Fleet Planning Under Demand Uncertainty

A Reinforcement Learning Approach

M.C.T.C. de Koning

Technische Universiteit Delft

TUDelft

Fleet Planning Under Demand Uncertainty

A Reinforcement Learning Approach

by

M.C.T.C. de Koning

Student number:	4179072	
Project duration:	May 2, 2019 – March 6, 2020	
Thesis committee:	Dr. ir. B. F. Lopes dos Santos,	TU Delft, supervisor & chair
	Dr. M. A. Mitici,	TU Delft, examiner
	Dr. N. Yorke-Smith,	TU Delft, examiner

Cover frontpage:
`https://stmed.net/sites/default/files/airport-wallpapers-28369-9089125.jpg`.

TU Delft

Preface

This report is the final deliverable for my Master of Science graduation thesis on fleet planning under demand uncertainty. This research topic is a continuation of a series of research thesis' in optimising fleet planning using alternative solving methodologies. Over the course of one year, I have worked on a very relevant airline planning problem and immersed myself in the cutting edge of artificial intelligence. The combination of these two fields of research proved to be a complex task which challenged me every day. I am grateful to have worked on a topic which triggered my curiosity, required a steep learning curve, and allowed me to excel in my personal, and intellectual growth. The research and completion of this thesis was not possible without the help of certain people whom I am grateful for.

To start, I would like to thank my thesis supervisor, Dr Bruno Santos, for his guidance throughout my graduation project. Since the beginning of this master program, I have enjoyed very much your courses, lectures, and our constructive meetings. I am therefore grateful to have worked together on this project. On several occasions, you were able to ask me the questions about my work which steered me in the right direction.

Secondly, I would like to thank Daniel Marta. Our meetings together concerning the implementation and guidance on deep reinforcement learning were of great value for my work. You were able to shed some light on hard to grasp concepts and had a realistic view on how to approach the learning problems.

Thirdly, I would like to thank Sydney Cohee and Luca de Laat, who both gave important feedback on my thesis and article. I'm sure my work would not have become such a beautiful story without your help.

Finally, I would like to thank my parents who have stood by my side the whole duration of my studies becoming an aerospace engineering. I'm grateful for their emotional and financial support which allowed me to develop myself personally and professionally.

M.C.T.C. de Koning
Delft, February 21, 2020

List of abbreviations

ADP	Approximate Dynamic Programming
AD	Average Demand
BOE737	Boeing 737-800
BOE753	Boeing 757-300
BTS	Bureau of Transportation and Statistics
CVaR	Conditional Value-at Risk
DD	Deterministic Dynamic
DDQN	Dueling Deep Q-Network
DMD	Dominant Market Demand
DND	Dominant Network Demand
DNN	Deep Neural Network
DOC	Direct Operating Cost
DOT	Department of Transportation
DP	Dynamic Programming
DQN	Deep Q-Network
DS	Deterministic Static
FAM	Fleet Assignment Model
FPM	Fleet Planning Model
GA	Genetic Algorithm
GDP	Gross Domestic Product
IP	Integer Programming
IQR	Interquartile Range
LP	Linear Programming
LPM	Lower-Partial Moment
MC	Monte Carlo
MDP	Markov Decision Process
MILP	Mixed-Integer Linear Programming
MIP	Mixed-Integer Programming
NN	Neural Network
OD	Origin-Destination
OR	Operations Research
ORD	Chicago O'Hare International Airport
OU	Ornstein-Uhlenbeck
POMDP	Partially Observable Markov Decision Process
RL	Reinforcement Learning
RPM	Revenue Passenger Miles
SD	Stochastic Dynamic
SDI	Stochastic Demand Index
SFO	San Francisco International Airport
SQG	Stochastic Quasi-Gradient
SS	Stochastic Static
TD	Temporal Difference

List of Figures

1 Airline planning framework from aircraft perspective, focus of this research is highlighted in blue (adapted from Santos (2017a) and Bouarfa et al. (2014)). ix

1.1 The airline planning process (Belobaba et al., 2015). 39
1.2 Problem formulation and perspectives map. 41

2.1 Generic decision tree, with decision nodes (squares), outcome nodes (circles), decisions and random outcomes (lines) (Powell, 2011). 45

4.1 Knowledge gap mind map. 56

1.1 General solution method diagram of Reinforcement Learning set-up (Sutton and Barto, 2018). 73
1.2 Example of a Fully Observable MDP (François-Lavet et al., 2018). 74
1.3 Temporal Difference (left) and Monte Carlo (right), with all hybrid TD-methods in the middle (Sutton and Barto, 2018). 75
1.4 Taxonomy of Reinforcement Learning strategies (adapted from Odonkor and Lewis (2018)). 76
1.5 Example fully connected network with one hidden layer (adapted from François-Lavet et al. (2018)). 78
1.6 Cheat sheet of the most widely used activation functions (Li et al.). 79
1.7 Diagram illustrating of the learning process of the Reinforcement Learning agent (adapted from Pytorch (2020)). 80

2.1 Mean-reversion model parameter calculation through linear regression. 88
2.2 Example of the historical cumulative transported passengers with 50 cumulative sampled demand trajectories in the 10-airport network using independent routes predictions (left); Histogram of the cumulative predicted demand values at 2024 (right). . . . 89
2.3 Example of historic market demand values with 50 sampled demand trajectories for the route SFO-ORD (left); Histogram of the predicted demand values at 2024 (right). These sample trajectories are for the Average Demand scenario (Part I). 90
2.4 Example of historic network demand values with 50 cumulative sampled demand trajectories for network (left); Histogram of the cumulative predicted demand values at 2024 (right). These sample trajectories are for the Average Demand scenario (Part I). . 91

3.1 Schematic architecture of the reward function calculation. The agent feeds the FAM block from the left with the state s_t and action a_t. The experience $< s_t, a_t, s_{t+1}, r_t >$ is stored in the replay buffer, and the next state s_{t+1} returned to the agent. 94
3.2 Simplified representation of Fleet Assignment Mode and Fleet Planning Model. 95
3.3 Profit of action space in the first period for 50 independent sampled demand values of a network with 4 airports. 101
3.4 Profit of action space in the first period for 50 independent sampled demand values of a network with 7 airports. 101

3.5 Profit of action space in the first period for 50 independent sampled demand values of a network with 10 airports. . . . 102
3.6 Example of a box plot with the interquartile range. . . . 102
3.7 IQR range of fleet decision procentual profit of optimal profit for various airline network sizes. . . . 103

4.1 Training score of DQN and averaged-DQN. . . . 107
4.2 Estimated error of the Q-function for DQN and averaged-DQN. . . . 107
4.3 Training score of DQN for various learning rates. . . . 109
4.4 Estimated error of the Q-function for various learning rates. . . . 109
4.5 Training score of DQN for various learning methods. . . . 110
4.6 Estimated error of the Q-function for various learning methods. . . . 110
4.7 Training score of DQN for various number of hidden layers. . . . 111
4.8 Estimated error of the Q-function for various number of hidden layers. . . . 111
4.9 Examples of adjusted non-linear reward functions. . . . 112
4.10 Training score of DQN for various non-linear reward functions. . . . 113
4.11 Estimated error of the Q-function for various non-linear reward functions. . . . 113
4.12 Training score of DQN for various lower bounds. . . . 114
4.13 Estimated error of the Q-function for various lower bounds. . . . 114

List of Tables

1.1 List of features tested with explanation. 82
1.2 High performing feature combinations from the Genetic Algorithm optimisation . . . 83

3.1 Decision variables, constraints, and runtime for a FPM/FAM model with two types of aircraft and varying size of periods and airport network. 104
3.2 Decision variables, constraints, and average runtime for a FPM model with two types of aircraft and varying size of periods and airport network. 105

4.1 Testing results of DQN vs Average-DQN. 108
4.2 Testing results of DQN for various learning rates. 108
4.3 Testing results of DQN for various learning methods. 109
4.4 Testing results of DQN for various number of hidden layers. 111
4.5 Testing results of DQN for various non-linear reward functions. 112
4.6 IQR of the lower bounds of the reward function. 113
4.7 Testing results of various lower bounds. 114

Contents

Preface ii

List of abbreviations iii

List of Figures iv

List of Tables vi

Introduction ix

I Scientific Article 1

II Literature Study
Previously Graded Under AE4020 37

1 Introduction 39
1.1 Context . . . 39
1.2 Review Structure . . . 41

2 Problem Formulation & Solution Techniques 42
2.1 Deterministic models: . . . 42
2.2 Stochastic Models: . . . 43
2.2.1 Two-Stage Stochastic Modelling: . . . 43
2.2.2 Multi-Stage Stochastic Programming:. . . . 44
2.2.3 Dynamic Programming: . . . 45

3 Modeling Perspectives: 49
3.1 Demand Uncertainty: . . . 49
3.2 Fleet Leasing: . . . 50
3.3 Robust Optimisation . . . 51
3.4 Fuel Price Uncertainty . . . 52
3.5 Maintenance . . . 53

4 Knowledge Gap 54

III Research Methodologies
Previously Graded Under AE4010 57

IV Thesis report 70

1 Reinforcement Learning 72
1.1 Context . . . 72
1.2 The Markov Decision Process . . . 73
1.3 Learning Strategies . . . 74
1.4 Value-based Learning . . . 77
1.5 Deep Q-network . . . 80

1.6 DQN in the Fleet Planning Problem . . . 81
1.6.1 State space . . . 81
1.6.2 Action Space . . . 84

2 Ornstein-Uhlenbeck Forecasting Model 85
2.1 The Mean-reverting Ornstein-Uhlenbeck Process . . . 85
2.2 Adapted Demand Forecasting Model . . . 88
2.3 Application in the Fleet Planning Problem . . . 91

3 Environment Model 93
3.1 Simulated Experience . . . 93
3.2 Airline Profit Optimisation. . . . 95
3.2.1 Decision Variables . . . 96
3.2.2 The Non-Decision Variables . . . 96
3.2.3 Objective Function . . . 97
3.2.4 Constraints: . . . 97
3.3 Fleet Assignment Model . . . 99
3.4 Reward function . . . 100
3.5 LP Matrix Size and Computational Times . . . 103

4 Sensitivity Analysis 106
4.1 Averaged-DQN: . . . 107
4.2 Learning rate: . . . 108
4.3 Monte Carlo vs Temporal Difference: . . . 109
4.4 Number of Hidden Layers . . . 110
4.5 Non-linear Reward Function . . . 112
4.6 Lower Bound . . . 113

5 Conclusions & Recommendations 116

Bibliography 120

Introduction

The fleet planning process is the strategic decision process when, which, and how many vehicles should be acquired to maximise profits in the coming years. These decisions are of uttermost importance to ensure a low operational cost and a competitive position in the airline market. The ownership of aircraft is an expensive financial investment and lease contract often last multiple years. In Figure 1 the fleet planning process is shown as the first step of the airline planning process. Consequently, the wrong decision in the fleet planning step has a large effect throughout the whole planning process. Fleet planning is therefor considered to be a long-term strategic planning process.

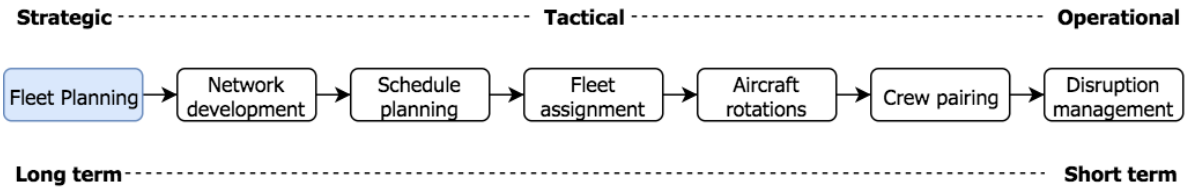


Figure 1: Airline planning framework from aircraft perspective, focus of this research is highlighted in blue (adapted from Santos (2017a) and Bouarfa et al. (2014)).

The fleet planning problem is thoroughly discussed in the preceding Literature Study summarized in Part 2 of this report. Here, current research in fleet planning was subdivided into two major fields: The modelling method and the modelling perspectives. The first research field focuses on innovative methods to converse the real-life fleet planning processes in a mathematical model accompanied by a solution method (e.g., deterministic or stochastic modelling). The second subfield is the modelling perspectives adding information to the mathematical model which can solely or combined add predictive power to the fleet planning process (e.g., demand and fuel uncertainty).

All deterministic models hold in common that they assume we live in perfect predictably world; consequently, all variables take a fixed value and the solution has one fixed outcome. As a result, a deterministic model cannot model for uncertain parameters, which are often depicted as probability distributions. To the contrary, stochastic models consider some parameters to be unknown; the solution can be viewed as a probability distribution of different outcomes (Smelser and Baltes, 2001). The fleet planning problem, which optimizes the future fleet decisions, is subjected to a large number of uncertainties (e.g. air-travel demand, fuel prices, or aircraft failures). As a result, research has shifted toward more complex stochastic methodologies incorporating these uncertainties in the fleet planning process (List et al., 2003; Hsu et al., 2011; Khoo and Teoh, 2014; Repko and Santos, 2017). Nowadays, both the deterministic and stochastic problems are most commonly solved using Operation Research (OR) methodologies (e.g., Simplex method, Branch & Cut, or Heuristics). However, the computational expensive of solving these problem increases exponentially with the increase of variables. The size of fleet planning models and solution methods are therefor still very limited, and more sophisticated modelling techniques have trouble find an application in the airline business due to their complexity. As a result, this research aims to explore and improve the modelling method of the stochastic fleet planning problem under demand uncertainty.

In the Artificial Intelligence (AI) field, Machine Learning (ML) research field has shown remarkable results in recent years. Moreover, the subfield Deep Learning (DL) has shown success in pre-

dicting and learning from large datasets through non-parametric functional approximations. Research in human-level control of games and robots (Mnih et al., 2015), image and speech recognition (Radford et al., 2019) have already experienced big improvements. However, in the Optimisation Research field, the novelty of AI has not been deeply rooted yet, and opportunities exist to improve the established optimisation techniques.
Indeed, artificial intelligence is finding applications in the assisting or replacement of conventional operational optimization approaches. Reinforcement Learning (RL) (or Approximate Dynamic Programming (ADP)) is a learning method/strategy which has gained a lot of momentum in the artificial intelligence community recently. The RL/APD problem can be formalized as a Markov decision process (MDP), where the at every discrete time-step the environment of the problem is observed, an action is taken according to a policy, and the environment transitions to a new state (Buşoniu et al., 2010).

A scenario tree with aircraft fleet decisions as actions under uncertain demand is currently the state-of-the-art approach in modelling and solving the fleet planning problem (Hsu et al., 2011; Khoo and Teoh, 2014; Repko and Santos, 2017; Requeno and Santos, 2018). With model-based ADP Requeno and Santos (2018) were able to find near-optimal solutions in a reasonable time for much larger trees compared to a backwards dynamic programming approach Nonetheless, the method is still very computationally expensive and only viable for a very small set of airports and simple network architectures. Moreover, all the fleet planning research incorporating the uncertainty over multiple periods make use of a scenario tree to represent the evolution of the unknown parameter. The uncertainty and the transition from one state to the next one are discretised in a small set of realisations to keep the decision tree from growing too large. This is referred to as a model-based approach as the transition model (scenario tree) is assumed to be known. As a result, due to the aggregation of estimated demand predictions, extreme demand growth outcomes and the demand growth in between the fixed growths are disregarded. Consequently, the solution will only be usable if demand evolves in the small set of scenarios and will not be robust to handle disruptive growths/shrinking of demands. An alternative solution to model-based learning is model-free learning. Here, the learning model has no knowledge of the underlying transition model and learns the adaptive fleet policy directly from interaction with the real environment and the sampled demand evolutions.

The main objective of this research is therefor to contribute to the development of adaptive policies by the generation of a model-free Approximate Dynamic Program (or Reinforcement Learning program) in a fleet planning environment subjected to uncertainty. This by generating model where (a) an agent in the form of a neural network learns the optimal dynamic fleet policy, by interaction with (b) an artificially created feedback environment, (c) under uncertain demand. Following the objective, the research question is defined as:

Can a model-free reinforcement learning model learn the long-term dynamic policy in a fleet planning problem under demand uncertainty.

From the main research question, multiple smaller sub-questions are devised. Investigating and answering all sub-questions will solve the main research question.

- Which elements and modelling techniques from literature can be used to design a model-free reinforcement learning program that successfully predicts the optimal policy?
- How should the reinforcement learning architecture be designed to learn the fleet planning policy?
- How can the sub-components be implemented in a model and trained efficiently?

- What is the performance of the reinforcement learning model and how does it perform against conventional fleet planning methods.

This report will be split into four parts. Part I will contain the integral scientific paper. Part II depicts a summary of the author's literature study which was previously graded under the course AE4020. The Research Methodologies report graded under AE4010 is included as Part III. Finally, Part IV shows the Thesis Report subdivided in four chapters. The first chapter expands on the Ornstein-Uhlenbeck Forecasting model used to sample air-travel demand. The second chapter explains the reinforcement learning process and the Deep Q-Network. The third chapter describes the Environment Model and reward generation. In chapter 4, a Sensitivity Analysis of the (hyper)parameters of the reinforcement learning model. Finally, in chapter 5 the conclusions and recommendations are discussed.

I

Scientific Article

Fleet planning under Demand Uncertainty: a Reinforcement Learning Approach

Mathias C.T.C. de Koning[1]

Delft University of Technology, Netherlands

Abstract

This work proposes a model-free reinforcement learning approach to learn a long-term fleet planning problem subjected to air-travel demand uncertainty. The aim is to develop a dynamic fleet policy which adapts over time by intermediate assessments of the states. A Deep Q-network is trained to estimate the optimal fleet decisions based on the airline and network conditions. An end-to-end learning set-up is developed, where an optimisation algorithm evaluates the fleet decisions by comparing the optimal fleet solution profit to the estimated fleet solution profit. The stochastic evolution of air-travel demand is sampled by an adaptation of the mean-reversion Ornstein-Uhlenbeck process, adjusting the air-travel demand growth at each route for general network-demand growth to capture network trends. A case study is demonstrated for three demand scenarios for a small airline operating on a domestic US airport network. It is proven that the Deep Q-network can improve the prediction values of the fleet decisions by considering the air-travel demand as input states. Secondly, the trained fleet policy is able to generate near-optimal fleet solutions and shows comparable results to a reference deterministic optimisation algorithm.

Keywords: Fleet Planning Problem, Dynamic Fleet Policy, Deep Q-network, mean-reversion Ornstein-Uhlenbeck

Nomenclature

$\mathcal{T}$	=	set of discrete time periods in the finite time horizon T, $\{0, 1, ..., T\}$;
$\mathcal{E}$	=	set of episodes, E being the total number of episodes;
$\mathcal{S}$	=	set of infinite states representing the airline resources and network conditions;
$\mathcal{R}$	=	continuous set of possible rewards, $r \in \mathbb{R}$;
$\mathcal{A}$	=	set of actions/fleet decisions, A being the total number of fleet decisions;
$\mathcal{N}$	=	set of routes in the network, N being the total number of routes in the network;
$\mathcal{M}$	=	set of market in the network, M being the total number of markets in the network;
$\mathcal{K}$	=	set of aircraft types, K being the total number aircraft types available;
$\mathcal{I}$	=	set of airport in the network, I being the total number of airports available;

[1]Author of this article, supervised by Dr. ir. B.F. Lopes dos Santos

1. Introduction

The fleet planning process of an airline is widely considered the most important long-term decision to ensure future profitability (Belobaba et al., 2009; ICAO, 2017). It is a strategic decision process which is concerned with the acquirement or disposal, the quantity, and the composition of the fleet in future years. In the airline business, fleet planning is considered to be the first step in the airline planning process and addressed via either one of two approaches: the *top-down* or the *bottom-up* approach. The first approach, top-down, estimates the required fleet based on a high-aggregate level analysis. Forecasters estimate the most-likely expected aggregate demand growth and the expected gap in capacity. The future gap in capacity is the amount of aircraft capacity needed to maintain profitable operations. This method is the most common approach in the contemporary airline business as it is a simple method and can be calculated using basic spreadsheets (Belobaba et al., 2009). However, this method is very sensitive to the forecast of aggregated demand; it does not account for the possible deviations from the estimated demand and accompanied fleet.

An alternative method to the top-down approach is the bottom-up approach. This approach uses a detailed modelling method to match the expected demand on a route-level to an operational flight network. By matching the demand and supply on a more detailed level, the fleet size and fleet composition can be modelled more accurately and efficiently. The bottom-up modelling approach can incorporate multiple variables and complexities to represent a more detailed planning model where the fleet decision can be tailored more precisely to the expected demand. Adding these complexities increases the size of the bottom-up models as they grow exponentially with the addition of more variables. Increasing the amount of detail has proven to be a fruitful approach to increase future profits, operational efficiency, and robustness. As a result, this thesis project will elaborate on bottom-up approaches. The remainder of this section will provide an introduction to the fleet planning problem using bottom-up approaches.

Background

In the 1950s Operational Research (OR) gained a lot of momentum as a research field led by the need of industry to optimize production methods to increase the efficient usage of resources (Hillier, 2012). Some of the first authors to address the operational fleet planning were Kirby (1959) and Wyatt (1961). They elaborated on short-term leasing of rail-cars to fulfil the temporal shortage of fleet due to excessive demand. Furthermore, Dantzig and Fulkerson (1954) and Bartlett (1957) investigated the number of ships needed to operate in a fixed schedule. The OR models were still very simple and only considered the fleet planning for a single time period. With the advent of increasing computational power in the 1970s, the size and complexity of the fleet planning problem increased. Shube and Stroup (1975) introduced the first multi-period bottom-up fleet planning model to generate long-term strategic decisions over multiple years using a Mixed Integer Linear Programming (MILP) model. This method is relatively simple to implement and further increases in computational power, meaning larger networks and multiple replacement strategies were able to be modelled using this technique (Bazargan and Hartman, 2012). However, the air-travel demand coefficients are deterministically fixed, and the solution to the problem is therefor an optimal solution for one evolution of demand. Ignoring the stochastic nature of variables such as air-travel

demand can result in unfit fleet planning solutions, and thus should be characterized as probability distributions (Kall and Wallace, 1994).

Stochastic modelling allows us to capture one or more variable's stochastic nature into the fleet planning model. In two-stage stochastic models, a sub-set of the decision variables is chosen, and after the uncertainty is revealed the remainder of the variables are determined. A recourse action is performed which can be seen as a corrective measure against the infeasibility that arises due to the realisations of the uncertainty (Sahinidis (2004)). Several publications demonstrate that two-stage stochastic modelling is a suitable approach to model uncertain perspectives into the fleet planning problem. Oum et al. (2000) utilized the two-stage stochastic approach to model the optimal mix between leased aircraft and owned aircraft under demand uncertainty. In a case study, the researches obtained data from 23 different international airlines and proved that a mix of leased and owned aircraft mitigates the cost of an airline under uncertain demand at an optimal mix of around 40-60% leased aircraft of the total fleet. List et al. (2003) employed the stochastic modelling method to create a more robust fleet plan under two uncertain parameters: future demand and future operating conditions. In the recourse function, a partial moment of risk is incorporated to decrease the effect of extreme outcomes of the optimal solution. The authors show that a trade-off between the fleet decisions and the accompanying risk of insufficient resources can induce high cost on airline operations. However, having only considered three markets and a homogeneous fleet, the authors report a high computational effort to compute fleet decisions. Listes and Dekker (2002) argue that fluctuation of demand induces low load factors across the network. They propose a model which tackles the operational flexibility through a demand scenario aggregation based approach to find a fleet composition which appropriately supports dynamic allocation of capacity. The fleet composition problem is modelled as a two-stage stochastic multi-commodity flow problem across a time-space network. Despite its advantages, only one period with a fixed flight schedule is optimized, limiting the planning horizon of usage as a long-term strategic planning tool. Moreover, all aforementioned two-stage stochastic models are limited to a single period planning horizon, and therefor not suitable to create long-term fleet policy.

Multi-stage stochastic programming, such as Dynamic Programming (DP), allows a sequential decision problem to be subdivided into sub-problems which are solved recursively (Bertsekas et al., 1995). In the airline industry, Hsu et al. (2011) use a DP model to simulate the optimal replacement strategy subjected to a Grey topological demand model. The planning model incorporates the trade-off between ownership, leasing of aircraft, and aircraft maintenance to increase the flexibility and better match short-term variations in demand. In a follow-up research Khoo and Teoh (2014) argued that the Grey topological demand model is too limited, as it does not capture disruptive events influencing the air-travel demand. They propose a novel model incorporating the stochastic demand index (SDI) in a step by step procedure using Monte Carlo simulations. In a more recent study, Repko and Santos (2017) employ a different approach to modelling a multi-period fleet plan under uncertainty. The problem is modelled in a scenario tree approach where the nodes represent the fleet decisions and the branches the demand revelations. Using MILP the ideal fleet composition is derived for each scenario, and with the accompanied probability of each scenario, the op-

timal fleet composition for each period is calculated. Sa et al. (2019) take a more detailed approach on the generation of the fleet compositions and future demand. They propose the generation of a set of demand scenarios which are derived by sampling long-term travel demand from the mean-reverting Ornstein-Uhlenbeck process. Secondly, fleet portfolios are used to compare different fleet composition to demand scenarios. However, as the size of the airline's network, planning horizon, and the number of demand revelations increases, the amount of states grows exponentially. As a result, solving the portfolio, or the scenario tree backwards, becomes too computationally expensive; Powell (2011) refers to this as the *curse of dimensionality*.

Approximate Dynamic Programming (ADP) brings a novel solution to the curse of dimensionality by stepping forward through the scenario tree and calculating the values of the state-action transitions recursively. ADP approximates the value-function of decision nodes in the scenario tree using the Bellman equation (Bellman, 1954). ADP, or often referred to as Reinforcement Learning (RL), has been implemented by multiple research communities (e.g., Control Theory, Artificiality Intelligence, and Operations Research) (Powell, 2009). Consequently, different names are adopted to describe the same process. However, in this paper a distinction is made between the two names: RL will refer to the *model-free* approach and ADP to the *model-based* approach. The ADP method has already proven to be useful in solving resource allocation, and vehicle routing problems in the transport industry. Lam et al. (2007) used ADP to model operational strategies for the relocation of empty containers in the sea-cargo industry. Novoa and Storer (2009) examine ADP approaches in modelling single-vehicle routing problem with stochastic demands. And Powell (2009) uses ADP to build an optimizer simulator for locomotive planning. With ADP, near-optimal solutions are found for the fleet size and planning of the vehicles. The model-free, reinforcement learning, approach has found a recent surge in popularity due to the novel improvement made in long-term decision making. Moreover, the use of deep neural networks as function approximators in RL has proven highly beneficial in term of performance and practicability to learn decision processes (Mnih et al., 2015).

Paper Contribution

This research contributes to the development of dynamic policies by the generation of a model-free reinforcement learning program in a fleet planning environment subjected to uncertainty. This will be achieved by developing a model where (a) an agent learns the optimal dynamic fleet policy, by interaction with (b) an artificially created feedback environment, (c) under uncertain air-travel demand.

The agent gives the optimal decisions in acquiring or disposing a number of aircraft at a given time using the current state and the values of the state-action transition. The environment converts the decision of the agent into the next state and calculates a meaningful reward of the state-action combination in a reasonable time. The agent learns from the experiences saved from previous agent-environment interactions. The agent-environment interaction is submitted to a change in air-travel demand over time which resembles the real-life nature of demand evolvement in the physical world.

The paper offers three main contributions:

1. Although state-of-the-art stochastic modelling methods have previously been used to explore and solve operations optimisation problems, we use a reinforcement learning approach to create a dynamic fleet planning policy. This work is the first to employ a model-free learning algorithm to learn the optimal strategic long-term airline fleet policy under air-travel demand uncertainty.
2. In an End-to-end learning set-up, a neural network is trained using stochastic samples of air travel demand to predict the impact of the fleet decision on the future operational profit. As a result, the trained neural network can be used easily as a predictive model for forecasters and managers without retraining.
3. The work proposes a new sampling strategy of the air-travel demand adapted from the Ornstein-Uhlenbeck forecaster employed by Sa et al. (2019). The air-travel demand growth at each route is adjusted for general network-demand growth to capture network trends.

Report Structure
The remainder of this paper is organized as follows: Section 2 describes and formulates the fleet planning problem as a Markov decision process. Section 3 elaborates on the reinforcement learning process. The air-travel demand sampling using the adapted Ornstein-Uhlenbeck mean reversion process is described in Section 4. Section 5 presents the training environment and reward generation. In Section 6 a proof-of-concept case study for the fleet planning problem is proposed, and in Section 7 the training and testing results are presented. Finally in section 8 the concluding remarks are depicted.

2. Problem Formulation

In the airline business, one of the main factors of success can be measured by matching the supply and demand as closely as possible (Dožić and Kalić, 2015). Consequently, the optimal fleet to fulfil the future air-travel demand across the operating network is crucial to ensure profitable future airline operations. Unfortunately, aircraft are high-capital investments and require intensive usage to capitalize a profit. Moreover, aircraft are not directly at the airline's disposal in times of high demand and must be acquired/disposed long in advance. A careful planning decision process is therefor needed to predict the most feasible fleet decisions based on the possible evolutions of air-travel demand.

The aim is to develop an optimal dynamic solution tool, which allows a re-assessment of the fleet decisions as time progresses. As a result, the fleet planning process can be represented as a sequential decision process with discrete time intervals, formalized as Markov Decision Process (MDP). The fleet planning problem is represented as a finite horizon MDP where at every discrete time-step $t \in \mathcal{T}$, a state $s \in \mathcal{S}$ is observed, a decision-maker takes an action $a \in \mathcal{A}$, and the state transitions to a new state $s' \in \mathcal{S}$ under a stochastic process, and a reward to the decision is generated $r \in \mathcal{R}$. To create the optimal fleet decision plan under the given airline and network conditions, a policy π is developed which represents a probability distribution over the fleet decisions or actions given the state.

2.1. State Space

The state of the fleet planning problem at a given period t is a collection of parameters or features, which holds the information of the airline and network for the decision-maker

to estimate the optimal action a_t. In this paper, two ensembles of features are created to train two different policies: a *static* fleet policy and a dynamic fleet policy. The static fleet policy will be trained 'blind' to the air-travel demand. This means that a fleet policy will be generated which is static over time, and independent on the intermediate evolution of demand. The policy is referred to as being Stochastic Static (SS) as it is static and trained on stochastic air-travel demand.

The dynamic fleet policy refers to a MDP where the air-travel demand features are included and the fleet decision are determined based on the current air-travel demand in the network. The policy is referred to as Stochastic Dynamic (SD) due to its dynamic fleet decisions and training on stochastic air-travel demand. The states of the MDP can be defined as:

$$s_t^{SS} = \left(t, ac_{own}^t\right) \tag{1}$$

$$s_t^{SD} = \left(t, ac_{own}^t, d_t\right) \tag{2}$$

Where $t \in \mathcal{T}$ is the period of the fleet planning problem. $ac_{own}^t = \left[ac_{own}^{t,k}\right]_{k\in\mathcal{K}}$ the amount of aircraft k owned in period t, and $d_t = [d_{m,t}]_{m\in\mathcal{M}}$ is a vector listing the market's demand value at period t. The market m replaces two Origin-Destination routes with the same airports, as it is assumed that air-travel demand is similar because the majority of passenger book round trips.

The size of the state vector S^{SS} and S^{SD} is dependent on the number of aircraft types considered and one entry for the time period. For the SD policy, the state vector size is extended with number of markets in the network:

$$S^{SS} = 1 + K \tag{3}$$

$$S^{SD} = 1 + K + M \tag{4}$$

2.2. Action Space

The actions of the fleet planning problem are defined by the decision to either acquire or dispose aircraft, the amount of aircraft, and of which type. The action vector can be defined as:

$$a_t = (ac_{acq}^t, ac_{dis}^t) \tag{5}$$

With $ac_{acq}^t = \left[ac_{dis}^{t,k}\right]_{k\in\mathcal{K}}$ the amount of aircraft k acquired in period t, $ac_{dis}^t = \left[ac_{acq}^{t,k}\right]_{k\in\mathcal{K}}$ the amount of aircraft k disposed in period t. Let's assume that at each stage t for each aircraft type k, aircraft are either acquired, disposed, or no action performed. Furthermore, the amount of aircraft bought or disposed is limited per aircraft type f_{max}^k. The size of the action space initially grows exponentially with increasing amount of aircraft types K and the maximum number of aircraft acquired or disposed per aircraft type f_{max}^k: $A = \prod_{k=1}^{K} 2 \cdot f_{max}^k + 1$. In order to keep the action space from growing to large, it is assumed that only one fleet action is taken for a single aircraft type each period t. The discrete action space and size therefor becomes:

$$\mathcal{A} = \left\{0, [-F^k, F^k]_{k \in K}\right\} \tag{6}$$

$$A = (2 \cdot f^k_{max} \cdot K) + 1 \tag{7}$$

Where $F^k = [1, ..., f^k_{max}]$, and each action $a_t \in \mathcal{A}$ is mapping of the input states.

2.3. Transition function

The transition function defines how the system transitions from state s_t to state s_{t+1}. In the fleet planning problem the demand growth of each market Δ^m_{t+1} is the stochastic variable which defines the demand growth at each market at the next state. As a result, the transition function and next state are defined as:

$$s_{t+1} = \begin{bmatrix} t+1 \\ ac^{t+1}_{own} \\ d_{t+1} \end{bmatrix} = \begin{bmatrix} t+1 \\ ac^t_{own} - ac^t_{dis} + ac^t_{acq} \\ d_t \cdot \Delta_{t+1} \end{bmatrix} \tag{8}$$

where, $d_{t+1} = [d_{m,t} \cdot \Delta_{m,t+1}]_{m \in \mathcal{M}}$. The observed change of demand growth in the market is the realization of the uncertain parameter in the MDP. The evolution of the growth of the demand in each market is the result of independent sampling, and is outlined in Section 4.

2.4. Value-based Optimisation

With MDP tuples the fleet planning problem can be optimized by finding the optimal policy π^* which maximizes the expected reward. The reward r_t is an evaluation of the action a_t given the state s_t, the next state s_{t+1}, and mapped through a reward function $r_t = R(s_t, a_t, s_{t+1})$. The goal of the decision maker is formalized in terms of the reward received (Sutton and Barto, 2018). In Section 5 a more in depth analysis of the reward is given. For now, the discounted return G_t of the finite horizon fleet problem is defined as a sum of all future rewards from t to T of the MDP interactions:

$$G_t = R_{t+1} + \gamma R_{t+2} + \cdots + \gamma^{T-1} R_T = \sum_{k=0}^{T} \gamma^k R_{t+k+1} \tag{9}$$

The discount factor γ $(0 \geq \gamma \geq 1)$ discounts future returns. Returns which are expected to contribute in the future are considered to be less value than current rewards. In order to learn the optimal sequence of actions resulting in the highest cumulative reward, a value function $V^\pi(s)$ is introduced. The value function approximates the value of being in a state s under a policy π as the expected discounted in state s, depicted in Equation 10. From the value function, optimal expected return or optimal value function $V^*(s)$ is defined in Equation 11.

$$V^\pi(s) = E_\pi [G_t | s_t = s] = \mathbb{E}\left[\sum_{k=0}^{T} \gamma^k r_{t+k} | s_t = s, \pi\right] \tag{10}$$

$$V^*(s) = \max_{\pi \in \Pi} V^\pi(s) \tag{11}$$

The policy which maximizes the value function at state s is the optimal policy π^*. To create meaningful rewards, the goal of the fleet planning process must be first identified. Airlines

can choose to optimize their fleet for maximization of transported passengers, adaptability of fleet in the network, minimize cost, etc. In this work it is assumed that the goal is to maximise the profit of future operational years. Thus, the policy represents the fleet decisions of the fleet planning problem which maximizes the airlines' current- and future profits.

3. Deep Reinforcement Learning

To optimise the goal and policy of the fleet planning problem, a learning algorithm is devised. In this chapter, a Reinforcement Learning (RL) process is proposed to learn the optimal policy by iteratively explore the MDP and learn to make the correct sequence of fleet decisions given airline and network conditions. The model-free RL method defines itself by fully separating the transition function from the agent's optimisation process. Contrary to model-based programming, where the transition function is known and used to estimate the optimal policy, the model-free agent is only able to observe the transitioned state after the action is taken.

3.1. Q-learning

Q-learning is the most widely known form of reinforcement learning technique developed by Watkins (1989). The Q-value function $Q(s, a)$ is introduced by extending the value function with the action value. The Q-value function represents the expected value (or total discounted reward) of performing an action a given the state s. An agent updates the Q-table with experiences $< s, a, s', r >$ gathered from interaction with an environment. At every step t, an action is chosen based on the current Q-value function, and the result of that action is observed as an experience. With this experience, the Q-value corresponding to the state-action pair is updated using Equation 12.

$$Q^{\text{new}}\left(s_t, a_t\right) \leftarrow Q\left(s_t, a_t\right) + \alpha \left[r_{t+1} + \gamma \max_a Q\left(s_{t+1}, a\right) - Q\left(s_t, a_t\right)\right] \tag{12}$$

The optimal policy of a Q-learning model can be deducted from the optimal Q-value function as it is the action which maximizes the expected return over time, represented by Equation 13. However, if greedy-policy is maintained throughout the learning process, the policy always exploits the current value function to maximize immediate reward. Higher Q-values remain hidden behind unexplored state-action pairs and the model quickly gets stuck in a local optimum. To incentivize the exploration of unvisited state-actions, an ϵ-greedy algorithm is introduced. At every decision with a probability of ϵ, a random action from the action set equal probabilities is chosen. During the learning process, initially the ϵ parameter will be high to investigate the state-action space, and progressively will decay.

$$\pi^*(s) = \underset{a \in \mathcal{A}}{\text{argmax}} Q^*(s, a) \tag{13}$$

A Q-table provides a simplistic method for assessing and storing the Q-values. However, as the number of actions and states increases, lookup-tables require an increasing amount of memory storage. As a result, more functional approximators are better suited for control problems of modern-day size. Requeno and Santos (2018) observed in a similar fleet planning problem that the relationship between the airline profits and the number of aircraft owned

is a non-linear concave function. Consequently, a non-linear function approximator will be most suitable to approximate the value function, such as a neural network.

3.2. Deep Q-network

In deep reinforcement learning the Q-function is approximated by a neural network (NN). The parameters θ of the NN are trained using stochastic gradient descent to minimize a loss function. Deep Q-network (DQN) (Mnih et al., 2015) is a value-learning deep neural network which brought novel solutions to the shortcomings of the original Neural Fitted Q-learning which suffered with slow and unstable convergence properties (Goodfellow et al., 2016). DQN learns the optimal Q-value function with a neural network approximation by minimizing the loss function $L_i(\theta_i)$:

$$L_i(\theta_i) = \mathbb{E}_{<s,a,s',r>\sim\mathcal{U}(\mathcal{M})}\left(r + \gamma Q\left(s', \arg\max_{a'} Q(s', a'; \theta); \theta_i^-\right) - Q(s, a; \theta_i)\right)^2 \tag{14}$$

Next to a policy network with parameters θ, a target network is created with parameters θ^-. The parameters of the target network are used to estimate the expected target values and are updated every C amount of episodes, to minimize the divergence of the estimation and the updated parameters. Secondly, a replay buffer is created where the experiences of the previous iterations are stored. At the time of training, a batch of random experiences are uniformly sampled from the buffer and used to optimize the policy network's parameters. This technique increases the learning speed of the parameters and allows for less variance. Finally, DQN clips the rewards between -1 and $+1$ to ensure more stable learning (François-Lavet et al., 2018).

In Figure 1, a representation of the reinforcement learning model is depicted. At the start of a training episode e, the demand growth d_t is sampled for the periods $T+1$. This information is available to the environment but not the agent. The reinforcement learning loop is visible as the interaction between the agent (DQN with replay buffer) and the environment. The RL loop is iterated T-times until the finite-horizon is reached. After the RL loop is terminated, if the final episode E is not reached, the RL parameters are reset to the initial state, a new demand growth trajectory sampled, and the process repeated.

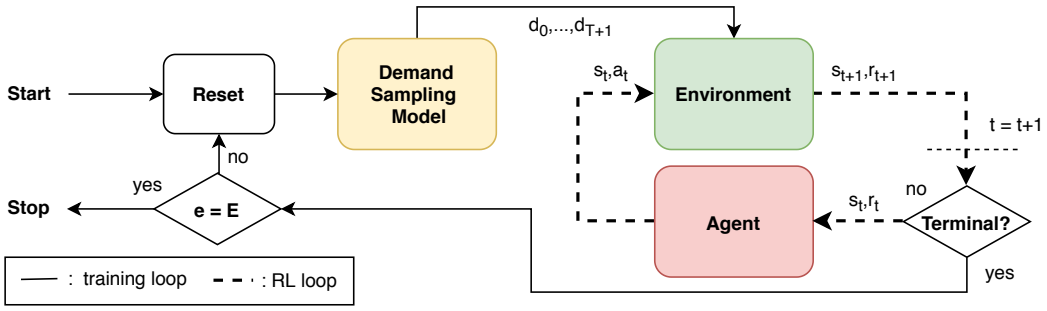


Figure 1: Diagram illustrating the architecture reinforcement learning loop (agent-environment) and the training loop for E episodes.

4. Sampling Stochastic Air-travel Demand

The stochastic nature of the MDP is formalized by a model sampling the air-travel demand values in every episode e. The sampling of a market m is a trajectory of air-travel demand growths from $t = 0, ..., T + 1$, based on the historical characteristics of the air-travel demand.

The mean-reverting Ornstein-Uhlenbeck (OU) process (Uhlenbeck and Ornstein, 1930; Vasicek, 1977) is an autoregression model which has found applications in modeling evolutionary processes in biology (Martins, 1994), diffusive processes in physics (Bezuglyy et al., 2006), and simulating volatility models of financial products Barndorff-Nielsen and Shephard (2001). Recently, using the mean-reverting time-dependent process as a sampling model for demand was successfully implemented by Sa et al. (2019) for a portfolio-based fleet planning model, and will form the basis of our demand sampling model.

4.1. Mean-Reverting Ornstein-Uhlenbeck process

In this section, the OU process as a sampling model for demand growth introduced following the notation Sa et al. (2019). The mean-reverting process is a stochastic differential equation of the growth of the air-travel demand x_t, which can be discretised to estimate the next air travel demand growth x_{t+1}:

$$x_{t+1} = x_t + \lambda (\mu - x_t) + \sigma dW_t \tag{15}$$

The term $\lambda (\mu - x_t)$ describes the mean reversion process often called the *drift term*. λ is 'the speed of the mean reversion' describing how fast deviation reverts back to the mean, and μ, 'long term mean growth rate', can be interpreted as the mean air-travel demand growth which the model will approach in the long term. Finally, the process becomes stochastic by the inclusion of $dW = W_{t+1} - W_t$ which has the Normal$(0, 1)$ distribution, and is referred to as the 'shock term'. The term σ influences the impact of the disruptions and can be interpreted as the volatility of the change in the growth of demand.

