y-axis is the total profit the model is able to earn over this period

---

### 6.1.2 Model scalability - runtime

Model runtime is not as obvious a measure for an ADP/RL algorithm as it is for a MIP model. A MIP model is less tuneable and will generally take a relatively specific amount of time to reach a certain optimality gap (we use 1% gap). ADP/RL is a heuristic, which does not provide an optimality gap feedback during runtime, so it is generally not possible to know how close to optimality the algorithm and hence when to stop it. As will be discussed further in the next chapter, ADP/RL runtime can be seen as the time spent on each function training/iteration multiplied by the number of trainings/iterations run, the latter of which in turn is a decision totally up to the modeler and is dependent on how well and quickly the algorithm converges, which varies significantly with a large set of algorithm hyper parameters/variants (such as the learning rate). In the algorithm configuration applied for most of our results, which we found to give the most promising results, we typically applied  $\sim 50,000$  iterations. For a different configuration of algorithm parameters, fewer iterations might be sufficient. In the case of our ADP/RL algorithm then, we define the runtime as the time it took to run the amount of iterations which seemed to give the best results given the specific algorithm configuration applied, i.e. a more subjective measure.

Nevertheless, the runtime of our algorithm is an important measure because the ability to solve full-scale salmon production scheduling problem instances in an acceptable time was one of the main motivations for applying more advanced optimization methods than MIP. Recalling from our project thesis, and confirmed with other salmon production scheduling researchers' experiences, MIP models become computationally intractable already for 3 or 4 fish tank, detailed problem instances.

Some insight into the scalability of our ADP/RL algorithm runtime versus a MIP model for a 5-year production schedule is presented below:

Solution time in seconds			
Method	1 tank	2 tanks	10 tanks
MIP	< 100	< 2000s	Unsolvable*
ADP/RL	< 50 000 (1 per iteration)	< 100 000 (2 per iteration)	< 1 500 000 (30 per iteration)

Table 6.2: Illustration of total solution time in seconds (and time per iteration for APD/RL model) for different problem instances. Only illustrative, as multiple problem parameters and ADP/RL algorithm parameters greatly affect the results

\*Not solvable within reasonable time (more than a week)

We point out how this optimization problem is a long-term strategic planning problem and not an operational problem. In other words, this problem is not intended to be solved every day, instead it might go months between each time the production schedule has to be reviewed due to e.g. changes in exogenous circumstances. Therefore, after discussing with industry professionals, we believe a solution time of less than 1-2 days would be acceptable.

Also, while there is limited room for speeding up the MIP model, as the Gurobi solver is highly optimized already, there are many ways to vastly speed up our ADP/RL model, for example by something as easy as writing the model in a faster programming language like C++ instead of Python. We will discuss this further in the next chapter.

---

## 6.2 Benchmarking - deterministic optimization

The following table compares how much the ADP/RL model is able to earn over a 40-month planning to the optimal schedule found by the MIP model for a 1-tank and 2-tank facility, respectively.

Profits from deterministic solution		
Method	One tank	Two tanks
MIP	139 MNOK	268 MNOK
ADP/RL	118 MNOK (85%)	211 MNOK (79%)

Table 6.3: Summary of profits earned in the deterministic, constant-price problem version by the MIP and ADP/RL models for the one and two tank problems, respectively

## 6.3 Benchmarking - stochastic optimization

The following graphs are scatter plots of the rolling horizon model and ADP/RL model performances (profits earned) evaluated over a set of 1,000 price realizations generated by the price model. On the y-axis is the total, discounted, cumulative profits earned by the model for the respective sample, and on the x-axis is the average price in the sample.

### Rolling horizon

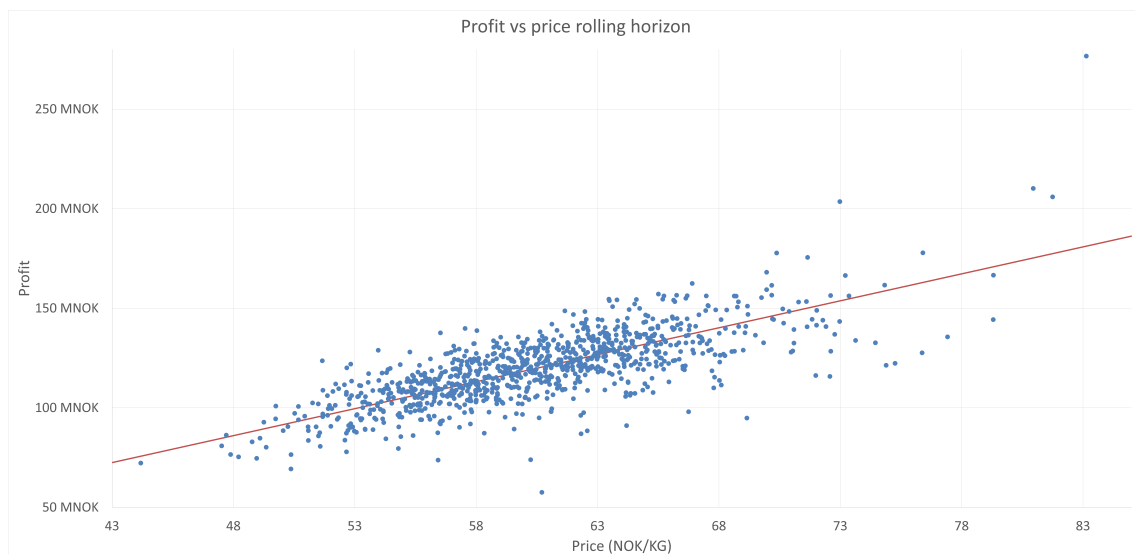


Figure 6.3: Profit vs price for one tank stochastic solution. Regression line plotted in red

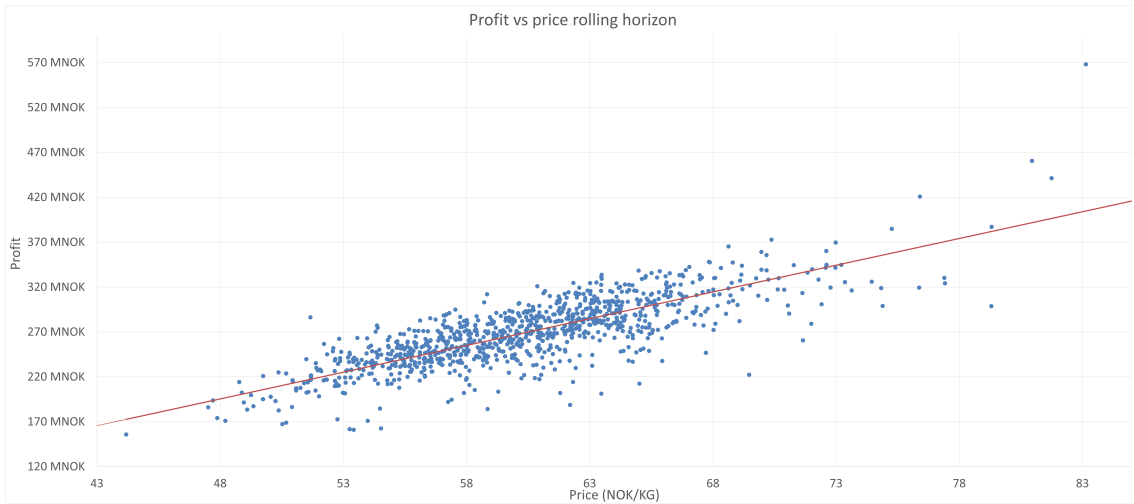


Figure 6.4: Profit vs price for two tanks stochastic solution. Regression line plotted in red

### ADP/RL

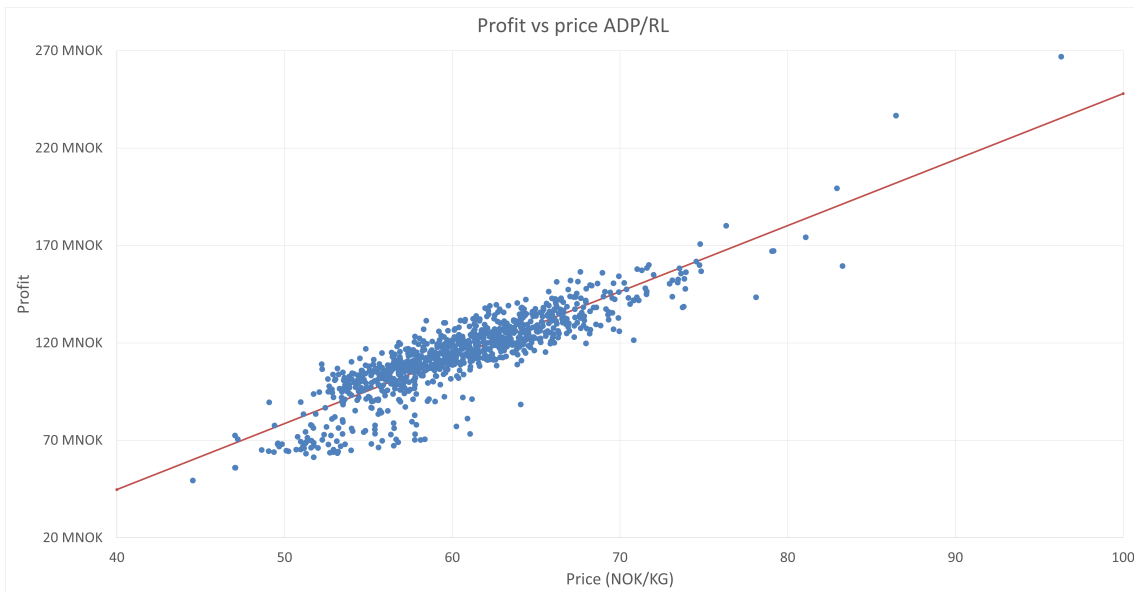


Figure 6.5: Profit vs price for one tank stochastic solution. Regression line plotted in red

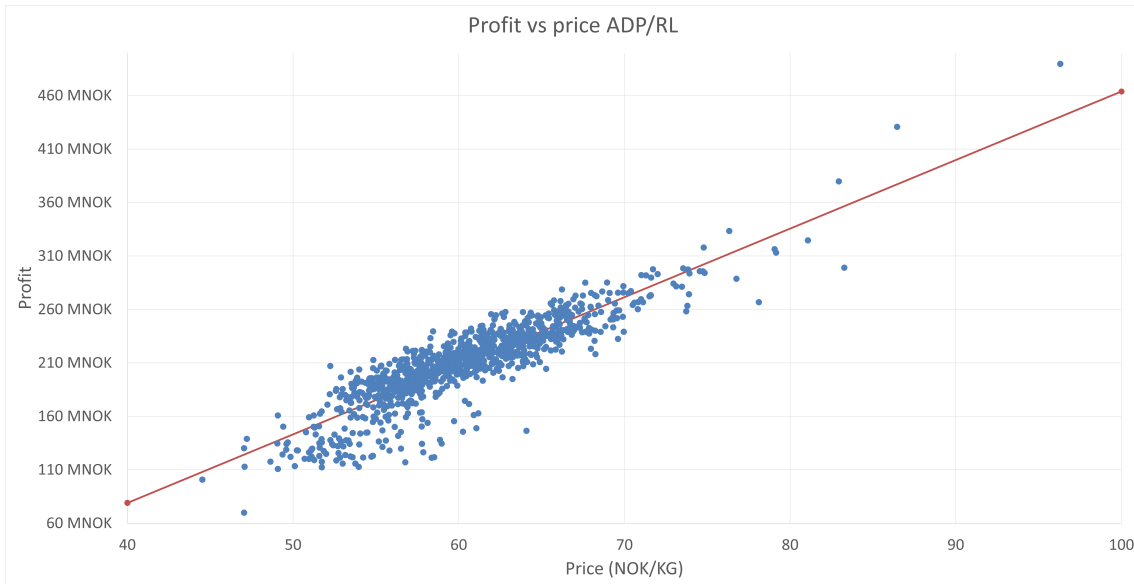


Figure 6.6: Profit vs price for two tanks stochastic solution. Regression line plotted in red

Profits from stochastic solution			
Method	Tanks	Avg profit	Regression line slope
<b>Rolling horizon</b>			
	One	120 MNOK	2.7 MNOK
	Two	270 MNOK	6.0 MNOK
<b>ADP/RL</b>			
	One	114 MNOK (95 %)	3.4 MNOK
	Two	210 MNOK (78%)	6.4 MNOK

Table 6.4: Comparison of profits earned by rolling horizon and ADP/RL models, averaged over 1,000 price realizations

### Comments on benchmarking results

The ADP/RL model earns on average 95% and 78% of the rolling horizon model when evaluated over (the same) 1,000 price realizations for the single tank and multitank problems, respectively.

That the ADP/RL model is closer to the optimal solution for the single tank problem than the multitank problem is in line with expectations. As will be discussed further in 7.3, the multitank problem has an additional feature that in order to exploit the facility capacity to the maximum, the schedules need to asynchronize the individual tank cycles. This is an additional challenge for the ADP/RL model.

The fact that the stochastic ADP/RL model performs better than the deterministic ADP/RL model in the single tank problem in relative(!) terms is attributed mainly to the weaknesses of the rolling horizon model versus the MIP model (see section 7.5). However, it could also suggest that the ADP/RL is able to take advantage of the price variations quite good in the single tank problem, at least better than in the multitank problem.

The regression slopes were calculated to indicate to what extent each model is able to benefit from a higher average price. The number can be interpreted as the additional profit the model earns for 1 NOK/KG higher average price. We initially suspected that the rolling horizon model was better

---

able to take into account the price variations, in fact we have qualitatively seen that it is, so it is therefore assumed that the reason the ADP/RL model has higher regression derivatives is due to a worse fit of the regression line to the ADP/RL results. It can be seen from the ADP/RL plots that for a small subset of the samples, the model seems to score significantly worse compared to the otherwise linear trend, something which "disturbs" the regression line and causes the higher regression line slope.

## 6.4 Value of stochastic solution

To investigate the value of stochastic solution (VSS) with respect to price uncertainty, we use the rolling horizon model. This is because in its current state, this model does not only give closer-to-optimal solutions than the ADP/RL model but also seems to be more sensitive to price variations/better at taking the price variations into account. The idea is to investigate whether the stochastic solution provided by the rolling horizon model in fact earns more than the deterministic solution methods when evaluating over the same 1,000 realized price samples.

For this particular part of the master thesis, we ask the interested reader to also read our project thesis leading up to this master and to see the results in context with each other. The project thesis included an in-depth discussion of the effect of price seasonality on the optimal production schedule for a salmon producer. Here, the large influence of the starting point of the year on the quantitative results, as well as the relative effect of different seasonal amplitudes are discussed, factors which are very much relevant for the particular quantitative results in this section as well. However, as this section has only been a minor focus in this master thesis in order to tie up a loose end from the project thesis, we will not be going thoroughly through this here, as that requires lengthy, in-depth discussions.

We test three cases:

- I. Deterministic solution with respect to a constant price. This solution is the closest (not completely equal) to the equivalent solution simply maximizing biomass output instead of profits
- II. Deterministic solution with respect to the price expectation given the price in the first period, using the forward curve model
- III. The rolling horizon model, solving each subproblem with respect to the relevant price expectation, implementing the first two steps, then seeing how the price develops and repeating. This is the same as was used in the previous section to benchmark the stochastic ADP/RL model

Average earnings over 1,000 price realizations		
Case	1 tank	2 tanks
I	121 MNOK	213 MNOK
II	126 MNOK	270 MNOK
III	120 MNOK	270 MNOK

Table 6.5: VSS research results using MIP and rolling horizon averaged over 1,000 price realizations

### Implications

The following are the key takeaways from the above results.

- The single tank problem results suggest that the value of stochastic solution is very low. The seemingly slightly negative VSS is attributed to sources of error and a small loss of optimality in the rolling horizon model compared to the MIP model.

- 
- The results suggest that the VSS is larger for the multitank problem than for the single tank problem, which is in accordance with the results from the project thesis. It takes very large price variations for the optimal single tank schedule to deviate from a continuous production cycle starting "straight away". For the multitank problem however, the coordination aspect creates an interesting trade-off between utilizing the MAB quota and receiving high prices already at quite small price variations.
  - For the 2-tank problem results, the difference between (I) and (II) indicate a large value for the salmon farmer to time production against price seasonality, which is a confirmation of the results from the project thesis.
  - The VSS seen by comparing (II) and (III) for the 2-tank problem indicates a low value of stochastic solution given that the salmon farmer is already planning to harvest as much biomass as possible during the high price season.

Critical reflections regarding the validity of these results will be discussed in the next chapter.

## Chapter 7

# How ADP/RL works for salmon production scheduling

### 7.1 Vast action space - Computational time

The action space for a multitank facility grows exponentially with the number of tanks. This is a challenge because our algorithm chooses a decision given a particular state by evaluating each individual relevant decision "manually". The computational time increases greatly with the number of relevant decisions to evaluate, as will now be explained. Our algorithm iteratively gathers training data by "playing salmon farming episodes", trains on this data and then performs this for many iterations over and over. The total computational time is thus a function of 1) the time spent on data gathering and value function training within one iteration respectively, and of 2) the total number of iterations needed before convergence is satisfying. Of the time spent one episode, > 95% of the time is spent on data gathering, i.e. "playing salmon farming". Within the data gathering, the RL agent moves forward in time, and for each time step (and state) the agent is in, it first has to generate the relevant potential decisions to consider and then evaluate each of these. Each decision is evaluated using the Bellman equation as explained in earlier chapters. Within each such timestep, generating the relevant solutions takes < 10% of the time, depending on the amount of decisions to generate and evaluate. While one decision evaluation takes vanishingly little time (0.002 seconds with our simple, not-particularly-fast Python implementation), the time to evaluate all relevant decisions grows large when the number of decisions increases exponentially. Even such a coarse decision space division as only five relevant decisions, would for a 10 tank problem give theoretically  $5^{10}$ , almost 10 million, decisions to evaluate.

This challenge remains by and large unsolved for our application of reinforcement learning. There are however several approaches to tackling this. These can be divided into three main categories: 1) Reducing the number of decisions to consider (at potential expense of optimality), 2) selecting decision in a smarter way than manually evaluating each one or 3) speeding up the computation either algorithmically or programming wise.

#### 7.1.1 Reducing the number of decisions

To reduce the number of decisions to evaluate, one can make the decision space coarser, i.e. restricting the possible model decisions. For example, instead of considering every number in the interval  $[0, Maxfish]$  (Where *Maxfish* is the maximum number of fish we consider relevant to even consider, which again is a function of the size and density restriction of the fish tank), one can only consider every 5% increment in that interval. If *Maxfish* would equal 100,000 fish, only considering 0 fish, 10,000 fish, 20,000 fish and so on greatly reduces the number of decisions from considering 0, 1, 2... fish. Other smarter divisions of the intervals are also applicable, if one knows where in the interval the optimal amount of fish is likely to be. The same goes for the harvesting



---

decision, where it is natural to aggregate up to only considering harvesting whole weightclasses or whole sales classes, rather than individual fish. Generally, we can call this different aggregations of the decision space. This sort of simplification does however go at the expense of optimality, because the theoretically optimal amount of fish could have been 11,253 fish, an alternative which would not have been considered. There is therefore a clear trade-off between increasing solution quality/optimality and reducing computational time by doing this. We will however argue that some level of aggregation will always be applicable, because real life production does not make it practically possible to make decisions at individual fish level. For example, when harvesting, salmon farmers actually sort the fish by lowering a masked net into the tank with a finely set mask size, such that only fish below a certain size can go through. Both because the salmon farmer does not know the exact distribution of the fish batch and because not all the fish might swim through the net, the farmer can only approximately adjust the size and number of fish harvested. Also when ordering fresh smolt to release in a tank, the delivery of smolt from the smolt producer usually has some error margin in terms of number of smolt. For these reasons, we argue that some level of aggregation is always applicable and that there is only a balance/trade-off to be done. This is however an easily changed parameter in the algorithm and therefore something which the salmon farmer can decide in each specific situation, depending on how fast the solution is needed.

The number of decisions can also be reduced by excluding certain "irrelevant" decisions, such as harvesting fish before they are sales ready. This is the natural starting point when restricting the decision space, and is what we mean by the term "relevant decisions".

As one decision is a combination of (sub-)decisions for each tank, one can also reduce the number of potential decisions to consider perhaps even more by excluding what combinations of decisions are considered. For example, one can exclude combinations of single tank decisions where one is harvesting more fish from a tank with less biomass than from one with larger biomass. The question is then whether or not this excludes the potentially best solution, and while it is very difficult to prove that such a combination is never optimal, we will intuitively argue that for the vast majority of practical situations, this can safely be excluded without measurable loss of optimality.

### 7.1.2 Smarter selection of decisions

In our algorithm, we generate all relevant decisions we want to evaluate and then evaluate each of this manually by calculating their Bellman equation value. As earlier stated, this corresponds to the general policy class of value function approximation policies, as given by W. B. Powell (2016) in his overarching framework for sequential decision making problems. There are other imaginable ways of doing this.

One approach would be to solve the subproblem in each timestep as a mixed integer program. This requires a parametric value function which is linear in the decision variables (or at least one that the solver can tackle) so that it can be put in the objective function of the program. A neural network is, to our understanding, not applicable as a value function in such a case, while a piecewise linear value function might be. When the number of relevant decisions to consider grows very large, this might be preferable. This also has the advantage of always being able to solve (the subproblem) to optimality, given the current parameters of the value function. This remains an interesting option for future research.

Another potential approach would be to look to other sequential decision making policies which select decisions by (in principle) taking the derivative of the value function with respect to the decision variables and setting equal to zero. Such a policy would also belong to the class of value function approximation policies. In our case, the challenge in doing this is that we do not have a closed-form, concave value function which is differentiable in all points, because of the non-convexity of the decision space. By this we mean the binary nature of harvesting versus release, which is decided by whether or not the tank is empty. This was one of the arguments for choosing ADP/RL (specifically a value function approximation policy). We are not aware of any other sequential algorithms which guarantee convergence for such mixed integer non-linear problems without proven convexity of the value function, and ADP/RL therefore remains as an applicable heuristic.

---

In his sequential decision making framework, W. B. Powell (2016) mentions three other high-level classes of policies. These are cost function approximation policies, direct, deterministic lookahead policies and policy function approximations and they might all be interesting to explore for this problem as well. We believe however that cost function approximation policies and lookahead policies are the most promising options, given that policy function approximations would quickly become both complex and limiting for the multitank problem.

### 7.1.3 Speed-up of model

An advantage of our algorithm is that there are several ways to speed it up. This section is not considering speed-up by making the algorithm converge faster and therefore need fewer iterations (which would be an obvious way to speed up the algorithm). This will be considered in the next chapter, and is considered a separate problem entirely, and what this thesis is mainly about. Here we only consider practical, implementation-related ways of speeding up the algorithm, *given* that the algorithm converges "satisfyingly", so the number of iterations are given.

One interesting approach is what is called distributed RL. By this we mean taking advantage of several computational nodes at once, performing certain tasks in parallel. This is done in many advanced reinforcement learning applications where training typically takes days or weeks. There are several ways to distribute a RL model, but in our case it would be most beneficial to gather training data by deploying several RL agents in parallel. Since data gathering is what takes up the majority of the time in one iteration, the algorithm might keep the value function and the value function training on one central node, while multiple agents are deployed to other computational nodes where they "play salmon farming episodes" and continually sends data to the value function training node. When the value function training node has received enough data to calculate new targets, the function parameters could be updated and sent back to the agents. This would be interesting in order to simply speed up the algorithm, but would probably require significant work on its own.

Our code is written in Python and without much programming knowledge from before. Python is known for not being particularly fast, and there is probably large speed-up potential to be gained by e.g. writing the algorithm in C++ and optimizing it for speed. For a practical, commercial implementation of the algorithm, this would be recommended.

## 7.2 Algorithm convergence - overall performance

In this section we will discuss the main factors for algorithmic convergence and performance, i.e. how profitable production schedules the model produces. This is based on our now relatively extensive knowledge about both salmon production scheduling optimization in general and also about reinforcement learning algorithms and their "behaviour".

Broadly speaking, the biggest, most widely applicable and most favoured branch within reinforcement learning (and ADP) is about solving Markov decision processes by some sort of value iteration while approximating the value function (i.e. a value function approximation policy in Powell's framework). At the very core of making such algorithms converge and perform are the following four key elements:

- Exploring all of the potentially "interesting" states and visiting them "enough" times so that the training algorithm can fit well to the true values of these states
- Making sure the targets provided to the training algorithm are sufficiently "good"/correct, at least in the long run. This involves making close to "optimal" subsequent decisions after visiting an interesting state in order to make the target actually resemble the true value of that state, which can be conflicting with the first point (exploration vs exploitation)
- Choosing a value function shape which is able to (sufficiently) capture the qualities of the optimal value function

- 
- Designing a training algorithm which, given a set of states and corresponding targets, is able to adjust the weights of the function approximator so that the function actually gets better at estimating the true optimal value of being in a state

These four elements are basically what reinforcement learning and approximate dynamic programming revolves around. "Solve these, and you have a functioning algorithm". Much harder in practice than in theory. Given that these elements are in place, the value function will converge towards values of states which are such that the decision making in each time step (which takes into account the values of resulting states) is done so that the highest values are reached in the long run, which is the goal of the algorithm. Sadly, reinforcement learning in this general form does not offer any convergence guarantees. Such guarantees would need to be built on a long line of assumptions regarding algorithmic details, which in turn basically comes back around to solving these four elements. Reinforcement learning and approximate dynamic programming is therefore a class of methods we find have promising features in theory, but which often prove highly challenging in practical implementation.

The first (and second) element is directly linked to the famous problem of exploration vs exploitation, which was discussed in chapter 3.3.2. Deciding on an exploration strategy is an essential part of advanced reinforcement learning, which often makes the problem one of off-policy learning. The second element is related to the first, because the sampling policy (/behaviour policy) influences the targets calculated (because these are calculated from "backed-up values"). For the fourth element, almost all reinforcement learning algorithms use some sort of gradient descent algorithm, as described earlier. PyTorch and TensorFlow are software packages intended to deal with this point (see 3.1.2). The question is then not only which specific variant of these to use (assuming differentiation is done correctly), but also of deciding on loss functions, learning rates and other hyper parameters. This is much harder in practice than in theory, and is largely a trial-and-error issue which is known to all practitioners of reinforcement learning. The specific training algorithm, i.e. what types of targets to use and how often to update the value function approximation, is also a challenge for the fourth element. This involves deciding on what type of Temporal Difference learning with bootstrapping or Monte Carlo methods (if the problem is episodic) to use.

Our problem-specific versions of these four challenges, as well as how we have tried to tackle them, will now be discussed.

## 7.2.1 Hyperparameter tuning

Typical hyper parameters in an ADP/RL algorithm include the following:

### Learning rate

Reinforcement learning algorithms typically need very low learning rates, lower than what is ideal for supervised learning. This is because of the moving targets and that one wants the algorithm only over time to adjust to the trends it finds while "playing the game". We found that a too high learning rate both caused oscillatory, unstable performance and possibly also made the network unable to capture the finer non-linearities (the concavity) of the optimal value function. We found that a learning rate in the order of  $1 \times 10^{-3}$  to  $1 \times 10^{-5}$  seemed to give more robust results for between 1,000 and 100,000 training iterations.