The μ, λ, σ parameters are referred to Ornstein-Uhlenbeck parameters and are deducted from historic data by approximation of the linear relationship between growth in demand x_t and the change of the growth in demand y_t with linear regression fitting. The linear regression of the historic data x_t, y_t reveals the regression coefficients for the slope b and the intercept a of the fitted data to calculate λ, μ, and σ (Chaiyapo and Phewchean, 2017).

Once the Ornstein-Uhlenbeck parameters are calculated for every route using the historic demand growth, numerous growth trajectories can be estimated for every route by independently and iteratively estimating the next demand growth using Equation 15.

4.2. Adapted Demand Sampling Model

Sa et al. (2019) assume that because air-travel demand tends to correlate to the GDP, the same way the stock of financial markets correlates to GDP, the OU process is a proven predictor of air-travel demand. In the research of Sa et al. (2019), the air travel demand trajectories for multiple routes are sampled independently from each other. Due to the normal distribution of the shock term, the aggregate shock term over multiple routes will converge to zero. As a result, the stochastic element is visible on individual markets; however, this effect is mitigated in the cumulative demand growth of the network and behaves therefor

as a linear extrapolation. Secondly, the OU process assumes a fixed long-term drift rate. Consequently, the long-term cumulative demand growth always converges to a single value. The result is a demand trajectory which converges to quasi-identical solution each sampling, and does not fully represent the stochastic behaviour of the real-life air-travel demand.

In this work, it is assumed that an overarching network demand growth exists which influences all routes equally. This growth is influenced by economic growth, fuel prices, airline branch reputation, etc. These factors influence all market demand growths. Secondly, changes in emerging economic markets or demographics of the area locations of airports, influence the willingness to fly certain markets. These factors are specific to independent market growths. Finally, it is assumed that the long-term mean growth is not a fixed value. For the same reason yearly growth changes, long-term mean growth can also diverge from the intended path due to numerous economic, environmental, and political factors.

An adapted demand forecasting model is proposed, where overarching network demand growth prediction δ_{t+1} is added to each market growth prediction $x_{m,t+1}$. Before sampling air-travel demand trajectories on a market level, a cumulative air-travel demand growth trajectory is sampled using the OU parameters derived from the cumulative historic air-travel demand in the network. By averaging the predicted network growth rate (δ_t^n) -which is equal for all markets in period t- and the predicted growth market rate, a semi-independent market demand growth $x_{m,t+1}$ for market m, is sampled:

$$\delta_{t+1} = \delta_t + \lambda\left(\mu' - \delta_t\right) + \sigma dW_t) \tag{16}$$

$$x_{m,t+1} = \frac{1}{2}(\delta_{t+1} + x_{m,t} + \lambda_m\left(\mu'_m - x_{m,t}\right) + \sigma_m dW_{m,t}) \tag{17}$$

$$\begin{aligned} \text{where} \quad & \mu' \sim \mathcal{N}(\mu, \sigma_\mu^2) \\ & \mu'_m \sim \mathcal{N}(\mu_m, \sigma_{m,\mu}^2) \end{aligned} \tag{18}$$

The normal distribution of the long-term mean growth μ' and μ'_m is depicted in Equation 18. Every simulation of an air-travel demand trajectory, a realisation of the normal distribution is drawn. The μ and μ_m represent respectively the estimated long-term mean growth for the network, and for every market m. The σ_μ^2 and $\sigma_{m,\mu}^2$ are the variance terms of the long-term mean growth of respectively the network and markets s. The latter variance terms represent how much dispersion is present in the mean growth over time. However, for the network and every market, only one historical long-term demand is present; hence, it is impossible to calculate the variance. In Section 6 three demand scenarios are represented to test the sensitivity of the variance parameters.

The air-travel demand sampled using Equation 17 is sampled for every time step t which consists of n_{years} years. However, the adapted OU sampling model is discretized in yearly demand growth predictions. As a result, there is a misalignment in time horizon between the RL loop and the demand sampling model. Consequently, the time horizon for the sampling model is changed to $Y = n_{years} \cdot (T + 1)$ years, and the periodical change in growth $\Delta_{m,t}$ is related to the yearly demand growth $x_{m,t+1}$ in Equation 19. Note that sampling horizon $(T + 1)$ is longer than the time horizon T to accommodate for the revealing of the final air-travel demand d_{T+1} and evaluation of the final action a_T in the RL loop.

$$\Delta_{m,t} = \prod_{i=1}^{n_{years}} (1 + x_{m,y+i}) \quad \text{where} \quad y = t \cdot n_{years} \tag{19}$$

5. Training Environment

The RL agent is tasked with learning the environment's dynamics in order to maximize the expected future reward. The purpose of the environment is thus twofold: transitioning to the next state s_{t+1}, and evaluating the action a_t in the form of a reward r_t. In the fleet planning problem, the environment represents the airline's resources and the air-travel demand network. The transition to the next state is explained in Section 2 and the accompanied revealing of the uncertainty in air-travel demand in Section 4. The remainder of this section will explain the training strategy of the agent-environment interaction and generation of the reward r as an evaluation of the fleet decisions.

5.1. Training Strategy

To successfully train the RL model, a meaningful reward function $R(s_t, a_t, s_{t+1})$ needs to be created which measures the goal of the fleet planning problem: maximise the expected profit. Every fleet decision influences the future profitability of the airline, which can be measured using a Fleet Assignment Model (FAM). The FAM optimizes the airline's profit (C_{FAM}) in period t by matching the flight frequency of aircraft types in the fleet to the sampled air-travel demand. However, the profit under the fleet decision can not be transformed directly into a reward as there is not a frame of reference to say how good or bad this decision was compared to other possible fleet decisions. Moreover, the sampling of demand influences the height of the optimal achievable profit severely, thus making the optimal profit fluctuate throughout the episodes.

An *oracle* is introduced as the optimal fleet decision and profit over the network and period to evaluate the periodical fleet decisions. This larger optimisation algorithm, Fleet Planning Optimisation Model (FPM), optimizes the frequency of flights in conjunction with the optimal fleet. At the start of each episode, after sampling the air-travel demand, the FPM calculates the optimal frequency, fleet composition, and consequently the accompanied profit (C^t_{FPM}) in each period as an upper bound to the FAM.

Figure 2 shows a schematic representation of the full reinforcement learning model with all sub-component interactions. At the start of the training process, the OU parameters are estimated, and the RL model parameters reset to $t = 0$. After initialization (or resetting after each episode), the air-travel demand for each market m is sampled for the full time horizon $T + 1$, resulting in a set of demand vectors $d_0, ..., d_{T+1}$. With the sampled future demand, the optimal fleet and periodical profit C^t_{FPM} can be calculated in the FPM. In addition to this, the optimal fleet decisions can be stored as experiences in the replay buffer to increase the size of the replay buffer and learning experiences of the model.

Now, RL loop is initiated the initial state s_0 is used to predict the first fleet decision a_0 in the policy network. With the fleet decision, and the first vector of the demand matrix d_0,

the FAM is optimized for the optimal fleet usage over the network and the profit C_{FAM}. In the reward function, the reward is calculated by comparing the profit of the fleet decision to the optimal profit in the corresponding period C^t_{FPM}. The resulting experience is stored in the replay buffer. The RL model transitions to the next period. If the final period T is reached, or the action a_t is infeasible, the RL loop is terminated and the network parameters are updated.

The dotted lines show the training process of the policy Q-network. For every experience sampled from the replay buffer the DQN loss is computed as depicted in Equation 14, and the weights are optimized by using stochastic gradient descent to minimize the loss. The target values of the loss function are estimated using a one-step Temporal Difference (TD(1)) method, and every $C = 10$ episodes the target network is updated with the learning parameters from the policy Q-network.

If the terminal episode E is reached, then the training process is stopped and the learned parameters of the DQN-model can be used to estimate the optimal dynamic fleet policy.

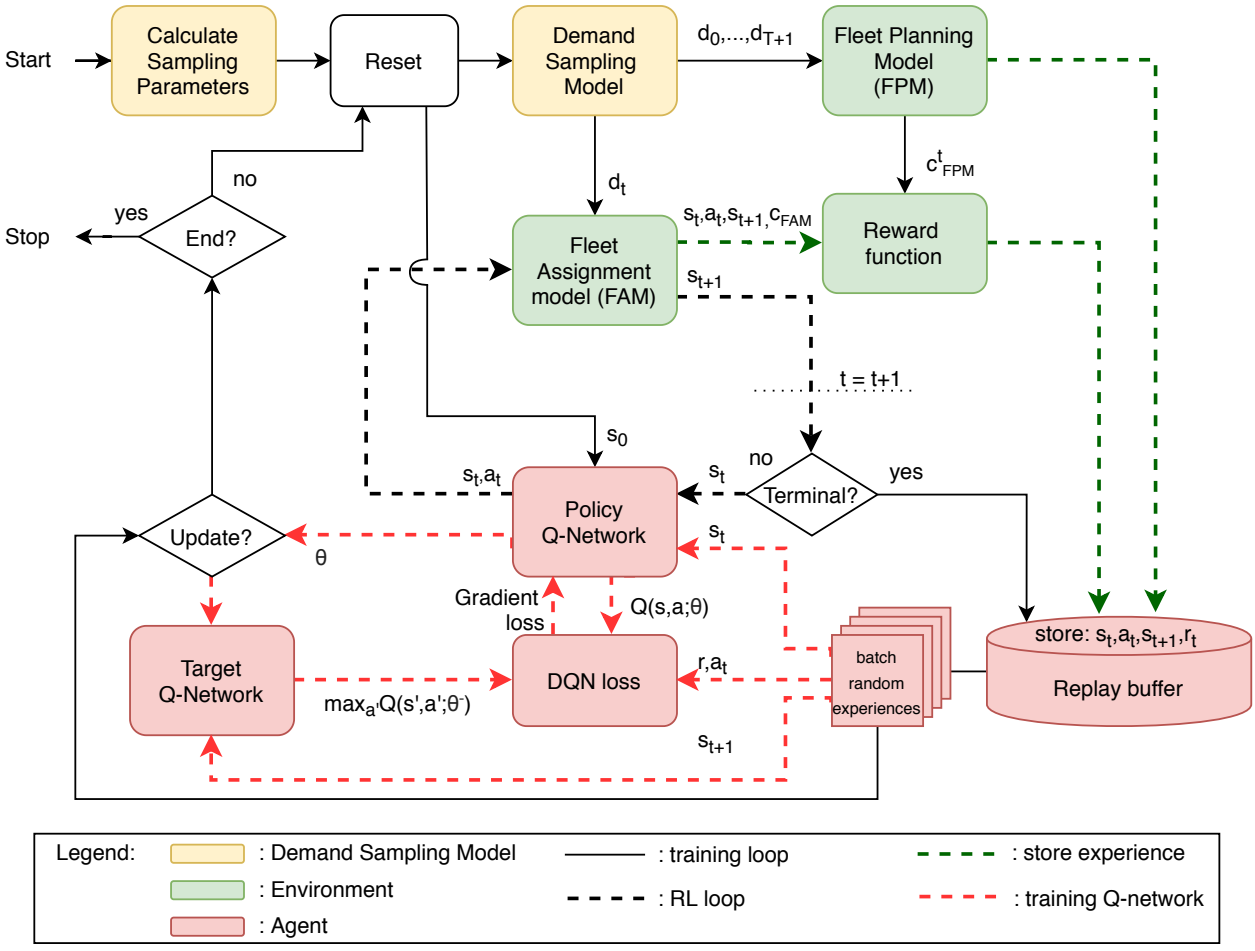


Figure 2: A diagram of the interaction of the Demand Sampling Model (yellow), the Environment (green), and the Deep Q-learning Model (red).

5.2. Fleet Assignment/Planning Model

The Fleet Planning Model is in practice an extension of the Fleet Assignment Model over multiple periods and with the fleet decisions as extra decision variables. A hub & spoke network with the possibility for point to point operations is assumed to be the operational network of the airline. The FPM and FAM are explained in conjunction in the remainder of the section. At every stage, the differences between the two models will be highlighted in the text. The FPM and FAM are both MILP's, which are adapted from Santos (2017).

The decision variables:

There are four types of decision variables: the direct passenger flow, the transfer passenger flow, the number of flights with a specific aircraft type, and the decision variables related to the fleet decisions which include the amount of aircraft owned, amount of aircraft disposed, and the amount of aircraft acquired. The first three types are used in the FAM without the periodical dependency. All decision variables are used in the FPM as described below.

x_{ij}^{t}	passenger flow non-stop from origin airport i to destination airport j in period t
w_{ij}^{t}	passenger flow from airport i to airport j that transfers at the hub in period t
$z_{ij}^{k,t}$	amount of aircraft operating from from airport i to airport j in period t
$ac_{own}^{k,t}$	amount of aircraft owned of type k in period t
$ac_{dis}^{k,t}$	amount of aircraft disposed of type k in period t
$ac_{acq}^{k,t}$	amount of aircraft acquired of type k in period t

The non-decision variables:

Next to the decision variables, a set of non-decision variables are required to define the airline environment and to create the objective function and constraints. All variables below are the fixed values; meaning they do not vary over episodes e.

c_{var}^{k}	variable cost of operating an aircraft of type k per flown km in [dollar/miles]
c_{own}^{k}	cost of owning an aircraft of type k each year in [dollar]
c_{dis}^{k}	cost of disposing an aircraft of type k each year in [dollar]
n_{week}	number of operating weeks per year
n_{year}	number of years in one period
OT_{ij}	average time to fly leg ij in [hours]
TAT_{k}	turn around time of aircraft type k in [hours]
OH_{k}	the maximum operation hours of an aircraft type k per week in [hours]
D_{ij}^{t+1}	demand between airport i to airport j per year in period $t+1$ [pax]
$dist_{ij}$	distance between airport i to airport j [miles]
s^{k}	seats in aircraft type k [pax]
g_{ij}	$g = 0$ if a hub is located at airport i, j, $g = 1$ otherwise
R^{k}	range of aircraft type k in [miles]
F_{init}^{k}	initial owned fleet for type k at initial period $t = 0$

The objective function

As established at the beginning of this section the objective function is to maximise the yearly profit of the airline in future years, which is a combination of maximizing the revenue while

minimizing the expenditures of future operations. The objective function implemented in the FPM is depicted below in equation 20. The revenue is assumed to be solely due to ticket sales on direct and indirect flights. The costs are broken down into operational costs, including those of operating the flights $z_{ij}^{k,t}$, and fixed costs related to the ownership and disposal of aircraft. The fleet costs are unrelated to the actual operational costs of the network, in other words the airline has these costs whether the aircraft is operated or not. The disposal costs of aircraft can be interpreted as fine due to a breach of the lease contract.
The FAM uses a similar objective function. However, the summation over periods is removed, and the fleet costs become a fixed value.

$$\text{Maximized Profit} = \text{Maximized Revenues} - \text{Minimized Costs}$$

$$\begin{aligned}\text{Maximized profit} \;=\; & \underbrace{\sum_{t\in\mathcal{T}}\sum_{i\in\mathcal{I}}\sum_{j\in\mathcal{I}}\left[Fare_{ij}\cdot(x_{ij}^{t}+w_{ij}^{t})\right]}_{\text{Revenue}} - \underbrace{\sum_{t\in\mathcal{T}}\sum_{i\in\mathcal{I}}\sum_{j\in\mathcal{I}}\sum_{k\in\mathcal{K}}\left[C_{var}^{k}\cdot dist_{ij}\cdot z_{ij}^{k,t}\right]}_{\text{Operational Costs}} \\ & \underbrace{-\sum_{t\in\mathcal{T}}\sum_{k\in\mathcal{K}}\left[C_{own}^{k}\cdot ac_{own}^{k,t}+C_{dis}^{k}\cdot ac_{dis}^{k,t}\right]}_{\text{Fleet Costs}}\end{aligned} \tag{20}$$

The constraints:
The objective function is subjected to a set of constraints:

$$x_{ij}^{t}+w_{ij}^{t}\leq D_{ij}^{t+1} \quad ,\forall i,j\in\mathcal{I},t\in\mathcal{T} \tag{21}$$

$$\sum_{j\in N} z_{ij}^{k,t}=\sum_{j\in N} z_{ji}^{k,t} \quad ,\forall i\in\mathcal{I},k\in\mathcal{K},t\in\mathcal{T} \tag{22}$$

$$w_{ij}^{t}\leq D_{ij}^{t}\times g_{ij} \quad ,\forall i,j\in\mathcal{I},k\in\mathcal{K},t\in\mathcal{T} \tag{23}$$

$$\sum_{i\in N}\sum_{j\in N}(OT_{ij}+TAT_{k})\cdot z_{ij}^{k,t}\leq OH_{k}\cdot ac_{own}^{k,t} \quad ,\forall k\in\mathcal{K},t\in\mathcal{T} \tag{24}$$

$$x_{ij}^{t}+\sum_{m\in\mathcal{I}} w_{im}^{k,t}(1-g_{ij})+\sum_{m\in\mathcal{I}} w_{mj}^{k,t}(1-g_{ij})\leq\sum_{k\in\mathcal{K}} z_{ij}^{k,t}\cdot s^{k} \quad ,\forall i,j\in\mathcal{I},t\in\mathcal{T} \tag{25}$$

$$\sum_{k\in K} z_{ij}^{k,t}\leq a_{ij}^{k}=\begin{cases}1000, & \text{if } dist_{ij}\leq R^{k}\\ 0, & \text{otherwise}\end{cases} \quad ,\quad \forall i,j\in\mathcal{I},k\in\mathcal{K} \tag{26}$$

$$ac_{own}^{k,t}+ac_{dis}^{k,t}-ac_{acq}^{k,t}=ac_{own}^{k,t-1},\quad \forall t=(1,\ldots,T),k\in\mathcal{K} \tag{27}$$

$$ac_{own}^{k,0}+ac_{dis}^{k,0}-ac_{acq}^{k,0}=F_{init}^{k},\quad \forall k\in\mathcal{K} \tag{28}$$

$$x_{ij}^{t}\in\mathbb{R}^{+},w_{ij}^{t}\in\mathbb{R}^{+},z_{ij}^{k,t}\in\mathbb{Z}^{+},ac_{own}^{k,t}\in\mathbb{Z}^{+},ac_{dis}^{k,t}\in\mathbb{Z}^{+},ac_{own}^{k,t}\in\mathbb{Z}^{+} \tag{29}$$

The first set of Constraints (21) dictates that the sum of direct and indirect passengers transported over a route ij does not exceed the demand in period $t+1$. The demand matrix $D = \{d_1, ..., d_{T+1}\}$ is shifted one time step. As these sets represent the revealed air-travel demand in the environment. The set of Constraints (22) ensures the aircraft balance in the airports because the operated network is repeated on a weekly basis. At the beginning and end of each week, the amount of arrived aircraft in each airport must equal the amount of departed aircraft. The set of Constraints (23) allows indirect passengers only to be transferred via the hub airport. The fourth set of Constraints (24) limits each type of aircraft k to be operated no more than the maximum allowance of operating hours per week. The amount of aircraft owned of type k will define how many times routes can be operated. The set of Constraints (25) ensures that the amount of non-stop and transfer passengers in each route is lower than or equal to the maximum amount of seats on a route. Similarly to (24), the amount of available seats is dependent on how many aircraft of type k operate that route. The range of Constraints (26) ensures that flights can only be operated by aircraft types which have sufficient range capability. The set of Constraints (21,22,23,24,25,24,26) are used in both the FPM and FAM, although the dependency of periods is removed in the FAM. Because every set of constraints is repeated for $t \in \mathcal{T}$ in the FPM, the FAM will have $T+1$ times fewer constraints.

The set of Constraints (27 and 28) are added to the FPM to extend the single period fleet planning model to a multiple period fleet planning model. The set of Constraints (27) dictate that the amount of aircraft owned, plus aircraft disposed, and minus aircraft acquired in period t must equal the aircraft owned in the previous period $t-1$. Constraint (28) initializes the period zero with the initial fleet F^k_{init}.

To conclude, the integrality and non-negativity Constraints (29) are depicted. Both of the decision variables related to the transported passengers (x_{ij} and w_{ij}) are assumed to be positive real numbers. As a result, these variables are continuous, because the influence on the profit is marginal, and the computational load is decreased. The decision variables related to the aircraft fleet and flight frequency are positive natural numbers.

5.3. The reward function:

The set of solutions $(C^0_{FPM}, ..., C^T_{FPM})$ for every optimal action a^*_t in $\mathcal{T}$, represents the upper bound of the profit C_{FAM} for an air-travel demand trajectory $\{d_0, ..., d_{T+1}\}$ in an episode. In the RL loop, after the agent's fleet decision, the FAM optimizes profit (C_{FAM}) given the fleet decision a_t and revealed demand d_{t+1}. The reward function transforms the calculated profit into a reward.

If the profit of the FAM is equal to the profit of the FPM ($C^t_{FPM} = C_{FAM}$), the fleet decision a_t of the agent is the optimal fleet decision a^*_t. Consequently, the accompanying reward is $+1$. However, a meaningful reward needs to be established for all sub-optimal fleet decisions and profits. Obviously, the worst decision should receive a reward of -1, but calculating the profits to all possible fleet decisions in the action space $\mathcal{A}$ to establish a profit range is a cumbersome task that is too computationally expense. Moreover, the worst fleet decision in the action space $\mathcal{A}$ can often yield a profitable airline. Hence, a lower bound $(lb) \in (0, 1)$ is established which is a percentage of the optimal profit. The rewards of the sub-optimal fleet decisions are obtained by mapping the profit C_{FAM} on a linear function between the optimal profit (C^t_{FPM}) where $r_t = +1$, and lower bound ($C^t_{FPM} \cdot lb$) where $r_t = 0$.

If the profit C_{FAM} is lower than $C^t_{FPM} \cdot lb$, the reward immediately becomes zero. Finally, the agent is punished for fleet decisions which are infeasible. If the fleet decision by the agent disposes more aircraft than available, the reward is (-1).

$$r_t = \begin{cases} -1, & \text{if } ac^{k,t}_{own} - ac^{k,t}_{dis} < 0 \\ \dfrac{C_{FAM} - lb \cdot C^t_{FPM}}{(1 - lb)\, C^t_{FPM}}, & \text{elseif } C_{FAM} > lb \cdot C^t_{FPM} \\ 0, & \text{else} \end{cases} \tag{30}$$

By increasing or decreasing the lower bound the aggressiveness of the rewards can be tuned. However, the learning of the agent can be very sensitive to the lower bound. If the lower bound is too low, all fleet decisions will have a reward close to +1, and the agent will be unable to learn the optimal policy. If the lower bound is too high, the rewards will be very sparse. As a result, the agent would have little rewards to learn from and have trouble converging to an optimal policy. The lower bound is therefor a hyperparameter of the model which is tuned to achieve the best learning.

6. Experimental set-up

6.1. Case Study

As proof of concept, a case study is conducted using the proposed methodology. The aim is to mimic the real life fleet planning process of a small airline operating on a domestic airport network in the United States (US). Ten major US airports are included in the network comprising 90 possible routes. Two aircraft types will be considered in the case study: A Boeing 737-800 (BOE738) and a Boeing 757-300 (BOE753). These aircraft are typical examples of narrow-bodies aircraft commonly chosen by airlines to operate shorter domestic flights. A planning horizon of 10 years is assumend with a fleet decision every 2 year resulting in 5 time periods for the RL loop. In Table 1 the case study parameters are depicted.

Table 1: Case study parameters

Notation	Definition	Value
E	# Episodes	5000
T	# Time horizon	5
Y	# Planning horizon	10
N	# Routes in network	90
M	# Markets in network	45
K	# Aircraft types	2

In Table 2, the aircraft parameters are displayed for two aircraft types commonly operated in domestic networks. The BOE738 is a newer narrow-body aircraft with a higher ownership cost. The BOE753 is an older type of aircraft thus having a lower ownership cost, yet it is more expensive to operate per flown mile due to higher fuel costs. There is not a high initial cost for the acquirement of the aircraft, as it is assumed the aircraft are leased on a yearly basis. If a lease contract is broken, the airline is assumed to pay an extra year in the form of a disposal cost. In addition, it is assumed that all flights have a load factor of $LF = 85\%$, and the aircraft operations are continued for $n_{week} = 50$ per year.

Symbol	s^k	v_c^k	$range^k$	OH^k	TAT^k	c_{var}^k	c_{own}^k	c_{dis}^k
Units	–	$\frac{hour}{week}$	$miles$	$\frac{hour}{week}$	$hour$	$\frac{USD}{mile}$	$\frac{USD}{year}$	$\frac{USD}{year}$
BOE738	162	543	3582	77	1	0.13	3.05E+06	3.05E+06
BOE753	243	530	3357	80	1.5	0.14	2.4E+06	2.4E+06

Table 2: Aircraft-related parameters

6.2. Demand Model parameters

The air-travel demand on routes is very difficult to measure as it is dependent on various parameters such as the air-fare, time of year, special events, etc. Except from surveying, the only data which resembles the air-travel demand on routes is the number of passengers who actually travelled. Consequently, it is assumed that the historical transported passengers represents the historical air-travel demand, and can be used as a predictor of future air-travel demand values.

The historical travelled passenger data is extracted from the Bureau of Transportation Statistics (BTS), part of the United States Department of Transportation (US DOT). In the TransStat database, the T-100 Domestic Market (U.S. Carriers) database contains the historical monthly market data of all US airlines. It is important to note that the historical travelled demand data is the market data between two airports, meaning it contains all passengers transported between two airports directly and indirectly by all US airlines.

With the adapted demand sampling model, infinite variations of demand matrices based on estimated parameters can be sampled. The variance of the shock terms $(dW_t, dW_{m,t})$ are defined by the Wiener process and therefor initially defined as $\mathrm{N} \sim (0, 1)$. However, the estimation of OU parameters on the markets shows very high historical estimation errors σ due to historical variability in transported passengers. As these estimation errors produce unreasonable shocks and demand growths in the prediction of air-travel demand, the standard deviation σ of the normal distribution's disruption effect is lowered my multiplying it with a predefined smoothing factor $\eta \in (0, 1)$ and $\eta_m \in (0, 1)$ for respectively the network and market prediction. In Equation 31 and 32, the adaptations for the Ornstein-Uhlenbeck sampling model are visualized.

$$\delta_{t+1} = \delta_t + \lambda\left(\mu' - \delta_t\right) + \eta\sigma dW_t) \tag{31}$$

$$x_{m,t+1} = \frac{1}{2}(\delta_{t+1} + x_{m,t} + \lambda_m\left(\mu'_m - x_{m,t}\right) + \eta_m\sigma_m dW_{m,t}) \tag{32}$$

The normal distributed long-term growth rates $(\mu' \sim \mathcal{N}(\mu, \sigma_\mu^2), \mu'_m \sim \mathcal{N}(\mu_m, \sigma_{m,\mu}^2))$, replace the previously fixed long-term mean growth rates. In Section 4, it was established that the variance terms $(\sigma_{\mu g}^2, \sigma_\mu^2)$ define the amount of dispersion of the long-term mean growth, and are difficult to define because of the lack of data. Due to the uncertainties in the variance terms of the long-term mean growth and the smoothing factor of the shock term, a set of demand scenarios generated to observe different sampling behaviours and investigate the sensitivity to the DQN-model. Three demand scenarios are generated: Average Demand (AD), Dominant Network Demand (DND), and Dominant Market Demand (DMD). In Table 3 the variance values and smoothing factors for the demand scenarios are depicted.

	η_m	η	$\sigma^2_{m,\mu}$	σ^2_{μ}
Average Demand	0.5	1	0.005	0.005
Dominant Network Demand	0.1	1	0.005	0.05
Dominant Market Demand	1	0.1	0.05	0.005

Table 3: Smoothing factor of the shock term and Variance of long-term mean growth for the three demand scenarios: Average Demand (AD), Dominant Network Demand (DND), Dominant Market Demand (DMD).

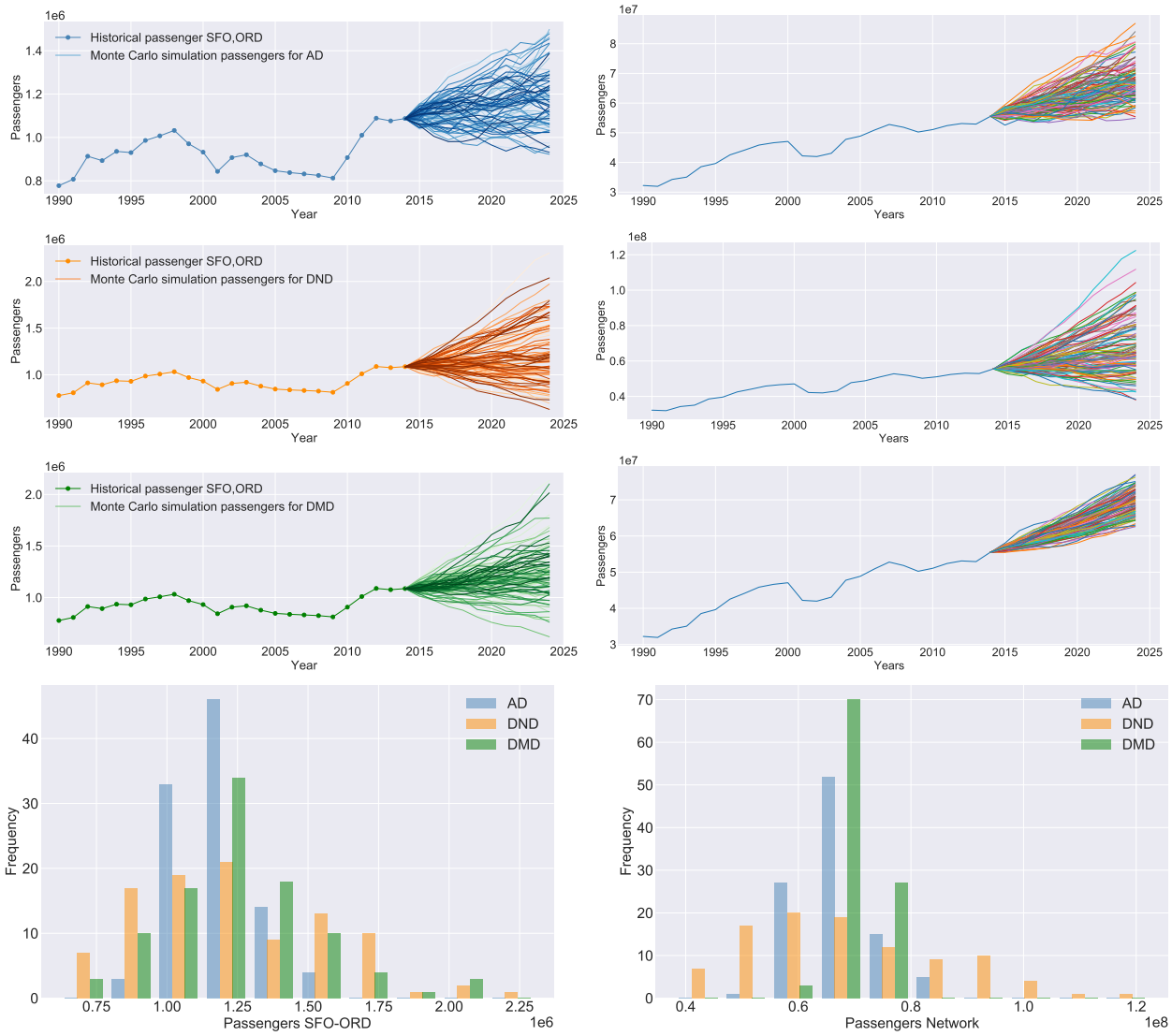


Figure 3: Simulations of stochastic demand trajectories for the market SFO-ORD for the three demand scenarios (left), and the corresponding cumulative network demand for the three demand scenarios (right); the histograms show the distributions of the predicted demand in period $T + 1$.

Figure 3 shows a simulation of the 50 trajectories sampled using the three different demand scenarios. The top row shows the simulation of the AD scenario, the second row the simulation of the DND scenario, and third row the simulation of the DMD scenario. Finally, the histogram on the bottom row shows a comparison of the demand predictions in period

$T + 1$ for the three demand scenarios. The left side shows the simulation of the market SFO-ORD, the right plots show the cumulative simulated demand in the network.

The AD scenario employs a smoothing factor $\eta_m = 0.5$ to compensate for the high shock terms in the market scenarios. Because, the historical network growth is a summation of multiple historical markets, the estimation error for the network σ is lower and does not require smoothing. The variance terms of the network and markets are assumed to equal 0.005, which translates in a long-term deviation of the mean growth of 0.5% for 68% of the demand trajectories. The latter parameters showed the most reasonable stochasticity and growth behaviour outlined in Figure 3. Both the market demand and the network demand shows a long-term mean growth which is reasonably distributed, and a stochastic behaviour both on a market level and a network level.

The DND scenario employs a higher smoothing factor for the market and a higher variance for the network long-term mean growth. The result depicted in Figure 3 (row 2). This demand scenario has a very strong stochastic behaviour and high dispersion on the network's long-term mean growth sampling, contrary to the market's long-term mean growth sampling. As a result, the market's growth is dominated by the network long-term mean growth sampling. The sampled cumulative demand trajectories behave less stochastic than the AD scenario, but the spread of the air-travel demand at the modelling horizon is large due to the increase in variance in the normal distribution sampling of the long-term mean growth rate of the network σ_μ^2.

The DMD scenario samples a future air travel demand which is highly market dependent and has little influence on network growth. The smoothing factor of the market is low and the variance is high, contrary to the network high growth smoothing factor, accompanied by a low variance term. As the influence from the network growth and stochasticity is very limited, this demand scenario resembles the use-case of Sa et al. (2019). In Figure 3 (row 3) an example of 50 sampled demand predictions of the route SFO-ORD (left), and the resulting cumulative network demand (right) is depicted. It is clearly visible that the divergent and stochastic behaviour of the market demand samples is mitigated in the cumulative network demand trajectories.

6.3. Hyperparameter Tuning

The Q-function approximator of the DQN is a fully connected feed-forward neural network. To estimate the appropriate fleet action, the state of the environment s_t is normalized, and fed to the input layer of the neural network. The input vector consists of 48 state values for the SD policy and 3 state vectors for the SS policy, and every value corresponds to a single neuron. The input layer is fully connected to two sequential hidden layers, each with 64 neurons, with a (non-linear) *elu* activation function. After the two non-linear hidden layers, an extra hidden layer with a linear activation of the weights is added. When performing a regression problem, such as approximation of the value of the action in the output layer, a conversion of the non-linear outputs of the elu layers to a real value is needed (Goodfellow et al., 2016). The final hidden layer is fully connected to the output neurons, which correspond to the size of the action vector. It is important to note that the DQN-model was tuned for the SD policy only, and for both policies the same hyperparameters were used. In Table 4 all the relevant hyperparameters of the DQN are shown:

Hyperparameter	Value
Multi-step returns	1
Learning rate [α]	0.001
Discount rate [γ]	0.95
Exploration [ϵ]	$1 \rightarrow 0.05$
Maximum memory length	100K experiences
Batch size	64 experiences
Hidden layers	3 layers
Dense size	64 neurons
Lower bound [lb]	0.985

Table 4: DQN hyperparameters

6.4. Training methodology

The RL model is run for $E = 5000$ episodes, which was determined iteratively to be a sufficient length to learn the fleet policy. Every episode, five periods are considered as five fleet decision points, and consequently RL loop is represented by five agent-environment iterations. Every period is assumed to consist of $n_{year} = 2$ years. At the beginning of each period a fleet decision is taken, with an assumed delivery time of one year. Consequently, the new fleet composition is assessed (the reward) in the two consecutive years after the first year has ended, and the aircraft delivered. For a five period time horizon with two years in each period, 11 years need to be sampled. As a result the sampling horizon becomes $Y = 1 + (n_{\text{years}} \cdot (T + 1))$. In Figure 4 a schematic representation demand revelation with actions and rewards for a finite horizon of 5 periods is illustrated.

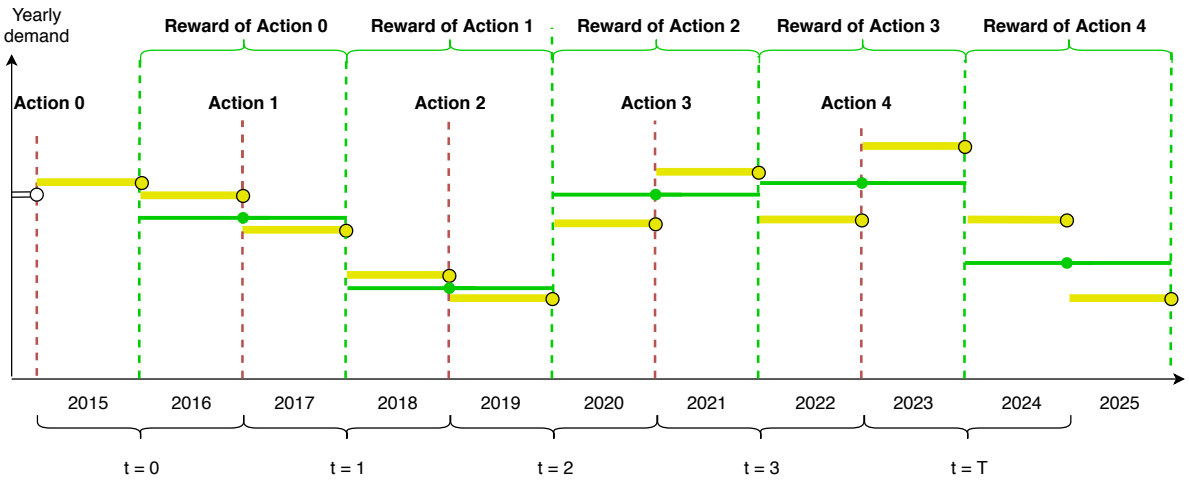


Figure 4: Example of one episode of sampled yearly demand (yellow dots) over a finite-horizon with fleet decisions (red) and calculation of the corresponding reward (green).

During the RL loop experiences are stored in the replay buffer. If the time horizon is reached, the policy NN is trained on the 64 randomly selected experiences from the replay buffer. The generation of experiences is computationally expensive due to optimisation of the FPM in the RL loop. In order to increase the efficiency and sample generation, at the start

of the learning process, multiple RL loops are run for one demand sample to increase the exploration of the state-action space and the usage of the FPM optimisation. In Figure 5 it is visible that at the start of the learning process five iterations of the RL loop are performed which decreases exponentially to one iteration from episode 2500 to the end.

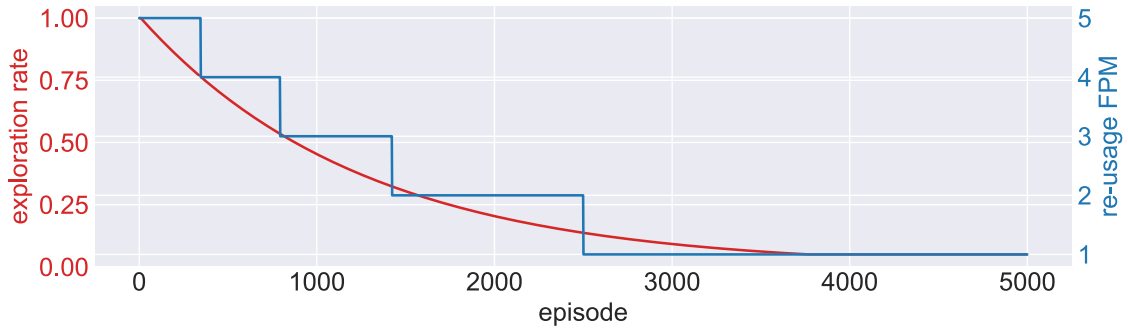


Figure 5: Exploration rate (red), and the amount re-usages of the FPM in one episode (blue).

Every episode the FPM optimizes the optimal weekly operational network, with the fleet evolution and the accompanied profit. As the number of considered airports in the network increases, so does the computational time of the optimisation algorithm. Consequently, the training becomes a very slow process, as the optimisation algorithm struggles to decrease the optimality gap between the primal and dual problem of the MILP. A solution is to increase the MIP gap and let the optimisation stop earlier. As a result, during training, the MIP gap of the FPM and FAM is set to $1E-3$. A side-investigation showed that the average impact of decreasing the gap from $1E-4$ to $1E-3$ on the optimal solution's reward function is less than 1%. This error-margin is assumed to be acceptable.

The performance of the training process is measured by the training score. The training score is measured by the average reward in the RL loop, by dividing the accumulated reward over five periods by the optimal achievable accumulated reward $R_{optimal} = 5$. The final 25% of the episodes, at every target NN update, the trained model is validated using a validation set of 25 air-travel demand evolutions. The validation score of the NN shows an unbiased evaluation of the model performance on the training data. Similar to the training process, the RL loop is run and the agent predicts the optimal fleet decision for each period t; however, the air-travel demand is not sampled randomly iteratively inserted from the validation set. The accumulated reward divided by $R_{optimal} = 5$ to obtain the validation score of one demand trajectory; this process is repeated for all 25 demand evolutions to calculate an average and variance of the validation score for a NN.

After the training process, the *Average Validation Score* and *Average Variance Validation Score* shows the average score and average variance over all the validated NN. The latter is an indication of the average variance of the NN's performance on the validation set. *Variance of Average Validation Score* shows the variation of the *Average Validation Score* and is and an indication of the performance dispersion of over the trained NN's. Finally, the trained NN with the highest score on the validation set is selected as the optimal fleet predictor based on the best training score and is then used for evaluation.

7. Result Analysis

7.1. Evaluation methodology

To evaluate the trained fleet policies SS and SD, two conventional fleet planning policies, referred to as the Deterministic Static (DS) and Deterministic Dynamic (DD), are developed. The DS and DD policies represent the fleet planning problem as a deterministic bottom-up approach and utilize a deterministic sampling of the air-travel demand to optimize an FPM and predict the optimal fleet composition. The deterministic demand of the two models is estimated using the adapted Ornstein-Uhlenbeck process without the stochastic shock and stochastic drift term. The result is a quasi-linear extrapolation of the historic long-term mean growth. An example of such a sampling process is visible in Figure 6.

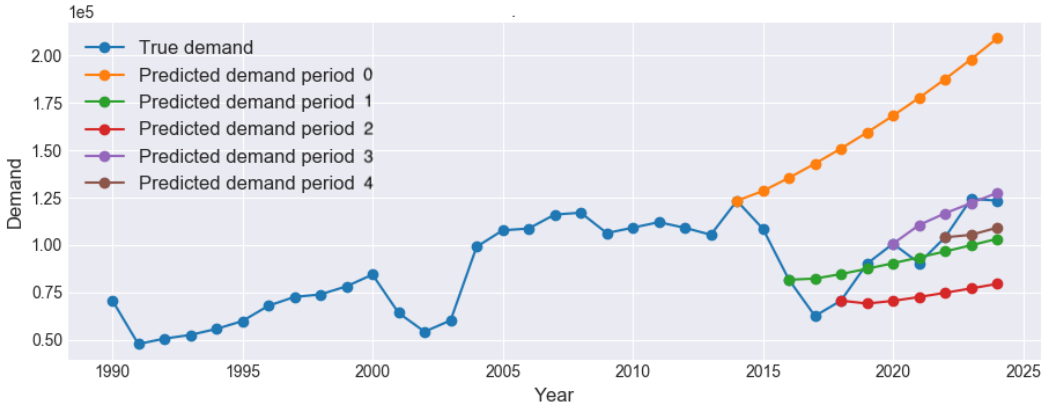


Figure 6: Example of a stochastic demand sampled for the market 'SFO-ORD' (blue), and intermediate non-stochastic demand samples throughout the episodes.

Here, the blue line consists of the historic yearly demand and a stochastic sampled demand from 2014, representing the true demand evolution of an episode. When the first fleet decision is taken in $t = 0$, the demand is predicted with the deterministic sampling policy, and the FPM calculates the optimal fleet decisions for the next 5 periods. The DS policy applies the 5 fleet decisions calculated at period $t = 0$ as a static policy, and follows that policy without adaptations to the revealed demand values. On the contrary, the DD policy allows for the fleet policy to be updated dynamically. At every period, after the true demand values of the previous period are revealed, a new deterministic demand evolution is predicted as well as an FPM optimized, spanning the remaining periods, to obtain the dynamic fleet decisions.

The DS policy is comparable to the Stochastic Static (SS) policy as both policies are not dependent on the evolution of the sampled demand. Both static policies represent a fleet planning process where a long-term fleet policy is generated which is not updated over time. The DD policy is comparable to the Stochastic Dynamic (SD) policy as both the policies represent a fleet planning process where the optimal fleet plan is re-optimized over time to generate a dynamic long-term fleet policy.

To evaluate and compare the stochastic and deterministic models three evaluation-sets (one

for each demand scenario) are constructed containing 50 air-travel demand predictions including the optimal fleet decisions and profit. The trained fleet policy showing the highest Validation Score is utilized as the optimal network and is compared to their deterministic counterparts. Similarly to the validation process the *Evaluation Score* and *Variance Evaluation Score* are calculated for the evaluation data-set of 50 samples.

7.2. Average Demand

Figure 7 shows the training score and validation score of the two stochastic models. It can be noted that the training score suffers from high variance. This is attributed to the assumption that the MDP is a fully observable process. The Markov Property assumes that the current state is a sufficient statistical of the future Silver (2018). It can be argued that the current state of the air-travel demand is not predictively sufficient of future observation of air-travel demand. This is because the action of a state is evaluated after two revelations of growth in air-travel demand, and can abruptly change due to the shock term of the OU sampling process. The prediction of the action is therefor the most-likely action under the probability distributed evolution of the demand. However, the 'noise' due to unpredictable and volatile revelations of air-travel demand induces a large amount of variance in the model.

Due to the variance, the training score and validation score is smoothed out using a moving average over 20 samples to visualize the trend. In the early stages of training, the training score is low for both stochastic models. As the number of episodes increases and the exploration decreases, the NN of the DQN's are trained and more increasingly the greedy policy is employed which leads to an increasing training score. It is clearly visible that the inclusion of the current demand for markets increases the training score of the DQN-model; the validation score confirms this behaviour. This is an indication that the SD NN can detect patterns in the demand values, and the dynamic policy chooses more optimal fleet decisions compared to the static policy.

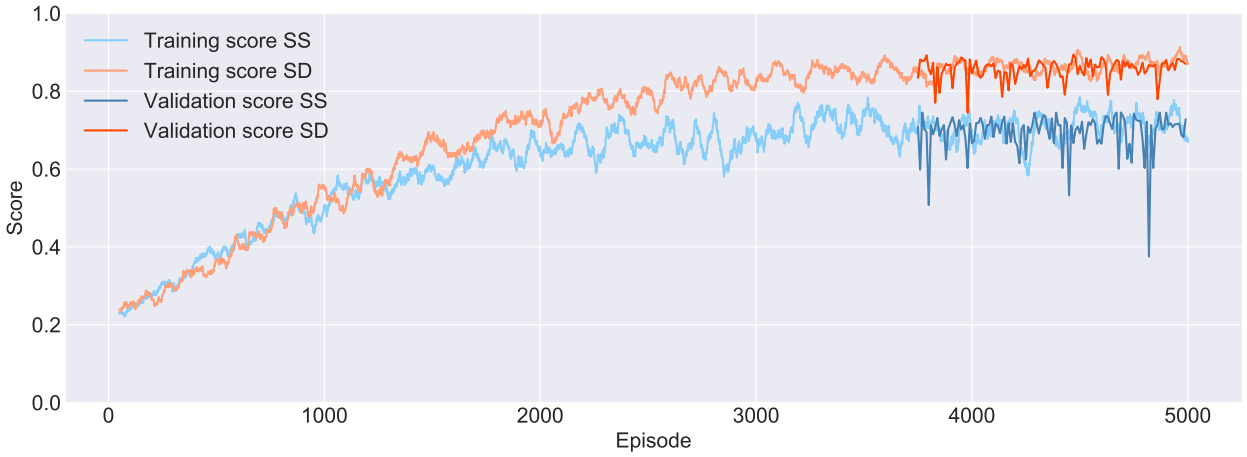


Figure 7: Training score and Validation Score of SS and SD policy network for Average Demand scenario.

In Table 5, the validation and evaluation scores are displayed. The Average Validation Score shows that the stochastic models perform slightly sub-optimal to their deterministic counterparts. Again it is clearly visible that the dynamic policies out-perform the static

policies. Moreover, the Variance Validation Score and Variance of Average Score improves with the addition of intermediate evaluations of the fleet decision based on the air-travel demand features (SD policy).

The Evaluation Score shows better results than expected w.r.t. the Average Validation Scores. The SS policy outperforms the DS counterpart both on Evaluation Score and Variance Evaluation Score. The SD policy performs not considerably worse than the DD benchmark for both the Score and Variance.

	DS	SS	DD	SD
Average Validation Score	73.68%	69.44%	88.86%	85.68%
Average Variance Validation Score	5.762	4.72	0.906	1.16
Variance of Average Val. Score	-	25.68	-	6.656
Evaluation Score	73.86%	74.86%	90.59%	88.81%
Variance Evaluation Score	6.615	6.597	0.522	0.691

Table 5: Average Validation Score of Average Demand scenario

A similar behavior is visible in Figure 8 and Figure 9. The relative testing error shows the percentage difference between the profit using fleet prediction models (C_{FAM}), and the optimal (true) profit solution (C^t_{FPM}). In all four policies, the relative errors from the optimal solution are equal for the first period. The reason behind this is that the initial state is always the same state, and as a result, the policies will always predict the same fleet decision.

The SS policy shows the exact predictions and errors as the DS in periods one through three. However, in the fourth period, the mean and spread decrease slightly; wherein the fifth period, the spread of the error of the SS policy starts to increase again compared to the DS. Overall, the SS policy shows slightly better results than the DS policy.

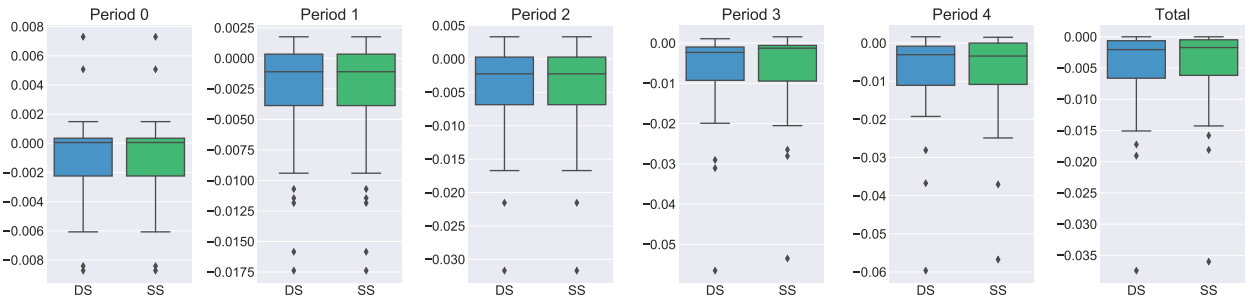


Figure 8: The relative testing error of the Deterministic Static (DS)(blue), and the Stochastic Static (SS)(green) policy to the optimal solution profit for Average Demand scenario.

Figure 9 shows the comparison of the dynamic policies. Where the stochastic policy shows improved prediction for periods three and four, the deterministic policy performs better in period two and five. The difference is more pronounced in the comparison of the total profit errors where DD outperforms SD. However, the range of the y-axis reveals that the difference in error is small therefor both policies show similar results.

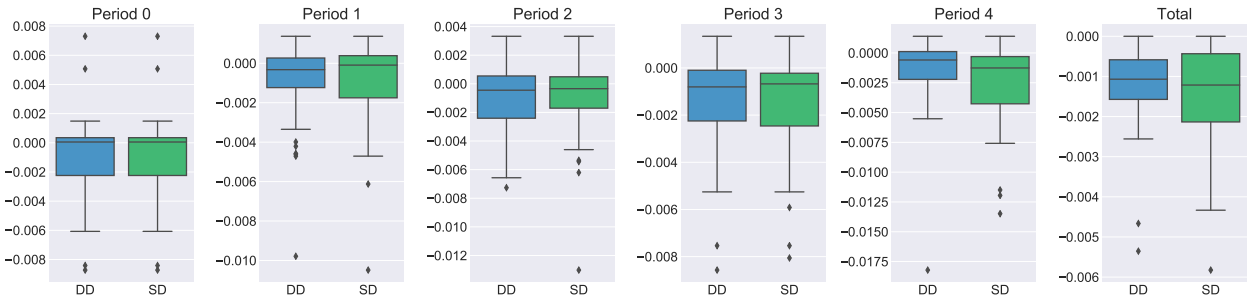


Figure 9: The relative testing error of the Deterministic Dynamic (DD)(blue), and the Stochastic Dynamic (SD)(green) policy to the optimal solution profit for Average Demand scenario.

7.3. Dominant Network Demand

In Figure 10, the training score and validation score of the two stochastic models are presented. Both the SS and SD training score increases over the episodes but show a much lower score compared to the training score of the AD scenario (Figure 7). Moreover, the difference between the dynamic and static training score is much higher than in the AD scenario. This behaviour is attributed to the DND scenario's air-travel demand which -at the end of the modelling horizon T- has much larger dispersion than the AD scenario. As a result, the static policies, which are not updated iteratively on the evolution of air-travel demand, perform inferior to the dynamic counterpart. Moreover, the experiences at the end of the time horizon will have very contradicting actions to the same state, thus inducing a lot of noise and training variance. This is visible in the validation score of the SS method which shows a high variance compared to the SD approach (Variance of Average Val. Score).

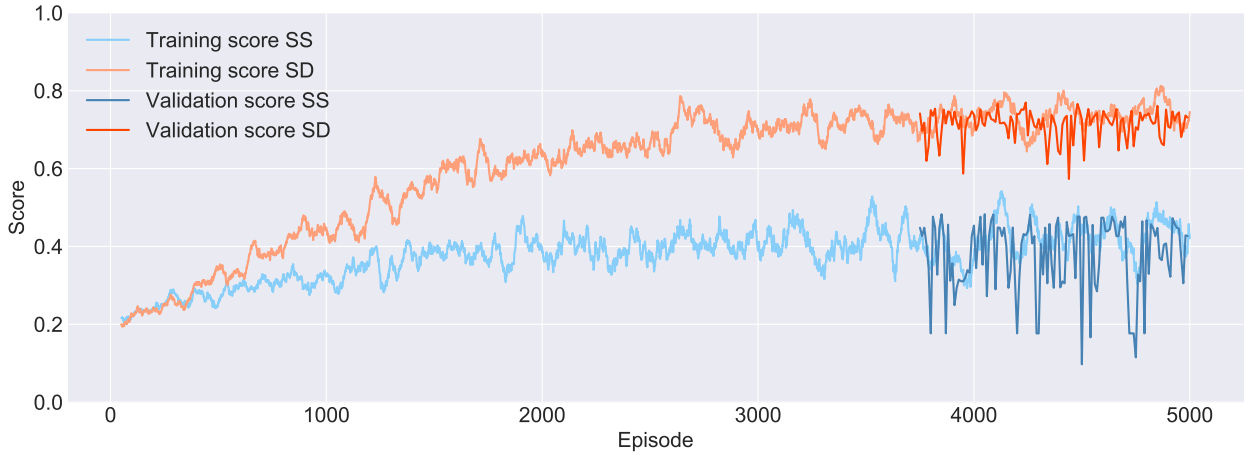


Figure 10: Training score and Validation Score of SS and SD policy network for Dominant Network Demand scenario.

In Table 6 the validation and evaluation scores and variances are presented. Next to the high Variance of Average Validation Scores, also the Average Variance Score is notably larger than in the AD scenario. The Average Validation Score of the DS approach seems to outperform the average validation score of the SS method by 10%. However, the Evaluation score of SS outperforms the DS evaluation score, but with a higher variance. This again

shows high variance in the NN training and dependence on the model selection as the final predictor of the fleet.

The dynamic policies too show a high variance as well as lower validation and test scores compared to the AD demand scenario. However, the SD trained NN outperforms the DD policy both in the validation- and evaluation score and variance. Especially the Average Variance Score is notable lower for the SD policy. Because the DD policy samples the demand based on the average growth of the historic transported passengers (as depicted in Figure 6), it overestimates (or underestimates) the mean growth of demand in the extreme demand trajectories. The SD policy adapts better to the demand trajectories and is able to outperform the DD policy both on evaluation and validation score.

	DS	SS	DD	SD
Average Validation Score	47.24%	37.84%	64.91%	71.39%
Average Variance Validation Score	12.363	8.26	7.63	3.35
Variance of Average Val. Score	-	88.027	-	13.761
Evaluation Score	43.09%	46.45%	68.44%	71.2%
Variance Evaluation Score	7.491	8.075	5.471	3.557

Table 6: Average Validation Score of Dominant Network Demand scenario

In Figure 11 and 12, the most notable trend is the lower variance of the SS and SD policy in the early stages of the planning horizon. The lower variance is a direct result of training the stochastic policy on sampled evolutions of demand. Training on this evolutions yields initial fleet decision which is better over a wider range of demand evolutions. Towards the end of the planning horizon, the error unfortunately increases for the SS policy. As a result, the total relative error from the optimal solution is very similar for both static methods with a slightly increased variance for the SS model.

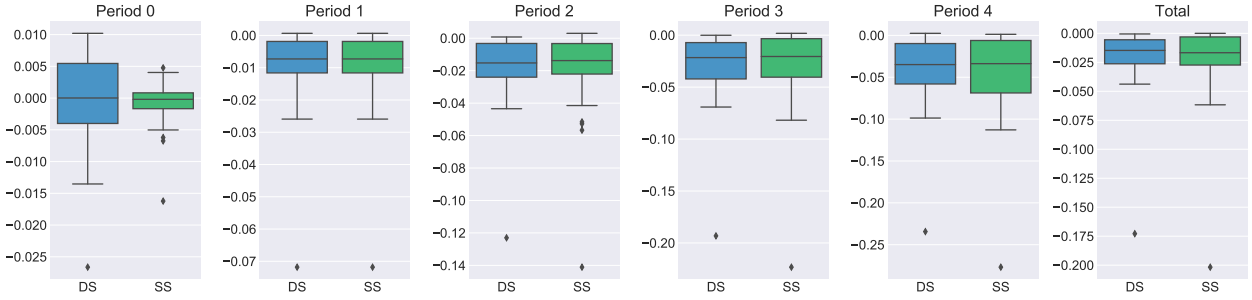


Figure 11: The relative testing error of the Deterministic Static (DS)(blue), and the Stochastic Static (SS)(green) policy to the optimal solution profit for Dominant Network Demand scenario.

Figure 12 shows a significantly improved performance for all periods except period two, which could be attributed to an over-fitting of the NN in that period. The total relative error of the SD is again similar to the DD with a lower variance for the SD.

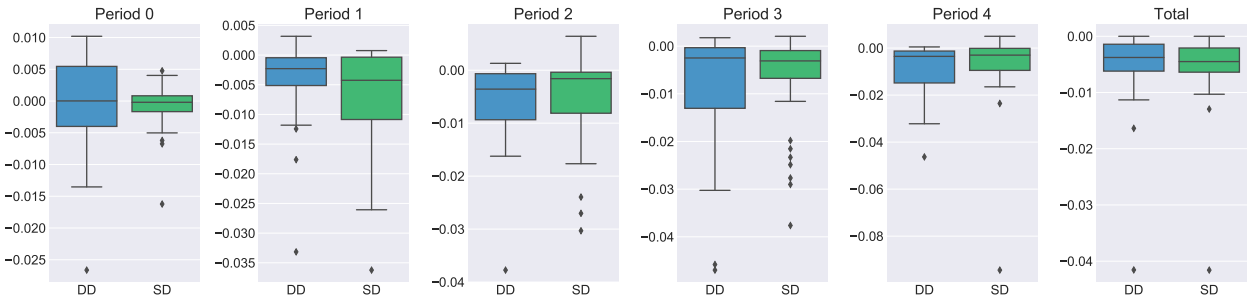


Figure 12: The relative testing error of the Deterministic Dynamic (DD)(blue), and the Stochastic Dynamic (SD)(green) policy to the optimal solution profit for Dominant Network Demand scenario.

7.4. Dominant Market Demand

In Figure 13 the training score of the RL model for both the SS and SD policy under the DMD scenario is shown. It is immediately notable that both the SS and SD policy converges to a very high and a similar training score. Moreover, both the training and the validation scores show low variance over the validated networks. The difference in performance scores between the static (SS) and dynamic (SD) policy is the lowest of all three tested demand scenario's. Because of the mitigating effect of sampled multiple markets, the dispersion of cumulative demand in the network is low, and the optimal fleet decision does not deviate much. Consequently, the static policy is as almost as good an estimator as the dynamic policy.

Figure 13: Training score and Validation Score of SS and SD policy network for Dominant Market Demand scenario.

In Table 7 the validation and evaluation scores and variances are shown. The stochastic trained policies show compatible results to the deterministic counterparts. The SS model's Average Validation Score is practically equal to the DS score but with a lower variance score. The SD shows a slightly sub-optimal Average Validation Score and Variance to the DD model. The evaluation of the models displays similar results, with the stochastic trained DQN-models matching the deterministic optimized method on both Average Score and Variance Score. The comparable results between the stochastic and deterministic are again related to the

demand scenario. In DMD, the influence of the growth sampling of the network is very little on the episodic sampling of air-travel demand, compared to the sampling of independent market growths. The resulting divergence of network growth over these samples is therefor small, and the optimal fleet decision converges to a small set of actions which always score high. This is attributed to the fact that a small growth in one market can be compensated with growth in another market.

	DS	SS	DD	SD
Average Validation Score	87.18%	87.34%	97.78%	95.61%
Average Variance Validation Score	2.158	1.41	0.065	0.09
Variance of Average Val. Score	-	14.983	-	1.229
Evaluation Score	85.83%	87.83%	96.74%	96.86%
Variance Evaluation Score	2.382	2.519	0.123	0.054

Table 7: Average Validation Score of Average Demand scenario

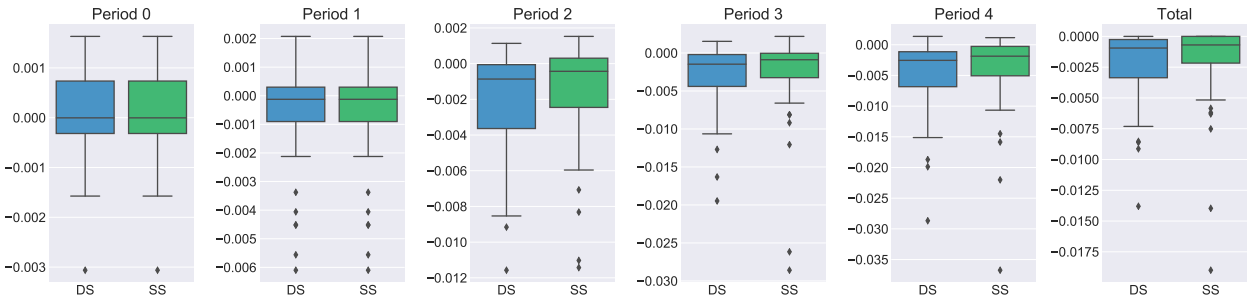


Figure 14: The relative testing error of the Deterministic Static (DS)(blue), and the Stochastic Static (SS)(green) policy to the optimal solution profit for Dominant Market Demand scenario.

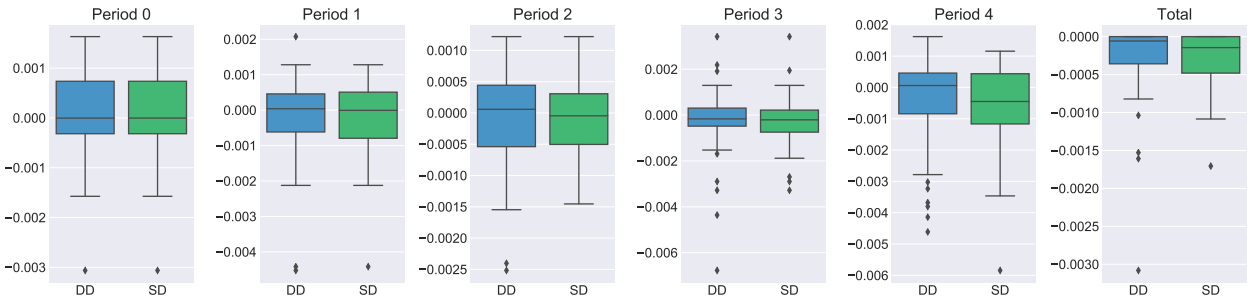


Figure 15: The relative testing error of the Deterministic Dynamic (DD)(blue), and the Stochastic Dynamic (SD)(green) policy to the optimal solution profit for Dominant Market Demand scenario.

In both Figure 14 and 15, the static and dynamic approach are shown respectively. The relative error of the SS is proportional to the DS method aside from a few outliers in the SS prediction that induce a higher variance of the evaluation score. The dynamic policies show comparable relative errors and variances too. The relative error on the total profit is very small from the optimal solution with minor deviations between the DD and SD.

7.5. Discussion

In the three demand scenarios, both the Validation- and Evaluation Score/Variance of the dynamic policy using air-travel demand as a feature show improvement over the static policy. Both policies were trained in a similar method with the only difference being the number of input features in the neural network. As the Static Stochastic (SS) policy was trained on only the fleet and period features, and the Dynamic Stochastic (SD) included features about the current state of the air-travel demand in the network, it was found that re-evaluation of the current air-travel demand increases the predictability of the optimal fleet decision and thus the performance of the fleet plan.

The SD policy showed overall a lower variance than the SS policy. As the SD policy can adjust the fleet decision based on the current demand, the SS policy trains for the best policy over the aggregate of air-travel demand evolutions. Moreover, the evaluation of the SS policy outperforms its deterministic reference policy (DS) for every demand scenario.

The SS policy validation score shows a lot more variance compared to the SD policy. This attributed to the training on very conflicting experiences; where the similar input states yield different fleet actions, which increases the noise during the training process. Moreover, the SS policy and SD share the same neural network architecture which is probably over-fitted for the SS due to its small input vector and the less complex policy it is trying to approximate.

The AD scenario shows the largest stochasticity in the market and network demand evolution out of the three scenarios due to the low smoothing the shock term and low variance of the long-term demand growth. As a result, the SD and SS model have more trouble learning as the stochasticity in the air-travel demand features creates noise in the training process. As a result, only in the AD scenario, the SD policy shows near-optimal results to the deterministic reference policy (DD).

The amount of divergence of the air-travel demand influences the learning of the policies. This is clearly demonstrated in the comparison of the DMD and DND scenario performance. The DMD scenario shows a high divergence of cumulative network demand, which resulted in the lowest prediction scores of all policies. Here, growth of the air-travel demand changes rapidly due to the large variance in long-term mean air-travel growth. This affects the predictability of the optimal fleet decisions, most notably the score and variance of the static policies as they can only generate one fleet plan for all sampled demand values. The DND scenario shows the highest prediction scores for both the static and dynamic policies. Due to the sampling of multiple market demand evolutions, the mean growth always converges to the long-term mean growth. Due to the network optimisation, and the re-assignment of fleet, the optimal fleet decisions converge to a small diverging set of fleet plans.

As pointed out earlier the training time of the neural network is highly dependent on the optimization time of the FPM and FAM. Multiple measures were taken to reduce the training time by increasing the MIP gap, re-using the sampled air-travel demand trajectory, and storing the optimal experience from the FPM. In Table 8, the training time of the NN's for the two stochastic policies are shown. Here it is clearly visible that the FPM and FAM optimisation are responsible for the long runtime and not the back-propagation of the NN for the SS and SD policy.

Nonetheless, after training the NN, the generation of the fleet decisions for a sampled tra-

	Total	FPM	FAM	SS NN	SD NN
AD [min]	1130.32	467.47	627.14	16.04	18.01
DND [min]	1082.12	429.96	616.14	16.21	18.18
DMD [min]	1159.95	459.15	665.2	16.04	17.96

Table 8: Training time of the two DQN's for the SS policy and the SD policy for the three different demand scenarios.

jectory of demand takes less than a second compared to several minutes for the deterministic counterpart (dependent on the MIP gap of the optimizer). As a result, the trained neural network could become an interesting tool for fleet planners and managers in the airline business to assist the decision process and quickly asses the fleet plan for new air-travel demand predictions. However, if the airlines' operated network changes, or the aircraft/route/demand characteristics change, a retraining of the DQN's is necessary.

8. Conclusion

The aim of this research was to contribute to the development of a dynamic fleet policy by the generation of a model-free reinforcement learning program in a fleet planning environment subjected to air-travel demand uncertainty. With this research, it is demonstrated that a RL program can be used to estimate the dynamic policy based on the air-travel demand. The proposed RL program (a) learns the optimal fleet policy and aggregates for demand uncertainties over time; (b) contains a neural network that gives good approximations of the future profit of fleet decisions; (c) has a demand forecasting model that samples realistic air-travel demand trajectories and trends; (d) results in a model that may be utilized to generate the fleet prediction for unseen air-travel demand trajectories and thus act as a reliable tool for the airline business to predict the solution to the arduous long-term fleet planning problem almost instantly.

This work shows for the first time the usage of a model-free learning algorithm with a neural network as a function approximator to learn the optimal strategic long-term airline fleet policy under air-travel demand uncertainty which completely replaces the optimisation process of the fleet problem. Using an end-to-end strategy, the fleet decisions of the agent are evaluated by comparing the profit of the predicted action using a Fleet Assignment Model (FAM) optimisation to the profit of the optimal action using a Fleet Planning Model (FPM) optimisation. At every episode, a trajectory of air-travel demand is sampled for each market using an adaptive Ornstein-Uhlenbeck forecaster based on the historical demand. With a case study, three demand scenarios are created, and two fleet stochastically trained policies (Stochastic Static (SS) and Stochastic Dynamic (DS)) are developed and evaluated against two deterministic fleet planning policies (Deterministic Static (DS) and Deterministic Dynamic (SD)).

The results showed that both of the stochastically trained policies were able to predict viable fleet plans which showed comparable or better results than the deterministic optimisation methods. However, the performance of the stochastic trained model decreased slightly with increasing stochasticity. This was attributed to the increased noise due to the fact that

the state is not sufficient statical of future demand. The SD policy, using the air-travel demand as in the input states, outperformed the SS policy, without the demand in the input state consistently over all the different scenarios tested. This proves that the neural network learned from the inclusion of air-travel demand as input feature.

Although the proposed methodology is not flawless, it employs a large benefit over the deterministic approach. Once the DQN-model is trained, generating new fleet decisions for demand trajectories can be generated almost instantly. As a result, airline fleet planners and managers can use this tool to quickly asses the composition of their fleet and when to acquire or dispose of aircraft given air-travel demand trajectories.

This research opens opportunities for future work. It can be argued that the current state of the air-travel demand is not predictively sufficient for future observation of air-travel demand; therefore the assumption that the MDP is fully observable may be rejected. In future work, it should be investigated on a more detailed level if POMDP representation of the fleet planning problem is a viable solution and could potentially improve fleet predictions. Secondly, this research only considers the air-travel demand as a stochastic parameter. In future work, other or more uncertain parameters (e.g. fuel price, aircraft failure, competition, etc.) should be included to better simulate the stochastic nature of the fleet planning process. Finally, the generation of the reward using the FPM and FAM optimisation process has proved to be the bottleneck for upscaling fleet planning problem to larger networks. Consequently, future work should investigate and develop faster methods to generate a meaningful reward function.

References

P. Belobaba, A. Odoni, C. Barnhart, The global Airline Industry, 2009.

ICAO, Aviation data and analysis seminar: Fleet planning and airline route evaluation, `https://www.icao.int/MID/Documents/2017/Aviation%20Data%20and%20Analysis%20Seminar/PPT4%20-%20Fleet%20Planning.pdf`, 2017. [Online; accessed 29-01-2020].

F. S. Hillier, Introduction to operations research, Tata McGraw-Hill Education, 2012.

D. Kirby, Is Your Fleet the Right Size?, Journal of the Operational Research Society 10 (1959) 252–252.

J. K. Wyatt, Optimal Fleet Size, Journal of the Operational Research Society 12 (1961) 186–187.

G. B. Dantzig, D. R. Fulkerson, Minimizing the number of tankers to meet a fixed schedule, Naval Research Logistics Quarterly 1 (1954) 217–222.

T. E. Bartlett, An algorithm for the minimum number of transport units to maintain a fixed schedule, Naval Research Logistics Quarterly 4 (1957) 139–149.

D. P. Shube, J. W. Stroup, Fleet Planning Model, Winter Simulation Conference Proceedings (1975).

M. Bazargan, J. Hartman, Aircraft replacement strategy: Model and analysis, Journal of Air Transport Management 25 (2012) 26–29.

P. Kall, S. W. Wallace, Stochastic Programming Second Edition, 1994.

N. V. Sahinidis, Optimization under uncertainty: state-of-the-art and opportunities, Computers & Chemical Engineering 28 (2004) 971–983.

T. H. Oum, A. Zhang, Y. Zhang, Optimal demand for operating lease of aircraft, Transportation Research Part B: Methodological 34 (2000) 17–29.

G. F. List, B. Wood, L. K. Nozick, M. A. Turnquist, D. A. Jones, E. A. Kjeldgaard, C. R. Lawton, Robust optimization for fleet planning under uncertainty, Transportation Research Part E: Logistics and Transportation Review 39 (2003) 209–227.

O. Listes, R. Dekker, A scenario aggregation based approach for determining a robust airline fleet composition, Technical Report, 2002.

D. P. Bertsekas, D. P. Bertsekas, D. P. Bertsekas, D. P. Bertsekas, Dynamic programming and optimal control, volume 1, Athena scientific Belmont, MA, 1995.

C.-I. Hsu, H.-C. Li, S.-M. Liu, C.-C. Chao, Aircraft replacement scheduling: a dynamic programming approach, Transportation research part E: logistics and transportation review 47 (2011) 41–60.

H. L. Khoo, L. E. Teoh, An optimal aircraft fleet management decision model under uncertainty, Journal of Advanced Transportation 48 (2014) 798–820.

M. G. Repko, B. F. Santos, Scenario tree airline fleet planning for demand uncertainty, Journal of Air Transport Management 65 (2017) 198–208.

C. A. Sa, B. F. Santos, J.-P. B. Clarke, Portfolio-based airline fleet planning under stochastic demand, Omega (2019) 102101.

W. B. Powell, Approximate dynamic programming : solving the curses of dimensionality, Wiley, 2011.

R. Bellman, The theory of dynamic programming, Bulletin of the American Mathematical Society 60 (1954) 503–515.

W. B. Powell, What you should know about approximate dynamic programming, Naval Research Logistics (NRL) 56 (2009) 239–249.

S. Lam, L. Lee, L. Tang, An approximate dynamic programming approach for the empty container allocation problem, Transportation Research Part C: Emerging Technologies 15 (2007) 265–277.

C. Novoa, R. Storer, An approximate dynamic programming approach for the vehicle routing problem with stochastic demands, European Journal of Operational Research 196 (2009) 509–515.

W. B. Powell, Approximate Dynamic Programming-I: Modeling, Technical Report, 2009.

V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, S. Petersen, C. Beattie, A. Sadik, I. Antonoglou, H. King, D. Kumaran, D. Wierstra, S. Legg, D. Hassabis, Human-level control through deep reinforcement learning, Nature 518 (2015).

S. Dožić, M. Kalić, Three-stage airline fleet planning model, Journal of air transport management 46 (2015) 30–39.

R. S. Sutton, A. G. Barto, Reinforcement learning: An introduction, MIT press, 2018.

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

L. Requeno, B. Santos, Multi-period adaptive airline fleet planning problem, Submitted to: *Transportation Science* (2018).

I. Goodfellow, Y. Bengio, A. Courville, Deep Learning, MIT Press, 2016. `http://www.deeplearningbook.org`.

V. François-Lavet, P. Henderson, R. Islam, M. G. Bellemare, J. Pineau, et al., An introduction to deep reinforcement learning, Foundations and Trends® in Machine Learning 11 (2018) 219–354.

G. E. Uhlenbeck, L. S. Ornstein, On the theory of the brownian motion, Physical review 36 (1930) 823.

O. Vasicek, An equilibrium characterization of the term structure, Journal of financial economics 5 (1977) 177–188.

E. P. Martins, Estimating the rate of phenotypic evolution from comparative data, The American Naturalist 144 (1994) 193–209.

V. Bezuglyy, B. Mehlig, M. Wilkinson, K. Nakamura, E. Arvedson, Generalized ornstein-uhlenbeck processes, Journal of mathematical physics 47 (2006) 073301.

O. E. Barndorff-Nielsen, N. Shephard, Non-gaussian ornstein–uhlenbeck-based models and some of their uses in financial economics, Journal of the Royal Statistical Society: Series B (Statistical Methodology) 63 (2001) 167–241.

N. Chaiyapo, N. Phewchean, An application of ornstein-uhlenbeck process to commodity pricing in thailand, Advances in Difference Equations 2017 (2017) 179.

B. Santos, Lecture Notes: Airline Planning and Optimization, [Accessed on: 2019/10/15], Technical Report, Delft University of Technology, Faculty of Aerospace Engineering, 2017.

D. Silver, Ucl course on rl: Lecture 2 markov decision processes, `http://www0.cs.ucl.ac.uk/staff/d.silver/web/Teaching.html`, 2018. [Online; accessed 18-October-2019].

II

Literature Study

Previously Graded Under AE4020

1 Introduction

1.1. Context

Fleet planning is the first step and is considered the most important long-term strategic decision in the airline planning process. It can be defined as: **The strategic decision process of when, which, and how many vehicles will be acquired to maximise profits in the coming years.**

In Figure 1.1, the airline planning process is depicted schematically. The planning process begins a long-term strategic process at the top of the chart and evolves as time progresses to a short term tactical decision. The fleet acquired in the strategic part of the planning process will have an effect throughout the complete planning cycle. It will decide how many passengers can be transported, which frequency it can operate, the level of luxury it can offer the customer, and which routes it is able to operate. Here, matching the size of the fleet and the demand in the network of an airline is a very complex but crucial step. Because of the minimal profit margins on which airlines operate, grounding aircraft because of little demand can induce enormous cost. Consequently, sudden changes in customer behaviour, fuel prices, or competition can change the optimal aircraft configuration of an airline drastically. Airlines are thus subjected to these future uncertainties which propagate throughout the planning process.

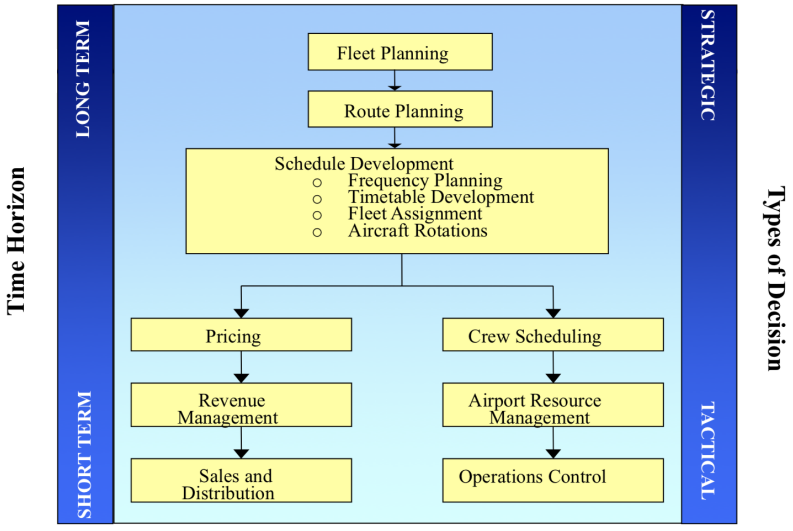


Figure 1.1: The airline planning process (Belobaba et al., 2015).

The airline's fleet can change in roughly two manners. First, older aircraft can be replaced by newer aircraft, which can have a positive influence in term of the efficiency, the flexibility, and the adaptability of the fleet. Secondly, the fleet's size can be either increased or reduced to meet future demand. Either way, the airline will employ an evaluation process to decide which and how many aircraft it will acquire and retire. Generally, two approaches are used to assess the financial and economic viability of a fleet plan (Belobaba et al., 2015): the macro approach, and the micro approach.

The macro approach:
The macro approach or the "top-down" approach, uses modelling at a high-aggregate level with simple spreadsheets. The fleet planners will estimate the expected future demand and available capacity of the airline on a network level. The created lack in capacity is the amount of aircraft capacity needed to be acquired to maintain optimal future operations. The top-level approach is relatively simple and is highly sensitive to the estimations made by the forecasters. However, due to the high level of uncertainty accompanied by it and practical considerations such as available landing slots, geopolitics, and available aircraft deliveries, the top-down approach is more commonly used in the airline industry (Belobaba et al., 2015).

The micro approach:
The micro approach or the "bottom-up" approach, uses a detailed modelling technique that tries to match the expected demand and capacity on a flight level in order to generate a specific aircraft fleet plan. This model considers predictions about demand in OD markets, aircraft constraints, expected changes in fuel cost, etc. By adding these complexities a more detailed planning model is generated and the fleet can be tailored more precisely to the future airline expected demand. Moreover, by modelling the fleet plan for multiple scenarios a more robust and complete representation of the future OD-market can be made. However, adding these complexities and uncertainties to the model is very difficult and often escalates the size and computational time

Because of the complexity accompanied by incorporating uncertainties in fleet planning models and the speculative nature of uncertainties, the top-down approach continues to be the conventional strategy used by the airline industry (Belobaba et al., 2015). However, with the advent of ever-increasing computational capacity and research, the employment of more complex models have become viable and should be considered by every airline. This literature study is mainly focused on the bottom-up approach to capture these uncertain scenarios in the long-term strategic planning process.

The literature review of this thesis aims to synthesize the state of the art findings of the past research performed on the fleet planning problem. More specifically, it focuses on the bottom-up approach to capture these uncertain scenarios in the long-term strategic planning process. At the end of the literature study, it should be visible which areas have already been investigated, where the knowledge gap is, and what state of the art techniques can be used to contribute with scientific research.

The work described in this chapter is a summary of the literature study, which was previously submitted and graded under AE4020

1.2. Review Structure

This part is subdivided into two subsections visualized in Figure 1.2. Chapter 2 will discuss the different modelling methods and the associated solution techniques shown on the right side. This section elaborates on the various method of translating a real-life operational process into a mathematical model with possible approaches in solving the model. Chapter 3 will discuss the different model perspectives encapturing the uncertainties. These perspectives are shown on the left side of the tree in Figure 1.2. To conclude, the knowledge mind map deducted from the literature research will be presented in Chapter 4.

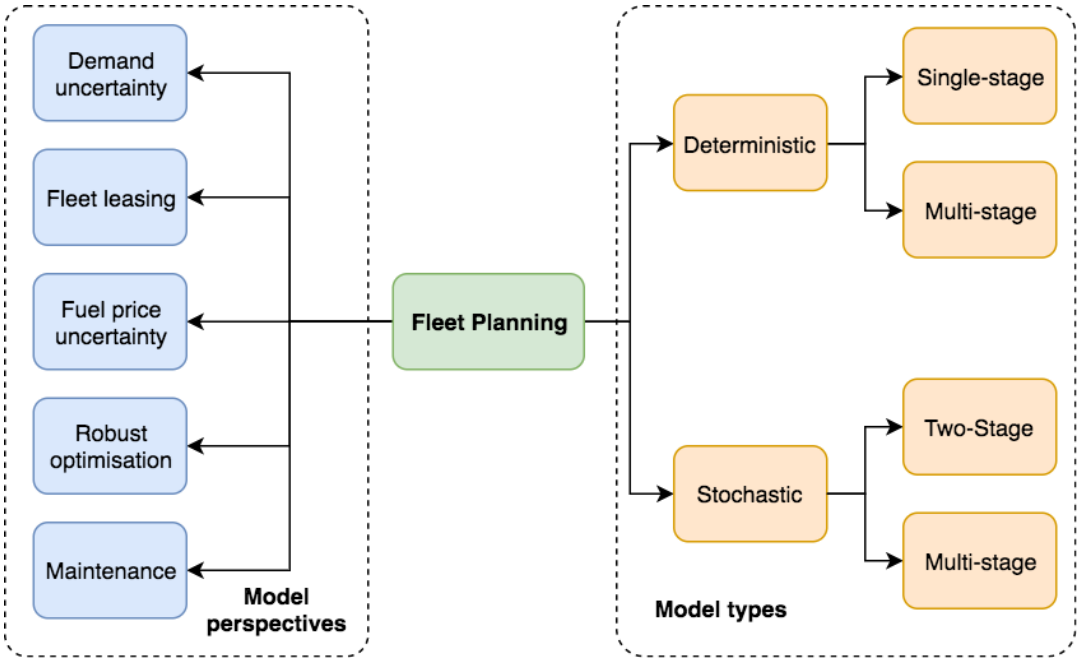


Figure 1.2: Problem formulation and perspectives map.