### Number of iterations/training episodes

The number of iterations to train the network are closely related to the applied learning rate. The following graph shows the improvement of the model for a 2-tank facility over 12,000 episodes: This is the same type of graph that was explained in 6.1.1. First, the "score" in this graph actually means the total (discounted) profits the agent was able to earn in each episode, meaning that the model was able to earn at most  $\sim 700$  MNOK during this training. Second, the large

variance in the agent performance is largely due to the random starting state for each episode and due to the random exploration in the sampling policy. Each episode here only lasts 52 months, meaning that the agent only has time for 2-3 full batch cycles. Both how many batches the agent has time for and how soon the first revenue comes in has a large influence on the total profits of the episode because of the discounting of the profits. This was explained in more detail in 6.1.1. The important point however is to see how the algorithm, for this set of hyper parameters and this sampling policy, behaves and actually "learns somethings". The slow, but steady improvement in performance (the trendline/moving average line one can imagine being drawn in the plot) is what we want to see. This graph indicates that more iterations should be run with the same hyper parameters, which is what we did. When it comes to the specific slope of the improvement, this is very problem specific and algorithm specific, and remains one of the "black box workings" of reinforcement learning algorithms. In our case, there is a definitive limit to how much it is theoretically possible to earn during an episode, which probably lies around the maximum profit earned here ( $\sim 700$  million NOK). Naturally, improvement towards this theoretical optimum slows as one gets closer. While understanding exactly how convergence/learning will occur for a given set of algorithm and problem parameters is virtually impossible, investigating this development graph can give important clues as to the tuning of the algorithm.

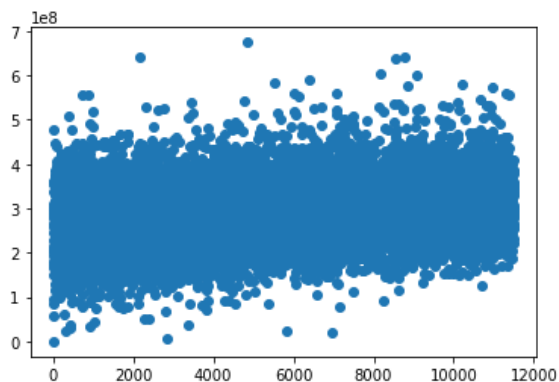


Figure 7.1: Algorithm score development graph example for one run of 12,000 episodes

### Momentum factor

For the training of the neural network, i.e. the adjustment of the network parameters, momentum is a potentially interesting factor to add to the function parameter updates. Momentum basically adds another term to the parameter update which is a small weighted average of the previous parameter updates, in addition to only the gradient term, intended to smooth out learning and avoid local optima. We found that when we did not apply any smarter exploration algorithms to the algorithm and/or the learning rate was too high, momentum could help avoid the algorithm too quickly converging towards a negative fish value if the agent received large penalties. However, once we had stabilized the algorithm and found satisfying performance in other ways, we found that adding momentum to the parameter updates seemed to have a counterproductive effect on the learning. This could be because of a relatively large amount of "noise" in the targets, which when combined with momentum seemed to make the algorithm get "stuck" without improvements quite easily.

### Discount factor

The discount factor is one of the most important "tools" for making reinforcement learning algorithms work. Because of our training algorithm design, and the way we calculate our targets, the discount factor basically decides how much weight to put to each direct profit earned in the subsequent time steps. It is well known that lowering discount factor in many cases makes ADP algorithms converge more quickly (W. Powell (2011)). This depends on the nature of the problem but is due to the fact that while all future rewards should be counted in the decision, often satisfyingly good decisions are found easier (when the value function estimates are inaccurate) when putting more weight on short-term rewards. It is important to note that the discount rate both affects the decision making in each step (through the Bellman optimality criterion) and the value function training targets. While the discount rate in problems like the salmon production problem have linkage to the concept financial time-value of money, including a discount factor less than 1

---

is a general way for ADP/RL algorithms to make the value function (and the Bellman equation) converge to a finite sum.

In the case of salmon production scheduling, an interesting dynamic is the following: The value function should in theory incorporate all future profits earned starting in a state and behaving "optimally". In salmon production however, it is often more useful to think in terms of *batch cycles* than one step profits. The value function for the single tank problem (and the multi tank problem) can therefore be divided in two terms: first, the value of the fish that are currently in the tank, i.e. the currently "active" batch, and second, the value of all the subsequent batches. While the order of magnitude of the value function depends on all the future batches as well, the *changes* in the value function for different states is mostly related to where in the cycle the *current* batch is. And more, the optimal decisions to perform in a given state seems to by and large be guided by estimating the correct value of the *current* batch and is less dependent on estimating the correct value of the subsequent batches. Because we know that batches typically take 15-22 months, adjusting the discount rate just so that the harvesting revenue at the end of the batch gets sufficient "weight" but without getting too much "disturbed" by the subsequent batches seemed to give more robust and quick convergence. A discount factor in the order of 0.92-0.97 seemed to work best. The obvious drawback with adjusting the discount factor using in the algorithm away from the correct financial discount rate is that in theory, the optimal production schedule is dependent on the financial discount rate. (E.g. higher discounting means one wants to harvest a batch sooner.) In ADP/RL models where the discount factor is adjusted, this effect is lost. Luckily, the loss is quite minimal in our case. Nonetheless, ADP/RL modelers typically distinguish between the theoretical financial discounting rate and the algorithmic, practical "future significance" discounting factor.

### 7.2.2 Exploration vs exploitation strategy

The innovative and problem specific exploration strategy we have achieved the most promising and robust results with was explained in 5.3.3.

The count-based exploration strategy we implemented had the effect of always making sure that the algorithm had tested the different alternatives "enough times" and that at least no alternative was left unexplored, which is what one generally wants. However, the great ADP/RL challenge of exploration vs exploitation is not only about ensuring sufficient exploration (that would have been easy), but about ensuring the proper *balance* between exploration and exploitation. In order to estimate the true value of a state, the sampling policy ideally should make a long line of optimal decisions *after* visiting the state one wants to investigate, because the optimal value is by definition dependent on subsequent optimal actions. Our exploration strategy does not involve any randomness in it's typical sense, instead it simply for a long time (many iterations) makes sure all alternatives are "tested" a certain number of times. The balancing act therefore comes in tuning the sizes of the exploration bonuses and, more importantly, the  $\epsilon$  parameter which determines what share of time the exploration sampling policy is used vs the exploitation sampling policy (i.e. the greedy policy). This is similar to the issue of the standard  $\epsilon$ -greedy policy and is closely related to the algorithm learning rate as well. While the problem specific exploration strategy we have designed has been a successful measure in this thesis, further development is needed and extensions or improvements of the strategy, for example involving some element of randomness, might speed up or even improve convergence.

### 7.2.3 Value function shape

As discussed, we have focused our efforts on two types of value function approximations, both being parametric functions. Two questions arise when deciding on such a value function: deciding what should be the inputfeatures to the function (the "x") and deciding what basic shape the function shall have.

The "custom" value function approximation has the main advantage of faster training, i.e. faster

---

convergence. The idea is that we can "skip" a lot of the training a neural network requires only to start approximating the shape of our problem-specific optimal value function, which we have already decided on when we design our own value function. In the case of a custom value function, the training time can also be considerably shortened because one then has the ability to initialize the parameters of the function to something that one guesses is close to the optimal/true value function. In other words, we can initialize the value function to something we believe is at least "pretty good". There are two main drawbacks to using a custom value function however. The first and most fundamental is related to how such a pre-defined function will always remain limited to the shape it has. When the true shape of the optimal value function is both unknown, highly complex and most likely not continuous at all points, it is rather hard to "guess" the optimal function shape and the output space of the value function approximation chosen will (most likely) be a restricted set which does not contain the exact true, optimal value function. If one uses a custom value function, one typically hopes one is able to decide on a function shape which captures the qualities of the optimal value function sufficiently to make (quite) good decisions. This is the case for most ADP applications. The second drawback is related to practical implementation of the function and attempting to train its parameters. Because the gradients of the loss function with respect to the parameters typically have very varying orders of magnitude (at least when the value function shape is not linear in the input features, a linear value function being a poor fit for our particular problem), actually calculating the parameter updates is challenging and causes for poor performance of the algorithm. This latter problem should however be solvable beyond what we have been able to do, possibly with more programming expertise at hand.

A neural network has one theoretical and one practical implementation-related advantage over fixed-design value functions, but also introduces new parameters to decide on, with limited theoretical foundation, in the implementation of the algorithm. The theoretical advantage is related to the *universal approximation theorem* (see 3.1.1), which means that the value function in theory can approximate any function shape. In other words, we need not worry about finding the optimal value function shape, but we do also not need to worry about whether or not the shape of the function becomes a "limitation" for the function. With a neural network, you can in theory always increase either the depth or the breadth of the network and then you are sure that at least it is theoretically able to capture the shape of the value function. The practical advantage when it comes to implementation is related to the extensive, well-developed software framework packages by e.g. PyTorch, which makes training both simple and more easily optimized.

We have found that we are able to train a neural network much more robustly than a custom value function in the case of our problem. While the neural network requires many more training episodes, it converges more robustly and satisfyingly, and still has about the same computational time as using a custom value function because our implementation of the custom function in Python is not able to take advantage of the very fast underlying code for neural networks which are built in the PyTorch tool we used. After analyzing how the parameter values change as the training goes along for the custom value function, we are also more convinced that neural networks have more potential in terms of reaching optimality for the salmon production scheduling problem, mostly because of the unknown, complex shape of the true optimal value function, which most likely does not have an ordered, "pretty" shape.

For both types of value function approximations however, the modeler needs to decide on what should be the input variables to the function. In our problem formulation, a state contains information regarding how many fish are in each weight class in each tank. For many ADP/RL applications however, one attracts certain "features" from the state information from which one believes the optimal value can be calculated. In the case of salmon production scheduling, there are two natural alternatives we have tested. The first is using as input the numbers in the state information directly, i.e. having one input variable for each weight class for each tank, in addition to the salmon price at the relevant time and the time of the year (in the stochastic model only). The second is calculating from these numbers the total number of fish in the batch, the average weight and the total biomass and making the value function a function of these. In the case of the neural network we have experimented with different options for input features and found that a smaller network with only one hidden layer and a breadth of 20-60 nodes in the hidden layer (depending on the number of tanks) with the number of fish, average weight and biomass of each tank, along with the total facility biomass, the current salmon price and the time of the year as

---

input features performs best, suggesting that the true optimal value function could be a function of these features alone. When using only these features, the network could be smaller (and ergo have fewer parameters), which seemed to make the training able to concentrate on extracting useful information from these features alone and thus not need more detailed information about each weight class. An important realization here could be that having one input variable for each weight class in each tank gives a very large theoretical input space. However, because in any relevant parts of the state space we only ever have fish in a few weight classes at a time (because the fish have a rather narrow distribution around an average weight and fish never are in e.g. all weight classes at the same time), we only ever visit a very small part of this theoretical input space which the neural network "trains to handle". This suggests that a deeper network is needed when using these as input features, because a deeper network (meaning more layers) is more able to make higher-level abstractions of the input features, and the network needs to understand that it should look at the combinations of the different weight classes of fish and not necessarily "base the calculation" on e.g. single weight class variables. Deeper networks are however more challenging to train. Also, exactly because the fish always are in some sort of distribution and hence always only are in a few neighbouring weight class of all the possible weight classes, it is natural to assume that the neural network will be able to extract quite much information about the value of the state from the number of fish and the average weight alone, without being explicitly "told" the distribution of the fish, at least enough information to be able to make sufficiently good decisions (which is the RL goal). Further research, which might get to an even finer/deeper level of value function tuning closer to the optimal value function, could experiment with giving the network additional features with information regarding the fish distribution, such as variance or median weight as well.

We would like to stress here that deciding on the depth and breadth of the neural network applied is one of the elements which cause reinforcement learning experts to say that RL in many ways is as much an art as a science. While the theoretical foundation for understanding how neural networks learn is solid, actually reasoning the exact size and type of neural network to work best for any given problem is next to impossible. Practitioners play by certain guidelines, as we have been presented with by amongst others professor Keith Downing at NTNU. Nonetheless, what turns out to actually work best for a particular problem and coordinating all the different decisions of a reinforcement learning model is something that needs a combination of both deep problem understanding and thorough RL modeling experimentation.

## 7.2.4 Value function training algorithm

By the value function training algorithm we mean both how the targets for the value function are designed and calculated, how often the function parameters are updated and the method applied to adjust the function parameters in each iteration. This was explained in chapter 5.3.3.

The targets we feed to the neural network for each training sample were explained in 5.3.3. We found that our version of the TD( $\lambda$ )-algorithm was much more efficient in training than the classical TD(0)-learning which is typically used for illustrations of RL-algorithms. At the same time, because we have a varying bootstrapping length, we found that not only the first time steps in an episode could be used for training, but also using the later time steps where the bootstrapping term "comes sooner" is beneficial. Our findings suggest that having a varying bootstrapping length gives both faster initial learning and less bias (as inspired by Monte Carlo approaches), but also more stable learning will less variance than having no bootstrapping would give. Our approach is made possible because of the infinite horizon nature of the problem.

Choosing training samples as well as the best optimizer algorithm is another element to the "art" of reinforcement learning. We found that using a mini-batch training algorithm where we choose almost all of the experiences gained during the relevant episode while at the same time having an episode length which gives time for 2-3 full batch cycles of harvesting (typically 55-60 months) is the most efficient. It does not put too much weight on one episode, but still makes the experiences from the episode relevant enough by at least having more than one full batch cycle. (What we want the network to "focus on" is how the batch of salmon develops and gives a profit). We also found that the RMSprop optimizer by PyTorch gave the most robust learning. We cannot exclude that other algorithms could not potentially help find better optima, but the way RMSprop uses

---

a decaying learning rate seemed to be the "safest" alternative in avoiding divergence. However, while choosing and tuning the specific network optimizer did show performance differences for our problem as for most RL problems, we found this had less effect on our problem than what many RL modelers experience. We therefore do not believe this will be the most important area of research going forward.