2

Problem Formulation & Solution Techniques

2.1. Deterministic models:

With the increasing popularity of the Operational Research field in the 1950s, researchers started to investigate how fleet and demand could be matched as efficiently as possible to increase operational profits. The first models considered a single period, homogeneous fleet and were solved analytically, hence the models were very small. Dantzig and Fulkerson (1954) were the first to model the minimum amount of vehicles needed to operate a fixed schedule in a point to point network in a nautical tanker problem. However, with a single-stage model, the problem is modelled as if it is not dependent on time. The solution of the decision variables is considered to be valid now and in the future. With the advent of more computational power and more sophisticated solution algorithms, adding multiple stages became a feasible option.

With the development of more advanced solution methods for OR problems and more computational power, the possibilities and complexity in modelling fleet problems increased accordingly. Whereas previously, the model tried to find an optimal fleet for one single stage and could be seen as tactical decision making, the multi-stage model comprises the possibility to generate long-term strategic planning for fleet planning. These models employ a more bottom-up approach where they will try to match an expected demand at flight level. By modelling the most efficient fleet over this network, a fleet size can be generated.

Shube and Stroup (1975) were the first to create a multi-stage fleet plan. In contrast to the single-stage model, which focused on matching incoming and outbound traffic at node level, the multi-stage model, as described by Shube and Stroup (1975), tries to model the non-homogeneous fleet planning of a period of 10 years to the expected demand. By creating ten periods, for each year one, a detailed plan is modelled which specifies when aircraft should be acquired and disposed. Shube and Stroup (1975) proposed a linear program with an objective function minimizing the direct operating cost (DOC), reducing the amount of money spent on new equipment, and decreasing the DOC and the debt created by acquiring new aircraft.

Solution Techniques:
The simplex method is one of the oldest methods to solve small scale linear problems (Nash, 2000). However, the need for algorithms solving MILP problems brought scientific innovations such as Benders decomposition and Dantzig-Wolfe decomposition (Benders, 1962; Dantzig and Wolfe, 1960).

Together with newer algorithms such as Branch and Bound (Dakin, 1965) and Branch and Cut (Gomory, 1963), these algorithms comprise a research branch called 'divide and conquer' as they divide the problem into multiple sub-problems to solve the main problem. However, deterministic algorithms optimize a problem by restricting the solution space until the global optimum is found. When dimensions increase, these algorithms cannot find a solution in a reasonable time. A solution to this is heuristics, where algorithms such as simulated annealing are designed to such a manner they search the solution space to find the global optimum. However, convergence to the optimum is not guaranteed. Moreover, deterministic models, including heuristics, do not take into account the uncertainty perspective.

Conclusion:
What deterministic models hold in common is that they assume a perfectly predictable world. As a result, a deterministic model cannot model for uncertain parameters which are often depicted as probability distributions. A way of solving this is inserting the mean values of historical data or other estimates of the unknown parameters in the deterministic models (Kall and Wallace, 1994). The result is a single solution to a highly uncertain problem. However, by ignoring the stochastic nature of the uncertainty variables, the solution of the deterministic models can be very misleading and incomplete.

2.2. Stochastic Models:

As discussed at the end of the previous subsection, stochastic models differ from deterministic models as they do not assume that the world is fully predictable. Instead, in stochastic modelling, a set of parameters will be assumed to be unknown and will take form as probability under a distribution. As a result, the general formulation is written in Equation as synthesized by Kall and Wallace (1994).

$$\begin{aligned} \min_{x} \quad & g_0(x,\xi) \\ \text{s.t.} \quad & g_i(x_i,\xi) \leqslant 0,\ i = 1,...,m \\ & x \in X \subset R^n. \end{aligned} \tag{2.1}$$

Here, ξ is a random vector which varies over a set $\Xi \subset R^n$. In other words, it is assumed that a family of "events" and the corresponding probability distribution of P is given. Similar to the deterministic modelling approach, the stochastic modelling approach can be subdivided into two parts. The first method is for a single-stage and referred to as *Two Stage Stochastic Modelling*. The second method considers multiple stages therefor called *Multi-stage stochastic programming*.

2.2.1. Two-Stage Stochastic Modelling:

Introducing uncertainty in the models means the solving algorithms have to adapt to handle the uncertainty. In general three approaches to model two-stage stochastic models can be identified: *Robust stochastic optimization, Probabilistic programming, Stochastic programming with recourse.*

Robust stochastic optimisation is a common approach to minimize the risk by including a variability measure f such as the variance of the cost of the second stage as a recourse action. If the value is high, the optimisation will try to reduce the 'variance-cost' on behalf of the second-stage variables expected cost and vice versa.

The probabilistic or chance-constraint approach is a second method often used to create robustness in the system. Indeed, the infeasibilities generated in the second-stage problem by choosing the first-stage decision variables are penalized. Usually, by doing this the confidence level above

a certain level is ensured. This method is usually employed for financial problems and only used in fleet planning to increase the robustness of the problem.

In stochastic programming with recourse, variables of the model are separated into two sets: the first stage and the second stage variables. The first stage variables have to be chosen before the uncertainty is revealed. The second stage variables are now chosen which are submitted to the random events. These variables are usually seen as a corrective measure, or recourse action, against the infeasibility arising due to realisations of the uncertainty (Sahinidis, 2004). Because of this uncertainty, the first stage variables are chosen to minimize their cost and the expected cost of the second stage variables. This method is mainly used in the fleet planning problem, where the fleet's decision variables are first-stage variables and the demand the uncertainty which is revealed after the first stage variables are set.

Solution Techniques:
Solution techniques in two-stage stochastic modelling are often modified from the deterministic solution techniques. is a variation on Bender's decomposition devised by Van Slyke and Wets (1969). Similar to Bender's decomposition the problem is divided into a 'master problem' and 'sub-problems'. However, the 'master problem' is now the first stage problem and the 'sub-problem' the second-stage problem. The master problem is solved to obtain first stage solutions which are used to solve the second stage problem which generates feasibility and optimality cuts for the master-problem. The L-Shaped algorithm tries to approximate the non-linear recourse term in the objective function (Xu, 2008).

Later Ermoliev (1969) devised a different approach on solving the stochastic minimization/ maximization problem: the stochastic quasi-gradient (SQG) method. The idea behind SQC is that conventional decomposition methods require the computer to calculate at every iteration the feasibility of each decision variable to every constraint. This is computational very expensive but give the exact solution if the optimal solution is found.

With the advent of sampling methods from SQG and the L-shape method, Higle and Sen (1996) proposed a method where the latter two methods are combined in one approach: Stochastic decomposition (SD). The sampling method used here is called 'Interior' Monte Carlo sampling because the samples are generated inside the algorithm. Moreover, every iteration of the SD algorithm requires the solution of the sub-problem to approximate the recourse function by adding cuts to the master problem. In SD the calculation of the subproblem will use randomly generated samples drawn from the probability distributed variable w.

Conclusion:
Two-stage models bring a promising solution to deal with the uncertain parameters in a model. However, a two-stage model has -as the name suggests- only two stages and are therefor unsuited for long-term planning models. They all imply that at the beginning of the planning horizon the uncertainty is revealed once (Xie and Huang, 2018). When modelling over a longer period, the sequential decision-observation process over multiple periods becomes of high importance. A multi-stage stochastic programming method needs to be employed.

2.2.2. Multi-Stage Stochastic Programming:

Multi-stage models are basically an extension of the two-stage stochastic models, wherein the two-stage the maximum value of the next stage is considered, multi-stage programming model will try to model the maximum expected value over all future stages. In practice, a decision problem -such

as fleet planning- is represented as a sequence of decision and observations evolving over time. In equation 2.2 a finite horizon framework is considered, meaning the end state of the problem is defined, and the problem is optimized to achieve the highest return in the end state. Extending the general formulation of a two-stage formulation described in the previous subsection results in a nested formulation (Shapiro et al., 2009). Here χ_T is a set of feasible solutions for x_t, $w_1, w_2, ..., w_T$ is a random data, and f are continuous functions.

$$\min_{x_1 \in \chi_1} f_1(x_1) + \mathbb{E}\left[\inf_{x_2 \in \chi_2(x_1, w_2)} f_2(x_2, w_2) + \mathbb{E}\left[... + \mathbb{E}\left[\inf_{x_T \in \chi_T(x_{T-1}, w_T)} f_T(x_t, w_T)\right]\right]\right] \tag{2.2}$$

It can be readily seen that a model spanning over multiple stages and periods becomes computationally expensive very fast. However, by subdividing the large multi-stage problem into smaller subproblems, solving the equation become feasible. This method is called Dynamic Programming.

Solution techniques:
Decomposition techniques such as Nested Decomposition, Lagrangean Decomposition, and Quadratic Nested Decomposition are proposed to solve the multistage programs. Nevertheless, decomposition approaches in a large-scale scenario tree are not able to reach an optimal solution in a feasible time frame. Moreover, solving multiple-stage stochastic programs has become infeasible for real-life problems, so research resorts to heuristics and approximations. Consequently, little literature is available on multi-stage stochastic programming from the operations point of view. Nevertheless, the control theory perspective of Bellman poses a solution in solving these large-scale models: Dynamic Programming

2.2.3. Dynamic Programming:

Dynamic programming (DP) was introduced by Bellman (1954) to model a multi-stage sequential decision process. At every s_t a decision x_t is taken. This process is called a Markovian decision process as every decision at each state is not dependent on the previous stages. After the decision is taken the value of the uncertain parameter w is revealed and the state transitions to a new state. By maximizing the contribution over all stages T, the optimal cost can be calculated. The cost (or value) of being in state s_t can be calculated using the *Bellman equation*:

$$V_t(S_t) = \max_{x_t \in \mathbb{X}_t} \left(C_t(S_t, x_t) + \gamma \mathbb{E}\{V_{t+1}(S_{t+1}(S_t, x_t, W_{t+1}))|s_t\}\right) \tag{2.3}$$

The first term of Equation 2.3 shows the contribution being in state x_t and taking action x_t. The second term shows the expected value when transitioning to state s_{t+1} with the revealed demand w_{t+1}. Figure 2.1 shows a generic model tree with action nodes, decision nodes, outcome lines, and random outcomes. The decision tree discretizes the probability function described above. In this example, each state has two possible outcomes to the uncertainty and two resulting states.

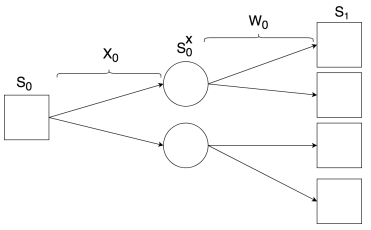


Figure 2.1: Generic decision tree, with decision nodes (squares), outcome nodes (circles), decisions and random outcomes (lines) (Powell, 2011).

The decision tree and the possible solution to the problem grows exponentially as the amount of states increases. Algorithms solving the model are computationally very expensive. Powell (2009)

poses that the procedure suffers from the *curse of dimensionality*. As can be seen from Figure 2.1 the number of states S_t, the number of actions a_t, and the number of random outcomes W_t grows exponentially, and are called the three curses of dynamic programming as the problem is being discretised. To solve the decision tree every state, over all periods, must be visited. Due to the exponential nature of the problem's state-, action-, and outcome space, the DP process becomes computational infeasible to solve in a reasonable time, even for relatively small problems. Because of these large state-action spaces, value functions and policies cannot be represented precisely and approximation becomes necessary.

Approximate Dynamic Programming:
In contrast to backward dynamic programming, Approximate Dynamic Programming steps forward through time instead of calculating the states recursively. However, this poses two problems when solving. First, when stepping forward the calculation of the value of future states is unknown. Hence, the exact value at the current state S_t is impossible to determine. Because of this, it is impossible to identify the optimal action. To solve this, an approximation of the value function is created thought iteration. The optimal action can be calculated by maximizing the approximated value function. Secondly, given the decision, the model can transition from state S_t to state S_{t+1}; however, uncertain parameters are often involved influencing the contribution function in S_t and the transition to S_{t+1}. A common strategy is to pick a realization (or sample) randomly from a distribution; this is referred to as a *Monte Carlo simulation*. The collection of these samples throughout the decision tree is called a sample path of the n-th iteration and denoted by w^n.

Using the notation of Powell (2011), a basic ADP program is summarized as follow. Initially, the value function approximation V_t is approximated for all t. A random sample path is generated for iteration $n = 1$. At every time-step $t = 1, ..., T$ the value of the action that maximizes the value function is calculated using the Bellman equation:

$$\hat{v}_t^n = \max_{a_t} \left(C_t(S_t^n, a_t) + \gamma \sum_{s'} \mathbb{P}(s'|S_t^n, a_t) \bar{V}_{t+1}^{n-1}(s') \right) \tag{2.4}$$

Here, $\bar{V}_t^{n-1}$ represent the approximation of the value function when transitioning to state S_{t+1}, multiplying this with the probability of transitioning to that state, summing over all possible states s' to which it can transition times a discount factor γ, entails the expected value of the future states. Together with the contribution of the current state, the value function of the visited state is updated with the calculated value approximation:

$$\hat{V}_n^t = \begin{cases} \hat{v}_t^n & S_t = S_t^n \\ \hat{V}_t^{n-1}(S_t), & \text{otherwise.} \end{cases} \tag{2.5}$$

Now, state transitions to the next state given the action a_t, the current state, and the sample realization of the sample path in $t+1$:

$$S_{t+1}^n = S^M(S_t^n, a_n^t, W_{t+1}(w^n)) \tag{2.6}$$

This is repeated for all time steps T and iterated N times until the end of the horizon. By doing so every iteration the approximated values of the time steps are updated. In this process, the ADP 'learns' to approximate the optimal strategy without visiting each state.

There are roughly two main approaches in ADP on deciding which action to take given the state: **Policy iteration** and **Value iteration**. Both use a policy π to select next actions, and both are called model-based approaches as the transition function is fully known during the learning process as

opposed to model-free learning methods. However, in value iteration, the model tries to calculate the optimal state-action function by updating the values function in every iteration. From the value function the optimal policy can be derived by selecting the action with the maximum value. Policy iteration works slightly differently as it tries the policy directly instead of first approximating the value function and then selecting the action. The values generated from the policy is used to update the policy function.

In the approximate value iteration, there are three common approaches in approximation value function: lookup-tables, parametric functions, and non-parametric functions (Powell, 2011). Lookup-tables require a lot of memory storage and may become too big when a big state-space problem is solved. Consequently, this literature study will focus on the functional approximators: **parametric** and **non-parametric**. As the name suggests, the parametric approach uses a global, parametric function to approximate the value function, whereas the non-parametric approach has a much weaker assumptions about the value function's form Urieli and Stone (2013).

Reinforcement Learning
An almost similar practice under a different name is Reinforcement Learning (RL) to ADP. However, where ADP described above was for a 'model-based' approach, RL is the general name for 'model-free' approaches in computer science. The difference between the two lie in the transition function. According to Powell (2009), model-free programming arises when we are not able to calculate:

$$a_t^n = \max_a (C(S_t^n, a) + \gamma \mathbb{E} V_{t+1}(S^M(S_t^n, a_t, W_{t+1}(w^n))))$$

In the equation above three main calculations need to be performed to solve the equation. First, the next state in the model: $S^M(S_t^n, a_t, W_{t+1}(w^n))$ is calculated. The transition function is often unknown or hard to model. If the transition function happens to be known, the uncertainty is often discretized in a few scenarios to limit the decision tree from growing too large and thus not fully capturing the stochastic nature of the uncertainty. Secondly, the computation of the expectation occurs. This becomes a problem when the underlying probability distribution of the exogenous problem is unknown and makes the calculation of the one-step transition function not possible. And thirdly, the cost function $C(S_t^n, a)$, or reward function, which can be unknown is attempted. A solution to solve these problems is to let the algorithm learn by interacting with the environment. With trial and error, different actions will give rewards or penalties to learn the best policy.

The combination of RL techniques and the novel discovered power of neural networks as function approximators has led to a popular approach in control theory and computer science after being considered a very slow training process. Researchers have shown with the success of Deepmind's AlphaZero and OpenAI'Dota bot the power and possibilities of model-free learning. At this moment the application of model-free reinforcement learning models has been applied mainly in human-level control environments, vision and speech recognition, and robotics (Mnih et al., 2015; van Hasselt et al., 2015; Schaul et al., 2016)). However, more research fields start to explore the applicability and power of neural networks and the reinforcement learning approach in a data-driven society.

Recent work on ADP
Although the Bellman equation has been invented in the 1950s the recent years have shown a spike in interest in DP and ADP in operational, financial, and control environments. In this chapter, the benefits of ADP have shown a clear potential to adapt in planning processes. Multiple authors have already applied the technique to optimize operational processes.

ADP has already been used to solve resource allocation and vehicle routing problems. Lam et al. (2007) applied ADP to model obtaining operational strategies for the relocation of empty containers in the sea-cargo industry. The authors show that the ADP with linear approximations under a Temporal Difference learning (Tesauro, 1995) framework for a two-port only configuration shows comparable results to the exact solution. Novoa and Storer (2009) examine ADP approaches in modelling single-vehicle routing problem with stochastic demand, and Powell (2009) uses ADP to build an optimizer simulator for locomotive planning. Where classic methods would involve large integer programming models, the ADP is able to find near-optimal solutions for the fleet size and planning of locomotives.

Recent work by Requeno and Santos (2018) employs the strategic fleet planning model in the airline industry using ADP. They observed that the calculated operational profit and the number of aircraft owned related to a non-linear concave function. As a result, a non-parametric Gaussian kernel regression is used to approximate this function more accurately. Moreover, Requeno and Santos (2018) are the first and only to model an airline fleet model using approximate dynamic programming approach to capture the effect of demand uncertainty.

Conclusion:
Dynamic programming, especially approximate dynamic programming, brings a novel solution from the control theory research area to find feasible solutions in operational problems. When considering a long-term time horizon the dimensions of a problem and the scenario tree increases exponentially in size and becomes insolvable with current hard- and software. Calculating the decision tree forward instead of backward brings new possibilities to large-scale programming models. Approximating the value function is in current literature the biggest research topic and hurdle to pass. Newly popularized methods, like reinforcement learning, and the use of a neural network can bring novel solutions ever so difficult designing of the parameters of polynomial functions. To the writer's knowledge, such an architecture has never been tested in a fleet planning problem.

3

Modeling Perspectives:

The modeling perspectives chapter elaborates on the various uncertainty parameters or perspectives which can be added to the fleet planning optimisation as an attempt to calculate a better representation of the real processes. Implementing additional decision variables or including stochastic variables to a model usually increases the computational time of the computational time and complexity of the model in exchange for a more detailed solution. In the following sections five perspectives in a fleet planning environment will be discussed: Demand Uncertainty, Fleet Leasing, Robust Optimisation, Fuel Uncertainty, Maintenance.

3.1. Demand Uncertainty:

Generally, research on fleet planning includes one or more of these perspectives with an appropriate model type. The model type can be seen as a 'means' to capture and investigate the various perspectives and considerations into a mathematical model. In fleet planning, demand uncertainty is unsurprisingly the most researched perspective in literature, as it is the uncertainty which has the highest impact on the strategic decisions.

Deterministic modelling of fleet planning optimisation problems originally relied on a deterministic demand prediction for which the optimal fleet is calculated. The result is a single solution completely dependent on the single predicted evolution of demand. Prime examples are Shube and Stroup (1975), New (1975), Bazargan and Hartman (2012).
Later, to create a set of stochastic solutions instead of one deterministic one, Sa et al. (2019) and Repko and Santos (2017) employed a scenario tree which formed a set of realization of demand under a stochastic probability. Whereas Repko and Santos (2017) used a predetermined set of probabilities and scenario realizations, Sa et al. (2019) employed an autoregressive process called the mean-reverting Ornstein-Uhlenbeck process to sample demand evolutions. Combined with discrete-time Markov chain transitions a set of demand scenarios were created. From this, a portfolio of realizations is solved using a MILP solver. Although this process captures the demand realizations in a more detailed manner and allows for a robust fleet selection, the calculation of the exact MILP optimisation is very computationally expensive, and the fleet composition is not adaptable over time with changes in air travel demand.

Furthermore, various stochastic programming approaches are employed to include the demand perspective in the fleet planning optimisation. A prime example of two-stage stochastic programming in a fleet planning perspective is Listes and Dekker (2002), where a model is introduced to tackle the operational flexibility under uncertain demand. They argue that random fluctuation in

demand create low average load factors and spilt passengers across a network. A solution to the random demand fluctuation is a fleet composition which appropriately supports dynamic allocation of capacity. With this, a strategic fleet planning model can be created depending on the flight schedules and accompanied stochastic demand. In other words, the authors propose a bottom-up approach where the strategic decision of the fleet composition is formulated in the fleet operation where the stochastic demand influences the capacity. Through implementing this flexibility across the network can be established. By employing the two-stage stochastic model, the second-stage variable's expected value is maximised over all possible scenario's. Using this method the operational flexibility under these scenario's is optimised. The authors are innovative by approaching this strategic problem from a more tactical level; but a significant drawback is the lack of a multi-periodicity. As the authors focus on fleet flexibility across the network at one period, they fail to acknowledge demand uncertainty over future years and the according fleet strategy.

In an attempt to model demand uncertainty in a multi-period, research turns to the multi-stage stochastic modelling process called Dynamic Programming (DP). The work by Hsu et al. (2011) uses a DP model to obtain the optimal aircraft replacement strategy in combination with aircraft leasing strategy. The model considers the demand to be uncertain and fluctuating over time. As a modelling solution, they employ Grey topological method with Markov-chain models to forecast the passenger demand and create randomness in demand. A few years later, Khoo and Teoh (2014) argue that the stochastic demand modelled is too limited. According to the authors, demand is more subjected to unexpected events and, not considering the possibility of disruptive events, the models do not resemble reality. They suggest a model incorporating the stochastic demand index (SDI). This is a step-by-step procedure to model the stochastic demand using Monte Carlo simulations. The model produces viable solutions for strategic long-term fleet decisions, although it needs improvement considering the integration of the service frequency and computational programming. The latter remains the bottleneck for DP models. As the scenario tree increases so does the computational time exponentially.

An attempt in reducing computational times for fleet planning models is performed by Requeno and Santos (2018). In her work, a model-based Approximate Dynamic Program is used to calculate the decision tree forward instead of backwards. As a result, value-approximations of the state-action spaces are learned to approximate the optimal solution in a reasonable computational time. In the transition function, air travel demand changes across the operational network and is chosen as the uncertain parameter; similar to Repko and Santos (2017), a symmetric demand scenario tree is created using predetermined demand changes and probability distributions. According to the author, the results have shown near-optimal solutions in reasonable computational times. However, it should be noted that the optimisation network used by Requeno and Santos (2018), as well as the demand tree are very small.

3.2. Fleet Leasing:

A convenient way to create flexibility in a fleet under variational demand is renting or leasing capacity when the owned capacity is not sufficient and disregarding the leased vehicles when the demand lowers. Prior research has proved a lot of benefits to the flexibility of the fleet when considering leasing structures. Next to fleet leasing, aircraft replacement strategies are also discussed in this section, as they closely relate to each other and are often modelled in conjunction to optimise operational profits.

Lease of aircraft

According to Gritta et al. (1994), the amount of leased aircraft by airlines in the US grew from 19% in 1969 to 54% in 1991. Leasing has major financial benefits to airlines in terms of debt/equity ratio on the balance sheet (Gritta et al., 1994). Besides, leasing aircraft creates more flexibility of capacity and the accompanied higher cost. Moreover, leasing aircraft separates the airlines, from the high financial risk accompanied by high capital investments, which is highly beneficial for airlines who only operate the aircraft. This sparked new research in terms of the optimal fleet mix of leased and owned aircraft.

Research by Oum et al. (2000) considered that in times of low demand, the leasing of aircraft will become cheap, and in times of high demand the leasing will be expensive, this enables the airlines and leasing companies to share the risk of uncertain demand. To prove the benefits of leasing lessors and lessees, Oum develops a two-stage stochastic model deriving the optimal demand for operating lease of aircraft. The results show that the optimal mix ranges between 40% and 60% of the total aircraft fleet. Oum et al. (2000) prove here that the mix of leased and owned aircraft mitigate the cost of an airline under uncertain demand. However, the corresponding fleet composition of the owned and leased aircraft is never discussed.

Aircraft replacement strategy

As discussed in Section 3.1, Hsu et al. (2011) introduced an aircraft replacement strategy including demand uncertainty, but also factors such as fleet commonality, purchasing price, maintenance, and economy of scale. In light of this work, Bazargan and Hartman (2012) created a strategic model for fleet acquisitions and disposals by minimizing the discounted cost of leasing or owning aircraft.

Bazargan and Hartman (2012) split the whole deterministic optimization problem up with variables for owned aircraft and variables for leased aircraft. Moreover, an index for the age of the aircraft is introduced as well as a binary decision variable to indicate if the airline has an aircraft of type k, in period j. As a consequence, the objective function becomes a massive summation over the fleet types K, all the possible ages with maximum age N, and the amount of periods T. Without going into depth in the actual formulas, it is fairly obvious that the amount of decision variable and the complexity of the model is very high. Interestingly, Bargazan shows in the analysis that results of computations over 11 years with just over 100 aircraft for 600 daily flights with a Cplex Solver. Unfortunately, computational times and the used processing power is not discussed.

Conclusion

The fleet leasing perspective can be viewed from multiple angles within the problem. Where in the early stages single-stage homogeneous fleet models are described, recent research has shown novel methods using multiple-stage heterogeneous fleet models mixing leased and owned aircraft. Hsu et al. (2011) and Bazargan and Hartman (2012) both address the replacement and leasing perspective in their models. However, both employ a different method; Bazargan and Hartman (2012) uses a deterministic model, and Hsu et al. (2011) a dynamic programming model. One way or another fleet leasing drastically increase the decision space of the programming models as the variables have to be duplicated. Therefore, state of the art programming models must be employed the keep solving time in a reasonable bound

3.3. Robust Optimisation

Robust optimisation can be viewed as incorporating uncertainty in optimisation models to hedge the solution from worst-case scenarios (Ben-Tal et al., 2009). A common approach is to minimize

the variance in a problem and creating a mean-variance. However, List et al. (2003) considers the mean-variance trade-off not always the best solution to create robustness into the model because variance gives equal weight to deviations above and below the mean value. This is illustrated by a simple example where the incorporation a term to minimize the cost variance across scenarios induced very strange and operational in-feasible solutions.List et al. (2003) therefor turned to the "downside risk" in financial investment portfolio's as presented in the work of Bawa (1975) and Fishburn (1977). They propose to, instead of minimizing the variance, to reduce the "one-sided" risk. These risk measures are borrowed from the financial industry where they are used to manage the risk in investment portfolios. In this section, two promising and widely used down-side risk measures are discussed: the lower-partial moment (LPM) and the conditional value-at-risk (CVaR)

Lower Partial Moment as Risk Reduction

List et al. (2003) proposes a robust optimization solution procedure to measure the impacts of uncertainty on fleet sizing and making a trade-off between costs and the accompanying risk. With the latter, an airline can determine its optimal choices for a predetermined risk acceptance level, which will probably not yield the optimal fleet, but the best choice under the uncertain demand values. In one-sided risk optimization, the goal is to control the likelihood that an objective function will grow to undesired values. In other words, by controlling the influence of the uncertainty on the second-stage decision variables, the risk of exceeding a preset threshold is minimized. Finally, the authors notice the problem formulated is large and will require a lot of computational effort to compute. That is why in the example the fleet is assumed to be homogeneous, only three markets are assumed, and the fleet is constant over a four-period planning horizon.

Conditional Value at Risk

A second method to decreasing the risk due to the uncertainty is employed by Naumann and Suhl (2013). In this work, he tries to model the fuel uncertainty in the two-stage stochastic model where a trade-off is made between fuel hedged and fuel bought at the stochastic price. A fuel overview of the model and discussing is shown in Section 3.4. In the constraints, a 'one-sided' risk measure similar to lower partial moment is employed: the Conditional Value at Risk (CVaR). The CVaR is the average value, of a percentage of worst-case scenarios usually defines by a percentage of the tail risk.

Conclusion:
It is clearly visible that the incorporation of risk-reducing elements in the stochastic models can mitigate the risk accompanied by the uncertainty in demand or fuel. It is interesting to note that the authors of both papers trying to mitigate the risk imported measure already used in the financial sector. The latter institutions have years of experience in generating investment portfolios with minimized risk. Secondly, both authors recommend extending the models to more sophisticated models with more variables, and thus higher dimensions, but at the same time warn about the accompanying need for better solution methods and more computational power.

3.4. Fuel Price Uncertainty

To the writer's knowledge, Naumann and Suhl (2013) is the only author who tried to model the effect of fuel price uncertainty in the strategic planning perspective, more specifically in the schedule design. The flight schedule decides if, and with which frequency, flights are performed in the network with a certain aircraft type. As a novelty, Naumann and Suhl (2013) consider the fuel hedging as a solution to minimize the financial risk. Complementary to the uncertainty in fuel price, the authors consider uncertainty in demand as stochastic too. Two scenario sets are created for both the fuel price and demand uncertainty, and are combined into a single set considering a demand for every

fuel price scenario. This occurs as a two-stage stochastic model is employed to deal with the uncertain parameters. In the first stage, the flight frequency and the amount of fuel hedged is decided, after the fuel price and demand is revealed, the passenger's flow is evaluated in the second stage.

Conclusion
Naumann and Suhl (2013) considers their work the strategic schedule design under the demand and fuel uncertainty. Arguably, the planning can be called 'strategic' as one period, a single year, is considered and no long term planning governing multiple years and fluctuating demands and fuel prices are not considered. Nevertheless, the conversion to a multistage model could employ multiple periods and model demand and fuel variation to strategic flight planning and according to fleet replacement. However, the number of decision variables will therefor increase drastically, along with the computational times.

3.5. Maintenance

All airlines are subjected to strictly regulated maintenance schemes, governing various types of maintenance checks, all with different time intervals and prices. Generally, when one considers the airline planning perspective, they readily place maintenance as a tactical decision of high influence in the network and scheduling design. These will be of high influence on the utilization rate of the aircraft and consequently on the efficiency of the airline's fleet (Clarck, 2007).

Bazargan and Hartman (2012) created a long-term fleet acquisition and disposal model to help airlines in their aircraft replacement strategy. This model is also discussed in section 3.2 governing leasing/owning strategies of aircraft. The model created by Bargazan and Hartman is innovative as it models the contemporary fleet replacement problem with a deterministic model. The objective function tries to minimize the total discounted cost while fulfilling the predetermined demand for wide- and narrow-body aircraft. One term considers the maintenance cost over the planning horizon by multiplying the amount of owned aircraft of type k and age i to the cost of operating and maintaining an aircraft of type k and age i, multiplied again, by the discount factor. The interesting value here is the cost of operating and maintaining an owned aircraft of type k.

Where Bazargan and Hartman (2012) modelled the deterministic environment, Hsu et al. (2011) consider the maintenance cost in a dynamic programming environment. Here, the cost of maintenance is divided into fixed maintenance cost, such as overhead; equipment; location cost; and variable maintenance cost, which vary with the status and number of aircraft. As discussed earlier in this chapter, Hsu et al. (2011) incorporate the leasing vs owning aircraft perspective in the dynamic model. By doing so, an insight is given into the interrelations between maintenance and leasing and the aircraft's age.

Conclusion:
Maintenance cost and the replacement threshold of aircraft have been covered in literature in a deterministic and stochastic manner. Both Bazargan and Hartman (2012) and Hsu et al. (2011) incorporate multiple modelling perspectives into one model, attempting to make them resemble the true process in airline operations. These perspectives such as maintenance, aircraft replacement strategies, and fuel costs all influence on another. To model the operational perspectives as correctly as possible, multiple perspectives should be incorporated; consequently, increasing the size of the models. Bazargan and Hartman (2012) and Hsu et al. (2011) is elaborate, but there remains room for improvement. More research regarding the influence of efficiency of newer aircraft on the threshold of maintenance cost needs to be performed to verify the finding by Hsu et al. (2011).

4

Knowledge Gap

In the previous chapters, the fleet planning problem is discussed from multiple perspectives, multiple modelling aspects, and various approaches. On a more advanced level, this literature review shows that researchers and companies have trouble dealing with uncertainty, as well as the curse of dimensionality. To structure all the covered literature subjects visually, a knowledge gap mind map is created in Figure 4.1. In the mind map, three distinct areas of research are visible. First displayed are the perspectives elaborately depicted in Chapter 3 with the possible underlying models. Secondly, the solving algorithms employed to solve the method in Chapter 2 are modelled. Thirdly, reinforcement learning or approximate dynamic programming is introduced, as discussed at the end of chapter 2. To ensure that the mind map is even more visually informative, a colour code is employed. This colour code weights the amount of research conducted on these methods from a fleet planning perspective. The green colour shows elements which are covered in literature frequently; the yellow colour shows elements which are partially covered; the orange areas show elements which are not or limitedly covered in a fleet planning environment. The yellow and orange areas show an opportunity to conduct research.

Throughout the literature review and the mind map, a clear division is made between the modelling perspectives and the optimisation methods. In practice, the modelling method and the optimisation method are related to each other and are difficult to separate; consequently, they are discussed simultaneously below.

Deterministic approach:
Short-term deterministic models were the first to address the fleet planning problem. These models were often employed in either a lease vs buying perspective (Kirby, 1959; Wyatt, 1961) or in a fixed time schedule problem (Dantzig and Fulkerson, 1954; Bartlett, 1957). The first long-term deterministic model solving the fleet planning model is Shube and Stroup (1975). Here the author modelled a full multi-stage fleet plan matching inbound and outbound traffic at the node level. However, these deterministic models do not have the possibility to employ uncertainty into the model. Moreover, deterministic models for multi-period planning suffer from long or even unfeasible computational times (Schick and Stroup, 1981; Sayarshad and Ghoseiri, 2009).

Stochastic approach:
Stochastic models, more specifically two-stage models, brought the novelty to modelling uncertainty by splitting up the decision variable and unveiling the uncertainty after choosing the first-stage variables (Shapiro and Philpott, 2007). Using this method, various perspectives were modelled and tested in the fleet planning model: Demand uncertainty (Listes and Dekker, 2002), fleet leas-

ing (Oum et al., 2000), risk optimisation (List et al., 2003; Naumann and Suhl, 2013), and fuel price uncertainty (Naumann and Suhl, 2013). However, the two-stage model lacks the ability to model long-term periods efficiently. Multi-stage models were introduced and tested on various perspectives such as demand uncertainty and fleet replacement strategies (Hsu et al., 2011; Khoo and Teoh, 2014). Nevertheless, fuel price uncertainty and robust modelling of an aircraft fleet over multiple stages remain unexplored in literature. Moreover, the maintenance perspective has been evaluated, but the work is questionable and could use a proper review to explore the relation and implication on other perspectives. Overall, the perspectives are investigated thoroughly until multi-stage modelling. The main reason for this is the ever-increasing computational power needed to solve these multi-stage models. Dynamic programming (backwards) brought a structured way to model sequential decision-state scenario trees but suffers heavily from the curse of dimensionality.

Model-based ADP
Approximate dynamic programming (ADP) is an optimisation method where instead of stepping backwards (calculating all states and action values recursively) the algorithm steps forwards in time, exploring the decision tree and approximating the value function. ADP or reinforcement learning (RL) can be roughly divided into two approaches to structuring the problem: Model-based or Model-free. The first one uses a predefined model created by the user often represented by a scenario tree. Here a transition function moving from one state to another must be defined and the underlying probability distribution known. In a scenario tree, the amount of dimensions is reduced by discretizing the probability function (Hsu et al., 2011; Requeno and Santos, 2018; Powell, 2009). Model-based ADP method in fleet planning has never been applied with a policy iteration approach. However, Requeno and Santos (2018) have recently built a model proving validity in a value iteration approach. This was only done considering demand uncertainty as a perspective and is therefor partly covered in the mind map.

Model-free RL:
On the other hand, the model-free method is part of a relatively old research field which has been revitalized in the present years. In the 1980s model such as Q-learning (Watkins, 1989) and SARSA (Rummery and Niranjan, 1994) were developed as simple model-free algorithms which are trained using the reward received from the environment through interaction with it. In recent years, with the dramatic increase in computational power and the accompanied popularization of neural networks, sparked research deep Q-learning and policy gradient methods. Here, instead of a learning function, a neural network is trained. These reinforcement learning models have shown tremendous achievements in learning optimal policies in a long-term strategic environment. Similar to the model-based branch, the model-free method can be divided into two subfields: value learning with deep Q-learning and policy gradients. Applying these novel algorithms to combinatorial optimisation methods is an even newer stream of research with few significant work done in the field.

Model-free programming in fleet planning
The discretization and generation of a fixed model with scenario tree is constraint by the curse of dimensionality in approximate dynamic programming. Moreover, designing the transition function in a dynamic programming model is a challenging task and remains subjected to design flaws influencing the outcome heavily. The implementation of a policy optimisation algorithm on a travelling salesman problem has already been proven successful (Kool et al. (2018)). However, the implementation of model-free algorithms in a fleet planning problem has never been done before and represents therefor a promising knowledge gap in the research field.

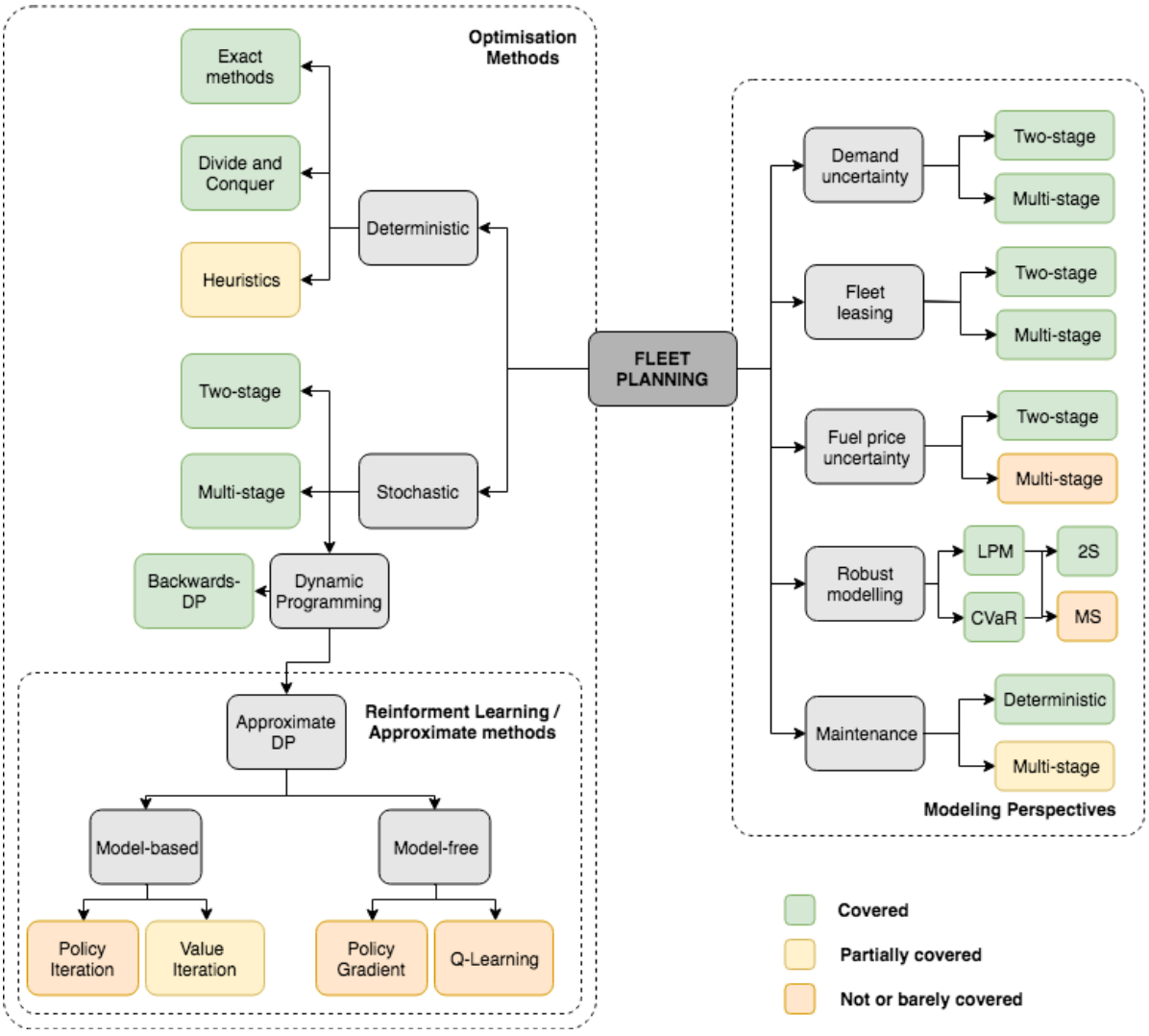


Figure 4.1: Knowledge gap mind map.

III

Research Methodologies

Previously Graded Under AE4010

Project Proposal and Plan: Fleet planning under Uncertainty

M.C.T.C. de Koning, 4179072
Control and Operations

February 20, 2020

Abstract

This project will focus on the very first step of the airline planning process: Fleet planning. The fleet planning process can be defined as the optimisation and decision process regarding acquiring/disposing, which and how many aircraft, at which time period, to fulfil demand and maximise profit of the airline. The aim of the project is to prove that model-free dynamic programming is a viable method in modelling the uncertain fleet planning process. This will be done by generating an environment model evaluating the agent's actions and thus learning the optimal policy. In this report, the research process, project structure and planning, as well the possible outcomes and results are discussed in the following sections. These model-free methods have shown remarkable results in learning optimal long-term policies and could therefor not only be implemented in the fleet planning process, but throughout the whole airline optimisation process.

1 Introduction

Fleet planning is part of the airline planning process and considered to be a long-term strategic procedure to assess if one or more aircraft should be acquired or disposed to ensure long-term operational feasibility of the routes and financial profitability of the airline. The fleet planning process is generally addressed in either one of two approaches. The first approach is the 'top-down' approach, where high-aggregate level, simple, spreadsheets are used. This approach is the most common method employed by airlines today. The technique is easy and cheap; nevertheless, it is highly sensitive to the estimations made by the forecasters. A second method is the 'bottom-up' approach where a detailed modelling technique is used to match expected demand and capacity on flight level. Using such an approach enables a more accurate and robust fleet plan; however, it is computationally expensive (Belobaba et al. (2009)).

Current literature employs various methodologies in solving the 'bottom-up' approach. However, contemporary multi-stage stochastic models become very large as they grow exponentially with new variables added and become insolvable (Sayarshad and Ghoseiri (2009)). Approximate Dynamic Programming (ADP) brings a novel solution to the problem. Such a program exploits the Markov Decision Process (MDP), which is a forward iterative manner to investigate the decision tree and find a near-optimal solution in a much shorter computational time. The latter is called 'model-based' because a transition function of the underlying model using the exogenous (uncertain) information must be designed. However, this transition function is often hard to model and the underlying distribution of the exogenous information unknown (Powell (2009)). An alternative approach is model-free APD or better known as reinforcement learning. The latter applied to the fleet planning problem comprises an interesting research gap.

Note: This part has been adapted from the original report submitted and graded under the course code AE4010. The adaptations are related to the development of the research question throughout the thesis work.

2 State of the art / Literature review

The fleet planning problem from a 'bottom-up' approach can be divided into two main questions: What is the methodology used to model the problem? And, what are the modelling perspectives? These perspectives differentiate themselves from the model types as they do not define the structure of the model, but try to capture decisions, uncertainty, or robustness into the model. In this literature review, an aggregate summary combining these two questions will be depicted below.

In the 1950s the most common approach in a fleet problem was using a deterministic one-stage model. Authors focused on short-term, tactical decisions in rail-car leasing to fulfil short-term excessive demand (Kirby (1959) ; Wyatt (1961)) and the number of ships needed to maintain a fixed schedule (Dantzig and Fulkerson (1954) ; Bartlett (1957)). However, with the advent of more computational power, the possibilities and complexity in modelling increased accordingly. To generate long-term decisions over multiple years, a multi-stage deterministic modelling approach is employed from the 1970s. Shube and Stroup (1975) is one of the first authors to model the fleet problem in a multi-stage environment. The objective function proposed contains three different objectives: minimisation of the direct operating cost (DOC), minimisation of money spent on new equipment, and minimisation of the debt created by acquiring new aircraft. By summing the objective function and constraints over multiple periods, a feasible long-term solution can be generated. However, two problems arise. First, the computational power required to calculate large models was often still not sufficient; the solution to this problem will be addressed later in this section. And secondly, the deterministic models have trouble accounting for uncertainty. The coefficients are deterministically fixed and are often estimated by aggregating over historical data, which ignored the stochastic nature of the variable and affect the solution gravely (Kall and Wallace (1994)).

A solution is stochastic modelling. Dantzig (1955) was the first to consider stochastic modelling and proposed dividing the linear program's decision variable into first and second stage variables. First, the first stage variables are fixed; whereafter, the uncertainty is revealed and the second stage variables are determined. A recourse action is performed which can be seen as a corrective measure against the infeasibility arising due to the realisations of the uncertainty (Sahinidis (2004)). Using this method various perspectives were modelled and tested in the fleet planning model: Demand uncertainty (Listes and Dekker (2002)), leasing vs. ownership (Oum et al. (2000)), risk optimisation (List et al. (2003) ; Naumann and Suhl (2013)), and fuel price uncertainty (Naumann and Suhl (2013)). However, these two-stage models lack the ability to model long-term periods efficiently. Multi-stage models are introduced and tested on various perspectives such as demand uncertainty and fleet mixture (Hsu et al. (2011) ; Khoo and Teoh (2014)). The latter multi-stage models use backwards dynamic programming (DP) to obtain a solution. The method exploits the Markov property which dictates that: *"The future is independent of the past given the presence"* (Markov, 1954). This enables the sequential decision problem to be subdivided into sub-problems, which can be solved recursively.

These large and long-term sequential decision problems is a big field of research in operational and control theory. However, these dynamic programs suffer from the three curses of dimensionality. The reason being that the number of states increases exponentially in relationship to (1) the state space, (2) the outcome space, and (3) the action space (Powell (2011)). The state of the art solution is approximate dynamic programming (ADP). By approximating the value-function and exploration of the decision tree, ADP can optimise the solution of the problem without calculating all the states. The work by Requeno and Santos (2018) is the only attempt to present of model-based ADP in the fleet planning literature. The solution is feasible but requires a defined transition model which is often difficult to define (Powell (2009)). Model-free ADP can provide a novel solution proven in state of the art artificial intelligence research. Instead of approximating the value function in a set scenario tree, the value function or policy is approximated by a neural network which is trained by interacting with an environment, this method is called Deep Reinforcement Learning (DRL). Research by Google DeepMind (Mnih et al. (2015); Schaul et al. (2015)) and OpenAI (Schulman

et al. (2017)) show amazing performances in obtaining long-term strategies, and outperform conventional reinforcement learning techniques. DRL as a model-free ADP technique shows therefor a promising solution to the fleet planning problem.

3 Research question, aims and objectives

3.1 Research objective:

The research has as an objective to contribute to the development of adaptive policies by the generation of a model-free approximate dynamic program in a fleet planning environment subjected to uncertainty. This by generating model where (a) an agent in the form of a neural network learns the optimal adaptive fleet policy, by interaction with (b) an artificially created feedback environment, (c) under uncertain demand.

3.2 Research framework:

Before the research questions can be created, the research framework is devised. The research framework shows roughly how the project objective described above can be achieved.

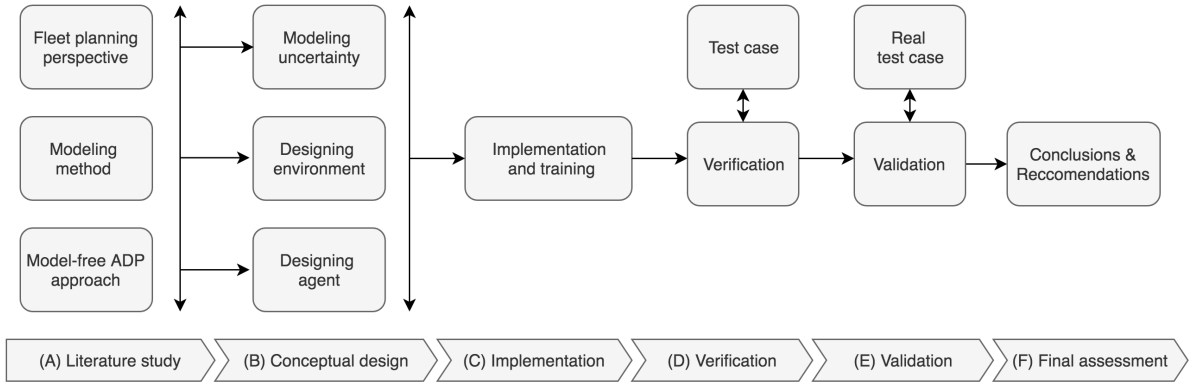


Figure 1: Research framework

The first part of the framework is the (A) Literature study where all relevant knowledge on multiple subjects is gained to define the most appropriate modelling methodology. The second stage is the (B) Conceptual design of the different sub-components. The uncertainty is modelled, the agent's network is designed, and an environment evaluating the agent's policy is created which has a suitable speed. All these sub-components are thoroughly unit tested using some off-the-shelf data. In the third part of the project (C), the sub-components are implemented and connected. The interaction of the components is investigated and corrected is needed. Next, the model is first (D) verificated and secondly (E) validated using a real test case. Finally, conclusions and recommendation are drawn, and (F) the Final assessment is performed.

3.3 Research question(s):

From the general outline proposed in the research framework and the**initial** research objective, the main research question can be devised:

Can a model-free reinforcement learning model perform better in predicting the adaptive policy in a fleet planning problem compared to a model-based approach.

Update: throughout the thesis work, the quantitative analysis of the model-free vs the model-based was

out of the scope, because no proof of concept was available. Consequently, the choice was made to make a proof-of-concept and to compare this to the contemporary method currently employed in the airline industry. Therefore, the research question becomes:

Can a model-free reinforcement learning model learn the long-term dynamic policy in a fleet planning problem under demand uncertainty.

More in-depth sub-questions are formulated which must be answered during the research project. Answering all these research questions results in answering the main research question. The sub-questions are related to the block sections shown in the research framework (Figure 1), and each sub-question contains multiple sub-sub-questions.

(A) Which elements and modelling techniques from literature can be used to design a model-free approximate dynamic program that successfully predicts the optimal policy?
   (a) What are the different fleet planning perspectives and what is their influence on the fleet planning process?
   (b) What are the modelling methods in fleet planning under uncertainty?
   (c) What is the difference between Model-free and Model-based approaches
   (d) What is the knowledge gap and research objective

(B) How should the learning agent and modelled environment architecture be designed to learn the policy
   (a) What is the most efficient way to accurately calculate the reward?
   (b) How should the demand distribution be modelled?
   (c) What is the best learning algorithm and how should the network be designed
   (d) How can the sub-components be verified?

(C) How can the sub-components be implemented in a model and trained
   (a) Does the sub-components work correctly without leakage of information?
   (b) Is the model converging to an optimal policy?
   (c) How does the model behave learning under uncertainty?
   (d) What is the learning time?

(D) How can the model be verified?
   (a) Does the model behave as expected?
   (b) Is there proper data available to verify, and how can it be used in testing?
   (c) How can a sensitivity analysis be performed?

(E) How can the verified model be validated?
   (a) Is there a real test case available to validate the model?
   (b) How can it be used in the validation of the model?

(F) What are the relevant conclusion and recommendations for future work?
   (a) Is the research question answered?
   (b) What are the conclusions?
   (c) What are the recommendations?
   (d) How can the results be reported in a structured manner?

4 Theoretical Content/Methodology

As section 5 will point out the test set-up; in this section, a more in-depth view is given in the theoretical basis of the research and how it differs from current approaches.

The current state of the art solution in solving the fleet planning problem is the method employed by Requeno and Santos (2018) which is Approximate Dynamic Programming (ADP). In equation 1 the Bellman equation is visible. This is the most important equation in dynamic programming as it defines the value of being in state t to be the cost $(C_t(S_t, a_t))$ of being in present state s_t, plus the expected value of moving to state s_{t+1}:

$$V_t(S_t) = \max_{a_t \in \mathbf{A}_t} \left(C_t(S_t, a_t) + \gamma E\left\{ V_{t+1}\left(S_{t+1}(S_t, a_t, w_{t+1}) \right) | s_t \right\} \right) \tag{1}$$

In ADP, this equation is used iteratively calculate throughout the tree, exploring state-action values. Because the program steps forward in time, the expected value of the next state V_{t+1} is unknown and is approximated. However, a secondary problem arises: the transition function $S_{t+1}(S_t, a_t, w_{t+1})$ is often very hard to model. In a fleet planning problem, where the state-action space is limited, the highest uncertainty lies in the unknown probability distribution of w_{t+1}. A solution to this is model-free ADP (or reinforcement learning model) were the program learning the value function or policy becomes an agent interacting with the environment and iteratively

The hypothesis is deducted from the research question and becomes therefor: *a model-free reinforcement learning model can better learn the dynamic policy in a fleet planning problem under demand uncertainty compared to conventional approaches.*

To accurately test this, first (1) a model-free reinforcement learning model must be created. And secondly (2) the performance must be compared to another state of the art solutions. The experimental set-up and performance evaluation is discussed in more depth in section 5. The first part of the methodology is encaptured by block (A), (B), and (C) in the research framework and comprises the generation of the building block for the experimental set-up. The biggest novelty lies in making a model-free reinforcement model which learns from the environment. In contemporary research a distinction between two types of models can be made: *Policy Optimization* and *Deep Q-learning*. Both techniques have benefits and drawbacks. In the literature study, a trade-off should be made and an optimal approach chosen.

To answer the research question, the model-free reinforcement model must be trained in a set-up as similar to previous research as possible so a comparison of the model-free versus conventional methods can be made and the hypothesis answered.

5 Experimental Set-up

To answer the research question and hypothesis, a research strategy has to be employed. Regarding the nature of our problem, quantitative comparison, an experimental set-up is the most viable approach to proof the concept. The experimental set-up in this research will be completely based on generating computer models and testing the interactions between them and comparing them to previous research. In figure 2, a typical reinforcement learning architecture similar to the experimental set-up employ in the research, is depicted. Two basic entities can be differentiated: the agent and the environment.

The agent can be viewed as the player in our experimental set-up. He will learn the policy by interacting with the environment. It does so with observed states and rewards. The agent has a neural network at which at each stage it receives the states from the environment, and it can predict the most likely action

corresponding to those states. After an episode (which is a set of sequential observation-action procedures) the environment returns a reward to the agent's neural network which it uses to update the parameters. The policy can be seen as 'what is the best action (acquire/sell) given the state of the network model'.

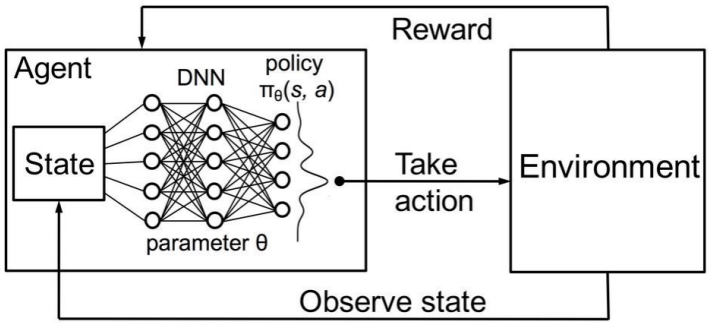


Figure 2: Reinforcement learning architecture Mao et al. (2016)

The environment can be viewed as the place where the agents 'live' in. The agents can interact with the environment through actions and receive a response in the form of a changing state or a reward. In our set-up, the environment is an artificial created model which uses the action of the agent and a defined probability distribution of the demand to calculate the profit in that situation. The model is a simple network optimisation problem solved and compared to the most optimal solution to generate a reward. The resulting state and rewards can then be fed back to the agent to learn from.

The most important aspect is the interaction between the agent and the environment and how the agent learns to adapt the policy to the uncertain behaviour in the environment. The policy will sample 'episodes' or trajectories which contains state-action pairs. At the end of the episode -which is the horizon of the problem- the agent will receive a reward to tune the parameters. A point of concern is the computational effort needed in the environment. Here the calculation of the next state must be performed efficiently as it might affect the learning rate of the model gravely. Sample efficiency is therefor of major importance when choosing the appropriate learning model.

Secondly, the environment created must resemble as closely as possible to the conventional methods. If the models differ too much the comparison of the two approaches might be biased and incorrect conclusions drawn. Moreover, the computational speed of the hardware of software used must be identical to make a qualitative comparison. The conventional method must therefor be re-run on the same computer as the novel architecture to correctly asses the performance.

6 Results, Outcome and Relevance

Data:

To simulate the environment and train the deep neural network, sufficient real-life airline data is required. To start with, basic airline and aircraft data is required. Airline data might consist of the routes the airline wants to operate, load factors, turn-around-time, block times, etc. Aircraft data will relate to the current fleet of the airlines such as the size of the fleet, the types of aircraft, range, fuel efficiency, maintenance requirement, etc. The exact data required will be vastly dependent on the depth of the model simulating the environment. It is important to note that if artificially created variables are used, one checks if these variables are available for a verification/validation test case.

A second important group of data is the parameters with prevail with uncertainty. Contrarily to the data

discussed above, these variables are sampled from a probability distribution where often little historical data is available. Examples of these stochastic variables are demand on the routes, fuel price, and competition. One must be careful in designing these distributions and sampling from them as they might be of high influence on the results. A thorough sensitivity analysis is advised.

Results:

The results of the experimental set-up should allow for a comparison between a model-based and the conventional methods answering the hypothesis. For this, metrics of performance should be generated. The overall solution of the objective function is probably the profit of the airline. However, the highest profit doesn't mean the best performance overall. Other considerations should be taken into account such as the optimality gap and learning rate.

Optimality gap.
An easy metric to compare the performance of different models. For this, the size of the fleet planning problem is reduced so it can be solved within a reasonable time. The optimality gap is the gap between the best solution and the best bound. Together with the time it takes to find the optimality gap, the optimal profit and fleet is compared between the conventional methods and model-free, with a simple solver such as Gurobi or CPLEX as a baseline.

Learning rate. A second popular measure is the learning rate. The learning rate is a measurement often used to compare the performance of neural networks. In Figure 4 an example of a learning rate comparison graph is shown. Each graph shows the learning performance of a different environment (in this case Atari 2600 games) and the various network architectures. Using this, the convergence speed and maximum score are combined in one graph. Comparing these graphs can show the (over)fitting and versatility of the different models.

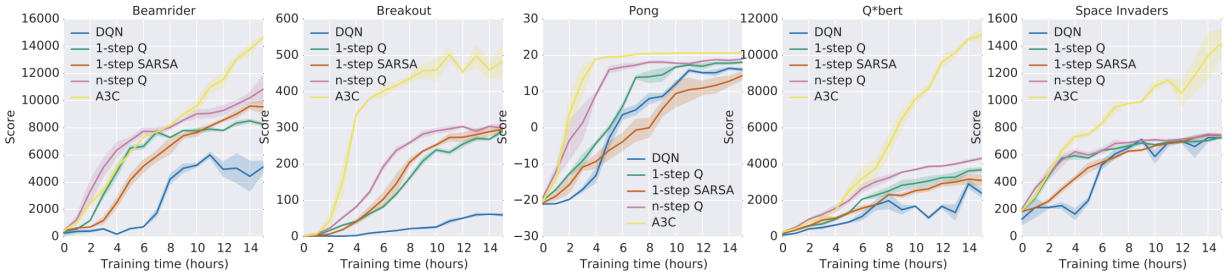
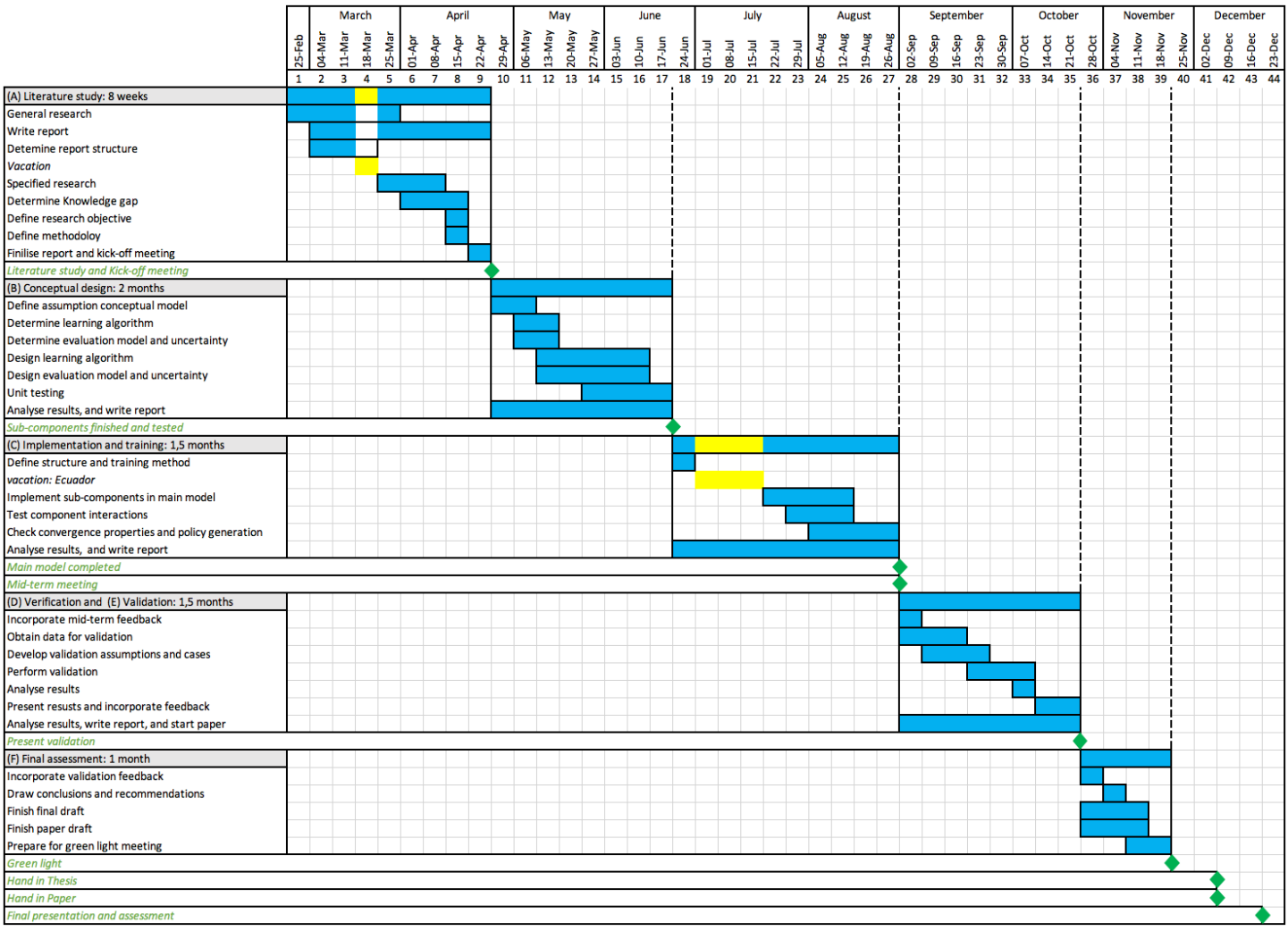


Figure 3: Example learning rate comparison for neural network architectures (Mnih et al., 2016)

7 Project Planning and Gantt Chart

On the next page, the Gantt chart of the research is shown. The Gantt chart results from the research framework (depicted in Figure 1) and gives a more in-depth overview of the different work packages. Analogous, the research framework the Gantt chart is divided into 5 work packages, except for verification and validations which is combined in one work package: (A) Literature study, (B) Conceptual design, (C) Implementation and training, (D) Verification, and (E) Validation, (F) Final assessment.

In the Gantt chart, two colours can be distinguished: blue and yellow. The colour blue shows normal work weeks, whereas the colour yellow shows holidays and consequently no work on the research will be performed. Finally, the green diamond shows the deliverables and important meetings.


March
April
May
June
July
August
September
October
November
December
25-Feb
04-Mar
11-Mar
18-Mar
25-Mar
01-Apr
08-Apr
15-Apr
22-Apr
29-Apr
06-May
13-May
20-May
27-May
03-Jun
10-Jun
17-Jun
24-Jun
01-Jul
08-Jul
15-Jul
22-Jul
29-Jul
05-Aug
12-Aug
19-Aug
26-Aug
02-Sep
09-Sep
16-Sep
23-Sep
30-Sep
07-Oct
14-Oct
21-Oct
28-Oct
04-Nov
11-Nov
18-Nov
25-Nov
02-Dec
09-Dec
16-Dec
23-Dec
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44
(A) Literature study: 8 weeks
General research
Write report
Detemine report structure
Vacation
Specified research
Determine Knowledge gap
Define research objective
Define methodoloy
Finilise report and kick-off meeting
Literature study and Kick-off meeting
(B) Conceptual design: 2 months
Define assumption conceptual model
Determine learning algorithm
Determine evaluation model and uncertainty
Design learning algorithm
Design evaluation model and uncertainty
Unit testing
Analyse results, and write report
Sub-components finished and tested
(C) Implementation and training: 1,5 months
Define structure and training method
vacation: Ecuador
Implement sub-components in main model
Test component interactions
Check convergence properties and policy generation
Analyse results, and write report
Main model completed
Mid-term meeting
(D) Verification and (E) Validation: 1,5 months
Incorporate mid-term feedback
Obtain data for validation
Develop validation assumptions and cases
Perform validation
Analyse results
Present resusts and incorporate feedback
Analyse results, write report, and start paper
Present validation
(F) Final assessment: 1 month
Incorporate validation feedback
Draw conclusions and recommendations
Finish final draft
Finish paper draft
Prepare for green light meeting
Green light
Hand in Thesis
Hand in Paper
Final presentation and assessment

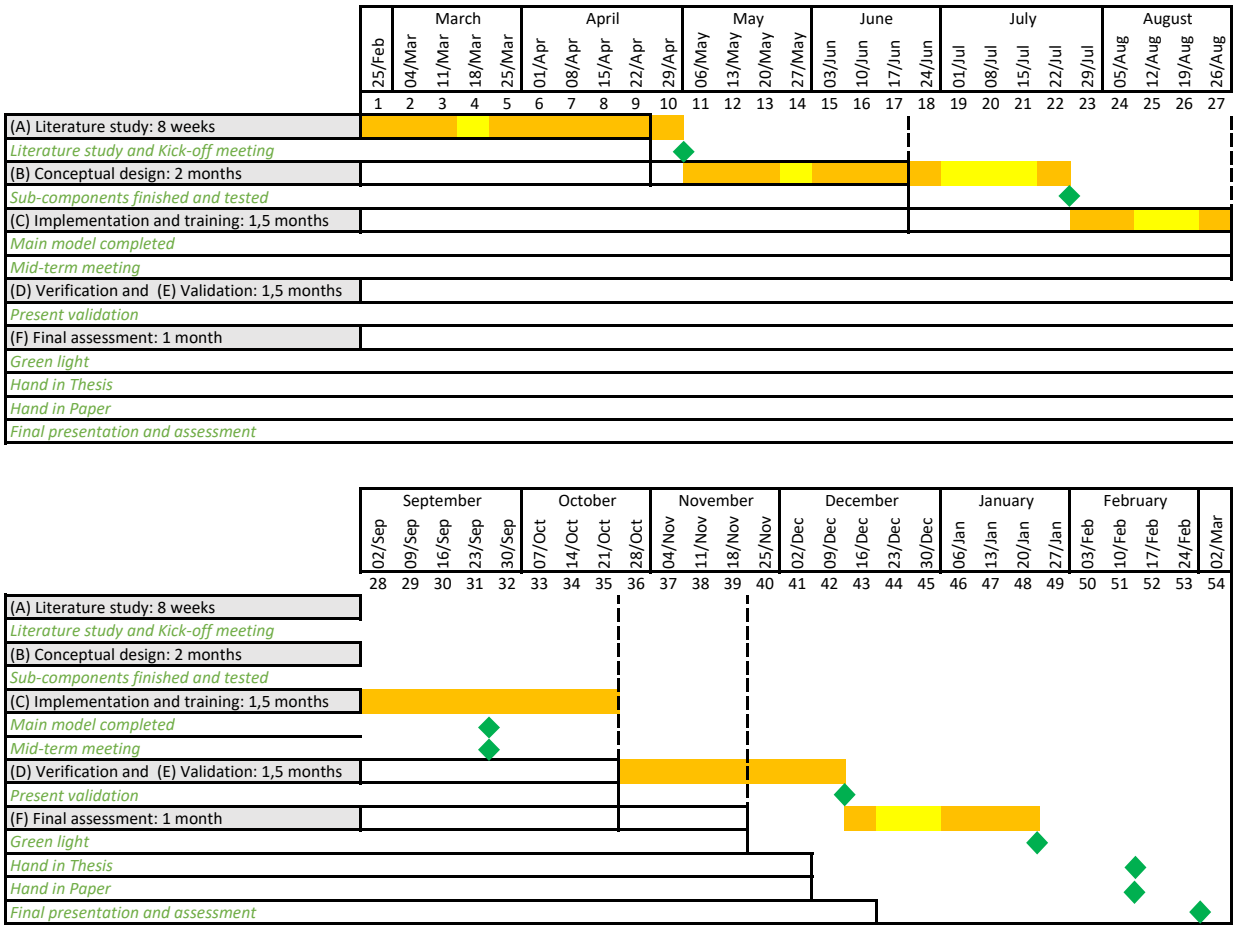


Figure 4: Resulting Gantt chart

EDIT

The second Gantt-chart shows the resulting timeline of performed as an assessment after completion of the research. The intended work time has exceeded the predetermined estimated work due to three reasons: First, the initial timeline was too optimistic and was short of the actually required thesis duration of 40 weeks. Secondly, more vacation was taken than anticipated. Where initially a total of 4 weeks was planned, in the end, more than double (9 weeks) was taken. And thirdly, the implementation and training phase proved harder than expected. Tuning a Deep Neural Network proved more difficult than anticipated and caused some delay. Overall I feel confident about the progress of the work and in the end and only exceeded the nominal project time by a couple of weeks.

8 Conclusions

With volatile fuel prices, uncertain demand, and high capital investments, the airline rely on very thin margins to run a profitable business. Fleet planning models which can predict robust long-term solutions and possible dangerous scenario's are therefor of uttermost importance in the airline industry. Model-based ADP approaches are limited as they require the use of a transition function and the fixed decision tree. Model-free methods do no use such a model architecture and should be able to learn very complex policies by simply iterating the agent-environment model.

The proposal of training a deep neural network sets a precedent never seen before in the fleet planning industry. These models have shown impressive learning performances in long-term and strategic planning problems (Schulman et al., 2017). The knowledge and applications of artificial intelligence techniques and more specifically reinforcement learning is stunning the world every day.

The research objective and plan show an achievable timeline to finish and answer the research question in a reasonable time. Challenges lie in the development of the environment's architecture, as this must be robust and fast. A second major challenge is the training and interaction of the agent and environment, as the performance of both is highly dependent on each other. And the final challenge, is the verification and comparison of the newly generated model-free method to the old model-based method, as the performance is a direct result of the environment model.

If the challenges can be overcome, modelling ADP model-free could be a viable method not only in fleet planning but in many other airline processes. The technology has proven itself in many other similar research area's and brings a solution in making long-term decisions in dynamic environments.

References

Bartlett, T. E. (1957). An algorithm for the minimum number of transport units to maintain a fixed schedule. *Naval Research Logistics Quarterly*, 4(2):139–149.

Belobaba, P., Odoni, A. R., and Barnhart, C. (2009). *The Global Airline Industry*.

Dantzig, G. B. (1955). Linear Programming under Uncertainty. *Management Science*, 1(3/4):197–206.

Dantzig, G. B. and Fulkerson, D. R. (1954). Minimizing the number of tankers to meet a fixed schedule. *Naval Research Logistics Quarterly*, 1(3):217–222.

Hsu, C.-I., Li, H.-C., Liu, S.-M., and Chao, C.-C. (2011). Aircraft replacement scheduling: A dynamic programming approach.

Kall, P. and Wallace, S. W. (1994). *Stochastic Programming Second Edition*.

Khoo, H. L. and Teoh, L. E. (2014). An optimal aircraft fleet management decision model under uncertainty. *Journal of Advanced Transportation*, 48(7):798–820.

Kirby, D. (1959). Is Your Fleet the Right Size? *Journal of the Operational Research Society*, 10(4):252–252.

List, G. F., Wood, B., Nozick, L. K., Turnquist, M. A., Jones, D. A., Kjeldgaard, E. A., and Lawton, C. R. (2003). Robust optimization for fleet planning under uncertainty.

Listes, O. and Dekker, R. (2002). A scenario aggregation based approach for determining a robust airline fleet composition. Technical report.

Mao, H., Alizadeh, M., Menache, I., and Kandula, S. (2016). Resource Management with Deep Reinforcement Learning. pages 50–56.

Markov, A. A. (1954). The theory of algorithms. *Trudy Matematicheskogo Instituta Imeni VA Steklova*, 42:3–375.

Mnih, V., Badia, A. P., Mirza, M., Graves, A., Lillicrap, T. P., Harley, T., Silver, D., and Kavukcuoglu, K. (2016). Asynchronous Methods for Deep Reinforcement Learning.

Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., Graves, A., Riedmiller, M., Fidjeland, A. K., Ostrovski, G., Petersen, S., Beattie, C., Sadik, A., Antonoglou, I., King, H., Kumaran, D., Wierstra, D., Legg, S., and Hassabis, D. (2015). Human-level control through deep reinforcement learning. *Nature*, 518.

Naumann, M. and Suhl, L. (2013). How does fuel price uncertainty affect strategic airline planning? *Operational Research*, 13(3):343–362.

Oum, T. H., Zhang, A., and Zhang, Y. (2000). Optimal demand for operating lease of aircraft. *Transportation Research Part B: Methodological*, 34(1):17–29.

Powell, W. B. (2009). What You Should Know About Approximate Dynamic Programming.

Powell, W. B. (2011). *Approximate Dynamic Programming: Solving the curses of dimensionality*, volume 703. John Wiley & Sons.

Requeno, L. and Santos, B. (2018). Multi-period adaptive airline fleet planning problem. *Submitted to: Transportation Science*.

Sahinidis, N. V. (2004). Optimization under uncertainty: state-of-the-art and opportunities. *Computers & Chemical Engineering*, 28(6-7):971–983.

Sayarshad, H. R. and Ghoseiri, K. (2009). A simulated annealing approach for the multi-periodic rail-car fleet sizing problem. *Computers & Operations Research*, 36:1789–1799.

Schaul, T., Quan, J., Antonoglou, I., and Silver, D. (2015). Prioritized Experience Replay.

Schulman, J., Wolski, F., Dhariwal, P., Radford, A., and Klimov, O. (2017). Proximal Policy Optimization Algorithms.

Shube, D. P. and Stroup, J. W. (1975). Fleet Planning Model. *Winter Simulation Conference Proceedings*.

Wyatt, J. K. (1961). Optimal Fleet Size. *Journal of the Operational Research Society*, 12(3):186–187.

IV

Thesis report

1 Reinforcement Learning

Generally, the Machine Learning (ML) field is subdivided into three classes: Supervised Learning, Unsupervised Learning, and Reinforcement Learning. All three methods have as purpose to learn a mapping of an input to an output. A supervised learning model is trained using a dataset consisting of input data and output data, referred to as labelled data. With the labelled data, a function can be learned by, feeding in the input data, observe the estimated output, and adjust the weights of the function by comparing the observed output to the actual (true) output labels. Unsupervised Learning models learn from datasets with only input data. Techniques such as data clustering and dimensionality reduction classify unseen data. Reinforcement Learning (RL) makes up the third category in the ML field. RL models initially have no input or output datasets; however, it is able to interact with an environment to learn a specific goal. By interacting with the environment the RL models receive an assessment of the predicted output in the form of a reward. This process is repeated thousands of times until the RL model learns the relevant dynamics of the environment to complete the predefined goal.

In the first section, the history of reinforcement learning and its applications in operational research is outlined. In the second section, the Markov decision process is discussed. That will be followed by the depiction of the learning strategies; value-based learning; and the introduction of the deep Q-network. In the final section, the application of the deep Q-network in the fleet planning problem is discussed.

1.1. Context

Reinforcement learning dates back as early as the 1980s with the Q-learning algorithm from Watkins (1989) as one of the first learning algorithms to combine temporal difference and optimal control threads (Sutton and Barto, 2018). At that time, the research field of RL had experienced and tremendous growth in the artificial intelligence subfield machine learning and neural networks (Sutton and Barto, 2018). Tesauro's TD-Backgammon learning program increased the popularity of RL even further, by matching the world best players in 1992 (Tesauro, 1995).

However, the initial popularity surge flattened due to the difficulty to upscale RL models to larger and more complex applications. Multiple techniques were devised to solve the generalization and abstraction problems (Ponsen et al., 2009). In 2013, Mnih et al. (2015) developed the first Deep Reinforcement Learning (DRL) method used to play multiple Atari games with human performance. The proposed technique enabled the approximation of high dimensional action-spaces and the growth of RL models to learn more complex problems. In the years hereafter, the research in DRL increased exponentially, transforming it in a state-of-the-art learning model achieving often superhuman performances such as defeating the world champions in the game of GO (Silver et al., 2017) and DOTA

2 (Berner et al., 2019).

Also in the field of operational research, artificial intelligence and RL has found applications solving computational expensive optimization problems. In combinatorial optimization generally three algorithmic structures are defined: end to end learning, learning of meaningful properties of optimization problems, and machine learning alongside optimization algorithms (Bengio et al., 2018). Recent attempts have proven fruitful in replacement of computationally expensive optimisation algorithms such as resource allocation and vehicle routing problems in the transport industry. Lam et al. (2007) used approximate dynamic programming to model operational strategies for the relocation of empty containers in the sea-cargo industry. Novoa and Storer (2009) examined a similar approach in modelling a single-vehicle routing problem with stochastic demands. And Powell (2009) uses approximate dynamic programming to build an optimizer simulator for locomotive planning. Finally, Requeno and Santos (2018) were the first to replace the optimisation of a fleet planning problem under uncertain demand by an approximate dynamic program. All aforementioned solutions employ a model-based learning algorithm, which leaves the model-free learning algorithm to solve the fleet planning problem unexplored.

1.2. The Markov Decision Process

To understand the principles of RL, an in-depth explanation of the agent-environment interaction is discussed in this section. For the fleet planning problem, it is assumed that the optimal fleet policy is calculated in a finite horizon problem using a discrete-time stochastic control process. A series of episodes E is constructed each episode with a finite set of discrete-time steps or periods $t = \{1, ..., T\}$. This is called an *episodic* Markov Decision Process (MDP).

At each period, the agent gets provided by the environment with a state $s_t \in \mathcal{S}$, the agent determines the best possible action $a_t \in \mathcal{A}$, and the environment then evaluates that action through a reward $r_t \in \mathcal{R}$ and returns the state transition $s_{t+1} \in \mathcal{S}$. In Figure 1.1 the simplified architecture is visible, showing the main interactions between the agent and the environment. Learning the optimal policy by accumulating the maximum reward through exploring agent-environment interactions is called reinforcement learning.

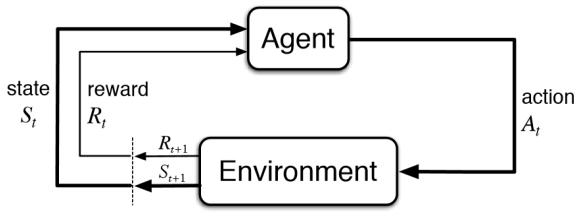


Figure 1.1: General solution method diagram of Reinforcement Learning set-up (Sutton and Barto, 2018).

The MDP is formally defined by a decision tuple of the form $< \mathcal{S}, \mathcal{A}, T, R, \gamma >$. Where, $\mathcal{S}$ is a finite set of states, $\mathcal{A}$ is a finite set of actions, $R(s_t, a_t, s_{t+1})$ is a reward function with a continuous set of possible rewards $\mathcal{R}$. γ is the discount rate: $\gamma \in (0, 1)$, and T is the transition matrix of the Markov process and can be defined as stochastic probability function of moving from a state s with an action a to a next state s' depicted in Equation 1.1.

$$T\left(s, a, s'\right) = P\left[s_{t+1} = s' | s_t = s, a_t = a\right] \tag{1.1}$$

At this point, it is important to note that the MDP in the fleet planning problem is assumed to be a fully observable MDP. This means that the agent act in the decision process contains full knowledge of each of the states. In other words, the agent can directly observe the states from the

environment which influences the policy, unlike a Partially Observable Markov Decision Processes (POMDP) where the agent only observes part of the states and derive a policy based on those observed. In a POMDP, the MDP tuple expands to a 7-tuple $< \mathcal{S}, \mathcal{A}, T, R, \Omega, \mathcal{O}, \gamma >$. A distinction is made between the actual states $\mathcal{S}$ and the observations of the model Ω, with a set of observation probabilities $\mathcal{O}$. a POMDP gravely increases the complexity of the system dynamics and RL models and is therefor out of the scope of this thesis project.

Figure 1.2 shows an example of a fully observed MDP used in the fleet planning problem. In each period t, the agent takes an action a_t. As a result a new state s_{t+1} is created in the environment together with the reward r_t.

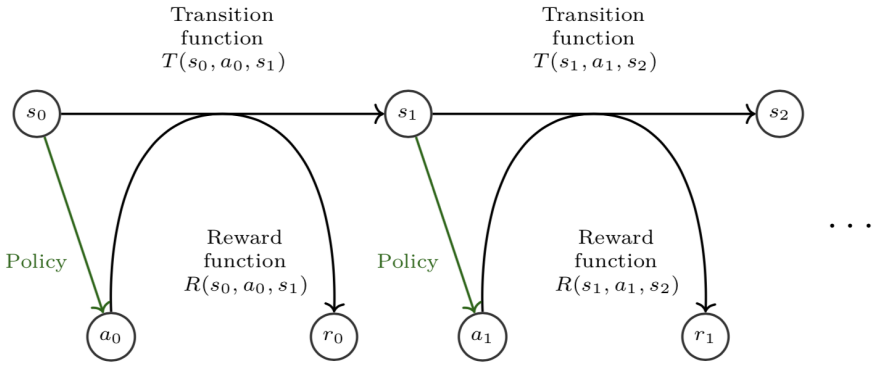


Figure 1.2: Example of a Fully Observable MDP (François-Lavet et al., 2018).

1.3. Learning Strategies

At every time step t, the agent performs a mapping from the states to probabilities of selecting an action a_t. The mapping of the agent is called the policy and denoted by π (Sutton and Barto, 2018). The agent's goal is to learn the policy, which maximizes the total amount of reward over the finite time horizon. In the fleet planning problem this can be interpreted as choosing the best possible fleet decision, given the current airline and network state, in order to maximise the profit of future operations.

Temporal Difference and Monte Carlo

Monte Carlo (MC) and Temporal Difference (TD) are the two main learning methods in RL. Figure 1.3 shows a simplified architecture comparison between TD (left) and MC (right). The diagram shows the top node which is to be updated with a trajectory of all transitions for one episode from the top to bottom. The white circles represent states, the black circles actions, and the square a terminal state. In TD, the value of a state is calculated using the current state and the maximum expected value of the next state. Monte Carlo on the other hand, uses all the accumulated rewards over the episode to calculate the value of a state. Consequently, MC can only be updated when the episode has ended and all rewards are known. This can be a long process, as very long-horizon models have to wait to commence the learning until the terminal state is reached.

Where MC suffers from high variance and no bias, the TD method has low variance, but high bias. Moreover, TD is very efficient in use, whereas MC has good convergence properties (Silver, 2018). In between TD and MC lies N-step TD which is a hybrid form of the two learning methods. Where in (1-step) TD only one reward and one estimate is used, 2-step TD uses two sequential rewards and one estimate. A hybrid version can compromise between the advantages and disadvantages of TD and MC. N-step returns can thus be defined as:

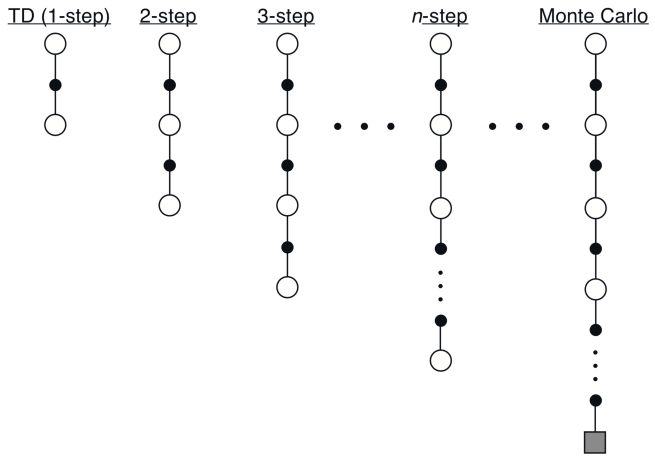


Figure 1.3: Temporal Difference (left) and Monte Carlo (right), with all hybrid TD-methods in the middle (Sutton and Barto, 2018).

$$\begin{aligned} G_t^{(1)} &= R_{t+1} + \gamma V(S_{t+1}) \\ G_t^{(2)} &= R_{t+1} + \gamma R_{t+2} + \gamma^2 V(S_{t+2}) \\ &\vdots \\ G_t^{(\infty)} &= R_{t+1} + \gamma R_{t+2} + \ldots + \gamma^{T-1} R_T \end{aligned} \tag{1.2}$$

Reflecting on the different methods described above and the implementation to the fleet planning problem brings no limitations to either one of the methods. The fleet planning problem has a relatively short horizon, thus MC becomes a viable option next to the original 1-step TD approach. Subsequently, the N-step trade-off is built as a parameter into the RL framework which can be tuned in the sensitivity analysis to achieve the most efficient and best convergence of the policy.