### 7.2.5 Constraint management challenges

One challenge specific to the application of ADP/RL to salmon production scheduling, is the management of the tank density and MAB constraints. Most reinforcement learning research does not come from an operations research background, and hence is not "used to" this type of constraint management. This is more common in the ADP community, but still, the nature of most ADP applications is such that the challenge is more about finding the optimal "balance" of the decision variables and less about managing feasibility within the constraints. In the application to salmon production scheduling however, the management of the tank density and MAB constraints is essential to the problem. This is because the optimal solution typically is the one which as closely as possible exploits these constraints, as these constraints limit the biomass output of the facility.

There are two main ways of modelling constraints, as hard constraints or soft constraints. We have tried both. Whether or not the constraints are hard or soft constraints in real world production is not straight forward to answer. Physically, it is for sure possible for the biomass in the tanks/facility to exceed the density or MAB restrictions, because these are merely restrictions imposed by the authorities to ensure fish welfare and sustainability, respectively. The individual, actual salmon producer's preferences then decides whether they should ideally be modelled as hard, unbreakable constraints or soft constraints with large penalties imposed when breached.

Modelling density and MAB constraints as soft constraints means implementing penalties in the direct profit function for choosing decisions which send the facility into a state in the next time period where the density and/or MAB constraints will be broken. In our model this proved to be a great challenge for the algorithm, in the sense that the convergence was highly sensitive to the magnitude of the constraint penalties. Too high penalties, and the value function often converged towards that "it is best to not release any fish at all and not risk a penalty" or, with too low penalties, the model seemed not to care about the constraints at all. This convergence is again strongly and in a highly complex way related to the learning rate and the exploration strategy used. For example, a high learning rate in combination with a high penalty would give the "negative experiences" too large impact on the value function parameters to say that the "value of fish" is low or negative. Higher penalties therefore suggests lower learning rates are necessary, but also that more clever exploration of different amounts of smolts released is necessary. Even experimenting with these relations however, the soft constraints version of the algorithm proved challenging. The balance here would probably need further in-depth research and more advanced exploration strategies.

Modelling as hard constraints means removing all decisions from the "relevant decisions" in each state which will lead to infeasibility in the next step. This of course prevents the model from ever breaking these constraints, but also limits the "learning potential" of the value function, in the sense that one ideally would like a value function which is so "smart" and sophisticated that it can "balance" around these constraints more delicately. It is also physically, in real production, more realistic in our understanding to model these constraints as soft constraints with large penalties. Our model has however performed better with hard constraints, more efficiently learning to release the "right amount of fish". In this case, we shape the direct profit function to impose a large penalty on harvesting fish before harvest ready weight (one should not harvest fish which will not be sold as food). If the model chooses to release too many fish (in the sense that the biomass restrictions will be met too early in the growth process), the model then has to harvest some of the fish before they are sales ready and then receives a penalty. This way, the model seemed to more efficiently and more robustly learn which amount of fish was the "right amount".



---

## 7.3 What characterizes the optimal solution and how ADP/RL can find it

Very broadly speaking, the optimal production schedule for a salmon farming facility is the one which exploits the tank density restrictions as well as the MAB quota to the maximum possible while at the same time taking into account the development of the salmon price. That is at least true in the problem formulation/version we are looking into. Producing and selling salmon has (historically at least) been highly profitable, and salmon farmers therefore generally want to produce as much biomass as possible and also to sell the salmon in the high price periods. The tank density restrictions and the MAB quota are therefore essential to the problem, as these are what limit the output of the facility.

Salmon farmers try on a weekly, monthly or yearly basis to figure out what is the best production schedule. There are then, as discussed also in 5.3.3, three key decision dimensions which by and large determine the production schedule:

- The number of smolts released in each batch
- How to harvest each batch, i.e. when to harvest, at what weight and in how many turns
- The coordination of the batch cycles of the different tanks, i.e. when fish are released in one tank versus in the others

These are the decisions which together with a growth model forms a production schedule and therefore these are the decisions the ADP/RL modeler needs to figure out how to solve. We will now, for each of these, discuss an ADP/RL model's ability to find the optimal decision with respect to these three dimensions.

### 7.3.1 The number of smolts released

One important insight is that if there was no density restriction or biomass restriction, there would, in each specific situation, be a theoretically optimal harvesting weight. This is because at one point, the feed cost and added time discounting of the harvesting revenue would outweigh the added biomass (which is marginally decreasing) from fish growth and the higher price per kg for higher sales classes (only up to ~6 kg). Additionally, the market does not necessarily want as large fish as possible, and most of the demand for salmon is for fish between 3 and 6 kg. So there exists a theoretical "optimal harvesting weight". However, in real land based salmon production, the tank density restrictions makes the situation more complex than this. When planning production of a single batch of fish, the salmon farmer then has three options. Theoretically, one could release so few fish in the tank that even at the optimal harvesting weight, the tank density restriction is not met. Alternatively, one could release more fish so that the fish would need to be harvested before this theoretically optimal harvesting weight (because the density restriction is being met). Finally, one could try to balance these so that the density restrictions are met exactly when the optimal harvesting weight is reached. It turns out, as all salmon producers have figured out and as our mixed integer programming solving a small problem version to optimality has showed, the balancing option is better. This is due to the marginally decreasing value of larger fish. This however involves a careful balancing in each situation, because the question of how many smolts to release directly relates to the weight of the fish when they have to be harvested. Too many fish, and some might even have to be harvested *before* they reach sales ready weight (~3 kg), and this is very undesirable both for ethical and economical reasons. (Economically because one then has had to pay smolt costs and feed costs for fish one cannot earn revenue from). The optimal solution would be to release so many smolts that the density restrictions are exactly met when the fish reach the weight where (ideally, according to the balance between added biomass and increasing feed costs described above) either the whole or the largest part of the batch is harvested.

The number of smolts therefore very directly relates to the utilization of the tank capacities and therefore to the optimal production schedule. One unequivocally essential aspect of the optimal

---

value function is then that it is concave in the number of fish in a tank. Our efforts have shown that the value function therefore needs to have some sort of non-linear, concave (at least in the number-of-fish dimension) shape. If a custom function with a pre-determined shape is applied, then one of the "tasks" of the ADP/RL algorithm is to move the (local) maximum ("top") of this function to where the optimal amount of fish is.

Neural networks on the other hand, have the interesting property that they can (in theory) approximate any function shape because of their inbuilt non-linearities (see background chapter). They are however initiated randomly. A neural network value function therefore has some other interesting requirements for the training. First, the network must "understand" that the value function should not be linear and/or constant such that all fish amounts are associated with either a positive or a negative value. One key learning has been that if the training algorithm gets too many targets with large negative values *early* in the network training (for example if it releases too many fish and gets a large penalty for either harvesting too early (hard constraints model) or for breaking the density constraint (soft constraints model)), then the network often very quickly converges towards parameters which say that all numbers of fish have a negative value. The ADP/RL agent then seems to quickly decide that "it is better not to release any fish at all and not risk the penalty". The exploration strategy therefore needs to, especially in the early iterations, explore "good" numbers of smolts released "enough" times.

Second, the network needs to learn about the convexity of the optimal value function, and to be (ideally) tuned very finely to find the maximum value with respect to the number of smolts released, which goes back to the discussion above. This brings us to the second key learning in this respect; how finely and robustly this function "top" is determined basically determines to a large extent how close to the optimal solution the ADP/RL algorithm will get. This is because, again, the amount of smolts released is the number one most important quality of the optimal production schedule. Both because of the fact that the targets are moving and dependent on how "well" the harvesting of the respective batch released is later performed, and because such finetuning of a neural network requires a lot of training iterations and very carefully tuned training hyperparameters, this proves to be highly challenging. This is also one of the reasons we during the work with this thesis were told by RL experts that "we should not have too high hopes of reaching "optimality"", because this sort of finetuning of the weights of a network with a RL algorithm (as opposed to supervised learning where the targets are standing still and are always "correct"(!)) is very challenging. If this is solved however, which there is no theoretical reason why it should not be, ADP/RL will most likely be able to perform very close to optimally (< 5% difference by our estimates), at least for the single tank problem (for the multitank problem, the coordination of the tanks makes for another challenging dimension). This seems therefore to be the foremost challenge of solving salmon farming with ADP/RL, and therefore will be a key focus area for future research.

### 7.3.2 Harvesting

The logic and reasoning in the above subsection is further complicated by the fact that the fish in a batch do not grow uniformly and their weights typically follow an approximate normal distribution. The "ideal" harvesting weight discussed above is therefore not straight forward when talking about a batch, because when the average weight of the batch is at the ideal weight, the largest fish will be much larger. Ideally, one would therefore harvest only the largest fish once they reach the ideal weight, and let the rest keep growing until that weight. This way one even further maximizes the value of a batch and the utilization of the density restriction. Following the logic from the above subsection, if there were no practical issues and no costs related to harvesting fish, one would ideally harvest a batch continuously, starting with the largest fish, once the tank density restriction is exactly met, and then continuously harvest the largest fish until there are none left such that one always "balances" at the density restriction. In real production however, this is of course not possible, both because of considerable practical efforts involved in harvesting, considerable setup costs and also due to the fish experiencing unwanted stress during harvesting. The "balance" in the optimal solution therefore is about deciding in how many turns to harvest the batch and how many and how large fish to harvest in each turn. Larger setup costs would imply fewer turns, and vice versa.

---

For an ADP/RL model to strike this balance, the value function would need to have learned very exactly the true value of having fish at each possible harvesting weight, which involves incorporating the fixed harvesting cost. Striking this balance close to optimally is highly challenging and increasingly challenging for the last "percentages" of optimality. This is probably the biggest challenge of the three dimensions discussed in this subchapter. When fish are ready for harvesting, the model in each time step has to make the decision between harvesting and having fewer (or no) fish left and not harvesting and having larger average weight and larger biomass in the next period. In other words, the model has to "resist the temptation" to receive a large revenue directly by harvesting now. This is in turn linked to the discount rate applied. This is one of the issues that probably becomes more challenging when using only number of fish, average weight and biomass as input features, instead of the full distribution. The algorithm then probably needs to many times experience having fish at different average weights to be able to understand that it then also receives higher prices (which it is also fed as input features). In other words, a deeper network might be necessary for the model to sufficiently learn how which price (i.e. which sales class) is relevant for different average weights. This, understanding when to combine the biomass with which sales class price, would probably require more levels of abstraction, which suggests a deeper network is appropriate, which in turn makes training possibly even more challenging. We believe there is much to gain in doing further research on different input features and different neural network sizes for our problem, but also on even smarter exploration strategies for making the model explore different average weights for harvested fish. Switching to mixed integer programming for the harvesting decision is another interesting option left for future research.

### 7.3.3 The coordination of different tanks - opening up for other methods?

Given the presence of a MAB quota which is granted to the facility by the authorities, the salmon producer also wants to utilize this capacity as much as possible. Given a set of fish tanks, each tank has its own schedule of batches, or batch cycles. In the case of only a single tank, and given a "start point" in time where the facility is empty, the optimal solution is to release fish straight away and maintain a continuous cycle of batches (assuming relatively small price variations). In the case of multiple tanks and an overarching MAB quota however, one can produce more biomass per unit of time over the next years if one starts the tank cycles out of synchronization, such that while one tank has harvest ready fish, the other tanks still have fish in the growing phase. Because of this feature, facilities are typically designed so that the sum of the tanks' density capacities (which translates into a biomass capacity given the tank volume), is higher than the MAB quota. This way, by having asynchronous production in the different tanks, one can utilize more of the MAB quota at all times. The question then is whether to for example to "distribute" the starting time of each tanks production cycle evenly over the next batch production lead time or instead to have almost all the tanks start their cycle immediately and then only put a few tanks on an opposite-phase cycle such that the MAB quota is respected. The issue is made even more complex by the variations in salmon prices, because given that one expects a high price season and a low price season, one might want to "concentrate" more of the tank cycles so that more tanks harvest during a certain period, even though this might mean slightly worse utilization of the MAB quota.

For an ADP/RL algorithm to solve this challenge, two factors need to be in place. First, the value function form and the algorithm needs to be such that it "understands" the symmetry between the tanks, and that the tanks are identical. Whether or not two tanks "switch place" should not have any effect on the value of the state. Second, it needs to understand that too much total biomass in the facility is not good, and learn to balance the MAB constraint the same way it learned how to balance the density constraints. While not exceeding the MAB, the more biomass the better, generally speaking. The latter will most likely be the greatest challenge for an ADP/RL algorithm to achieve the last few percentages of the optimal solution.

When it comes to the aspect of combining different, equal tanks in a schedule, the decomposed structure of the problem becomes apparent. This is what we exploited with the decomposition heuristic algorithm presented in chapter 5.3.2.

---

It is important to note that while the single tank optimization problem naturally lends itself to dynamic programming approaches, the advantage of such approaches when it comes to the multiple tank problem is not as apparent. The single tank problem is the typical control of an entity/resource over time which ADP/RL methods deal with. When introducing multiple tanks however, while there is an important element of coordinating the different tanks, the single tank problem remains more or less independent of the master problem. This is because our research so far has shown that the optimal solution (often, not always) has the quality of the individual tanks being initiated on a steady state "batch cycle" (releasing one batch, harvesting and then releasing next batch immediately in a steady state) and then the coordination of the tanks remain the same into the future.

One of our most promising suggestions for future research therefore is about how to exploit the decomposed nature of the problem even more while still applying the promising, more scalable qualities of the ADP/RL approach to the single tank sub problems. Hybrid approaches to this might even turn out to be more promising than applying ADP/RL to the multitank problem directly.

## 7.4 Value of stochastic solution

Solving a stochastic optimization problem is typically much harder computationally than solving the corresponding deterministic problem. This is because instead of merely finding a fixed solution value of all decision variables, one has to find an optimal *policy*, which is basically a given value for each decision variable for each possible scenario. This is also the case for the salmon production scheduling problem. As the focus on salmon price stochasticity has been a major focus in both this master thesis and the preceding project thesis, it is natural to ask about the actual value of stochastic solution. In other words, is it worthwhile to solve the stochastic problem, actively taking into account the price development, or is the salmon farmer basically just as well served with assuming either a constant price or a deterministic price expectation.

The results in 6.4 suggest that the value of stochastic solution is rather low compared to deterministic optimization with respect to the price expectation (i.e. the price forward curve), but high compared to deterministic optimization with respect to a constant price. The latter gap is however comprised mainly of the difference between deterministic optimization with respect to a constant price and a realistic price expectation assuming the correct price seasonality. Our results alone therefore suggest that the salmon farmer would be practically equally well off by optimizing deterministically with respect to a realistic price expectation based on a reasonable price seasonality.

These results are very dependent on the quality of the price expectation available to the salmon farmer. The price expectations given by our price model gives rather good ideas about how the price will actually develop, as can be seen in 5.2. In reality, there might be more uncertainty also related to the seasonality of the price than what is assumed in our price realizations. We know that salmon farmers use forecasted harvesting volumes as an important indicator of future salmon prices. Due to significant volatility from other sources as well, and the sometimes limited availability of this information, good price forecasts might not be available for much longer than a year into the future. Salmon price forecasting is very difficult, as discussed thoroughly also in the literature (Guttormsen (1999)). However, this does not only affect the advantage to optimizing deterministically with respect to the price expectation, but also to stochastic optimization, and the reason is the long production lead times of salmon. Therefore, the VSS given by comparing (II) and (III) in 6.4, might be larger in reality when less good price forecasts are available. Our results are therefore far from definitive regarding the VSS.

Based on insight in the problem and the optimal solution, the low VSS might be reasonable. A basic realization is that it is hard, or maybe even not desirable, to account today for (large) uncertainty 18 months into the future, i.e. when smolts released today will only be harvested after 15-22 months. Even for the multitank problem (the singletank problem is not really necessary to consider here, because no actual salmon farmer only has one tank), the mechanisms which might provide stochastic flexibility are mainly the coordination of the different tanks and the

---

harvesting timing. However, given the amount of fish which have already been released when these two mechanisms become "relevant" (18 months after), the flexibility left to the salmon farmer is greatly reduced, because of amongst others the density and the MAB restrictions. Again, the reader is referred to our project thesis for in-depth discussions on this matter. The main insight we would like to point out from this is that one batch of salmon alone is so valuable and so profitable that producing as much biomass as possible is often the best solution, even in the face of (small) price variations. Only at relatively large price variations ( $> 10\%$ ), the optimal production schedule is significantly changed.

Qualitative inspection of the ADP/RL model shows that the stochastic version of the model is having trouble actually taking into account the price variations. As mentioned earlier, the addition of the time of year and the salmon price as input features into the neural network (or any other value function approximation) suddenly makes the ADP/RL agent's challenge much harder. This is mainly because the correlation between these input features and the value of the state is totally different and much weaker than the correlation between the features related to the fish themselves and the same value. While a neural network in theory is perfectly capable of understanding such "differences in correlativity", in practice this makes the learning much more difficult. It can even have undesirable consequences on convergence and overall performance according to our experience. After all, the job of the neural network is to find the true correlation between each state feature and the value of the state. The findings suggest that deeper networks might be preferable, to allow more levels of abstraction.

So, what does this mean for future research? The results given here have too many uncertainties and error sources to be conclusive in that stochastic solution is not worthwhile to a salmon producer. Also, experienced salmon farmers we have discussed this with state that this issue is of great importance to them and that they continuously and actively try to "harvest their fish when prices are high". Our conclusion will therefore be that more research is needed. Our literature search revealed a growing amount of literature on both the salmon price and on salmon production scheduling optimization, but we see a lack of researchers looking at the intersection, which is what motivated this thesis in the first place. And even if the value of stochastic solution is relatively low, the value of developing high-quality, reliable price forecasts and using these in deterministic optimization models is very large.

## 7.5 Critical reflection

### 7.5.1 Results validity

Because of the combination of a short time-horizon and end-of-horizon challenges in the subproblem, the rolling horizon model sometimes makes some odd choices, such as delaying release of smolt overly long in order to time for price seasonality. In other words, the rolling horizon model is not perfect.

Furthermore, as we also showed in the project thesis, such quantitative experiments are very sensitive to the length of the planning period and the start month of the planning period, because small differences in the models' choices might make it so that one model has time for one more harvesting during the planning period. One batch harvesting alone brings in so much revenue that this makes a large difference in the profits earned. Especially in the 2-tank problem, selecting the appropriate planning period in order to exclude such errors and let all models have "time for" equally many batches harvested is challenging and an important source of error in such experiments.

For these two reasons in particular, both the VSS results and the stochastic benchmarking results need to be taken not as definitive measures in any way, but as indicators. For the benchmarking of the ADP/RL model, this is not very important as it was only ever meant to give an idea of how good the ADP/RL model was performing. We have, as discussed throughout this chapter as well, also qualitatively assessed the performance of the ADP/RL model, and the assessments are in line with the quantitative results in chapter 6. For the VSS results however, this makes us unable to draw definitive conclusions. Nevertheless, we have analyzed the results and tried to minimize such

---

error sources and therefore believe the results' implications for the VSS have merit.

Coming from the linear programming community, it is also important to point out how the ADP/RL model is not like a typical or exact optimization model with a steady convergence that runs until some optimality gap measure is reached and then returns a result where the quality of the result is known. Instead, ADP/RL is a heuristic. As was discussed also earlier in this chapter, convergence is (seemingly) highly random, and knowing how many iterations and other hyper parameters are needed to give the best results is almost impossible. Any small change in parameters might give a different result and require a different number of iterations to reach the global optimum reachable by the ADP/RL agent, or cause the model to get stuck in some local minimum. The performance of our ADP/RL model is hence not a fixed measure either. Rather, the ADP/RL results presented here are some of the best results we have been able to produce, and only a few more weeks of tuning and testing might have given even better results.

## 7.5.2 General reflections

In the development of the ADP/RL model for salmon production scheduling in this thesis, there has been little literature and earlier research to lean on, as is often the case when working at the frontier of an academic area. The authors have worked by first developing a relatively wide and relatively deep knowledge on ADP and RL by deep diving into the works of field front figures such as Warren Powell and David Silver and then combining this knowledge with a knowledge of land based salmon production in order to design a novel solution method for the salmon production scheduling problem. As is also the inherent nature of ADP/RL as a field, the work has to a relatively large extent been based on trial and error. While efforts were made, amongst others in cooperation with dynamic programming expert and professor Qing Li from the Hong Kong University of Science and Technology, to e.g. develop theoretical derivations of the qualities of the theoretically optimal value function and to build more solid theoretical basis for a solution algorithm, this proved highly challenging. This is also perhaps the biggest weakness in our work. This is due to the highly complex, non-linear value function and non-convex decision space in the problem, as well as the fact that few have developed such theories before us so we had little guidance. We would like to stress however, that in the case of large-scale sequential decision making problems, there can be a significant difference between algorithms with theoretically proven convergence and practically efficient and applicable models. ADP/RL as a field has many times showed how convergence in a practical implementation will often require some element of trial and error tuning no matter the theoretical foundation. In our case, implementing the different types of value function approximations and deciding on their shape, finding the best algorithm hyper parameters and deciding on an "algorithmic" discount rate are examples of key modelling elements where we have had to (systematically) test the different options. Looking back, there are certain elements and alternatives that could have been tested even more thoroughly and systematically if the "plan" had been clearer at an earlier point. One example is more thoroughly testing the custom value function approximation implementation, which we in retrospect see has further potential we have not had time to investigate. There are other "leads" we have not had the resources to follow up as well, partly due to bad planning. In general, both the breadth and the depth of the ADP/RL field leaves us with no possibility of exploring all potential solution strategies to the degree of exhaustion we would have liked during a 6-month master thesis. This also presents the greatest "source of error" in our findings as well, the risk that there are algorithmic improvements we have not yet discovered, or algorithmic details which either prohibits or disturbs the convergence at some point. ADP/RL modellers can typically refine, adjust and tune complex models for almost indefinite time. However, we can safely say that we have opened the doors and started researching all the primary solution strategies (some basic and some more advanced) within the ADP/RL field that would be relevant to our problem, such as a problem-specific exploration strategy (exploring the three key schedule decision dimensions, see 5.3.3), a custom value function approximation and a decomposition model, as well as the go-to generalist neural network of the RL field. While we do not deem the ADP/RL model developed here good enough for use in real world production in its current state, future researchers might very well be able to achieve this.

While designing the trials and which algorithmic options to test however, we have to a large

---

extent used our problem specific knowledge and intuition. Our largest successes and algorithmic breakthroughs in terms of convergence have been where we have combined our now deep insight into the salmon production scheduling optimization problem with a gradually increasing insight into the nature of ADP/RL algorithms. Countless hours have been spent deep diving into solution logs, understanding how each value function target is derived and how it affects the update of the value function parameters and then adjusting the model sampling policy accordingly and other algorithm hyper parameters accordingly *given* the knowledge of how the fish grow and the price typically develops. This has been key to our work and will also likely be key to future research in the application of ADP/RL to salmon production scheduling.

# Chapter 8

## Conclusion

In the work with this master thesis we have built an ADP/Deep RL model based on n-step Temporal Difference (TD(n)) learning, testing both custom functions and deep neural networks as value function approximators and using a count-based, problem-specific exploration strategy resembling the Upper Confidence Bound method. The ADP/RL agent "plays the salmon farming game", exploring different numbers of fish released, different harvest weights and different tank coordinations. It is then *learning* from its experiences and the reward signals it receives, by iteratively improving its estimate of the true value of being in a given state and uses this value to make better decisions. To be able to solve the salmon production scheduling problem with respect to salmon price uncertainty, we have also developed a semi-parametric price model which both generates random price development samples and is able to calculate forward curves based on these samples. Finally, we have built a rolling horizon model based on mixed integer programming which is used to benchmark the ADP/RL model for small problem instances.

This master thesis had three primary goals. The first was a general, ambitious goal of building a scalable optimization model for producing high quality production schedules for the full-scale salmon production scheduling problem, intended for use in real world production with real value added. The second was related to the first and was about investigating whether approximate dynamic programming/reinforcement learning can serve this purpose. The third goal was determining the value of stochastic solution with respect to price uncertainty, or in other words, determining whether a salmon producer should actively take into account price variations when planning production.

The ADP/RL model is able to solve full-scale problem instances and scales better than the MIP benchmark for large problem instances. Furthermore, while the MIP benchmark is already rather optimized in terms of computational efficiency and runtime, there are a wide range of programming and implementation related improvements we have discussed which could be made to our ADP/RL model which would reduce computational time to a fraction of the current runtime. In terms of solution performance and quality, the ADP/RL model is also able to generate production schedules which are good enough that they exceed the level where the authors are able to visually, qualitatively say whether or not the schedule is optimal, at least for the single tank problem. Professor Keith Downing of NTNU, an experienced RL practitioner, implied early in the process that this application of RL is so complex compared to typical introductory RL models that merely producing such descent results would be a success. Nevertheless, we must conclude that the model performance is not good enough for practical real world application in its current state. The ADP/RL model earns  $\approx 80\%$  of the optimal production schedule for both 1- and 2-tank instances in the deterministic problem, which we consider the most relevant benchmark. We consider this gap too large and the model performance too varying for an actual salmon farmer to use the model as it is. The primary reason for this is that the ADP/RL model is less able to exploit the density and MAB restrictions of the facility than the MIP model and is therefore less accurate in the number of fish it releases in each batch and in the coordination of the different tanks, which in turn is the most significant influence on the overall profit earned. While it has less absolute impact on overall profit earned, the ADP/RL model also has trouble determining the optimal harvesting



---

weight with precision, which is the second most important reason for the optimality gap.

Zooming out and looking broadly at the application of ADP/RL to salmon production scheduling, we are nevertheless convinced of the potential for such methods in not only this field, but also in other biological herd management problems. With more resources, more time and more experience in the use and implementation of both custom function approximators, advanced neural networks and exploration strategies, future researchers can surely produce more high-performing, robust ADP/RL models than we have been able to. Especially when combining ADP/RL with the knowledge of the natural decomposed structure of our problem and possibly with hybrid methods integrating mixed integer programs, which we will discuss further in the future research section, models with high performance and real value added to an actual salmon producer can be built. Still, we would like to explain how we, like many others, have been partly victim to the hype surrounding reinforcement learning. The authors' key learning and takeaway regarding RL from this work is, like field experts like W. Powell (2011) and Irpan (2018) also proclaim, that RL in many ways is as much an art as a science. The largest and most famous successes of RL for extremely complex problems, such as Google DeepMind's AlphaZero, are so impressive that it is hard not to see RL as a group of methods which will bring profound breakthroughs to many fields in the years to come. While this vast potential, also for salmon production scheduling, can surely be realized, we are merely realizing that actually building such successful models are far more difficult and resource-consuming than one might get the impression of.

For the more isolated matter of determining the value of stochastic solution for the salmon production scheduling problem with respect to price uncertainty, our results suggest that the value of stochastic solution is relatively low, even for the multitank problem. On the other hand, depending on the amplitude of the price seasonality, planning deterministically with respect to a high-quality price forecast, i.e. harvesting as much as possible during the high price seasons, can be of great value compared to simply assuming a constant price. The value of generating high-quality price forecasts and applying these during scheduling is therefore large. The reader is referred to our preceding project thesis for in-depth discussions regarding the effect on optimal production scheduling of price seasonality.

## Future research

During our work we have discovered several promising leads for future research with respect to solving the salmon production scheduling problem.

The first and most problem specific lead is about exploiting the decomposed nature of the salmon production scheduling problem. As explained earlier, the different tanks in a facility are identical from an optimization perspective and the problem is easily seen as decomposed into a master problem coordinating the different tanks' production cycles and subproblems for each tank determining the specifics for each batch. While we have built an algorithm version (explained in 5.3.2) which exploits this structure by assuming that the value of the state of the full facility is nothing more than the sum of the values of the individual tank states, we have not devoted significant time to this, and several other options exist. This structure of the problem is so prevalent it is hard to ignore, and especially in terms of short runtimes, the exploitation of this structure would be one of the first places to start. Deep insight into the specifics and dynamics of salmon production is however required.