Policies:
In reinforcement learning, two major distinctions in policy generation can be distinguished:

- **Deterministic policy** is mainly used in an environment where the result of an action results in a known change of states. The policy is defined by $\pi(s)$ and the result of the mapping is one deterministic action a_t which has the highest value. Deterministic policy models usually represent the state-action by a value function to determine the best possible action, and are referred to as **value-based algorithms**. These algorithms are generally more simplistic and have good convergence properties but are limited to small discrete action spaces.

- **Stochastic policy** is more regularly used in an environment where the transition from states is not fully known and is represented by a probability. As a result, the optimal policy $\pi(a|s)$ is the probability of selecting the action $a_t = a$ if state $s_t = s$. The policy is a direct result of the gradient states-action function and therefor often referred to as **policy-based algorithm**. These algorithms generally allow for a larger discrete action space (or continuous action space), but have high variance and often converge to local maxima.

A final third policy generation method is a hybrid solution between deterministic and stochastic policies. Actor-critic methods use a value-based estimator as a critic to estimate the value of the state-action pair and an actor as the policy-based estimator to control the policy of the agent. The actor uses the 'critique' of the critic to asses the previous action.

Throughout all the reviewed literature there is no clear preference for value-based, policy-based, or actor-critic methods. The application of the best methodology is highly problem dependent.

Generally, value-based methods are simpler to employ and assess compared to policy-based methods and have a low variance and bias. However, they can only operate in small deterministic action, whereas policy-based methods are able to represent continuous actions spaces. In the fleet planning problem, the transition from one state to the next state is assumed to be fully observable, and only one deterministic fleet action can be chosen at each state. Moreover, the action space (fleet decision) is limited and discrete. The result is that the value-based method is the chosen framework to create a fleet policy.

Model-free vs Model-based

In Figure 1.4 the Reinforcement Learning taxonomy is depicted. Here, a secondary division in the RL fields is visible: model-based and model-free learning. Both models exploit the MDP to find the optimal policy to solve a problem. Model-based algorithm assumes that there is a known model of the environment in which the algorithm operates. The state-action space can usually be represented by a scenario tree which defines the possible scenarios or outcomes of state transitions and uncertainties. In other words, the transfer function T is known and the learning method is adapted to fit these state transitions. The state transitions and uncertainties are usually discretised in a small set of possible outcomes to keep the tree from growing too large. In model-free RL, the transfer function T is unknown to the agent, and there is no scenario tree. The RL model learns the optimal policy solely based on the MDP tuple, without additional information of the environment state transitions or probability distributions of uncertain parameters. Hence, the advantage of a model-free approach over a model-based is that no scenario model of the environment needs to be generated to learn the optimal policy. Consequently, the degree of success of the model-based approach becomes dependent on the ability to discretize an accurate model of the environment. In the discretization process, the true transition probabilities are aggregated and extreme outcomes disregarded. As a result, this research opted for a model-free RL approach to the fleet planning problem under uncertain air-travel demand.

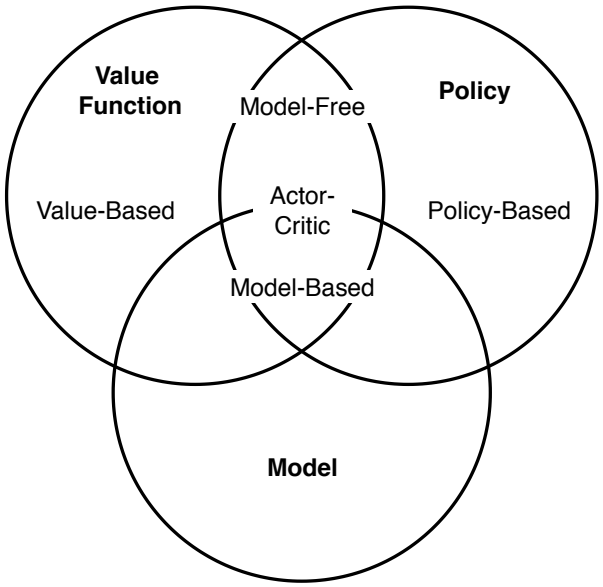


Figure 1.4: Taxonomy of Reinforcement Learning strategies (adapted from Odonkor and Lewis (2018)).

1.4. Value-based Learning

In the previous section it was established that the value-based method is the most logical approach to effectively acquire the optimal fleet policy. However, before diving deeper into the value-based methods, some important concepts and parameters are explained below.

The objective of a RL model is to find the optimal policy π^* which maximizes the discounted return. The discounted return G_t of a finite horizon problem can be defined as the sum of all future returns from t to T:

$$G_t = R_{t+1} + \gamma R_{t+2} + \cdots + \gamma^{T-1} R_T = \sum_{k=0}^{T} \gamma^k R_{t+k+1} \tag{1.3}$$

The discounted rate γ_t, $(0 \leq \gamma \leq 1)$, influences returns k-steps into the future, as expected short-term reward is considered a more valuable long-term reward when the discount rate is high. On the contrary, in value-based learning models, the model tries to learn the estimated value of being in a state using the discounted return. The value $V^\pi(s)$ of being in state s can be defined as the expected discounted return for a given state in Equation 1.4, and for a state-action pair in equation 1.4 called the Q-function.

$$V^\pi(s) = E_\pi [G_t | s_t = s] \tag{1.4}$$

$$Q^\pi(s,a) = E_\pi [G_t | s_t = s, a_t = a] \tag{1.5}$$

Finally, the optimal expected return $V^*(s)$ and the optimal Q-function $Q^*(s,a)$ are defined as:

$$V^*(s) = \max_{\pi \in \Pi} V^\pi(s) \tag{1.6}$$

$$Q^*(s,a) = \max_{\pi \in \Pi} Q^\pi(s,a) \tag{1.7}$$

Having defined all the basic concepts and functions of the value-based reinforcement learning approach the remaining part of this chapter provides a deeper understanding of the various methods available and the final method chosen for this research.

Q-learning

Probably the most widely known algorithm in RL methods is Q-learning. This algorithm was developed in the 1980s by Watkins (1989) and is one of the simplest forms of a RL model. At every step, the algorithm chooses the action that will maximise the expected return of future states under the greedy policy. The Q-learning algorithm uses the Q-function in a look-up table to approximate the expected state-action value $Q^*(s,a)$.

After the action, the agent receives *an experience* from the environment in the form of a tuple: $< s, a, r, s' >$. In other words: under state s, the agent chose action a, received reward r, and transitioned to state s'. This experience is very useful for the agent to update the Q-function in the look-up table. The new Q-value in Q-function is updated using *one-step TD* described by:

$$Q^{new}(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha \left[r_{t+1} + \gamma \max_a Q(s_{t+1}, a) - Q(s_t, a_t) \right] \tag{1.8}$$

The old Q-value is updated by the difference between the 'target value' and the old Q-value. The target value $(R_{t+1} + \gamma \max_a Q(S_{t+1}))$ can be defined as the sum of the reward and the maximum expected value or return. The learning rate dictates how much the Q-function is updated, at zero no learning will take place, at one the old Q-value is completely discarded. This process is repeated every period until $t = T$ is reached, yielding the end of an episode e; the final Q-values are updated,

the states are reset to s_0, and the process is repeated until the Q-function converges.

Despite the advantages, a problem arises when the optimal policy is used. After updating the Q-function for a positive experience (e.g. high reward), the next time the agent reaches a similar state, the same action will always be chosen as the updated Q-value is always higher. To prevent this, an ϵ-greedy algorithm is used to ensure that random actions are chosen to let the agent *explore* the state-action space in the early stages of the learning. As the learning progresses, the rate of exploration decreases and the algorithm will follow the optimal policy more (*exploit*).

The Q-learning algorithm forms the foundation for value-based model-free learning processes. However, when the state-action space grows, it becomes impossible to learn the Q-function for every single state-action pair.

Neural Network

Instead of representing every single state-action pair in a look-up table, the Q-value can be approached by a set of parameters θ. The value function approximated by this set of parameters can be described as a parametric or a non-parametric function. Non-parametric function approximators such as neural networks (NN) have proven to be an effective approach for the approximation of value functions due to their ability to represent nonlinear functions and because they are not dependent on the initial model design (de Koning, 2019). Moreover, the work of Requeno and Santos (2018) showed that non-linear function approximators are needed in the ADP fleet planning problem to represent the value function. In this section, the working principles of NN as function approximator is discussed.

A neural network maps a set of inputs x on a set of outputs y through a function $\{ : \mathcal{X} \rightarrow \mathcal{Y}$ parameterized by the weights $\theta \in \mathbb{R}^{n_\theta}$ ($n_\theta \in \mathbb{N}$) (François-Lavet et al., 2018):

$$y = f(x;\theta) \tag{1.9}$$

In a neural network, the multiple processing layers maps an input vector through layers to an output vector. At each layer, a non-linear transformation takes place, inducing different levels of abstraction (Olah et al., 2017; Erhan et al., 2009). In Figure 1.5 an example architecture of a neural network with one hidden layer is visible. This network is referred to as 'fully connected' because all outputs of the previous layer are the inputs for each neuron in the next layer.

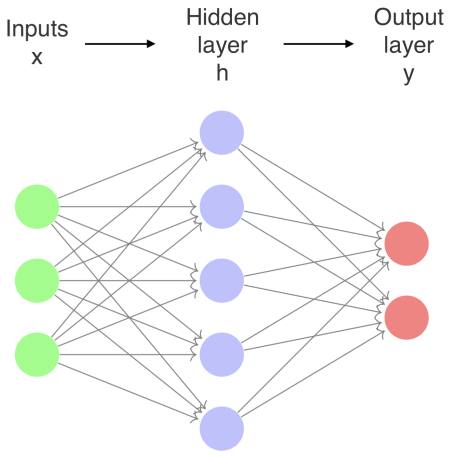


Figure 1.5: Example fully connected network with one hidden layer (adapted from François-Lavet et al. (2018)).

The first layer consists of all the input values often referred to as *the features*, with the size $n_x \in \mathbb{N}$. In the next layer a set of neurons with size $n_h \in \mathbb{N}$ linearly transforms the input values by multiplica-

tion with a weight or parameter matrix W_1 (size: $n_h \times n_x$) and a bias term b_1 with size n_h is added. The outputs of the hidden layer h with size $n_y \in \mathbb{N}$ are obtained by multiplication with a non-linear activation function A :

$$h = A\left(W_1 \cdot x + b^{(1)}\right) \tag{1.10}$$

$$h = A\left(\begin{bmatrix} \theta_{0,0} & \theta_{0,1} & \dots & \theta_{0,x} \\ \theta_{1,0} & \theta_{1,1} & \dots & \theta_{1,x} \\ \vdots & \vdots & \ddots & \vdots \\ \theta_{h,0} & \theta_{h,1} & \dots & \theta_{h,x} \end{bmatrix} \begin{bmatrix} x_0 \\ x_1 \\ \vdots \\ x_h \end{bmatrix} + \begin{bmatrix} b_0^{(1)} \\ b_1^{(1)} \\ \vdots \\ b_h^{(1)} \end{bmatrix}\right) \tag{1.11}$$

Finally, in the output layer, the last transformation is performed to obtain the output values y with W_2 having size $n_y \times n_h$ and b^2 size n_y:

$$y = W_2 \cdot h + b^{(2)} \tag{1.12}$$

In order to train this network, we initialize the weights, and start feeding the inputs through the network. The weights are tuned using back-propagation. A loss- or cost-function $L(\theta)$ is created which measures the error which the difference in the observed output, and the true value. This represent how good the neural network predicted the value given the input. In order to minimize the error, gradient descent is used to update the model's weights:

$$\theta \leftarrow \theta - \alpha \nabla_\theta L(\theta) \tag{1.13}$$

Here, α is the learning rate of the weights and defines how violently the weights are updated. If the learning rate is low the learning process will be slow; if the learning rate is high the learning becomes faster but can become unstable.

As discussed earlier in this section, the activation function is a non-linear function mapping which introduced a level of abstraction. Without the activation function, the mappings in the network are completely linear as the network becomes a simple linear regression model. Activation functions 'activate' neurons if a certain threshold is met, consequently introducing non-linearity and the possibility to model more complex behaviours. In Figure 1.6, a selection of activation functions is shown. Every function has both advantages and disadvantages; however, an in-depth analysis of the differences and their implications is out of the scope of this research.

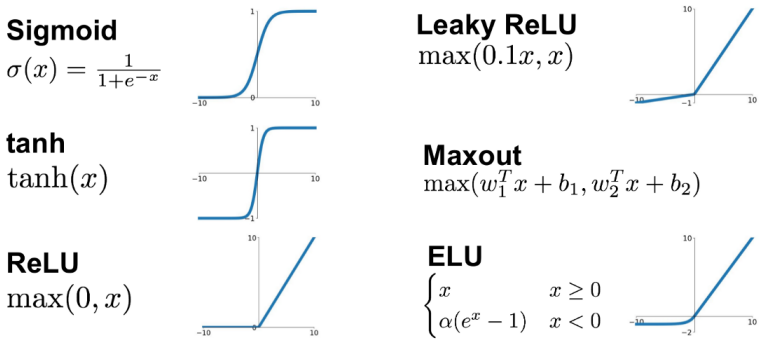


Figure 1.6: Cheat sheet of the most widely used activation functions (Li et al.).

The implementation of a neural network in a reinforcement learning environment has been applied in the 1990s by Tesauro (1995). However, the initial scientific enthusiasm was lost as the method could not be upscaled to larger and more complex problems due to instability in the learning process, high rewards, and auto-correlation of the observations (Casas, 2017). However, the

revival of the research field came with the development of Deep Q-network model of Mnih et al. (2015).

1.5. Deep Q-network

The Deep Q-network (DQN) combines all of the concepts described previously. A neural network is fitted to approximate the Q-function, by parametrization of the Q-values: $Q(s,a;\theta_i)$. Similar to Equation 1.8, the target of each iteration k can be defined as:

$$Y_{target} = r + \gamma \max_{a'} Q\left(s', a'; \theta_i\right) \tag{1.14}$$

To train the network the parameters are tuned using the loss function and a replay buffer. Every episode, the network stores the experiences $< s, a, s', r >$ in a replay buffer $\mathcal{M}$. When training the network, a mini-batch of experiences is uniformly drawn from the replay buffer ($U(\mathcal{M})$). Because the experiences of a trajectory of the MDP hold a large dependence on each other, training on these experiences introduces variance and instability (François-Lavet et al., 2018). By uniformly sampling over all previous experiences stored in the replay buffer, the mini-batch of experiences becomes an independent and identically distributed set increasing the stability of the learning process. The parameters are updated through stochastic gradient descent of the loss function. The loss function compares the expected value of the network to the true value, and is defined as a minimization of the squared loss:

$$L_i\left(\theta_i\right) = \mathbb{E}_{<s,a,s',r>\sim U(\mathcal{M})}\left(\underbrace{r + \gamma \max_{a'} Q\left(s', a'; \theta_i^-\right)}_{\text{target}} - Q\left(s, a; \theta_i\right)\right)^2 \tag{1.15}$$

The attentive reader has noticed that the parameter θ_i of the target in loss function has changed to θ_i^-. The DQN keeps a separate target network to generate the target values in order to solve the 'running target' problem. If a single network is used to predict the target values and update the weights, oscillations in the policy destabilize the learning process (Mnih et al., 2015). Every episode the *policy network's* parameters are updated using the experiences from the memory buffer. Every C iterations, the *target network* gets the updated weights from the learning network. Figure 1.7 shows the architecture of the DQN training process in a reinforcement learning set-up.

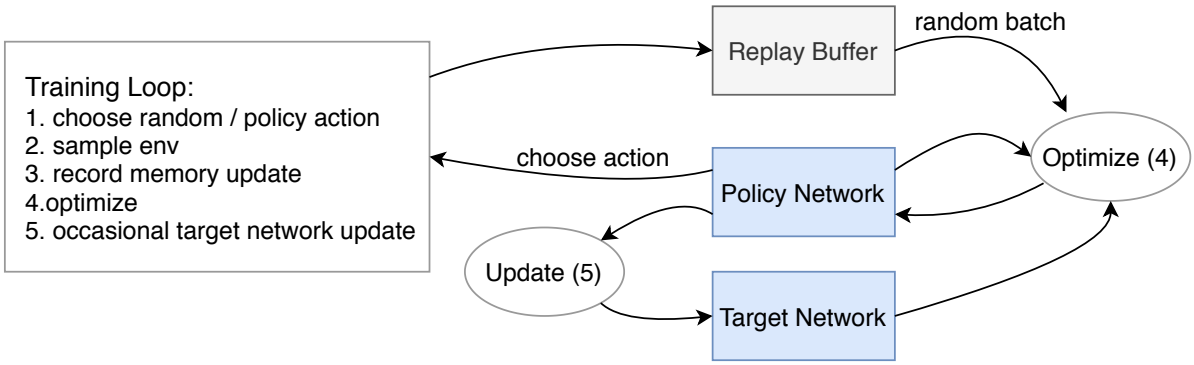


Figure 1.7: Diagram illustrating of the learning process of the Reinforcement Learning agent (adapted from Pytorch (2020)).

In the years after the original paper of Mnih et al. (2015), multiple researchers have contributed to the development of DQN with extensions improving the learning rate and stability. van Hasselt

et al. (2015) introduced Double DQN as a solution to the overestimation of value-actions in DQN. Schaul et al. (2016) improved DQN by rating the experiences used to train the network. Valuable experiences are sampled more frequently to increase the learning rate. Wang et al. (2016) introduced Dueling DQN (DDQN), which incorporates an actor-critic network. And finally, Hessel et al. (2018) combined all of the innovations above into a single model.

The extensions bring extra complexity to the learning and the understanding of the interaction between the artificial fleet environment and the agent. Therefore, the basic DQN model by Mnih et al. (2015) is used to learn the fleet policy of the fleet planning problem.

1.6. DQN in the Fleet Planning Problem

In this section, a more in-depth analysis is given of the application of a DQN model to solve the fleet planning problem. The fleet planning problem revolves around the question of how many, which type, and when to acquire or dispose aircraft, which is formalized as the action space of the MDP. Multiple external factors influence the fleet decisions, which are represented as the states of the MDP. In this research, the air-travel demand is assumed to be the only uncertain variable and sampled for every RL loop using the adapted Ornstein-Uhlenbeck process. The state transition is fully separated from the optimisation process, and thus not available to the agent. The final element of the MDP tuple (or experience) is the reward, which is an evaluation of the action given the current state, represented by the reward function. The DQN-model serves as the decision-maker in the fleet problem and is trained using the fleet planning experiences generated through observing the agent-environment interactions. The neural network approximates the state-action values representing the expected future profit is a fleet decision is taken given the state of the airline and demand in the network. In essence, the trained agent replaces the optimisation algorithm for fleet planning in an end to end learning set-up. To evaluate the fleet decisions the reward is generated by assessing the fleet planning problem given the revelation of the air-travel demand, which is discussed in chapter 3.

The remainder of this chapter elaborates on the specific formulation of the feature selection and action space, the normalization of input values, and the action space of the fleet planning problem.

1.6.1. State space

Feature Selection

The input parameters, features, or state, is the information given to the network which is transformed through the neural network to an output vector which represents a value prediction of the state. The neural network is trained to extract relevant information from the states to predict the most valuable fleet action. Consequently, the choice and design of the features are of uttermost importance to predict the optimal fleet decisions.

Based on the relevant variable needed to optimize a Fleet Planning Model (FPM) a set of features is generated which is tested using a Meta-learning model. In table 1.1, an overview of the tested features is depicted. It is important to note that the size of every feature is dependent on the number of aircraft types considered and the size of the airline network. The features *Fleet*, *Step*, *Demand_route*, and *Demand_route_total* are unprocessed features which are directly related to the environment. The remainder of the features are variables conceived from the Fleet Assignment Model (FAM) of the previous state (the FAM is discussed in Chapter 3). As a result, they give more in-depth information about the fleet planning problem; however, the FAM must be run previously at every step to generate the features.

Testing all possible combinations of features is a computationally expensive task. To speed up the feature selection process, a Genetic algorithm (GA) is employed to assess combinations of fea-

Fleet	The amount of aircraft of each type k at time t
Demand_route	The demand at each route n at time t
Demand_route_total	The cumulative demand at time t
Demand_slack	Returns the slack of the demand constraint. It represents the amount of demand unsatisfied at each route n at time t
Demand_slack_total	Returns the cumulative slack of the demand constraint. It represents the amount of demand unsatisfied at each route n at time t
Step	The time step of the state
Available_seats	The amount of seats available on each route n at time t
Unsatisfied_demand_ave	Average demand of previous period minus the available seats on each route n at time t on each route at time t
Unsatisfied_demand	The demand of previous period minus the available seats on each route n at time t
Duals_capacity	Dual variables of the relaxed LP of the capacity constraint at time t (how much increase in profit if amount of aircraft increases)
Duals_pass_flow	Dual variables of the relaxed LP of the passenger flow at time t (how much increase in profit if available seat on routes increases)

Table 1.1: List of features tested with explanation.

tures. The GA algorithm uses selection process similar to a natural selection process (Davis, 1991). A random combination of features is sampled and tested. The best performing feature combinations are saved and new feature combinations are formed by 'breeding' good performing features. In order to incentivize exploration, sometimes features combinations are 'mutated', e.g. a random selected feature replaces a good performing feature. This process is repeated multiple times, and every iteration is called a 'generation'.

In Table 1.2, the results of the feature testing are shown. To speed up the process, and network of four airports and time horizon of five periods is considered. It is assumed that the optimal feature combination is independent of the network size, and the same combination of features can be used in larger networks. Five generations of ten feature combinations are tested, resulting in 50 tested feature combinations. In the Table 1.2 all feature combinations with a testing score above 94% are shown. The last two rows show respectively the number of occurrences of this feature is a good performing combination and the percentage of the occurrences w.r.t. the total amount of high performing feature combinations.

It is clearly visible that four features stand out: *demand_route, duals_capacity, fleet, and step*. The latter two features are very small ones and require only a minimal amount of input neurons. However, demand_route and duals_capacity are also reasonably small for small networks; however, the input size increases drastically with the size of the network. As a result, only one feature employing information about the demand is imported. Because demand_route occurs more and is the highest performing combination with fleet and step, the combination of those three features is selected as the input vector. Moreover, the preprocessed features require the model to run the FAM to obtain the feature values at every iteration, increases the computational effort and complexity.

Data Normalization:
One of the most vital steps in machine and deep learning is data normalization. Because the NN is fully connected, each input value will pass through every neuron in the next layer. If the input scale between input neurons is large, one input neuron's change in values will have a larger influence on the NN as the other neuron. Consequently, the loss function and optimisation will be focused

Available seats	demand_route	demand_route_total	demand_slack	demand_slack_total	duals_capacity	duals_pass_flow	fleet	step	unsatisfied_demand	unsatisfied_demand_ave	
x					x			x			**94.62%**
	x				x			x			**94.24%**
							x	x	x		**94.66%**
x					x			x			**94.56%**
	x				x			x			**94.68%**
	x		x				x				**94.04%**
					x			x	x		**94.15%**
	x				x			x			**95.1%**
	x		x				x				**94.6%**
	x		x				x				**94.37%**
	x				x		x				**94.75%**
			x		x		x				**94.31%**
			x			x		x			**94.06%**
	x				x		x				**94.18%**
	x						x	x			**95.45%**
	x						x	x			**94.69%**
	x				x		x				**94.99%**
	x		x				x				**94.22%**
			x				x	x			**94.56%**
x	x						x				**95.12%**
3	**13**	**0**	**7**	**0**	**10**	**1**	**13**	**11**	**2**	**0**	
15%	**65%**	**00%**	**35%**	**00%**	**50%**	**05%**	**65%**	**55%**	**10%**	**00%**	

Table 1.2: High performing feature combinations from the Genetic Algorithm optimisation

around minimizing the error of the large input instead of the combination of those inputs. In order to avoid the bias towards features with high or low values, all input values are normalized using Equation 1.16.

$$x' = \frac{x - \bar{x}}{\sigma} \tag{1.16}$$

$$\text{with:} \quad \bar{x} = \frac{1}{N}\sum_{i=1}^{N} x_i$$

$$\sigma = \sqrt{\frac{1}{N-1}\sum_{i=1}^{N}(x_i - \bar{x})^2}$$

Here, $\bar{x}$ is the average and σ is the standard deviation of the all the input values $x_1, ..., x_N$ for a single neuron. However, because the RL process is a learning process which explores the state-action state iteratively, the input variables are unknown at initializations of the model. A solution comes with the greedy algorithm and replay buffer. As discussed earlier, at the beginning of the learning process the exploration of the model is very high. The agent will explore the state-action space by choosing random actions to fill up the buffer and start learning on an initial randomly distributed

state-action experiences. This randomly distributed set of experiences offer an opportunity to calculate the average and standard deviation of every input neuron. These normalization parameters are calculated after the size of the replay buffer exceeds a preset threshold, and are used throughout the training process in order to keep the policy from oscillating.

1.6.2. Action Space

The neural network maps the input states to a column-vector representing the predicted Q-values of the fleet decisions. Every neuron corresponds to a single fleet decision, and given the greedy algorithm, the neuron with the highest values represents the action with the highest expected profit given the state.

Let's define the decision vector a_t which holds the acquirement or disposal aircraft of each aircraft type k in period t, and is represented as:

$$a_t = (ac^t_{acq}, ac^t_{dis}) \tag{1.17}$$

With $ac^t_{acq} = \left[ac^{t,k}_{dis}\right]_{k\in\mathcal{K}}$, $ac^t_{dis} = \left[ac^{t,k}_{acq}\right]_{k\in\mathcal{K}}$. Every possible decision vector is therefor represented by an element in the output space $a_t \in \mathcal{A}$. To constraint the size of the action space $\mathcal{A}$, the assumption is made that for each aircraft type k only a single amount can be either acquired or disposed. For example, $a_t = ([1,0],[0,3])$ translates to the acquirement of 1 aircraft of the first type and the disposal of 3 aircraft of the second type. This strategy enables us to acquire or dispose multiple aircraft of multiple types at every time-step. Assuming a maximum amount of aircraft acquired/disposed f^k_{max}, the size of the action space A grows exponentially with an increasing amount of aircraft types k and the maximum number of aircraft acquired or disposed per aircraft type f^k_{max}:

$$A = \prod_{k=1}^{K} \left(2 \cdot f^k_{max} + 1\right) \tag{1.18}$$

The exponential growth of the action space poses a problem as it increases the action-space drastically which causes problems in value-based algorithms such as DQN. To decrease the action space and still predicts meaningful fleet decisions it is assumed only one fleet decision per aircraft type per time-step can be performed. Moreover, in the commercial airline business, fleet additions or disposals are often single aircraft type contracts and rarely entail deals acquiring multiple types of aircraft. The new action space grows linearly with the number of types of aircraft considered and the maximum number of aircraft acquired or disposed per aircraft type f^k_{max}:

$$A = (2 \cdot f^k_{max} \cdot K) + 1 \tag{1.19}$$

$$\mathcal{A} = \left\{0, [-F^k, F^k]_{k\in\mathcal{K}}\right\} \tag{1.20}$$

Where $F^k = [1, ..., f^k_{max}]$, and each action $a_t \in \mathcal{A}$ is mapping of the input states. To illustrate, an action space for two aircraft types (BOE738 and BOE753), $f^k_{max} = 3$, and with the linearly growing fleet decision set would yield the following action space:

$$\mathcal{A} = \left\{0, \underbrace{-1,-2,-3,+1,+2,+3}_{\text{BOE738}}, \underbrace{-1,-2,-3,+1,+2,+3}_{\text{BOE753}}\right\} \tag{1.21}$$

2

Ornstein-Uhlenbeck Forecasting Model

This chapter explains the working principles of the stochastic air-travel demand forecasting model. At the start of each episode e, the air-travel demand is sampled for every Origin-Destination (OD). The air-travel demand can be interpreted as the total amount of passengers who are willing to travel between the origin and destination airport for the average airfare. With the expected air-travel demand, an airline can optimise the fleet planning and ensure efficient future operations. Consequently, the expected demand is considered the most import uncertainty in the fleet planning process.

Forecasting of variables can be performed in various methods. A first method could be a random sampling of values. This method is often referred to as a random walk method. At every period t, it is assumed that the next predicted value holds no relation to the historical evolution of previous values. Being completely independent of the fluctuations of the historic values creates a completely random sampling. However, in this research, it is assumed that the evolution of the historic air-travel demand holds information to the prediction of the future demand in the form of an autoregression model.

Autoregression models assume that a trend is visible in the historic data which can be extrapolated to predict future values. One of the most common models is linear regression. Here, the coefficients of a linear function are fitted with the historic values by minimization of the residuals and used to predict future values. However, using simple linear regression yields a deterministic approach, where the simple regression model assumes that the current demand trend-line will be maintained in future years, which is ignores sudden disruptive changes in the market. In the past, events such as the introduction of low-cost carriers, terrorist attacks, economic growth/decline have proven to be influential on the air-travel demand. To simulate these disruptive events in the future, a stochastic element needs to be incorporated in the sampling of air-travel demand, thus a different method is proposed: The mean-reverting Ornstein-Uhlenbeck process.

In the first section, the mean-reverting Ornstein-Uhlenbeck process as demand growth forecasting model is introduced. In the second section, the adaptive demand forecasting model used is outlined. Finally, in section three, the application of the adaptive demand forecasting model in the fleet planning problem is discussed.

2.1. The Mean-reverting Ornstein-Uhlenbeck Process

The mean-reverting process, or regression to the mean, has roots in various research fields. The concept dates back to the late 1800s first established by Sir Francis Galton. He argued in a statistical series, that a very extreme outcome is more likely to be followed by a moderate one and will eventually revert to the mean value. This phenomenon has been observed in multiple research fields and

is often unawarely present in our daily lives. A prime example is the "hot hand fallacy", where a sudden increase or decrease of an outcome is often quickly assumed to be the new normal. However, future outcomes tend to revert to the mean value over time.

The mean-reverting has been used in modelling variables which tend to be cyclical while still holding their stochastic nature. It has been successfully applied in financial processes such as stock, commodities, and option prices, but more importantly, it has been used in air-travel demand prediction by Sa et al. (2019). The premise of his work has formed the basis of the demand representation in this thesis, so the resemblance will therefor be present. Sa et al. (2019) argue that the success of modelling financial product has been significant due to its correlation to the gross domestic product (GDP). In the airline business too, a strong correlation between GDP and air travel demand is visible (Pearce, 2014). Since both stock prices and air travel demand hold GDP as a correlating factor, the expectation that the mean-reverting process can be used in forecasting air travel demand is a promising method (Sa et al., 2019).

Mathematical Formulation:
Sa et al. (2019) outline the importance of first establishing which demand parameter should be considered in the reverting process: historical passenger data, historical passenger growth data, historical revenue-passenger-mile (RPM) data or historical RPM growth data. In a sensitivity analysis, it was concluded that historical passenger growth data holds the best fit and is thus chosen to model the reverting process.

The Ornstein Uhlenbeck process at every OD can be represented independently by a stochastic differential equation:

$$dx_t = \lambda\left(\mu - x_t\right)dt + \sigma dW_t \tag{2.1}$$

Discretizing the differential equation between t and $t+1$ yields:

$$x_{t+1} = x_t + \lambda\left(\mu - x_t\right) + \sigma dW_t \tag{2.2}$$

Equation 2.2 is often called the Vasicek model in finance describing the evolution of interest rates. The model describes x_{t+1} which is the next predicted demand growth between time t and $t+1$. x_t represents the growth between $t-1$ and t. The term $\lambda\left(\mu - x_t\right)$ describes the dynamics of the mean-reversion model and is often referred to as the drift term. Sa et al. (2019) compares the mean reversion to a physical spring-mass where the equilibrium represents the mean passenger growth. When the spring is stretched it wants to return to the equilibrium. In the mean-reversion process, the drift term mimics this behaviour and corrects a deviation from the mean towards the long-term mean growth. The μ parameter is the 'long-term mean growth rate' and can be interpreted as the mean value which the model will approach throughout time. The λ parameter is called the 'speed of reversion' and is characterized as the amount of time it takes to revert back towards the mean growth.

The last term is dW_t, and is called the Wiener process. The wiener process represents a simple Random Walks process and introduces the stochastic nature. Here, the increment $dW = W_{t+1} - W_t$ has the Normal$(0,1)$ distribution often called the 'shock term'. By multiplying this random stochastic number to the standard deviation deducted from the historic data, a disruptive growth of demand similar to the error deducted from the calculation of the drift term parameters can be introduced in the predicted values. The term σ can be interpreted as the volatility of the change in the growth of demand.

Calculating the parameters:
Before sampling of the demand values can start, the Ornstein-Uhlenbeck's parameters need to be

calculated. This process is only performed once as these parameters remain fixed throughout the sampling of demand. To deduct the process' parameters from the historic data, the linear relationship between historic yearly demand growth x_t and the change in yearly demand growth x_{t+1} is exploited in the form of a linear regression fitting of the historic demand growth data. Let's represent the mean-reversion equation as a simple linear function:

$$y = ax + b + \epsilon \tag{2.3}$$

$$\begin{aligned} \text{with:} \quad & y = x_{t+1} - x_t \\ & x = x_t \\ & a = \lambda\mu \\ & b = -\lambda \end{aligned}$$

Fitting the linear regression to the historic data x, y reveals the regression coefficients for the slope b and the intercept a of the fitted data. After some rewriting, the drift term parameters can be described in Equation 2.4 and 2.5.

$$\lambda = -b \tag{2.4}$$

$$\mu = -\frac{a}{b} = \frac{a}{\lambda} \tag{2.5}$$

Finally, the standard deviation error of the regression model can be calculated. This can be interpreted as the historic error between the drift term parameters and the true historic values. The standard deviation is calculated by taking the root mean square of the summation of the difference between the predicted change in demand growth and the actual change in demand growth:

$$\sigma = \sqrt{\frac{\sum(y - \hat{y})^2}{N - 1}} \tag{2.6}$$

$$\begin{aligned} \text{with:} \quad & \epsilon = y - \hat{y} \\ & \hat{y} = a + bx \\ & y = a + bx + \epsilon \end{aligned}$$

In figure 2.1 an example of an estimation process of the Ornstein-Uhlenbeck parameters is demonstrated between the airport SFO and ORD. The first plot shows the historic yearly demand values from 1990-2014 on the OD leg as a cumulative sum of all monthly transported passengers in that year. So, 2014 shows all passengers from January to December 2014. The second plot shows the passenger demand growth x_t as the percentage difference between two consecutive years. Once this is found, the change in demand growth can be calculated as the final parameter to the Ornstein-Uhlenbeck process. The final plot shows the demand growth data versus the change in demand growth data. The linear regression is displayed using the least-squares method, and the fitted line shows the result of the regression parameters.

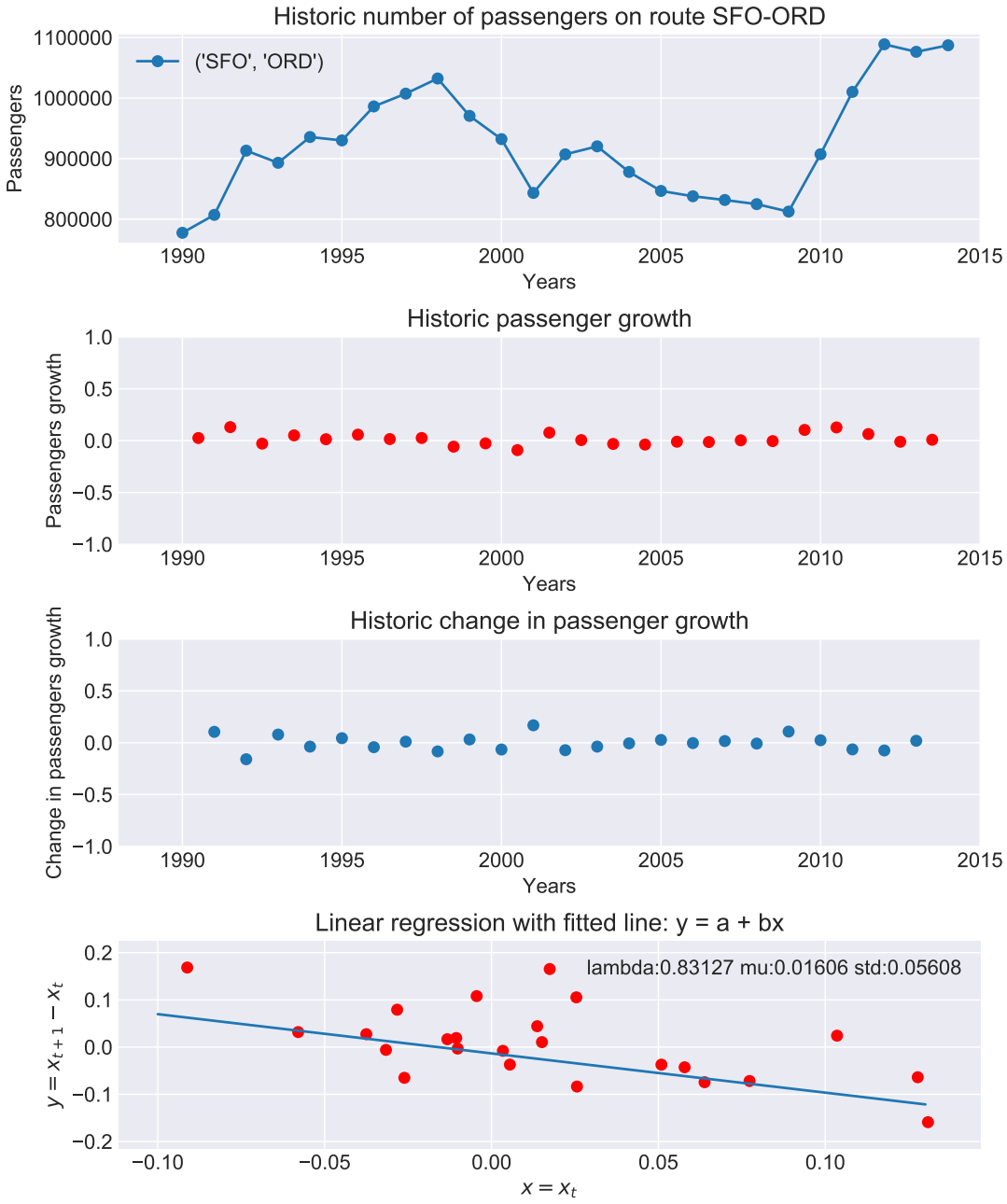


Figure 2.1: Mean-reversion model parameter calculation through linear regression.

2.2. Adapted Demand Forecasting Model

In the work of Sa et al. (2019), the parameters of the Ornstein-Uhlenbeck process are estimated using the historical evolution of the route-specific demand. For each route, the parameters are calculated, and demand trajectories are sampled to construct transition probability matrices and create a scenario generation model. However, the premise that passenger growth in route pairs grow independent of each other is incorrect and results in an unrealistic sampling process. In Figure 2.2 a result of such a sampling strategy for a ten airport network is visible. For every route, the Ornstein-Uhlenbeck parameters are estimated and used to sample 20 independent demand revelations. Figure 2.2 depicts the historical cumulative transported passengers and the cumulative predicted demand in the 10-airport network using the Ornstein-Uhlenbeck implementation of Sa

et al. (2019) .

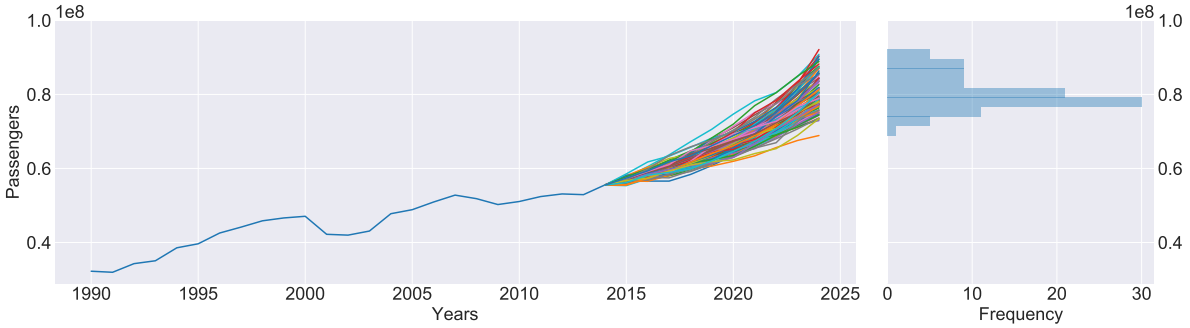


Figure 2.2: Example of the historical cumulative transported passengers with 50 cumulative sampled demand trajectories in the 10-airport network using independent routes predictions (left); Histogram of the cumulative predicted demand values at 2024 (right).

The sampled demand revelations all behave similarly to a quasi-linear regression model with a little spread and stochastic behaviour. The behaviour could be explained due to the fact that each year, the demand growth is sampled for every route, including the shock term which influence is determined by the sampling of the normal distribution. Because the shock term's normal distribution centred around zero, sampling the multiple demand growth for multiple routes results in an average shock across the network converging to zero, and behaving like a simple linear regression. In order to represent the demand revelation as closely as possible to the real physical process, some additional assumptions are made and a new adapted Ornstein-Uhlenbeck forecasting model is created.

To begin, it is assumed that an origin-destination pair or route is fully dependent on the destination-origin pair, and the two airports will from now on be referred to as a *market*. Investigation of the market data in the United States shows that market pairs' transported passengers are almost completely similar because the most common airline flights and tickets are round trips. Consequently, the Ornstein-Uhlenbeck parameters and sampling of the demand growth are calculated based on the average historic market demand.