As mentioned in section 3.3, this thesis has focused on what Warren Powell calls a *value function approximation policy*, which both is what ADP/RL typically refers to and what people mean when they talk about ADP/RL and also is the most common solution strategy therein (W. Powell (2011)). Another important lead for future research is about using other types of policies, in the general sense as used by W. B. Powell (2016). Especially what Powell has named cost function approximations and lookahead policies might be worth investigating, since we believe policy function approximations would probably be overly complex and constraining for the multitank problem (see also section 7.1.2). Designing clever algorithms which combine ADP/RL capabilities like estimating the true value of the standing biomass and the ability to adapt to stochasticity with mixed integer programming subproblems which are more fine-grained and better at exploiting production

---

restrictions appears to be one of the most promising ways to solve the salmon production scheduling problem when modeled as a Markov Decision Process. By finding clever ways to iteratively improve the state valuation, short horizon mixed integer programs with either one or a few time steps can be solved. Currently, the end-of-horizon valuation and ensuring production constraints are respected also for the last batch in the horizon is one of the primary challenges of using MIP alone, along with computational intractability, and this is why the merger of ADP/RL with MIP appears highly promising. This is among the options the authors would have explored themselves if more time was available and the model was to be improved in order to be of use to actual salmon producers.

Looking back at the problem description in chapter 2, there are certain problem features which have been left out of the work with this thesis. Further research efforts should also look into how to incorporate these. The first is about including different smolt types (smolt weights) and letting the model choose between each type to be released for each batch. This is rather straight forward, and while it has not been tested in our work, our model is perfectly capable of handling this already. The second, which is slightly more on the side, is about how some land based salmon producers plan to build facilities with the option to move fish between different tanks to better utilize total facility capacities. This greatly increases the complexity from an optimization perspective, but is also of high interest for future research.

Once again referring to how RL in many ways is as much an art as a science, future research will also be about simply building on the knowledge presented here and applying more time, more resources and more experience to the problem. The authors hope to have planted seeds for a new line of research applying ADP/RL methods to not only salmon production scheduling but also to other animal production and biological hard management problems as well. As land based salmon farmers face tighter margins and machine learning is further developed, both in terms of hardware capabilities and software capabilities, we expect the commercial interests in doing exactly this will only rise in the coming years.

# Bibliography

- Achiam, Joshua (n.d.). “Spinning Up in Deep RL, 2018”. In: URL <https://spinningup.openai.com/>.
- Andersen, Bendik Persch (2019). “Efficiency in the Atlantic Salmon Futures Market”. MA thesis. NTNU.
- Ankamah-Yeboah, Isaac, Nielsen, Max, and Nielsen, Rasmus (2017). “Price formation of the salmon aquaculture futures market”. In: *Aquaculture Economics & Management* 21.3, pp. 376–399.
- Arnason, Ragnar (1992). “Optimal feeding schedules and harvesting time in aquaculture”. In: *Marine Resource Economics* 7.1, pp. 15–35.
- Asche, Frank and Bjørndal, Trond (2011). *The economics of salmon aquaculture*. Vol. 10. John Wiley & Sons.
- Asche, Frank and Guttormsen, Atle G (2001). “Patterns in the relative price for different sizes of farmed fish”. In: *Marine Resource Economics* 16.3, pp. 235–247.
- Asche, Frank, Misund, Bård, and Øglend, Atle (2016). “Fish Pool Priser–Hva Forteller de oss om Fremtidige Laksepriser?” In: *Economics and Management* 20, pp. 312–323.
- Asheim, Leif Jarle et al. (2011). “Are prices or biology driving the short-term supply of farmed salmon?” In: *Marine Resource Economics* 26.4, pp. 343–357.
- Berge, Aslak (2021). “Pengejakt for landbasert laks: Verdien av tillit”. In: *iLaks*. URL: <https://ilaks.no/pengejakt-for-landbasert-laks-verdien-av-tillit/>.
- Bjørndal, T. (1988). “Optimal Harvesting of Farmed Fish”. eng. In: *Marine resource economics* 5.2, pp. 139–159. ISSN: 2334-5985.
- Bjørndal, Trond, Knapp, Gunnar A, and Lem, Audun (2003). *Salmon: a study of global supply and demand*. SNF/Centre for Fishery Economics.
- Bjørndal, Trond and Tusvik, Amalie (2019). “Economic analysis of land based farming of salmon”. In: *Aquaculture Economics & Management* 23.4, pp. 449–475.
- Bloznelis, Daumantas (2016). “Salmon price volatility: A weight-class-specific multivariate approach”. In: *Aquaculture economics & management* 20.1, pp. 24–53.
- Bravo, Fernanda et al. (2013). “Mathematical models for optimizing production chain planning in salmon farming”. In: *International transactions in operational research* 20.5, pp. 731–766.
- Brevik, E. et al. (2020). “Optimisation of the broiler production supply chain”. In: *International Journal of Production Research* 58:17, pp. 5218–5237.
- Cobo, Ángel et al. (2019). “A decision support system for fish farming using particle swarm optimization”. In: *Computers and Electronics in Agriculture* 161, pp. 121–130.
- Deaton, Angus and Laroque, Guy (1992). “On the behaviour of commodity prices”. In: *The review of economic studies* 59.1, pp. 1–23.
- Denstad, Are Gederaas, Ulsund, Einar Askevold, and Lillevand, Miriam (2015). “Production planning and sales allocation in the salmon farming industry”. MA thesis. NTNU.
- Ewald, Christian-Oliver and Ouyang, Ruolan (2017). “An analysis of the Fish Pool market in the context of seasonality and stochastic convenience yield”. In: *Marine Resource Economics* 32.4, pp. 431–449.
- Ewald, Christian-Oliver, Ouyang, Ruolan, and Siu, Tak Kuen (2017). “On the market-consistent valuation of fish farms: using the real option approach and salmon futures”. In: *American Journal of Agricultural Economics* 99.1, pp. 207–224.
- EY (2020). *EY Norwegian Aquaculture Analysis 2020*.
- Fang, Jiarui et al. (2013). “Sourcing strategies in supply risk management: An approximate dynamic programming approach”. In: *Computers & Operations Research* 40.5, pp. 1371–1382.

- 
- FishPool (2020). *Fish Pool Index Annual report 2020*.
- Forsberg, Odd Inge (1996). “Optimal stocking and harvesting of size-structured farmed fish: A multi-period linear programming approach”. In: *Mathematics and computers in simulation : transactions of IMACS* 42.2-3, pp. 299–305. ISSN: 0378-4754.
- Forsberg, Odd Inge (1999). “Optimal harvesting of farmed Atlantic salmon at two cohort management strategies and different harvest operation restrictions”. In: *Aquaculture Economics & Management* 3.2, pp. 143–158.
- Forsberg, Odd Inge and Guttormsen, Atle G (2006). “The value of information in salmon farming. Harvesting the right fish at the right time”. In: *Aquaculture Economics & Management* 10.3, pp. 183–200.
- Fuchigami, Helio Yochihiro and Rangel, Socorro (2018). “A survey of case studies in production scheduling: Analysis and perspectives”. In: *Journal of Computational Science* 25, pp. 425–436.
- Graves, Stephen C (1981). “A review of production scheduling”. In: *Operations research* 29.4, pp. 646–675.
- Guttormsen, Atle G (1999). “Forecasting weekly salmon prices: risk management in fish farming”. In: *Aquaculture Economics & Management* 3.2, pp. 159–166.
- Hæreid (2011). “Allocating sales in the farming of atlantic salmon: Maximizing profits under uncertainty”. MA thesis. Institutt for industriell økonomi og teknologiledelse.
- Hæreid, Schütz, and Tomasgard (2013). “A Stochastic Programming Model for Optimizing the Production of Farmed Atlantic Salmon”. In: *Stochastic Programming: Applications in Finance, Energy, Planning and Logistics*. World Scientific, pp. 289–311.
- Hersoug, Bjørn, Mikkelsen, Eirik, and Osmundsen, Tonje C. (2021). “What’s the clue; better planning, new technology or just more money? - The area challenge in Norwegian salmon farming”. In: *Ocean Coastal Management* 199, p. 105415. ISSN: 0964-5691. DOI: <https://doi.org/10.1016/j.ocecoaman.2020.105415>. URL: <https://www.sciencedirect.com/science/article/pii/S0964569120303227>.
- Hornik, Kurt, Stinchcombe, Maxwell, and White, Halbert (1989). “Multilayer feedforward networks are universal approximators”. In: *Neural networks* 2.5, pp. 359–366.
- Irpan, Alex (2018). *Deep Reinforcement Learning Doesn't Work Yet*. <https://www.alexirpan.com/2018/02/14/r1-hard.html>.
- Karp, Larry, Sadeh, Arye, and Griffin, Wade L (1986). “Cycles in agricultural production: the case of aquaculture”. In: *American Journal of Agricultural Economics* 68.3, pp. 553–561.
- Kennedy, John OS (2012). *Dynamic programming: applications to agriculture and natural resources*. Springer Science & Business Media.
- Khosravi, Abbas et al. (2011). “Comprehensive review of neural network-based prediction intervals and new advances”. In: *IEEE Transactions on neural networks* 22.9, pp. 1341–1356.
- Kratsios, Anastasis (2021). “The universal approximation property”. In: *Annals of Mathematics and Artificial Intelligence* 89.5, pp. 435–469.
- Kure, Endre Hegland and Frøystein, Kristian (2013). “Optimal Salmon Production: Producing Smolt Under Uncertainty”. MA thesis. Institutt for industriell økonomi og teknologiledelse.
- Langan, Tarald Bjørdal and Toftøy, Tor (2011). “Produksjonsoptimering innenfor lakseoppdretten-planlegging under usikkerhet”. MA thesis. Institutt for industriell økonomi og teknologiledelse.
- Lei, Chao and Ouyang, Yanfeng (2017). “Dynamic pricing and reservation for intelligent urban parking management”. In: *Transportation Research Part C: Emerging Technologies* 77, pp. 226–244.
- Leung, PingSun and Shang, Yung C (1989). “Modeling prawn production management system: a dynamic Markov decision approach”. In: *Agricultural Systems* 29.1, pp. 5–20.
- Lillestøl, J. (1986). “On the problem of optimal timing of slaughtering in fish farming”. eng. In: *Modeling, identification and control* 7.4, pp. 199–207.
- Lodhi, Mohsin (2015). “Import demand elasticities for farmed salmon in the European Union and United States.” MA thesis. UiT Norges arktiske universitet.
- Lohmander, P. (2007). “Adaptive optimization of forest management in a stochastic world”. In: *Handbook of operations research in natural resources*. Springer, pp. 525–543.
- Markets, DNB (2021). *Land-based and offshore farming, what will be the impact from new farming methods*.
- Mistiaen, Johan A and Strand, Ivar (1998). “Optimal feeding and harvest time for fish with weight-dependent prices”. In: *Marine Resource Economics* 13.4, pp. 231–246.
-

- 
- Mnih, Volodymyr et al. (2013). “Playing atari with deep reinforcement learning”. In: *arXiv preprint arXiv:1312.5602*.
- Monteiro, Marina Dietze (2020). “A Novel Semiparametric Structural Model for Electricity Forward Curves”. PhD thesis. PUC-Rio.
- Mowi (2020). *Salmon farming industry handbook*.
- Mowi (2021). *Salmon farming industry handbook*.
- Oglend, Atle (2013). “Recent trends in salmon price volatility”. In: *Aquaculture Economics & Management* 17.3, pp. 281–299.
- Olafsdottir, G et al. (2019). “Governance of the farmed salmon value chain from Norway to EU”. In: *Aquaculture Europe* 44.1, pp. 5–17.
- Papageorgiou, Dimitri J et al. (2015). “Approximate dynamic programming for a class of long-horizon maritime inventory routing problems”. In: *Transportation Science* 49.4, pp. 870–885.
- Pascoe, Sean, Wattage, Premachandra, and Naik, Debananda (2002). “Optimal harvesting strategies: practice versus theory”. In: *Aquaculture Economics & Management* 6.5-6, pp. 295–308.
- Pathak, Deepak et al. (2017). “Curiosity-driven Exploration by Self-supervised Prediction”. In: *CoRR* abs/1705.05363. arXiv: 1705.05363. URL: <http://arxiv.org/abs/1705.05363>.
- Pindyck, Robert S (1990). *Inventories and the short-run dynamics of commodity prices*. Tech. rep. National Bureau of Economic Research.
- Powell, W.B. (2011). *Approximate Dynamic Programming: Solving the Curses of Dimensionality*. Wiley Series in Probability and Statistics. Wiley. ISBN: 9781118029169. URL: <https://books.google.no/books?id=RS53Dyx8wcUC>.
- Powell, Warren B (2016). “A unified framework for optimization under uncertainty”. In: *Optimization Challenges in Complex, Networked and Risky Systems*. INFORMS, pp. 45–83.
- Rivera, Arturo E Pérez and Mes, Martijn RK (2017). “Anticipatory freight selection in intermodal long-haul round-trips”. In: *Transportation Research Part E: Logistics and Transportation Review* 105, pp. 176–194.
- Russell, Stuart and Norvig, Peter (2002). “Artificial intelligence: a modern approach”. In:
- Rynning-Tønnesen, Christine and Overaas, Sivert Thorvik (2012). “Production optimization in the salmon farming industry: Ordering smolt under uncertainty”. MA thesis. Institutt for industriell økonomi og teknologiledelse.
- Salas, Daniel F and Powell, Warren B (2013). “Benchmarking a scalable approximate dynamic programming algorithm for stochastic control of multidimensional energy storage problems”. In: *Dept Oper Res Financial Eng*.
- Schwartz, Eduardo and Smith, James E (2000). “Short-term variations and long-term dynamics in commodity prices”. In: *Management Science* 46.7, pp. 893–911.
- Silver, David (2015). *Lectures on Reinforcement Learning*. URL: <https://www.davidsilver.uk/teaching/>.
- Silver, David et al. (2017). “Mastering the game of go without human knowledge”. In: *nature* 550.7676, pp. 354–359.
- Solheim, Magnus and Trovatn, Ola (2019). “The economic attractiveness of land-based salmon farming in Norway: a comprehensive presentation of an emerging industry”. MA thesis.
- Sparre, Per et al. (1976). “A Markovian decision process applied to optimization of production planning in fish farming.” In:
- Stikholmen, Nils-Arne (2010). “Produktivitetsutvikling over tid i oppdrett av laks: en studie av perioden 2001-2008 med bruk av DEA og Malmquistindeks”. MA thesis. Universitetet i Tromsø.
- Stray, Tharald Jørgen (2019). “Application of deep reinforcement learning for control problems”. MA thesis. NTNU.
- Sutton, Richard S and Barto, Andrew G (2018). *Reinforcement learning: An introduction*. MIT press.
- Taube-netto, M. (1996). “Integrated Planning for Poultry Production at Sadia”. In: *Interfaces (Providence)*, 1996-02 26(1), pp. 38–53.
- Yin, Jiateng et al. (2016). “Energy-efficient metro train rescheduling with uncertain time-variant passenger demands: An approximate dynamic programming approach”. In: *Transportation Research Part B: Methodological* 91, pp. 178–210.
- Yu, Run, Leung, PingSun, and Bienfang, Paul (2006). “Optimal production schedule in commercial shrimp culture”. In: *Aquaculture* 254.1-4, pp. 426–441.