Secondly, it is assumed that a demand trend prevails on two levels: a network level, which influences all markets, and the market independent trend only influencing itself. As shown previously, in a network of independent markets, due to the law of large numbers, the stochasticity in the network decreases as the amount of airports increases. Moreover, it is safe to assume that higher level (global) trends exist in air-travel demand. Increased fuel prices or taxes can suddenly increase the airfare and thus decrease the overall air-travel demand. To incorporate this behaviour, a network trend is estimated over the airline network, in addition to estimating the parameters on a market level.

Thirdly, contrary to the original Ornstein-Uhlenbeck sampling process, it is assumed that the long-term mean growth rate is a normal distribution instead of a constant value. It represents the uncertainty that the current mean value will prevail in the future, and replaces it as a probability distribution of the long-term mean growth.

Using the cumulative transported passengers over the network, the network's Ornstein-Uhlenbeck parameters λ, μ, σ can be estimated. The same process is repeated for Ornstein-Uhlenbeck parameters $\lambda_m, \mu_m, \sigma_m$ for every market m based on the market's demand history.

Two mean-reversion equations are introduced in Equation 2.7 and 2.8. Both equations are similar to the original mean-reversion equations, although with some minor alterations. In both the equations, next to the already stochastic shock term, the long-term growth mean parameter is re-

placed by a normal distribution to simulate the dispersion of long-term growth. Before sampling the market demand trajectories, the cumulative network growth trend δ_{t+1} is sampled using Equation 2.7. This is the network growth which will influence all the market demands.

Next, for market m, the demand growth $x_{m,t+1}$ is sampled using Equation 2.8, which is an average of the network growth trend δ_{t+1} and the market dependent trend. By averaging, it is assumed that the influence of the network growth is equal to the predicted growth of the market.

$$\delta_{t+1} = \delta_t + \lambda\left(\mu' - \delta_t\right) + \sigma dW_t) \tag{2.7}$$

$$x_{m,t+1} = \frac{1}{2}(\delta_{t+1} + x_{m,t} + \lambda_m\left(\mu'_m - x_{m,t}\right) + \sigma_m dW_{m,t}) \tag{2.8}$$

$$\begin{aligned} \text{where} \quad & \mu' \sim \mathcal{N}(\mu, \sigma^2_\mu) \\ & \mu'_m \sim \mathcal{N}(\mu_m, \sigma^2_{m,\mu}) \end{aligned} \tag{2.9}$$

The shock terms dW_t and $dW_{m,t}$ for respectively the network and market, are sampled for every period t in every demand trajectory; on the contrary, the long-term mean growth μ' and μ'_m ,depicted in Equation 2.9, are sampled once for every demand trajectory. The variance of the shock terms, are defined by the Wiener process and are in principle equal to one. The μ and μ_m variables represent respectively the estimated long-term mean growth for the network, and for every market m. The σ^2_μ and $\sigma^2_{m,\mu}$ are the variance terms of the long-term mean growth of respectively the network and markets s. The latter variance terms represent how much dispersion is present in the mean growth over time. However, for the network and every market, only one historical long-term demand is present; hence, it is impossible to calculate the variance. The result is that a long-term mean growth variance is assumed to be equal to 0.005 for both the network and markets. In the article in part I, three demand scenario's are created to test the variance parameters.

Now, the next air-travel demand can be predicted using Equation 2.10. Because every market exists for two routes the same growth trajectory is used. d_{t+1} is the future demand value at period $t+1$ for a single route, d_t is the demand value of the current period t and x_{t+1} the predicted growth in demand from t to $t+1$.

$$d_{t+1} = d_t \cdot (1 + x_{t+1}) \tag{2.10}$$

Where $d_{t+1} = \left[d_{n,t+1}\right]_{n \in N}$ is a vector of the air travel demands sampled for every route n at period t.

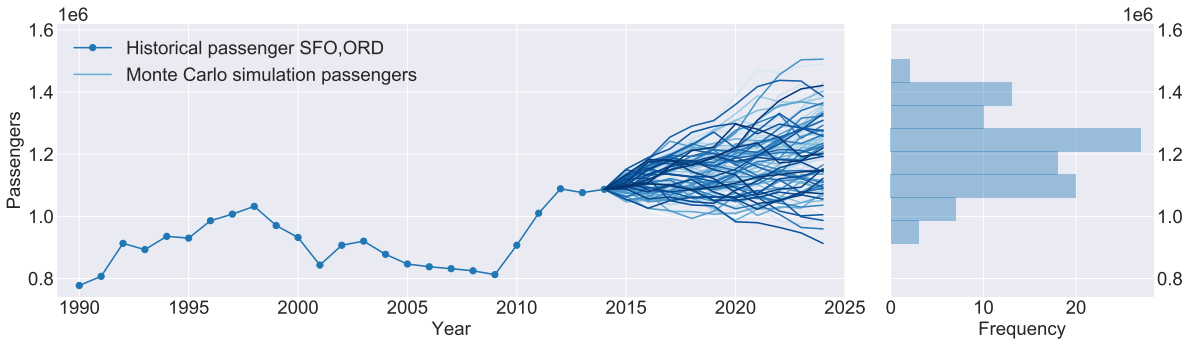


Figure 2.3: Example of historic market demand values with 50 sampled demand trajectories for the route SFO-ORD (left); Histogram of the predicted demand values at 2024 (right). These sample trajectories are for the Average Demand scenario (Part I).

Figure 2.3 (on the left) shows an example of the demand prediction above. From 1990-2014 the yearly historic demand values are shown. After 2014, 50 lines of future demands are sampled using the Ornstein-Uhlenbeck process for the route SFO-ORD. The histogram on the right represents 50 simulations of the predicted demand in the year 2024. Distribution of the predicted demand values shows a normally distributed behaviour around the 1.2E6 passengers per year. It shows that the Ornstein-Uhlenbeck process is able to sample demands where distribution is centred around the expected linear regression of the historic demand values, but with a normal distributed stochastic nature.

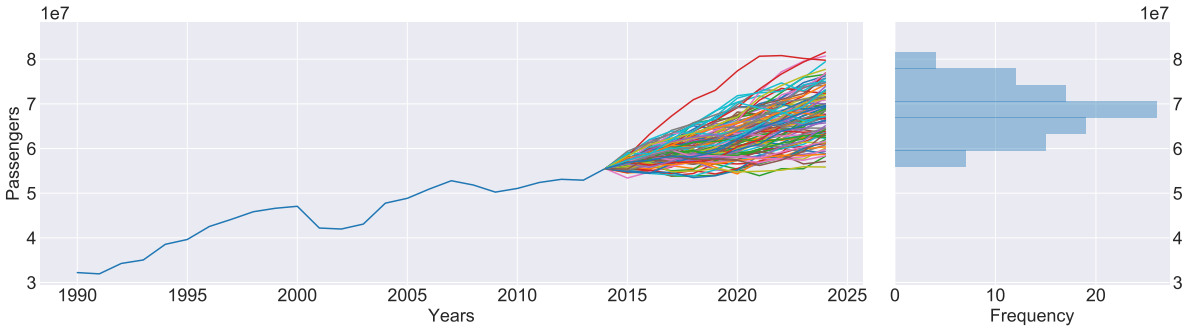


Figure 2.4: Example of historic network demand values with 50 cumulative sampled demand trajectories for network (left); Histogram of the cumulative predicted demand values at 2024 (right). These sample trajectories are for the Average Demand scenario (Part I).

In Figure 2.4 the cumulative predicted air-travel demand of the semi-dependent markets is depicted. It is clearly visible that compared to Figure 2.2, the dispersion of the final air-travel demand values is much wider, and disruptive events on a network level are apparent. Still, due to the law of large numbers, part of the stochasticity is lost on a network level. By changing the standard deviation of the network and the markets, the stochasticity of the air travel demand may be controlled. A greater distribution results in a greater representation of disruptive events.

2.3. Application in the Fleet Planning Problem

The result of the adapted Ornstein-Uhlenbeck forecasting model is an air-travel demand simulator which samples unique yearly stochastic trajectories. However, in the fleet planning problem, the uncertain variable is the demand growth from period t to $t+1$. The time-horizon discretization of the fleet planning problem does not correspond to the discretization of the demand simulation. For example, in the case study, a fleet decision is taken every n_{years} years, but the demand simulation is discretized in yearly demand growths. Let's assume $\Delta_t = [\Delta_{m,t}]_{m \epsilon \mathcal{M}}$ is a vector comprising the growth of demand of every market. Where the time-horizon of the fleet planning problem is discretized in T periods, the time-horizon of the demand sampling model $Y = n_{years} \cdot (T+1)$ years.

$$\Delta_{m,t} = \prod_{i=1}^{n_{years}} \left(1 + x_{m,y+i}\right) \qquad \text{where} \quad y = t \cdot n_{years} \tag{2.11}$$

Equation 2.11 shows the sampled demand growth values $\Delta_{m,t}$ in every period and route, as a product of all the sampled yearly growths in that period. Finally, the sampled demand matrix D_N is constructed. The process of predicting the demand growths using Equation 2.11 for T periods, and Equation 2.12 depicts the adjusted demand forecast $d_{n,t+1}$ as a multiplication of the current demand in each route times the predicted demand growth of the corresponding market m. The

result is a demand matrix with N routes as rows and $T+1$ periods as columns depicted in Equation 2.13.

$$d_{n,t+1} = d_{m,t} \cdot \Delta_{m,t+1} \tag{2.12}$$

$$D_N = \begin{bmatrix} d_{0,0} & \cdots & d_{0,T+1} \\ \vdots & \ddots & \vdots \\ d_{N,0} & \cdots & d_{N,T+1} \end{bmatrix} \tag{2.13}$$

The sampling process is repeated at the start of every new episode e. The demand matrix D_N is available to the environment, and used to calculate the optimal fleet decision (moreover in Chapter 3). Every iteration t, the next demand vector for $t+1$ are revealed to the agent until the time-horizon is reached.

3

Environment Model

Model-free reinforcement learning generates a policy through iteratively training an agent by interacting an environment. The main propose of the environment is to evaluate the action chosen by the agent in the form of a reward. The reward is then used by the agent to tune the neural network parameters and to optimize the fleet policy. The environment also provides the state transition from state s to the next state s_{t+1} with the revealing of the uncertain parameters.

In the first section of this chapter, the context of the interaction between the reinforcement learning model and the environment will be further outlined and discussed. That will be followed by the depiction of the Fleet Planning and Fleet Assignment Models that evaluate the fleet decisions; an explanation of the reward function generation; and an analysis of the implications that the size of the LP-matrix has on the computational times and reward accuracy.

3.1. Simulated Experience

Reinforcement Learning (RL) has proven itself to be a reliable estimator of policies for different strategic planning and control environments (Mnih et al., 2015). The power of RL models lies in their ability to learn the complex dynamics of an environment by interacting with it. For example, when learning to play a video game, the agent perceives the pixels of the screen, act on the controls of the game, and receives rewards from the video game often in the form of a score. The RL model iteratively plays the video-game millions of times in order to learn the optimal policy which maximizes the score.

It goes without saying that in the case of a fleet planning problem the environment generating the reward and state transition cannot be learned from real experiences. Indeed, every airline has one fleet plan for a distinctive network it operates in. Converting this into a supervised dataset would result in a single episode of experiences which the RL would copy exactly every time and thus would not be able to adapt over time. As a result, simulated experiences are needed to learn from. Luckily, RL has already been proven to be a powerful tool in learning from simulated experiences (Sutton and Barto, 2018). Opposed to model-based methods, where a complete probability distribution of all state transitions is needed, model-free RL allows learning from a sampling of transitions.

The (simulated) experience is an MDP tuple and consists of four parts $< s, a, r, s' >$ which are saved every period of every episode in the replay buffer of the agent. As discussed in Chapter 2 the air-travel demands transition from state s to the next state s' is sampled using the mean-reverting Ornstein-Uhlenbeck process and requires no further attention in this chapter, as the action a follows the agent's prediction discussed in Chapter 1. The only variable which requires attention is the reward r and the accompanied reward function $R(s_t, a_t, s_{t+1})$. The reward function evaluates

has good the fleet decision a_t was given the state, which includes the current and next sampled air-travel demand revelation.

First, it must be defined what a good/bad fleet decision is. As pointed out earlier, the fleet of an airline has a large effect on future airline operations, the network, and financial health. A fleet must allow for versatility and flexibility in the network the airline wants to operate. A fleet must represent the airlines business model (low/legacy carrier) in terms of aircraft type, size, and interior configuration. Moreover, the size of the fleet determines the number of passengers that possibly can be transported and the market share that can be captured. All of the requirements above can be boiled down to one overarching goal: maximizing profit. Subsequently, let's assume that maximization of the future profit is the goal, and the reward is a measurement of this goal.

The fleet action determined by the agent together with the simulated air-travel demand (which is revealed in the environment model) allows for the calculation of the future generated profit using a Fleet Assignment Model (FAM). The FAM optimizes the weekly flight frequency of aircraft types operating between airports in order to maximise the profit. In other words, the model develops a weekly schedule of aircraft types between airports which generates the highest amount of profit. Other methods such as the assignment of specific aircraft (tail-number) on routes give a more detailed representation of the profit. The tail-assignment method may help with the misrepresentation of reality that the consolidation of tail number and aircraft types can produce through a less refined schedule; however, its computational time is too high and therefor disregarded.

Now, the FAM generates an estimation of the achievable profit of the airline with the fleet decision of the agent model. Unfortunately, the height of the profit is invaluable without a frame of reference. Due to the sampling of air-travel demand trajectories, variations of the available demand results in maximum achievable profits varying over the episodes; a frame of reference is needed defining how good or bad the action's profit is in respect to optimal profit, which yields from the optimal fleet decision. A solution could be to calculate all the hypothetical profits for all possible fleet decisions and define the reward based on the range of calculated profits. However, calculating all the hypothetical profits is a computationally expensive job. Moreover, the FAM is optimized to generate the highest profit regardless of future operations. As a result, fleet decisions which are disadvantageous to the related period, but are beneficial for future operations are not incentivized. Consequently, a multi-period Fleet Planning Model (FPM) optimisation is introduced. The FPM is similar to the FAM however, the model spans multiple periods and includes acquired and disposed aircraft in each period and the periodical profit. With the optimal achievable profit in each period as calculated in the FPM, the profits generated with FAM are evaluated in the form of a reward.

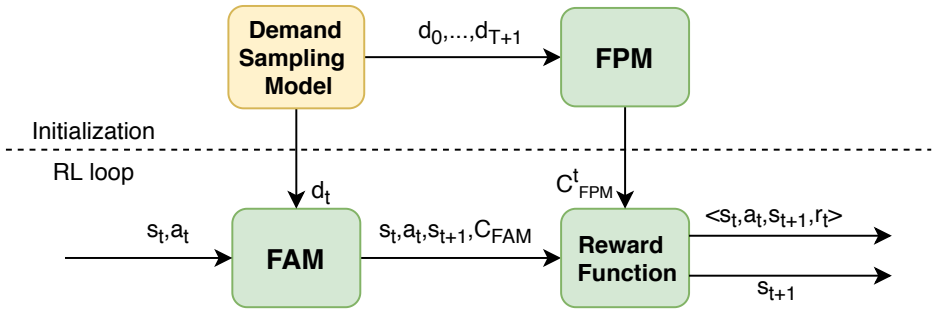


Figure 3.1: Schematic architecture of the reward function calculation. The agent feeds the FAM block from the left with the state s_t and action a_t. The experience $< s_t, a_t, s_{t+1}, r_t >$ is stored in the replay buffer, and the next state s_{t+1} returned to the agent.

In Figure 3.1, FPM optimisation and FAM optimisation block are visible in conjunction with their interaction with each other. The FPM uses the sampled set of air-travel demand $\{d_0, ..., d_{T+1}\}$ to calculate the optimal fleet decisions and corresponding profit for each period C^t_{FPM} at initialization, before starting the RL loop. These profits represent the frame of reference for the evaluation of the fleet decision in the RL loop. At each time step t, the profit C_{FAM} is calculated using the demand d_t and the fleet decision a_t, and evaluated to the C^t_{FPM}. In Section 3.4, a more elaborate explanation of the reward function is discussed. In the next section, the optimisation algorithms (FPM/FAM) are explained.

3.2. Airline Profit Optimisation

In Operations Research (OR), a real-life problem is idealized and transformed into a mathematical formulation and solved using an optimisation algorithm; It is a solution strategy of iteratively calculating a problem to converge towards the optimal solution. In Part II (Literature Review) the various modelling method and optimisation solutions for OR are outlined. Both the FPM and FAM are deterministic optimisation problems as they use the revealed air-travel demand as a deterministic variable. Deterministic optimisation methods can roughly be divided into three problem types: Linear Programming (LP), Integer Programming (IP), and Mixed Integer Linear Programming (MILP). In LP, the decision variables of the problem are continuous; in IP, the decision variables can take only an integer form; and in MILP, the decision variables can take either a continuous or an integer form. To illustrate, in the fleet planning problem, the amount of aircraft disposed cannot take a continuous form and is therefore always represented as an integer. It is important to note that IP and MILP programs require much longer computational times, as solving the problem requires extra computationally heavy techniques such as Linear Relaxation, Heuristics, and Divide and Conquer. The author's Literature Study (de Koning (2019)) expands thoroughly on these solving techniques. The deterministic optimisation algorithm design is not part of this research. Subsequently, an of the shelf commercial solver *Gurobi* is used.

In the FAM the goal is to maximise the profit given the aircraft fleet and the air-travel demand in a single period. The FPM shares a similar goal as the FAM; however, the aircraft fleet decisions are decision variables too, and the problem spans multiple periods. Consequently, the FAM combines multiple FAM models and includes the fleet decision into one large optimisation problem solving the entire fleet planning problem for a sampled demand trajectory. In Figure 3.2 a simplified representation of the FPM and FAM is shown. Indeed, at every period t, the FAM optimises the allocation of aircraft over the network. Optimising theses FAM-optimisations in conjunction with optimising the optimal fleet over multiple periods t results in the FPM.

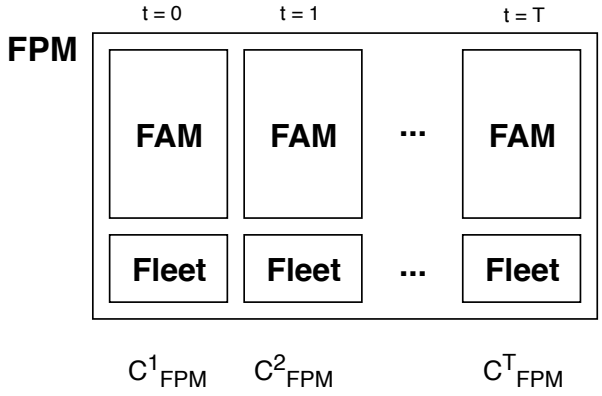


Figure 3.2: Simplified representation of Fleet Assignment Mode and Fleet Planning Model.

As a result, the FPM will be the most extensive and computationally expensive model. In the remainder of this section, the FPM's solution method is explained, adapted from Santos (2017b). The FAM will almost be identical to the FPM; however, all summations of multiple periods and constraints and decision variables related to the fleet are not considered. The FAM is outlined in Section 3.3. For the FPM, the following sets are defined:

$\mathcal{T}$ = set of discrete time periods in the finite time horizon T, $\{0, 1, ..., T\}$;
$\mathcal{K}$ = set of aircraft types, K being the total number aircraft types available;
$\mathcal{I}$ = set of airport in the network, I being the total number of airports available;
$\mathcal{N}$ = set of routes in the network, N being the total number of routes in the network;

3.2.1. Decision Variables

In an optimisation problem, the goal is to determine a set of parameters which maximizes (or minimizes) an objective function. In the Fleet Assignment Model (FAM) only three type of decision variables can be identified: The flight frequency between airports per aircraft type, the number of non-stop passenger, and the number of transfer passenger connecting through the hub airport. These decision variables are also employed for the FPM, together with the decision variables defining the number of aircraft owned/disposed/acquired in period t:

x_{ij}^{t} passenger flow non-stop from origin airport i to destination airport j in period t
w_{ij}^{t} passenger flow from airport i to airport j that transfers at the hub in period t
$z_{ij}^{k,t}$ weekly flight frequency between airport i and airport j in period t operated by type k
$ac_{own}^{k,t}$ number of type k aircraft owned in period t
$ac_{dis}^{k,t}$ number of type k aircraft disposed in period t
$ac_{acq}^{k,t}$ number of type k aircraft acquired in period t

3.2.2. The Non-Decision Variables

Next to the decision variables, a set of non-decision variables are required to define the airline environment and to create the objective function and constraints. All variables below are the fixed values; meaning they do not vary over episodes e.

c_{var}^{k} variable cost of operating an aircraft of type k per flown km in [dollar/miles]
c_{own}^{k} cost of owning an aircraft of type k each year in [dollar]
c_{dis}^{k} cost of disposing an aircraft of type k each year in [dollar]
n_{week} number of operating weeks per year
n_{year} number of years in one period
OT_{ij} average time to fly leg ij in [hours]
TAT_{k} turn around time of aircraft type k in [hours]
OH_{k} the maximum operation hours of an aircraft type k per week in [hours]
D_{ij}^{t+1} demand between airport i to airport j per year in period $t+1$ [pax]
$dist_{ij}$ distance between airport i to airport j [miles]
s^{k} seats in aircraft type k [pax]
g_{ij} $g = 0$ if a hub is located at airport i, j, $g = 1$ otherwise
R^{k} range of aircraft type k in [miles]
F_{init}^{k} initial owned fleet for type k at initial period $t = 0$

3.2.3. Objective Function

As outlined earlier, the goal or objective is to maximise the total profit of the airline within the time horizon. The profit of an airline can be defined as:

$$\text{Airline profit} = \text{Revenue} - \text{Costs} \tag{3.1}$$

The revenue of the airline can be modelled in different ways. Belobaba et al. (2015) approaches the revenue as the Revenue Passenger Miles (RPM: the amount of passengers times the travelled distance) times the *yield* (the revenue per passenger per flown mile). This is a good method when the fares of flights are not available. However, in our problem the average fare prices of each flight are available, extracted from the Bureau of Transportation Statistics (BTS), part of the United States Department of Transportation (US DOT). Equation 3.2 shows the objective function as a maximization of the yearly profit, where the revenue is represented as the summation over all routes and periods of the average fare $fare_{ij}$ of each route times the amount of non-stop passengers x_{ij}^t and transfer passenger w_{ij}^t travelled per week. To obtain the yearly profit the summation is multiplied with the number of operational weeks. Note that no fare price distinction is made for non-stop passengers and passenger travelling via the hub.

The costs are split up in two components: the operating costs and the fixed fleet costs. The operating cost is a summation over all periods, routes, and aircraft types of the variable operating cost per aircraft type C_{var}^k times the distance $dist_{ij}$ of each route times the flight frequency $z_{ij}^{k,t}$ of each type per period. Finally, two cost components for the fleet decisions are introduced. The summation is made over all periods and aircraft types, multiplying the ownership cost of an aircraft C_{own}^k and the amount of aircraft owned of each type in each period and multiplying the disposal cost of an aircraft type C_{disp}^k with the amount of aircraft of type k disposed at each period t.

$$\begin{aligned}\text{Maximize profit} = \quad & \sum_{t\in\mathcal{T}}\sum_{i\in\mathcal{I}}\sum_{j\in\mathcal{I}}\left[fare_{ij}\cdot(x_{ij}^t + w_{ij}^t)\right] \\ & - \sum_{t\in\mathcal{T}}\sum_{i\in\mathcal{I}}\sum_{j\in\mathcal{I}}\sum_{k\in\mathcal{K}}\left[C_{var}^k\cdot dist_{ij}\cdot z_{ij}^{k,t}\right] \\ & - \sum_{t\in\mathcal{T}}\sum_{k\in K}\left[C_{own}^k\cdot ac_{own}^{k,t} + C_{dis}^k\cdot ac_{dis}^{k,t}\right]\end{aligned} \tag{3.2}$$

Note that no cost-component is induced to the decision variable to acquire aircraft. It is assumed that an airline will either lease an aircraft or depreciate an aircraft over multiple years. The cost component of acquired aircraft is therefore captured by the fixed ownership costs.

3.2.4. Constraints:

Demand Constraint

Equation 3.3 ensures that the amount of passenger flow can not exceed the demand. The sum of non-stop passengers x_{ij} and passengers transferring at the hub w_{ij} with origin i and destination j, is smaller or equal than the demand between those airports. Note that the demand is shifted one period because the assesment of the fleet decision in period t is for the revealed demand in step $t+1$. This equation must hold for all airport origins i and destinations j in $\mathcal{I}$, and every period $t \in \mathcal{T}$. Subsequently, the number of constraints is: $C = N \cdot T$. Note that $N = I \cdot (I-1)$.

$$x_{ij}^t + w_{ij}^t \leq D_{ij}^{t+1} \quad , \forall i, j \in \mathcal{I}, t \in \mathcal{T} \tag{3.3}$$

Aircraft Balance Constraint

Equation 3.4 ensures that the amount of aircraft coming into an airport also has to leave the airport. This is necessary to ensure that the amount of aircraft at the beginning and end of the week is equal

and the schedule can be repeated weekly. The equation defines that sum of all inbound flights arriving at airport i departing from airports j is equal to the number of outbound flights from airport i to the airports j. This constraint must hold for all airports $i \in \mathcal{I}$, all aircraft type $k \in \mathcal{K}$, and all $t \in \mathcal{T}$. Subsequently, the number of constraints is: $C = I \cdot K \cdot T$.

$$\sum_{j \in \mathcal{I}} z_{ij}^{k,t} = \sum_{j \in \mathcal{I}} z_{ji}^{k,t} \quad , \forall i \in \mathcal{I}, k \in \mathcal{K}, t \in \mathcal{T} \tag{3.4}$$

Transfer Passenger Constraint
Equation 3.6 ensures that transfer passengers can not have the hub as origin or destination. The variable g_{ij} represents a matrix with ones, except if i or j is the hub airport. As a result, the right-hand side of the inequality becomes zero and consequently the transfer passengers for those airports. This constraint must hold for all airports $i, j \in \mathcal{I}$, all aircraft type $k \in \mathcal{K}$, and all $t \in \mathcal{T}$. Subsequently, the number of constraints is: $C = N \cdot K \cdot T$.

$$w_{ij}^{t} \leq D_{ij}^{t} \times g_{ij} \quad , \forall i, j \in \mathcal{I}, k \in K, t \in \mathcal{T} \tag{3.5}$$

Aircraft Capacity Constraint
Equation 3.6 ensures that the utilisation time of each aircraft per week is not exceeded. For every aircraft type and in each period, the possible weekly utilisation of an aircraft type is the amount of aircraft owned $ac_{own}^{k,t}$ multiplied with the block time per week OH_k. The summation of the average flight time $OT_{i,j}$ and turn around time $TAT_{i,j}$ times the number of flight legs for each route must be smaller than the possible weekly utilization time. This constraint must hold for all aircraft type $k \in \mathcal{K}$, and all $t \in \mathcal{T}$. Subsequently, the number of constraints is: $C = K \cdot T$.

$$\sum_{i \in \mathcal{I}} \sum_{j \in \mathcal{I}} \left(OT_{ij} + TAT_k\right) \cdot z_{ij}^{k,t} \leq OH_k \cdot ac_{own}^{k,t} \quad , \forall k \in \mathcal{K}, t \in \mathcal{T} \tag{3.6}$$

Capacity Verification Constraint
Equation 3.7 ensures that the amount of non-stop and transfer passenger on each route is lower than the maximum amount of seats on a route. The maximum amount of seats sold on a route is defined as the multiplication of the flight frequency $z_{ij}^{k,t}$ with the number of seats s^k. This must be larger than the transported non-stop and transfer passengers. Again the variable $g_{i,j}$ is used to allow only transfer passenger when flying from or to the hub. This constraint must hold for all airports $i, j \in \mathcal{I}$, and all $t \in \mathcal{T}$. Subsequently, the number of constraints is: $C = N \cdot T$.

$$x_{ij}^{t} + \sum_{m \in \mathcal{I}} w_{im}^{k,t}\left(1 - g_{ij}\right) + \sum_{m \in \mathcal{I}} w_{mj}^{k,t}\left(1 - g_{ij}\right) \leq \sum_{k \in \mathcal{K}} z_{ij}^{k,t} \cdot s^k \quad , \forall i, j \in \mathcal{I}, t \in \mathcal{T} \tag{3.7}$$

Range Constraint
Equation 3.8 ensures that the operational range of an aircraft is not violated on a route. In other words, if the aircraft type cannot fly a route, the weekly flight frequency over all periods should be zero. If an aircraft type can fly that route, the flight frequency is constraint with a Big M of 1000, otherwise zero. This constraint must hold for all airports $i, j \in \mathcal{I}$, and all $k \in \mathcal{K}$. Subsequently, the number of constraints is: $C = N \cdot K$.

$$\sum_{k \in \mathcal{K}} z_{ij}^{k,t} \leq a_{ij}^{k} = \begin{cases} 1000, & \text{if } dist_{ij} \leq R^k \\ 0, & \text{otherwise} \end{cases} \quad , \quad \forall i, j \in \mathcal{I}, k \in \mathcal{K} \tag{3.8}$$

Fleet Evolution Constraints
Up until here, all constraints involved optimisation of the assignment of the fleet in a network. However, the next constraints connect the FAM's for each period into a multi-period FPM with fleet decisions. Equation 3.9 ensures that the continuity of fleet between periods is satisfied. In other words,

the amount of aircraft owned $ac^{k,t}_{own}$ plus the amount of aircraft disposed $ac^{k,t}_{dis}$ min the amount of aircraft acquired $ac^{k,t}_{acq}$ must equal amount of aircraft in the previous period. This constraint must hold for all $k \in \mathcal{K}$, and $t = (1, ..., T)$. Equation 3.10, initializes the fleet in the first period. Subsequently, the number of constraints is: $C = T \cdot K$.

$$ac^{k,t}_{own} + ac^{k,t}_{dis} - ac^{k,t}_{acq} = ac^{k,t-1}_{own}, \quad \forall t = (1, \dots, T), k \in \mathcal{K} \tag{3.9}$$

$$ac^{k,0}_{own} + ac^{k,0}_{dis} - ac^{k,0}_{acq} = F^{k}_{init}, \quad \forall k \in \mathcal{K} \tag{3.10}$$

Fleet Decision Constraints

The following equations are used to match the fleet decision in the FPM to the action space of the agent, defined in Chapter 1. Equation 3.11 ensures that in every period one fleet acquirement or disposal is allowed for one type. The $min(x, y)$ function chooses the item with smallest value, the x or y variable. The summation of disposed and acquired fleet over all types must remain smaller than one. Because the fleet variables and integers, only one fleet decision variable in each period can be one or higher, or all zero. Finally, equation 3.12 ensures that the fleet decision in every period can not be higher than the maximum allowed acquirement or disposal of fleet f^{k}_{max}. These constraints must hold for in every period $t \in \mathcal{T}$. Subsequently, the number of added constraints is: $C = 2 * T$.

$$\sum_{k \in K} \min(ac^{t,k}_{own}, 1) + \min(ac^{t,k}_{disp}, 1) \leq 1, \quad \forall t \in \mathcal{T} \tag{3.11}$$

$$\sum_{k \in K} ac^{t,k}_{own} + ac^{t,k}_{dis} \leq f^{k}_{max}, \quad \forall t \in \mathcal{T} \tag{3.12}$$

Integrality and Non-negativity Constraints

In principle, Equation 3.13 ensures that all decision variables in the optimisation model are natural numbers or zero (positive and integer). All decision variables are integer and consequently, the model becomes an Integer Programming problem. As discussed earlier this requires the most amount of computational power and is therefor the slower. However, in practice, both passenger flow passengers are assumed to be continuous, because the influence on the profit is marginal, and the computational load is decreased.

$$x^{t}_{ij} \in \mathbb{R}^{+}, w^{t}_{ij} \in \mathbb{R}^{+}, z^{k}_{ij} \in \mathbb{Z}^{+}, ac^{k}_{own} \in \mathbb{Z}^{+}, ac^{k}_{dis} \in \mathbb{Z}^{+}, ac^{k}_{acq} \in \mathbb{Z}^{+} \tag{3.13}$$

3.3. Fleet Assignment Model

As pointed out earlier, the FAM is similar to the FPM; however, it does not span multiple periods and all fleet decision variables., and optimises the optimal fleet plan given the fleet composition to retrieve the profit. $(ac^{t,k}_{own}, ac^{t,k}_{dis}, ac^{t,k}_{acq})$ become constant variables $(ac^{k}_{own}, ac^{k}_{dis}, ac^{k}_{acq})$ defined by the action vectore of the agent. Moreover, the same objective funtion and constraint Equations 3.3, 3.4, 3.6, 3.6, 3.7, and 3.8 are used in the FAM model; however, only a single period is considered: $T = t$. Note that conform to the FPM, in Equation 3.15 the demand is the revealed air-travel demand d^{t+1}_{ij}. The full FAM model can be found below.

Objective Function:

$$\begin{aligned} \text{Maximize profit} = \quad & \sum_{i \in \mathcal{I}} \sum_{j \in \mathcal{I}} \left[Fare_{ij} \cdot (x_{ij} + w_{ij}) \right] \\ & - \sum_{i \in \mathcal{I}} \sum_{j \in \mathcal{I}} \sum_{k \in \mathcal{K}} \left[C^{k}_{var} \cdot dist_{ij} \cdot z^{k}_{ij} \right] \\ & - \sum_{k \in K} \left[C^{k}_{own} \cdot ac^{k}_{own} + C^{k}_{dis} \cdot ac^{k}_{dis} \right] \end{aligned} \tag{3.14}$$

Constraints

$$x_{ij} + w_{ij} \leq d_{ij}^{t+1} \quad , \forall i, j \in \mathcal{I} \tag{3.15}$$

$$\sum_{j \in \mathcal{I}} z_{ij}^{k} = \sum_{j \in \mathcal{I}} z_{ji}^{k} \quad , \forall i \in \mathcal{I}, k \in K \tag{3.16}$$

$$w_{ij} \leq D_{ij} \times g_{ij}, \forall i, j \in \mathcal{I}, k \in \mathcal{K}, t \in \mathcal{T} \tag{3.17}$$

$$\sum_{i \in \mathcal{I}} \sum_{j \in \mathcal{I}} \left(OT_{ij} + TAT_{k}\right) \cdot z_{ij}^{k} \leq OH_{k} \cdot ac_{own}^{k} \quad , \forall k \in \mathcal{K} \tag{3.18}$$

$$x_{ij} + \sum_{m \in \mathcal{I}} w_{im}^{k} \left(1 - g_{ij}\right) + \sum_{m \in \mathcal{I}} w_{mj}^{k} \left(1 - g_{ij}\right) \leq \sum_{k \in \mathcal{K}} z_{ij}^{k} \cdot s^{k} \quad , \forall i, j \in \mathcal{I} \tag{3.19}$$

$$z_{ij}^{k} \leq a_{ij}^{k} = \begin{cases} 1000, & \text{if } dist_{ij} \leq R^{k} \\ 0, & \text{otherwise} \end{cases} \quad , \quad \forall i, j \in \mathcal{I}, k \in \mathcal{K} \tag{3.20}$$

$$x_{ij}^{t} \in \mathbb{R}^{+}, w_{ij}^{t} \in \mathbb{R}^{+}, z_{ij}^{k} \in \mathbb{Z}^{+} \tag{3.21}$$

3.4. Reward function

Shaping the reward function $R(s_t, a_t, s_{t+1})$ of the RL process probably one of the more difficult practices in RL and vital to assure fast training and good convergence properties. The reward function relates the effect of the state-action pair to a value which is used by the agent to learn the optimal policy. Indeed, the reward value is the only measure for the agent to identify the value of an experience. The reward in DQN is 'clipped' between $-1, 1$ to ensure smooth learning without disruptive learning values. However, between these min/max values lies a range of possibilities. In this section, the profit of the FPM and FAM is translated to a reward function.

In the RL field, little literature exists regarding the shaping of reward functions for operational and control processes. However, there are some general rules to follow in order to obtain a reward function which will allow convergence and functionality to learn the policy. To start, commonly the +1 is considered to be the reward accompanied to the optimal decision and return. In the fleet planning problem, this occurs the moment the periodical profit of the FPM equals the profit of the FAM. All other fleet decisions will be less optimal and should have a lower reward.

If all sub-optimal rewards are set to zero, when exploring the state-action space, a plus one reward will be very rare, and model will take a long time learning and have trouble converging due to the sparsity of the reward. Indeed, a more graduate learning curve is needed which punishes really bad decisions but rewards near-optimal decisions. As a solution, the procentual relationship between the profit or contribution of the FPM and RAM could be used to assess the value of the fleet decision as a linear function ranging from zero profit (r=0) to the optimal profit (r=1):

$$r_t = \frac{C_{FAM}}{C_{PFM}^{t}} \tag{3.22}$$

Nevertheless, problems arise with the introduction of terminal decisions. A terminal decision is a fleet decision which causes the RL loop to terminate. An example would be to dispose three aircraft of type k while the current fleet only has two aircraft of type k. This action is infeasible and consequently, the network should be 'punished'. That is why the reward model checks the validity of the action first and assigns a -1 reward if the action is infeasible.

Lower bound of the Action Space

The procentual relationship between FPM and FAM described above induces another problem. In reasonably sized fleet problems the least optimal valid action still creates a very positive reward. Indeed, bad fleet decisions have an influence on the profit but do not make the profit diminish towards zero. As a result, in larger fleet planning problems, Equation 3.22 will always result in positive reward close to +1, making all fleet decision evenly valuable to the DQN-model.

Moreover, as pointed out earlier, due to the sampling of demand the maximum achievable profit and lowest valid profit fluctuate. Figure 3.3, 3.4, and 3.5 shows the profit calculated with a FAM for all (valid) fleet decisions for 50 air-travel demand samples for respectively four, seven, and ten airports.

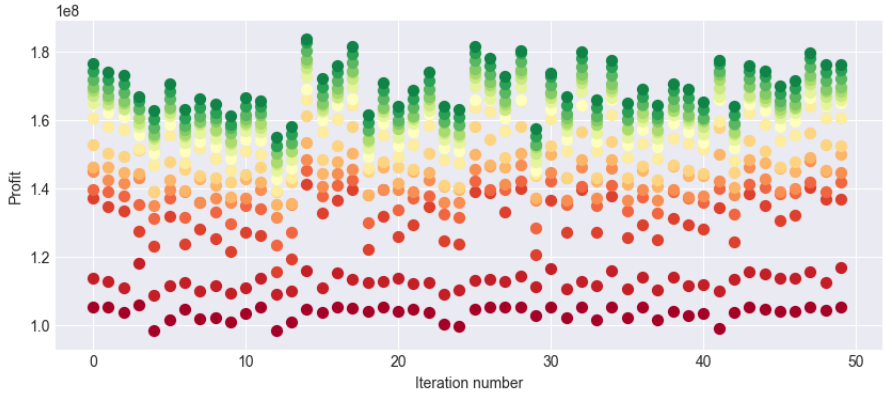


Figure 3.3: Profit of action space in the first period for 50 independent sampled demand values of a network with 4 airports.

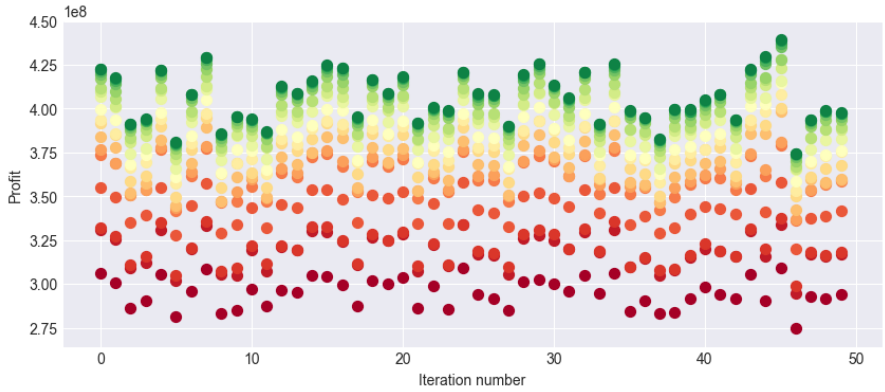


Figure 3.4: Profit of action space in the first period for 50 independent sampled demand values of a network with 7 airports.

In every iteration, the highest profit of the action space can be identified as the upper green dot, and the lowest profit the lower red dot. Two conclusions can be drawn. First, the worst valid action still creates a very positive profit, and thus a reward close to +1. Secondly, as the network grows the prescribed effect becomes even larger as the profit grows. A possible solution is setting a lower bound profit value. By subtracting the lower from the FAM and FPM contribution, the reward is calculated with respect to this lower bound. However, setting a single value lower bound creates a bias in the evaluation of the reward as iterations with low demand and low optimal profit are devaluated, and the reward of actions with high demand overestimated. To solve the bias, a lower bound as a procentual off-set from the optimal value is chosen and equation 3.22 becomes:

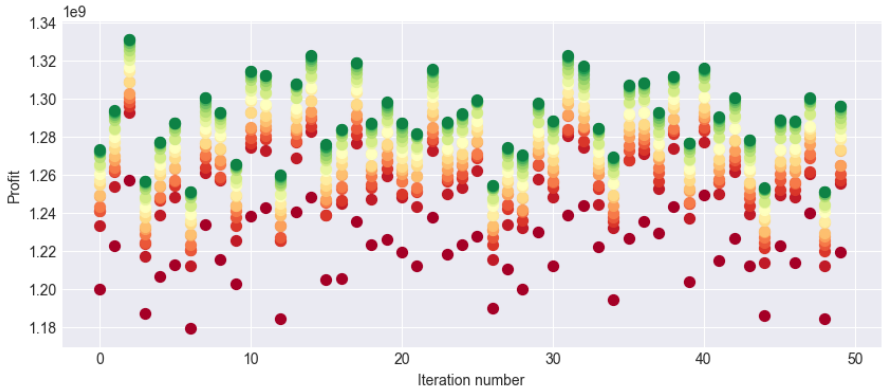


Figure 3.5: Profit of action space in the first period for 50 independent sampled demand values of a network with 10 airports.

$$r_t = \begin{cases} -1, & \text{if } ac_{own}^{k,t} - ac_{dis}^{k,t} < 0 \\ \dfrac{C_{FAM} - lb \cdot C_{FPM}^t}{(1-lb)\, C_{FPM}^t}, & \text{elseif } C_{FAM} > lb \cdot C_{FPM}^t \\ 0, & \text{else} \end{cases} \tag{3.23}$$

Here, lb is the lower bound for an airline network ranging between $(0,1)$. The value of the lower bound is a parameter of the model which can be increased and decreased in order to improve the learning. The lower bound is found empirically by assessing the profit range of the fleet actions for the considered fleet problem. As the size of the airline network increases, so does the fleet and the profit related to the airline operations (of course in real life, the airline's profitability is dependent on more variables, but in the FPM/FAM optimisation models this relationship holds). The relative impact of adding one aircraft or disposing one aircraft on the profit becomes smaller with increasing profitability due to air travel demand and network size.