# Appendix A

## Appendix

### A.1 Skretting growth table

Temperatur (°C)																						
gram	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	$\frac{H_{80}}{H_0}$	$\frac{A_{80}}{A_0}$
30	0,17	0,33	0,51	0,70	0,89	1,09	1,29	1,49	1,69	1,89	2,08	2,26	2,42	2,57	2,68	2,75	2,78	2,74	2,63	2,42	0,81	0,81
100	0,12	0,29	0,48	0,67	0,86	1,06	1,25	1,44	1,62	1,79	1,95	2,09	2,21	2,31	2,38	2,41	2,39	2,32	2,18	1,98	0,81	1,16
200	0,12	0,28	0,45	0,62	0,80	0,98	1,15	1,32	1,49	1,64	1,77	1,89	1,99	2,07	2,12	2,14	2,12	2,05	1,93	1,75	0,82	0,96
300	0,11	0,25	0,41	0,57	0,73	0,90	1,06	1,21	1,36	1,49	1,61	1,72	1,81	1,88	1,92	1,94	1,91	1,85	1,74	1,57	0,83	0,91
400	0,10	0,23	0,37	0,52	0,67	0,83	0,97	1,12	1,25	1,37	1,48	1,58	1,66	1,72	1,76	1,77	1,75	1,69	1,59	1,44	0,84	0,89
500	0,09	0,21	0,34	0,48	0,62	0,77	0,90	1,04	1,16	1,27	1,37	1,46	1,54	1,59	1,63	1,63	1,61	1,56	1,47	1,32	0,84	0,88
600	0,08	0,19	0,32	0,45	0,58	0,71	0,84	0,97	1,08	1,19	1,28	1,36	1,43	1,48	1,51	1,52	1,50	1,45	1,36	1,23	0,85	0,88
700	0,07	0,18	0,29	0,42	0,54	0,67	0,79	0,91	1,02	1,12	1,20	1,28	1,34	1,39	1,42	1,42	1,41	1,36	1,27	1,15	0,86	0,87
800	0,06	0,16	0,27	0,39	0,51	0,63	0,75	0,86	0,96	1,05	1,14	1,21	1,27	1,31	1,34	1,34	1,32	1,28	1,20	1,08	0,87	0,87
900	0,05	0,15	0,26	0,37	0,48	0,60	0,71	0,81	0,91	1,00	1,08	1,14	1,20	1,24	1,26	1,27	1,25	1,21	1,13	1,02	0,88	0,87
1000	0,05	0,14	0,24	0,35	0,46	0,57	0,67	0,77	0,87	0,95	1,03	1,09	1,14	1,18	1,20	1,20	1,19	1,15	1,07	0,97	0,88	0,87
1100	0,04	0,13	0,23	0,33	0,44	0,54	0,64	0,74	0,83	0,91	0,98	1,04	1,09	1,12	1,14	1,15	1,13	1,09	1,02	0,92	0,89	0,88
1200	0,04	0,12	0,22	0,32	0,42	0,52	0,62	0,71	0,79	0,87	0,94	1,00	1,04	1,07	1,09	1,10	1,08	1,04	0,98	0,88	0,90	0,88
1300	0,04	0,12	0,21	0,30	0,40	0,50	0,59	0,68	0,76	0,84	0,90	0,96	1,00	1,03	1,05	1,05	1,03	1,00	0,93	0,84	0,91	0,88
1400	0,03	0,11	0,20	0,29	0,38	0,48	0,57	0,65	0,73	0,80	0,87	0,92	0,96	0,99	1,01	1,01	0,99	0,96	0,90	0,80	0,91	0,88
1500	0,03	0,11	0,19	0,28	0,37	0,46	0,55	0,63	0,71	0,78	0,84	0,89	0,93	0,95	0,97	0,97	0,96	0,92	0,86	0,77	0,92	0,89
1600	0,03	0,10	0,18	0,27	0,36	0,45	0,53	0,61	0,68	0,75	0,81	0,86	0,89	0,92	0,94	0,94	0,92	0,89	0,83	0,74	0,93	0,89
1700	0,03	0,10	0,18	0,26	0,35	0,43	0,51	0,59	0,66	0,73	0,78	0,83	0,86	0,89	0,90	0,91	0,89	0,86	0,80	0,72	0,94	0,89
1800	0,03	0,09	0,17	0,25	0,33	0,42	0,50	0,57	0,64	0,71	0,76	0,80	0,84	0,86	0,88	0,88	0,86	0,83	0,77	0,69	0,95	0,89
1900	0,03	0,09	0,16	0,24	0,33	0,41	0,49	0,56	0,63	0,69	0,74	0,78	0,81	0,84	0,85	0,85	0,83	0,80	0,75	0,67	0,95	0,90
2000	0,03	0,09	0,16	0,24	0,32	0,40	0,47	0,54	0,61	0,67	0,72	0,76	0,79	0,81	0,82	0,82	0,81	0,78	0,73	0,65	0,96	0,90
2250	0,02	0,08	0,15	0,22	0,30	0,37	0,44	0,51	0,57	0,63	0,67	0,71	0,74	0,76	0,77	0,77	0,75	0,72	0,68	0,60	0,98	0,91
2500	0,02	0,08	0,14	0,21	0,28	0,35	0,42	0,48	0,54	0,59	0,64	0,67	0,70	0,72	0,72	0,72	0,71	0,68	0,63	0,56	1,00	0,92
2750	0,02	0,07	0,13	0,20	0,27	0,33	0,40	0,46	0,52	0,56	0,60	0,64	0,66	0,68	0,68	0,68	0,67	0,64	0,60	0,53	1,02	0,93
3000	0,02	0,07	0,13	0,19	0,26	0,32	0,38	0,44	0,49	0,54	0,58	0,61	0,63	0,64	0,65	0,65	0,63	0,61	0,56	0,50	1,04	0,94
3250	0,02	0,07	0,12	0,18	0,25	0,31	0,37	0,42	0,47	0,52	0,55	0,58	0,60	0,62	0,62	0,62	0,60	0,58	0,54	0,48	1,06	0,95
3500	0,02	0,07	0,12	0,18	0,24	0,30	0,36	0,41	0,46	0,50	0,53	0,56	0,58	0,59	0,60	0,59	0,58	0,55	0,51	0,46	1,08	0,96
3750	0,03	0,06	0,11	0,17	0,23	0,29	0,34	0,40	0,44	0,48	0,51	0,54	0,56	0,57	0,57	0,57	0,56	0,53	0,49	0,44	1,10	0,97
4000	0,03	0,06	0,11	0,17	0,22	0,28	0,33	0,38	0,43	0,47	0,50	0,52	0,54	0,55	0,55	0,55	0,54	0,51	0,47	0,42	1,12	0,98
4250	0,03	0,06	0,11	0,16	0,22	0,27	0,33	0,37	0,42	0,45	0,48	0,51	0,52	0,53	0,54	0,53	0,52	0,49	0,46	0,41	1,14	0,99
4500	0,03	0,06	0,11	0,16	0,21	0,27	0,32	0,37	0,41	0,44	0,47	0,49	0,51	0,52	0,52	0,51	0,50	0,48	0,44	0,39	1,16	1,00
4750	0,03	0,06	0,10	0,15	0,21	0,26	0,31	0,36	0,40	0,43	0,46	0,48	0,50	0,50	0,51	0,50	0,49	0,46	0,43	0,38	1,18	1,01
5000	0,03	0,06	0,10	0,15	0,20	0,26	0,31	0,35	0,39	0,42	0,45	0,47	0,48	0,49	0,49	0,49	0,47	0,45	0,42	0,37	1,20	1,02
5250	0,03	0,06	0,10	0,15	0,20	0,25	0,30	0,34	0,38	0,42	0,44	0,46	0,47	0,48	0,48	0,48	0,46	0,44	0,41	0,36	1,22	1,03

Figure A.1: Skretting growth table for atlantic salmon. Daily percentage growth for given temperatures and weights

---

## A.2 Numerical illustrative examples to optimal harvesting.

Here, we provide numerical practical examples of decisions faced by a salmon producer, and apply the theory of optimal harvesting discussed in 2.5.3 to these. In the first examples, we consider a constant salmon price for all sales classes, meaning the salmon price is the same at all times for all fish weights. We then move on to including price variations with both seasonality and fish size.

First, say we now consider a single batch rotation problem which takes the following form: Should the salmon producer harvest the batch of fish now or wait a time  $\Delta t$ ? Harvesting now gives the value of  $V(t)$  (given that we are at time  $t$ ), or incur extra feeding cost and harvest at  $t + \Delta t$  for the value of  $V(t + \Delta t)$ . The feeding cost is represented by  $C^f(t)\Delta t$ , where  $C^f(t)$  is defined as above. We here assume we can use discrete rather than continuous discounting. In this single batch rotation problem, we will then delay harvesting until  $\Delta t$  if

$$\frac{1}{(1+r)^{\Delta t}}V(t + \Delta t) - \frac{1}{(1+r)^{\frac{\Delta t}{2}}}C^f(t)\Delta t > V(t) \quad (\text{A.1})$$

where  $r$  is the discounting factor and, as before,  $V(t) = B(t)P(t) = Nw(t)P(t)$ . Feed costs are, simplified, discounted halfway into the time interval because these arise continuously. We restate that we only consider feed costs in this case, not harvesting costs or insurance costs, as these have a relatively small impact on the problem.

We will now see what this means in practical, numerical terms, when, as stated above, the salmon price is constant  $P(t) = P$ . The following is a numerical example intended to explore the optimal harvesting point at different discounting rates. Say we are at time  $\tilde{t}$  in the production, and we have a batch of 200,000 ( $N$ ) fish in the tank, all weighing 4,500 grams ( $w$ ). At 10 degree Celsius, it takes approximately 45 days to grow salmon 1 kg from 4,500g to 5,500g. Then the total biomass in the tank is equal to  $B = Nw$ . Say all the fish can either be harvested today (at  $\tilde{t}$ ) or after 45 days ( $\Delta t$ ), when the fish have grown 1 kg. Then  $\Delta t$  in equation A.1 is 45 days. Say the feed costs accumulated during these 45 days are  $C^f$ . Finally, say we have a mortality of around 3% each year, which is a realistic number for a land based facility. The *relative value of delaying harvesting of the fish until 45 days has passed*, i.e. the relative change in the present value of the batch is then

$$V_{delay} = \frac{\frac{1}{(1+r)^{\Delta t}}(V(\tilde{t} + \Delta t) - \frac{1}{(1+r)^{\frac{\Delta t}{2}}}C^f)}{V(\tilde{t})} \quad (\text{A.2})$$

If this value is positive, then it is profitable to delay harvesting. If it is negative, the salmon producer should harvest now.

The feed costs  $C^f$  is set using a feed cost of 15 NOK/kg and a FCR of 1.12 to be 16.8 NOK per fish (per kg) in the time interval  $[\tilde{t}, \tilde{t} + \Delta t]$ . The situation then is as follows:

	$t = \tilde{t}$	$t = \tilde{t} + \Delta t = \tilde{t} + 45$
P	50 NOK	50 NOK
Weight per fish ( $w$ )	4.5 kg	5.5 kg
Number of fish ( $N$ )	200,000	199,300
Biomass ( $B$ )	850 tonnes	1,050 tonnes
Nominal $V(t)$	42.5 MNOK	52.5 MNOK
Nominal feedcost accumulatd from last time point	-	3.30 MNOK

Table A.1: Single rotation numerical example input parameters

Let us now calculate what this decision looks like for different discount rates for one tank and one

single rotation.

Yearly discount rate $r$	$V_{delay}$ = Relative change in present value by delaying harvest compared to $V(\tilde{t})$ , absolute change in paranthesis (positive means delaying is profitable)
0%	14.4% (6.5 MNOK)
5%	13.7% (6.2 MNOK)
7.5%	13.4% (6.0 MNOK)
10%	13.1% (5.8 MNOK)
12.5%	12.7% (5.7 MNOK)
15%	12.4% (5.6 MNOK)
17.5%	12.1% (5.4 MNOK)
20%	11.8% (5.2 MNOK)

Table A.2: Value of delaying harvest as a function of different yearly discount rates in one single rotation example

All values are positive, meaning in each case it is positive to delay harvesting for 45 days until the fish have grown to 5.5 kg. In other words, the optimal harvesting weight in the single batch rotation problem with the parameters assumed here is larger than 4.5 kg, which is expected. The numbers are also positive with a good margin, meaning it is with great certainty more profitable to wait with the harvesting. As mentioned earlier, Mistiaen and Strand (1998) found that discount rates has implications for optimal harvesting value. We see here, however, as confirmed by Bjørndal (1988), that the effect is not very large, relatively speaking. However, the theory confirms that different salmon producers applying different discount rates will come to slightly different conclusions regarding optimal harvest times.

Now we will move into the much more interesting and much more realistic problem of a multiple batch rotation problem. We still only consider one tank/cage of fish. This means we will keep the parameters in this first example unchanged, but now the salmon producer has to take into account that delaying harvest will also delay the time until a new batch can be released and thus the time until all future cash flows are received. In other words, we must take into account *the opportunity cost of the fish tank!* We therefore now get the relative value of delaying harvest compared to harvesting now equal to (a rather simplified calculation admittedly)

$$V_{delay} = \frac{\frac{1}{(1+r)^{\Delta t}}(V(\tilde{t} + \Delta t) - \frac{1}{(1+r)^{\frac{\Delta t}{2}}}C^f - (1 - \frac{1}{(1+r)^{\Delta t}})FV}{V(\tilde{t})}} \quad (A.3)$$

Where FV is the perpetuity of all future cash flows from producing salmon in the given tank, where we simply assume that for all future batches the nominal value of harvesting will be equal to the average of the value at time  $\tilde{t}$ ,  $V(\tilde{t})$  and the value at time  $\tilde{t} + \Delta t$  will be  $V(\tilde{t} + \Delta t)$  (i.e. the fish are harvested at 5.0 kg in the "4.5 kg or 5.5 kg decision", see below). The last term thus is a simple measure of the effect on the present value of delaying all future harvests. The equation as a whole simply states that, under the assumptions made here, the salmon producer should compare (the discounted cash flow from harvesting at a later point minus the discounted accumulated feedcosts that will arise minus the cost of an additional discounting/delay of all future harvests from this tank) with the cash flow received from harvesting now. This is admittedly a very stylized example and a simple calculation, but it serves the purpose of showing the theory of optimal harvesting of a batch of fish in a single tank in the eyes of a salmon producer.

We will consider two decisions. First, say we have 200,000 fish in the tank at 3.5 kg at time  $\tilde{t}$ , and the decision is whether to harvest now or to wait until the fish have grown to 4.5 kg. This takes approximately 50 days at 10 degrees Celsius, as calculated by the Skretting SGR-table (see appendix A.1). Then, second, we will consider the case of either harvesting when the fish have reached 4.5 kg or delaying even longer until the fish have reached 5.5 kg weight. This will take approximately additional 45 days. The reason that the second time step is a few days shorter is



that, while daily weight gain in percentage is slightly higher for the first time period, the daily weight gain in absolute numbers is still a bit higher for the second time period. We still assume a constant price.

	$t_0 = \tilde{t}$	$t_1 = t_0 + 50$	$t_2 = t_1 + 45$
P	50 NOK	50 NOK	50 NOK
Weight per fish (w)	3.5 kg	4.5 kg	5.5 kg
Number of fish (N)	200,000	199,200	198,400
Biomass (B)	700 tonnes	896 tonnes	1,091 tonnes
Nominal $V(t)$	35 MNOK	44.82 MNOK	54.56 MNOK
Nominal feedcost accumulated from last time point	-	3.30 MNOK	3.27 MNOK

Table A.3: Multiple rotation numerical example input parameters

The results now look as follows for the most relevant discount factors

Yearly discount rate $r$	"3.5 kg or 4.5 kg" $V_{delay}$ = Relative change in present value by delaying harvest compared to $V(\tilde{t})$ , absolute change in paranthesis (positive means delaying is profitable)	"4.5 kg or 5.5 kg" $V_{delay}$ = Relative change in present value by delaying harvest compared to $V(\tilde{t})$ , absolute change in paranthesis (positive means delaying is profitable)
5%	2.6% (0.92 MNOK)	0.4% (0.19 MNOK)
7.5%	2.4% (0.84 MNOK)	0.2% (0.12 MNOK)
10%	2.2% (0.78 MNOK)	0.1% (0.05 MNOK)
12.5%	2.0% (0.72 MNOK)	-0.1% (-0.03 MNOK)
15%	1.9% (0.66 MNOK)	-0.2% (-0.1 MNOK)

Table A.4: Value of delaying harvest as a function of different yearly discount rates in multiple rotation example

These results are rather interesting, because they actually show that subject to the parameter values we have chosen (as realistic as possible) and the assumptions we have made, that if a 12.5% yearly discount rate is applied (which is a value called acceptable by a salmon producer we have talked to), the optimal harvesting weight for this batch is around 4.5 kg. The logic showing this is that the numbers show a positive change in present value by delaying harvest from 3.5 kg to 4.5 kg, but a negative change from 4.5 kg to 5.5 kg, meaning it is profitable to delay until 4.5 kg but not until 5.5 kg. If a salmon producer considering a single tank of fish and when to harvest this batch finds himself in this particular situation illustrated here, then our results suggest optimal harvesting weight is somewhere around 4.5 kg. We also note that if the producer chose a discount rate of 10%, the optimal decision would be to harvest at 5.5 kg. This is interesting, as discount rates might be slightly different between firms, as discussed in 2.5.3. With our stylized example situation here, we have thus shown that the discount rate might make a significant difference between firms' operations.

We restate once more that these calculations are subject to amongst others assumptions that harvesting costs and smolt release costs can be omitted due to restricted relevance to the results and that the cost of delaying all future harvests can be correctly calculated the way we have done here, as shown by equation A.3. We will note however, that adding harvesting costs and smolt release costs to the problem would effectively work to increase the optimal harvesting weight (push the optimal harvesting time later in time), although not much.

In the practical examples considered so far, the salmon price has been assumed to be constant at 50 NOK/kg for all sales classes and times. This is far from true in reality, as discussed previously. We will now move on to study the effect of *price variations* on these simplified versions of the optimal harvesting problem.

---

### A.2.1 Expected price variations - seasonality and fish size

As previously discussed, the salmon price primarily varies along two dimensions, looking apart from the stochastic variation. These two dimensions are a cyclical seasonality in the price with a price peak somewhere in Q1 as well as a price differentiation between different weight classes of sold fish, typically divided into sales classes of 3-4 kg, 4-5 kg and 5-6 kg. So, in a more realistic version of the optimal harvesting problem than what has been discussed so far in this chapter, when delaying harvesting the salmon producer can also expect a change in the price/kg she receives when harvesting.

Say we look at the same situation in the multiple batch rotation problem as in the previous section. Only now we will also include that the expected price will change during these time intervals. The change in price might be both because the fish reach a higher sales class, or because we for an example are in Q4 and the price is expected to rise due to seasonality. Larger fish are generally higher priced, although the opposite has occurred, as discussed in section 2.5.1. Thus, at certain times there might be no or even a negative price change when moving to a higher sales class, or we might be in a time of year when prices are stable or even declining. That means the effects of seasonality and price gaps between weight classes might work in the same direction or in opposite directions and they might theoretically work in both directions (price going up or down).

Which effect of seasonality or price gap between sales classes cause the price change is not strictly relevant for the sort of numerical calculations we are performing here. What matters is how large the effect of the price variation is in sum, and thus what is the total change in price/kg the salmon producer will receive. We will therefore now look at a range of price changes that might occur during the "harvest or wait" decisions we are considering.

Say we stick to the same parameters and the same decisions faced by the salmon producer as before, i.e. harvest at 3.5 kg or wait for 4.5 kg and subsequently, harvest at 4.5 kg or wait for 5.5 kg. We are still considering the optimal harvesting time in a multiple batch rotation setting, meaning we include both feed costs and the delay of future cash flows in the calculation. Equation A.3 is then still unchanged, only now the nominal value  $V(t)$  changes not only due to an increase in biomass but also due to a change in price. To make it even more interesting, lets also add yet another step. Once 5.5 kg is reached, the salmon producer can choose to harvest or to wait for the fish to reach 6.5 kg. This takes approximately 45 days at 10 degrees Celsius. In this situation, we have adjusted the feed costs slightly up to 18 kg and see how this affects the problem as well. The situation then looks as follows:

	$t_0$	$t_1 = t_0 + 50$	$t_2 = t_1 + 45$	$t_3 = t_2 + 45$
P(t)	50 NOK	$P(t_0)(1 \pm \delta)$	$P(t_1)(1 \pm \delta)$	$P(t_2)(1 \pm \delta)$
Weight per fish (w)	3.5 kg	4.5 kg	5.5 kg	6.5 kg
Number of fish (N)	200,000	199,200	198,400	197,600
Biomass (B)	700 tonnes	896 tonnes	1,091 tonnes	1,284 tonnes
Nominal V(t)	35 MNOK	$BP(t)$	$BP(t)$	$BP(t)$
Nominal feedcost accumulated from last time point	-	3.53 MNOK	3.50 MNOK	3.48 MNOK

Table A.5: Value of delaying harvest with different price changes in multiple rotation example input parameters

We will in the following calculations proceed with a yearly discount rate of 12.5%. This will be discussed further in the next section, but is mainly because this seems appropriate in terms of the capital risk land based salmon farmers face and because we have confirmed that this is an acceptable value with salmon producers we have discussed with. We saw, however, in the previous examples

that although the discount factor has an effect on the optimal timing, the effect is relatively small.

The price change scenarios analysed will be a range reflecting the most likely outcomes. We repeat that stochastic/random price variations are ignored in our calculations. A 45 or 50 day period is not a long period, hence a very large price variation due to seasonality is unlikely. However, moving up a sales class, the average over the last five years yields a price increase of almost 3% as presented in chapter 2. Extreme cases over the last year yield around 13% price gap from one sales class to another. As this is considered a rather extreme case, we test a range of price change in each time interval from -5% to +10%. Again, the important here is to see how price changes affect optimal harvest timing, not to test for all possible values. The results are presented in table A.6.

Price change per time interval	"Harvest at 3.5 kg or wait until 4.5 kg", relative change in present value by waiting	"Harvest at 4.5 kg or wait until 5.5 kg", relative change in present value by waiting	"Harvest at 5.5 kg or wait until 6.5 kg", relative change in present value by waiting
-5%	-4.5% (-1.6 MNOK)	-6.4% (-2.8 MNOK)	-9.0% (-4.4 MNOK)
-3%	-2.1% (-0.8 MNOK)	-4.2% (-1.8 MNOK)	-6.5% (-3.4 MNOK)
-2%	-1.0% (-0.3 MNOK)	-3.0% (-1.3 MNOK)	-5.3% (-2.8 MNOK)
-1%	0.2% (0.07 MNOK)	-1.8% (-0.8 MNOK)	-4.1% (-2.2 MNOK)
0%	1.4% (0.5 MNOK)	-0.6% (-0.3 MNOK)	-2.9% (-1.6 MNOK)
1%	2.6% (0.9 MNOK)	0.6% (0.3 MNOK)	-1.7% (-0.9 MNOK)
2%	3.7% (1.3 MNOK)	1.8% (0.8 MNOK)	-0.4% (-0.3 MNOK)
3%	4.9% (1.7 MNOK)	3.0% (1.4 MNOK)	0.7% (0.4 MNOK)
5%	7.2% (2.5 MNOK)	5.4% (2.6 MNOK)	3.2% (1.9 MNOK)
7.5%	10.2% (3.6 MNOK)	8.4% (4.1 MNOK)	6.2% (3.9 MNOK)
10%	13.2% (4.6 MNOK)	11.4% (5.6 MNOK)	9.2% (6.0 MNOK)

Table A.6: Value of delaying harvest as a function of percentage change in price per time period

As before, a positive change means the optimal choice is to delay harvest. As expected, in the most negative price change case, it is never profitable to delay harvesting, and in the most positive price change case, it is always profitable to delay harvesting. What is more interesting, is that in the -1% price change case, it is still optimal to delay harvesting from 3.5 kg to 4.5 kg, even though the producer will get a lower price/kg of biomass harvested. This can be understood as the percentage growth in biomass outweighing the percentage decline in price. In this case of -1% price change, which for example can represent a situation where we are in spring and approaching the low price season and there is no price difference between 3-4 kg fish and 4-5 kg fish, the optimal harvesting weight is around 4.5 kg. That is because it is not profitable to delay harvesting further when 4.5 kg is reached. Looking at the 1% price change case, the results suggest the optimal harvesting weight is when the fish is around 5.5 kg, by the same logic. The same is true in the 2% case, which means that when the fish have reached 5.5 kg, even though the salmon producer knows she will get up to 2% higher price per kg of biomass, she should harvest the fish now rather than wait. Finally, we see that the changes in profitable decisions with price variations are large, suggesting that when a salmon producer finds him/herself in this particular situation, wondering whether to harvest now or to let the fish grow further, the expected price increase or decrease is a very significant part of the decision.

Admittedly, these are stylized examples relying on several simplifications and assumptions and they only take into account what we deem the most impactful elements of the decision, which are feed costs and the delay of future harvests. For example, in reality the salmon producer's decision is not as simple as only harvesting at 1 kg intervals and in modern salmon production, partial harvesting is typically done, meaning the harvesting of a batch is done in several turns. Furthermore, in reality, several additional aspects of the decisions must be taken into account. This will be discussed in the last part of this chapter. We will nonetheless argue that these calculations provide meaningful insight to a salmon producer. Surely, any experienced salmon production planner have a gut-feel regarding this numbers, that might or might not correspond with the results above.

Our main point in presenting the theory and the numerical examples above have been to highlight the different relevant elements to take into account when deciding the optimal harvesting time of a batch of fish in a tank, and to explain in detail the salmon producer's decision problem. Although stylized, our examples are not especially unrealistic in terms of parameter values, meaning that the numbers calculated here might actually be applicable to real production subject to certain assumptions. It is even possible for each individual salmon producer to set up their own elaborate, specific calculations such as the above, to decide optimal harvesting times, involving more factors than done here. Following the example of Bjørndal (1988), it is even possible to calculate the globally optimal harvesting time exactly if one has an empirical weight function  $w(t)$  to base the calculations on.

### A.3 Seasonal component



Figure A.2: Plot of mean adjusted log prices of Salmon forwards and seasonal component represented with a sine/cosine function