To establish a set of lower bounds, all possible profits related to the fleet decisions, as calculated in Figure 3.3, 3.4, and 3.5, of the FAM optimisation (C_{FAM}) are divided by the optimal contribution C_{FPM}^t for every period t and saved to a dataset from which the interquartile range (IQR) is calculated. The IQR is a measure of the variability of the procentual score of sampled FAM contributions with respect to the demand sample's best action contribution FPM, often represented as a box plot. Figure 3.6 shows an example of a boxplot with the IQR. The IQR is the range of data point that that lies between the 25th percentile: the lower quartile(LQ), and the 75th percentile: upper quartile (UQ). The Lower Whisker LW and Upper Whisker (UW) represent the 'suspected' minimum and maximum of problem, which is 1.5 IQR from the respective quartile. The median is denoted as (MED)

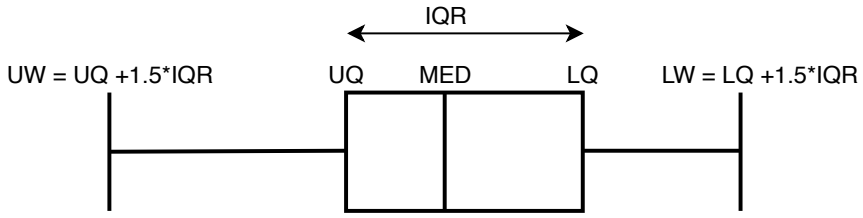


Figure 3.6: Example of a box plot with the interquartile range.

An example of the average IQR values over a network ranging from three to ten airports is visible

in Figure 3.7. It is clearly visible that the bigger the network grows, the smaller the IQR becomes. This behaviour can be explained due to the increased profits. As a result, the influence bad decisions on the profit become less prone as the network grows. The value of the lower bound and IQR range is thus dependent on the network size fleet planning model and should be empirically established before running the RL program. The IQR value choice influences the learning of the agent, and an optimal value can only be determined by trial and error. As a result in Chapter 4 as sensitivity analysis of the lower bound on the performance of RL program is depicted.

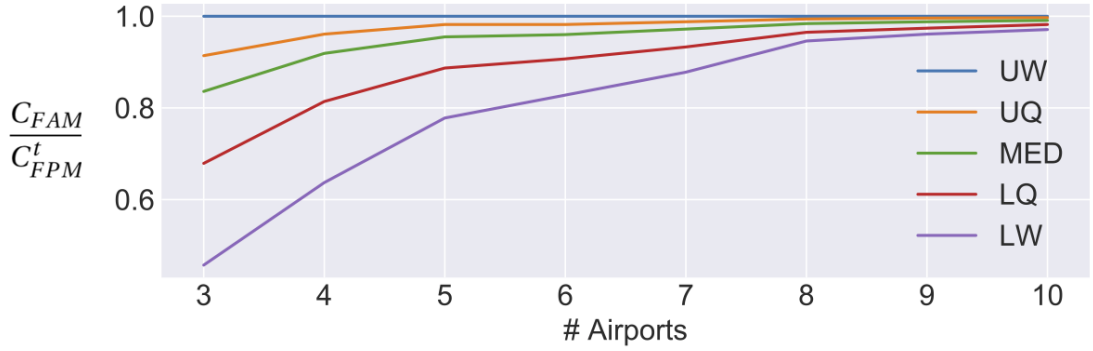


Figure 3.7: IQR range of fleet decision procentual profit of optimal profit for various airline network sizes.

3.5. LP Matrix Size and Computational Times

When training a model for E episodes and T periods, the FPM is run E times whereas the FAM $E \cdot T$ times.; however, the most computationally expensive part of training the RL model will be the optimisation of the FPM. This is related to the large LP-matrix of the FPM and the many integer decision variables. In this research little attention is given into the improvement of optimisation algorithms to solve the FPM and FAM. However, more focus is given to maximizing the sample efficiency and early stopping of the optimisation algorithm. Before continuing to the possible improvement of computational speed, let's investigate the computational expense of the FPM and FAM.

The MIP gap of a linear optimisation program can be defined as the optimality gap between the primal problem (the best-known solution) and the associated dual problem (the best bound) in a MILP optimisation. Because of the integer variables solving a MILP requires more complex methods with respect to simpler LP models, and rarely the problem is solved. In MILP optimisation, if the optimality gap of the incumbent solution becomes smaller than the predefined MIP gap, the model is assumed to be 'solved', but the exact solution lies somewhere in that gap.

To investigate the runtime of the FPM and FAM, the default MIP gap in Gurobi is set to $1E-4$, and the model is run with varying number of airports and a varying number of periods. In Table 3.1 the three tables are visible. The first and second table shows respectively the number of decision variables and constraints related to the number of periods T, and airports N. The attentive reader might notice that the size of the decision variable is not only dependent on the number of airports, periods, but also on the amount of aircraft type K considered. To decrease the complexity, in all the runtime tests the number of aircraft types is set to two. Increasing or decreasing the number of aircraft types considered increases or decreases the number of decision variables and constraint with roughly $25-30\%$, which is less influential compared to the number of airport or periods. The last table shows the average runtime of the various period-airport configurations for 50 air travel demand samples. Note that the maximum runtime of FPM optimisation is set to 500 seconds.

It is clearly visible that the runtime increases exponentially and does not increase linearly with

# Periods									# Decision Variables
FPM	5	170	290	450	650	890	1170	1490	1850
	4	136	232	360	520	712	936	1192	1480
	3	102	174	270	390	534	702	894	1110
	2	68	116	180	260	356	468	596	740
FAM		30	54	86	126	174	230	294	366
# Airports		3	4	5	6	7	8	9	10

# Periods									# Constraints
FPM	5	210	370	580	840	1150	1510	1920	2380
	4	168	296	464	672	920	1208	1536	1904
	3	126	222	348	504	690	906	1152	1428
	2	84	148	232	336	460	604	768	952
FAM		42	80	130	192	266	352	450	560
# Airports		3	4	5	6	7	8	9	10

# Periods							Average Runtime over 50 runs		
FPM	5	0.07	0.86	5.66	44.7	267.5	500+	500+	500+
	4	0.1	0.78	11.94	32.96	184.6	497.7	500+	500+
	3	0.08	0.5	8.6	8.62	3.49	8.6	17.36	74.75
	2	0.04	0.18	1.36	1.02	1.22	1.09	1.25	1.73
FAM		0.005	0.01	0.021	0.035	0.089	0.448	0.49	0.935
# Airports		3	4	5	6	7	8	9	10

Table 3.1: Decision variables, constraints, and runtime for a FPM/FAM model with two types of aircraft and varying size of periods and airport network.

increasing constraints or decision variables. The fleet planning environment considered in the case-study, comprising ten airports and five periods, consistently reaches a runtime of more than 500 seconds for an optimality gap of $1E-4$.

A solution to solve this problem is to increase the MIP gap and stop the optimisation early. The contribution gain often does not scale linearly with the extra computational time, and therefore decreasing the MIP gap or stopping early might be a viable solution. Although, increasing the MIP gap allows for errors in the solution of the PFM and FAM and ultimately the reward function, in exchange the computational time becomes far less. A new investigation is performed on the test case's variable settings (e.g. ten airports, two aircraft types, and five periods). For another 50 air travel demand samples, the FPM and FAM for the optimal fleet decisions are calculated. However, this time whenever Gurobi finds an incumbent solution, the incumbent profit, the runtime, and MIP gap is saved. With the saved data, the development of the profit over runtime or MIP gap can be reconstructed, but more importantly the relative reward error from the intermediate solution with respect to the optimal solution (MIP gap = $1E-4$) and 'true' reward. The results of this test are visible in Table 3.2. The upper table shows the average percentage error of the incumbent solutions with respect to the reward found if the MIP gap of $1E-4$ or a runtime limit of 500 seconds was reached. The lower table shows the accompanied average runtime.

The test shows a few interesting results. To begin, the FPM is the dominant factor in runtime and increases exponentially with a logarithmic decrease of the MIP gap. Moreover, for preset MIP gap of $1E-4$ for the FPM, the optimality gap almost never converges below the $1E-4$ and is always cut off at the time-limit at 500 seconds. On the other hand, increasing the optimality gap does not have a grave influence on the procentual error for the benchmark reward. Below a MIP gap of $1E-3$, the relative reward error remains below 1% which is an acceptable error to not affect the learning of the neural network. Moreover, relative error increases as the MIP gaps from the FAM and FPM

MIP gap FPM	RELATIVE ERROR FROM TRUE REWARD [%]					
0.1	127.570	127.259	127.045	121.047	108.854	46.532
0.01	17.161	16.859	16.653	10.837	-0.990	-52.299
0.005	6.563	6.263	6.057	0.258	-11.530	-59.612
0.001	1.260	0.960	0.754	-5.035	-16.806	-63.579
0.0005	0.505	0.206	0.000	-5.788	-17.556	-64.143
0.0001		-0.256	-0.462	-6.249	-18.015	-64.475
MIP gap FAM	0.0001	0.0005	0.001	0.005	0.01	0.1

MIP gap FPM	AVERAGE RUNTIME [seconds]					
0.1	2.15164	0.29768	0.2472	0.23513	0.16298	0.11194
0.01	2.30546	0.4515	0.40103	0.38896	0.31681	0.26576
0.005	2.19241	0.33845	0.28797	0.2759	0.20375	0.15271
0.001	7.6814	5.82744	5.77696	5.7649	5.69274	5.6417
0.0005	51.567	33.0274	32.5227	32.402	31.6804	31.17
0.0001	502.082	500.228	501.678	500.166	500.094	500.043
MIP gap FAM	0.0001	0.0005	0.001	0.005	0.01	0.1

Table 3.2: Decision variables, constraints, and average runtime for a FPM model with two types of aircraft and varying size of periods and airport network.

start to deviate from each other. Because the reward is a result of the profit comparison in the FAM and FPM, different MIP-gaps result in different level of detail. Consequently, the relative error above the diagonal is positive as the optimality gap of the FAM becomes smaller and the profit deviations grow, and vice versa below the diagonal.

To conclude, in order to achieve a meaningful reward in a reasonable time, a MIP gap below $1E-3$ for both the FPM is necessary, and consequently, the same MIP gap should be employed for the FAM. To maximise the training process, the MIP gaps are set to $1E-3$ with a time-limit of 5 seconds. If the optimality gap of the FPM does not decrease below the MIP gap within the time-limit, the MIP gap of the FAM is increased to the optimality gap of the FPM to minimize the error. During the validation and evaluation of the DQN-model, a more detailed reward is employed; the MIP gaps are set to $5E-4$ with a time-limit of 30 seconds. Again the MIP-gap of the FAM is set to the optimality gap of the FPM in every iteration.

As the size of the network I, the number periods T, and the number of aircraft types K increases, so increases the computational time of the reward function exponentially and the training process of the DQN-model. Moreover, the increased size of the network increases the amount of profitable route and the airlines' profit. As a result, the influence of fleet decisions become relatively small, and the reduction optimality gap of the FPM and FAM more crucial to ensure meaningful rewards.

4

Sensitivity Analysis

This chapter details the sensitivity analysis of the reinforcement learning model. It is important to investigate the sensitivity of the sub-components' results to variations in (hyper)parameters and input variables. In this chapter, various hyperparameters from the DQN-model and parameters from the environment are tested to assess and evaluate the influence on the learning performance.

The hyperparameters of the learning models are the parameters which define the structure, size, and learning methods. The correct hyperparameters can gravely contribute to the learning and decreasing the loss function. The loss function is defined by Mean Squared Error (MSE) of a predictor and consists of three elements: the bias, the variance and the error. The bias is the error between the average prediction and the actual values of the sample. When the bias is high, it usually indicates that the model is not able the represent the underlying pattern in the data: under-fitting. On the contrary, the variance of the model is prone to emerge in an aggressive learning setting with noisy data. The parameters are tuned to much towards the training dataset and over-fitting occurs. The noise is intrinsic to the data trained and tested on and cannot be improved by hyperparameter tuning (Weinberger, 2018).

In this section, multiple key (hyper)parameters for the DQN's neural network (NN) and environment are investigated repetitively. An ensemble of DQN-models is created with varying values for a (hyper)parameter and trained. During the training process, at every episode, after sampling an air-travel demand, the RL loop is run for every DQN-model, and trained using their own replay buffers. This ensures all DQN-model -with different (hyper)parameters- are trained using the exact same air-travel demand trajectories. Consequently, deviations in performance can be related to the (hyper)parameters instead of the variations in demand trajectories.

An evaluation set is used to evaluate the performance of each DQN-model at the end of the training. The evaluation-set consists of 50 independent air-travel demand trajectories and the corresponding Deterministic Static and Deterministic Dynamic Average Score and Variance of the Score. To evaluate the DQN-models, the parameters θ_e of the last 30 episodes with a 10 episode interval of each DQN-model is tested ($e = E-300, E-290, ...E$). The variance of the DQN scores over the 30 networks is referred to as the Variance of the Average Score.

To speed up the sensitivity analysis, the fleet planning problem is reduced from a multiple aircraft type problem to a single aircraft type problem. It is assumed that the analysis for the single aircraft problem holds for the multiple aircraft problem. However, the output (action) space will be smaller, and some non-linearity will disappear. Moreover, the policy of the DQN-models is Stochastic Dynamic policy tested for the Average Demand scenario.

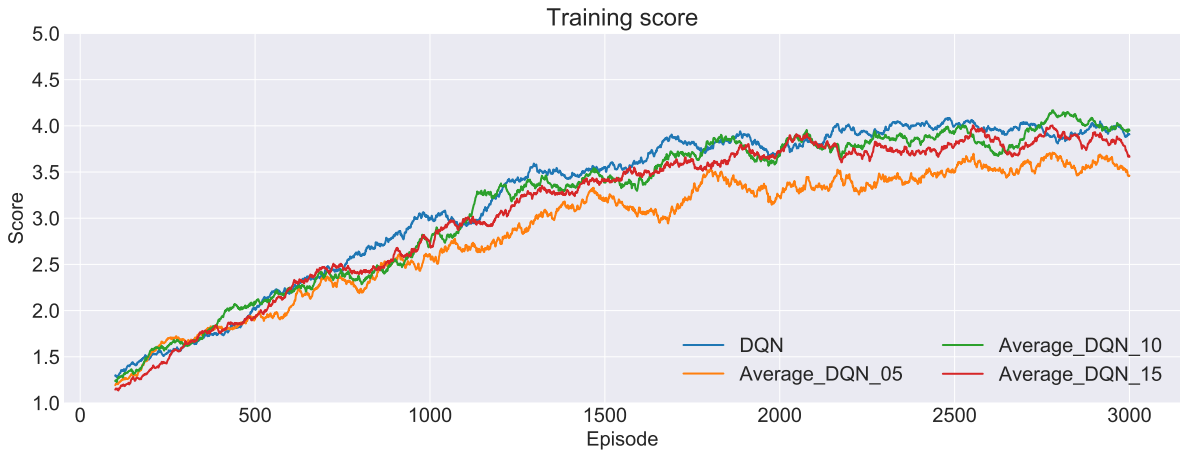


Figure 4.1: Training score of DQN and averaged-DQN.

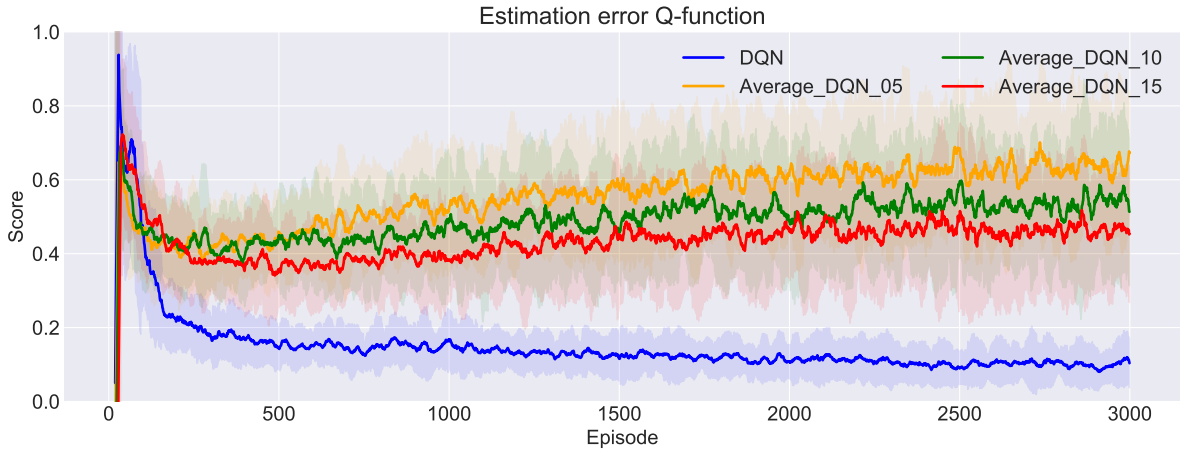


Figure 4.2: Estimated error of the Q-function for DQN and averaged-DQN.

4.1. Averaged-DQN:

An extension of DQN is tested to improve the learning stability and reduce the approximation variance of the target estimation (Anschel et al., 2017). Averaged-DQN achieves this by averaging the Q-value estimations over a set K previous NN, shown in Equation 4.1.

$$Q^A_{i-1}(s,a) = \frac{1}{K}\sum_{k=1}^{K} Q(s,a;\theta_{i-k}) \tag{4.1}$$

The Averaged-DQN is tested for $k = [5, 10, 15]$ NN, and measured against the Deterministic Static, Deterministic Dynamic, and the original DQN with a target network. The Average score is measured on 50 air travel demand samples over the final 10% of NN's with an interval of 10 NN's. Figure 4.1 shows the training score, and Figure 4.2 the training loss throughout the learning process. In Table 4.1 the Average Score, Average Variance Score, and the Variance of the Average Score across the tested networks is depicted.

The Averaged-DQN shows a lower Average Score w.r.t. the normal DQN-model; however, an improved Variance of the Average Score. This means that the change of the Average Score during the training of the neural network becomes more stable as the estimation of the Q-value is averaged of K-networks. However, the Variance of the Score within each test (on the 50 samples) becomes higher. The latter indicates that the variance is turned in an increased bias for the Averaged-DQN.

	DD	DS	SD			
			DQN	Ave-DQN(k=5)	Ave-DQN(k=10)	Ave-DQN(k=15)
Average Score	66.3%	84.46%	80.95%	70.37%	78.17%	78.64%
Av. Var. Score	6.833	1.274	1.48	4.95	2.5	3.12
Var. of Av. Score			2.721	0.612	0.713	0.103
Time [min]			22.15	25.59	43.8	58.72

Table 4.1: Testing results of DQN vs Average-DQN.

Figure 4.2 shows that the average estimation error of the target values is consistently higher for the Averaged-DQN, and confirms the increased bias and an under-fitted network.

Finally, the learning time of the DQN and Averaged-DQN grows with the amount of K-networks. As the number of networks -averaging the Q-values- increases, the amount feed-forwards increases and thus the time to train the ensemble of networks. As a result, the original DQN is employed in the fleet planning problem.

4.2. Learning rate:

After the feed-forward pass, the NN is updated through a backward pass, here the estimation error is calculated and the parameters adjusted using an optimization algorithm: *Adam* (Kingma and Ba (2014)). Adam is a combination of RMSprop and Stochastic Gradient Descent with momentum. The learning rate of the optimization algorithm defines how much the parameters of the NN are changed by the estimation error. If the learning rate is too small, the parameters do not update enough and the training process can become slow. If the learning rate is too high, the model changes to fast and keep overshooting the optimal set of parameters. To evaluate the learning rate impact on the results a logarithmic set of learning rates is tested and shown in Table 4.2:

	DD	DS	SD					
			0.1	0.01	0.001	0.0001	0.00001	0.000001
Average Score	66.3%	84.46%	23.14%	25.33%	77.34%	79.39%	71.26%	59.92%
Av. Var. Score	6.833	1.274	3.3	3.86	1.98	1.77	3.5	5.97
Var. of Av. Score			559.889	457.898	16.914	4.683	6.958	0.721
Time [min]			11.17	11.15	11.1	11.12	11.08	11.08

Table 4.2: Testing results of DQN for various learning rates.

A performance balance is found around $\lambda = 0.0001$. A higher learning rate deteriorates the Average Score drastically, but more significantly the Average Variance Score. At the same time, the Variance of the Average Score decreases with the decrease of the learning rate. This is explained by the weight updates which become smaller with a decreasing learning rate; hence lower Variance of Average Score accros the tested NN. However, a too small learning rate results in a very small learning of the training data and a model which is underfitted, visible by the Average Variance Score. A too high learning rate on the other hands overshoots the optimal network weights. As a result, a learning rate of $\lambda = 0.0001$ is chosen

In Figure 4.4 the average estimation Q-function shows clearly how $\lambda = 0.000001$ loss is biased and consequently under-fitted. The higher learning rates are not visible on this graph, due to the enormous estimation error.

Figure 4.3: Training score of DQN for various learning rates.

Figure 4.4: Estimated error of the Q-function for various learning rates.

4.3. Monte Carlo vs Temporal Difference:

In Chapter 1 the difference between Monte Carlo (MC) and Temporal Difference (TD) is explained. An analysis is made by testing the model for various n-step TD predictions. Because five periods are considered, TD(5) is equal to an MC prediction as no estimation of the Q-values is used. Gradually, decreasing the number of n-steps results in TD(1) which is the normal TD estimate.

	DD	DS	SD				
			TD(1)=TD	TD(2)	TD(3)	TD(4)	TD(5)=MC
Average Score	66.3%	84.46%	80.91%	79.84%	78.76%	78.96%	79.58%
Av. Var. Score	6.833	1.274	1.24	1.57	1.96	1.8	1.8
Var. of Av. Score			5.888	12.926	11.532	7.323	6.958
Time [min]			10.49	10.07	9.55	9.19	8.7

Table 4.3: Testing results of DQN for various learning methods.

In Table 4.3 the testing results are depicted. The Average Scores do not show major variations and are all around 80%. The Average Variance Score shows a favorability to the TD(1) or the MC learning method. The Variance of the Average Scores over the tested NN's shows the biggest difference with high variance for the n-step methods. Figure 4.5 show that all learning methods have sta-

ble learning curve and do not variate a lot. In Figure 4.6 the estimated loss is shown. The increasing MSE with increasing n-step needs some further investigation. The MSE of the prediction consists of the variance between the real predictions and the model predictions, the noise of the observations, and the bias of the predictions and the model. Moving from TD(1) to TD(5) the decreasing bias is traded for more variance. However, because the fleet planning problem already suffers from high variance, the introduced variance from the MC deteriorates the performance. As a result, one-step TD is employed as learning method.

Figure 4.5: Training score of DQN for various learning methods.

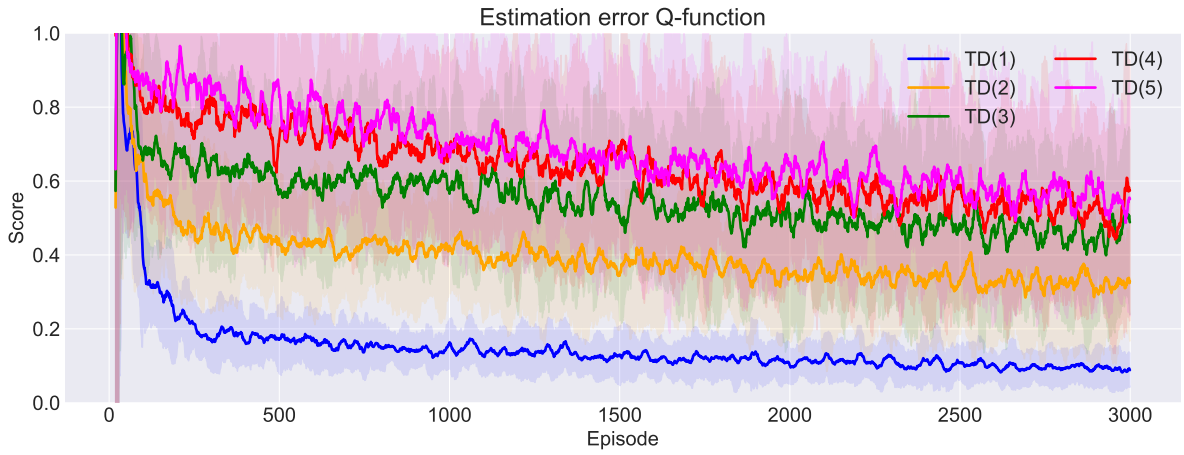


Figure 4.6: Estimated error of the Q-function for various learning methods.

4.4. Number of Hidden Layers

A conventional (feed-forward) NN has three types of layers: The input layer, the output layer, and the hidden layers. The hidden layers are what makes a NN a *deep* neural network (DNN). Increasing the number of hidden layers in DNN's usually increases the complexity of the problem the network can represent. However, increasing the number of layers can cause the network to over-fit the training data. Vice-versa, too little layers can result in a model under-fitting the data. Unfortunately, there are no rules, only guidelines in establishing the optimal number of hidden layers.

To investigate the optimal number of hidden layers h, an increasing range of hidden layers is tested. In the test set-up, next to the input layer and output layer, the last hidden layer is in every

test a linear activated fully connected layer, which has proven effective in value-function approximation Goodfellow et al. (2016). As an example, a test network with $h = 1$ has one input and output layer, one hidden layer non-linear activated, and one hidden layer linear activated.

	DD	DS	SD			
			h=1	h=2	h=3	h=4
Average Score	66.3%	84.46%	76.4%	79.38%	80.85%	81.5%
Av. Var. Score	6.833	1.274	2.6	1.74	1.62	1.51
Var. of Av. Score			8.767	6.406	7.737	1.885
Time [min]			8.47	9.69	10.69	11.63

Table 4.4: Testing results of DQN for various number of hidden layers.

Figure 4.7: Training score of DQN for various number of hidden layers.

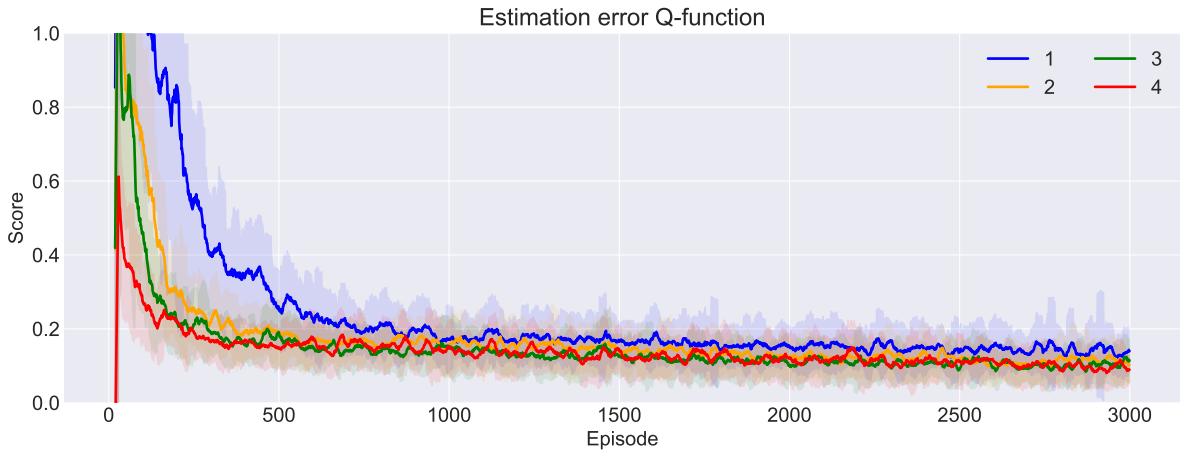


Figure 4.8: Estimated error of the Q-function for various number of hidden layers.

At first sight, increasing the amount of hidden non-linear layers in the NN improves the Average Score, and decrease the Variance Score and Test variance. Figure 4.7 shows that the training is stable for network with more than 1 hidden layer, and Figure 4.8 shows loss prediction errors which are fairly similar. However, the increased benefit of adding more layers is limited from $h = 2$ onward. An additional test is conducted by adding even more hidden layers $h = [4,5,6]$. Unfortunately this test showed that the Average Score does not further increase, and the Variances increase gravely.

Moreover, the amazing performance of the reference $h = 4$ showed too a much lower Average Score and higher Variance. An indication that the initially trained network is accidentally over-fitted to test data. As a result, to combat over fitting and increased variance, a network of $h = 2$ hidden layers will suffice.

4.5. Non-linear Reward Function

This is the first of two tested environment-parameters related to the reward function. In Chapter 3 an in-depth analysis of the generation of the reward function is given. The function follows a linear increase from a empirical determined lower bound ($r = 0$) to the optimal fleet decision profit ($r = +1$). Often, a non-linear reward function is employed to stimulate excellent decision more than sub-optimal decisions. In Figure 4.9 an example is visible for various adjusted rewards $r_{adj} = r^x$

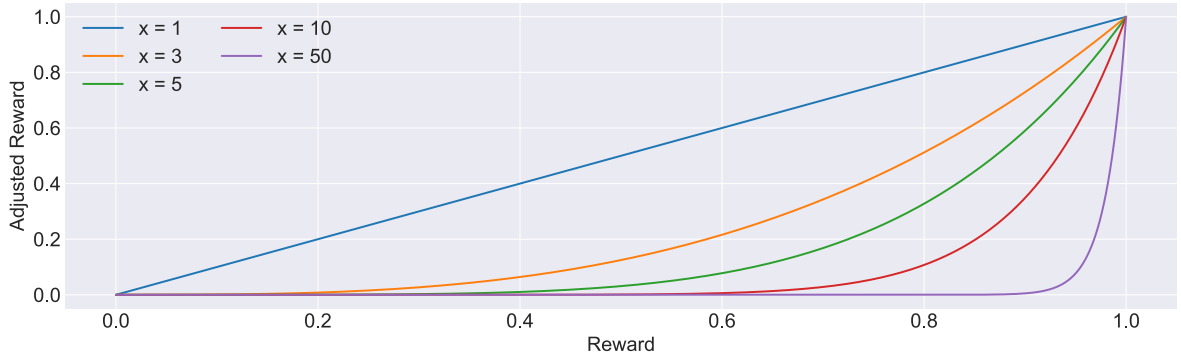


Figure 4.9: Examples of adjusted non-linear reward functions.

In Table 4.5 the testing results of the various non-linear reward functions are visible. The non-linear function $x = 3$ and $x = 5$ show similar results as the linear reward function in Average Score, Average Variance Score, and Variance of the Average Score. For $x = 10$ the Average Score seems to decrease, and at $x = 50$ almost 10% of the Average Score is lost. The training score in Figure 4.10 shows deceiving results, as the reward's non-linearity increases the training score becomes lower. This behaviour is due to the sparsity of rewards of increasing non-linear reward functions. Figure 4.11 shows increasing training loss for higher non-linear reward functions due to the increased variance induced by the sparsity of rewards.

The results suggest that $x = 1,3,5$ are the optimal adjusted reward parameters. The Average Scores are identical; however, the linear reward has a lower Variane of the Average Score, but the non-linear ones improves on the Average Variance Score. The choice is made to train the NN with the linear reward to decrease the variance during the training of the NN, and reduce the sparsity of positive rewards.

	DD	DS	SD				
			x=1	x=3	x=5	x=10	x=50
Average Score	66.3%	84.46%	79.67%	79.69%	79.61%	77.7%	70.55%
Av. Var. Score	6.833	1.274	2.6	1.74	1.62	1.51	2.82
Var. of Av. Score			5.458	6.546	7.737	1.885	45.097
Time [min]			10.09	10.07	10.08	10.07	10.09

Table 4.5: Testing results of DQN for various non-linear reward functions.

Figure 4.10: Training score of DQN for various non-linear reward functions.

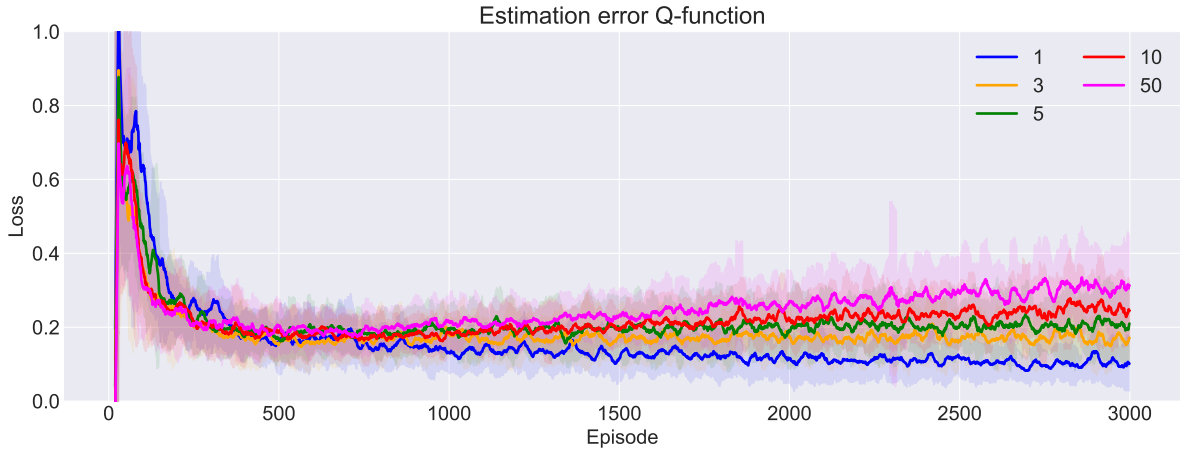


Figure 4.11: Estimated error of the Q-function for various non-linear reward functions.

4.6. Lower Bound

When comparing the profit of the optimal fleet decision C^t_{FPM} and the fleet decision of the agent C_{FAM}, the lower bound lb is a percentage of the optimal profit. The linear reward function starts to accumulate rewards if the agent's profit surpasses the lower bound profit. To establish a set of lower bounds an IQR range of the actions and the accompanying profits for different air travel demand is sampled. In Table 4.6 a set of lower bounds adopted from the empirically established IQR from all possible action reward is depicted. As an example: The LW (lower whisker) of lower bound means that almost all actions will receive a reward higher than zero. A higher lb such as the MED (median) means that half of fleet decision profits C_{FAM} will receive a reward higher than zero, and half of the fleet decisions will not. The UW (upper whisker) as lb results in a very sparse reward function where one if the action is the optimal action, a reward of +1 is received.

	UW	UQ	MED	LQ	LW	LB
lb	1.000	0.998	0.985	0.967	0.934	0.900

Table 4.6: IQR of the lower bounds of the reward function.

Table 4.7 shows the testing results of six identical DQN-networks trained from an environment

with varying lower bounds in the reward function. It is clearly visible that a lower bound in the extremes of the IQR performs poorly. If the lb is too high, the reward generation is too sparse and DQN-model struggles to learn and the variance increases. The MED lower bound is chosen for the fleet planning problem.

	DD	DS	SD					
			UW	UQ	MED	LQ	LW	LB
Average Score	66.3%	84.46%	74.08%	77.88%	79.42%	79.18%	76.39%	75.0%
Av. Var. Score	6.833	1.274	2.23	1.57	1.54	1.65	1.99	2.12
Var. of Av. Score			18.123	8.044	8.024	8.54	13.42	20.547
Time [min]			10.94	10.92	10.9	10.88	10.88	10.88

Table 4.7: Testing results of various lower bounds.

Figure 4.12: Training score of DQN for various lower bounds.

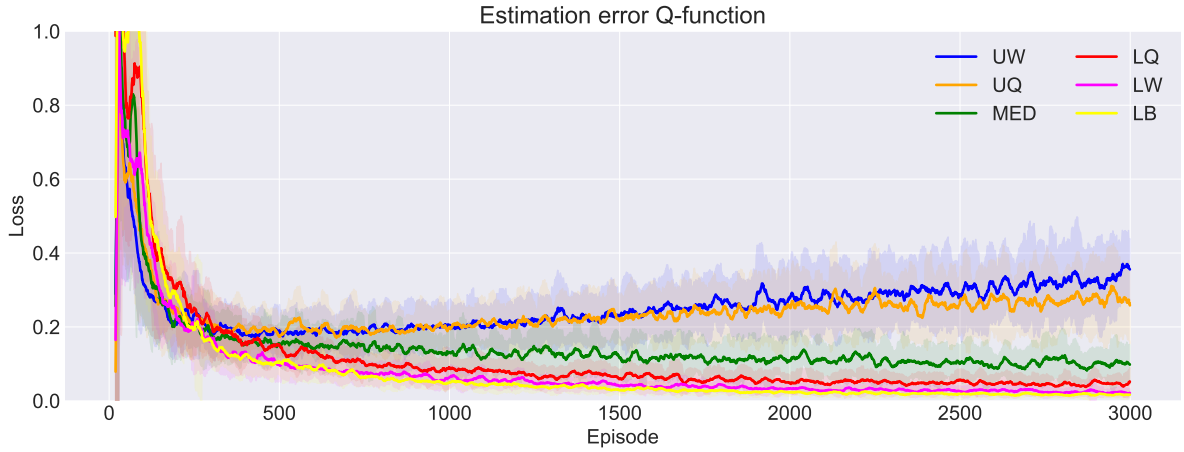


Figure 4.13: Estimated error of the Q-function for various lower bounds.

This behaviour is clearly visible in Figure 4.13 where the loss function of the UW and UQ lower bounds increases throughout the learning due to the increasing variance. Secondly, a low lower bound is also disadvantageous as the DQN-model can not identify the proper rewards. As too many sub-optimal actions receive a high reward, the gradient loss diminishes, and the model loses detail.

Again, in Figure 4.13 it is visible how the loss function approaches zero, not because the DQN has perfectly learned the policy; but because every all rewards to actions are approaching +1. Figure 4.12 again shows a deceiving image as the rewards of the actions is under- or overestimated for lower bounds too high or too low. As a result, the MED value of the IQR-range shows the best Average Score and Variance; and is therefor chosen as the lower bound in the reward function.

5

Conclusions & Recommendations

In this chapter, a look back is taken to the initial research question given the work presented. The purpose of this research is to answer the research question:

'Can a model-free reinforcement learning model learn the long-term dynamic policy in a fleet planning problem under demand uncertainty.'

From the main research question multiple smaller sub-question were devised. In the remainder of this chapter, a step-by-by assessment of the sub-question is given to evaluate if the main research question is solved.

1. Which elements and modelling techniques from literature can be used to design a model-free reinforcement learning program that successfully predicts the optimal policy?

In the initial phase of the thesis research, a literature study was devised, summarized in part II. A detailed overview is given of all the different techniques developed to solve optimization problems with, and without, uncertain parameters. Moreover, the implementation of these techniques on the fleet planning problem is investigated under various uncertainty parameters. Stochastic learning techniques can better capture uncertain parameters characterized as probability distributions. However, multi-period stochastic programs is still an active research stream due to their high complexity and exponential growth in size. New artificial intelligence techniques such as ADP/RL bring a novel solution to large scenario trees which are to computational expensive to solve. In recent years, model-free reinforcement learning has shown major success in modelling long-term strategic policies. The application of model-free modelling in the fleet planning problem has never been done before and therefore shows a promising research gap.

2. How should the learning agent and environment architecture be designed to learn the fleet planning policy?

Three sub-components are designed to model all the relevant aspects to the fleet planning problem and decision-maker: The reinforcement learning agent, the environment model, the air-travel demand forecaster. It is assumed the states represent a fully observable MDP, which implies that the observable state a sufficient statistic of the future to identify the appropriate actions. A value-based approach is chosen, where a deterministic policy devises a fleet decision/action given the states of the airline and network. The deep Q-network of Mnih et al. (2015) is adopted as a reinforcement learning agent. The neural network allows to learn non-linear approximations, and the replay buffer

increases the sample efficient learning from the experiences.

In the environment model, the state transition takes place and the action is assessed through a reward function. The reward is calculated by comparing the generated profit of the agent's fleet decisions with a Fleet Assignment Model (FAM) to the profit of the optimal fleet decisions calculated with a Fleet Planning Model (FPM).

The Ornstein-Uhlenbeck process is introduced as a forecaster of air-travel demand similar to Sa et al. (2019). Instead of building demand scenario's through simulations, the sampled demand is directly used as an evolution of demand for an episode in the reinforcement learning model. The mean-reverting Ornstein Uhlenbeck (OU) process is adapted to better resemble the stochastic nature of air-travel demand. Next to the sampled air-travel demand growth on every market is influenced by an overarching network demand growth, and the long-term mean growth value is replaced by a sampled value from a probability distribution to mimic the stochasticity of long-term air-travel demand.

3. How can the sub-components be implemented in a model and trained efficiently?

The reinforcement learning model is trained for a predetermined set of episodes. At the start of the training process, the OU parameters for the network and every market are estimated. Which will remain the same during the training process. At the start of every episode, the demand growth of every market is sampled, for the complete planning horizon. Next, the evolutions of demand are used to model the optimal fleet policy and accompanied profit with the FPM. The optimization of the FPM is the computational bottleneck of the training process. As the size of the fleet planning problem increases (increasing number of airport, aircraft types, and time-horizon), so does runtime increases exponentially. A side-investigation showed that the MIP-gap of $1E-3$ showed reasonable computational times around 5 seconds with little effect on the reward calculation.

The reinforcement learning loop is started by feeding the initial state through the neural network to obtain the state-action values. The state of the dynamic policy consists of the current period, the current fleet, and the current market demand. Using the ϵ-greedy policy, a fleet action is chosen and reported to the environment model. In the environment model, the reward of the fleet action is calculated by optimising the FAM with the fleet action and the revealed market demand and comparing the observed profit to the optimal profit calculated in the FPM optimisation. The reward, state, next state, and fleet decision/action is stored in the replay buffer, and the transitioned state revealed to the agent until the terminal state is reached. To increase the generation of experiences and the efficient usage of the FPM, in the early stages of the learning, multiple RL loops are run for a single evolution of demand. Because the exploration of policy is still very high, the actions are random, and the state-action space can be explored faster.

The network is trained using a mini-batch of samples uniformly drawn from replay buffer. The loss function is optimized using Adam, and a target network is used to stabilize the learning process, and every 10 episodes the target network is updated by the policy network. After training, the model transitions to the next episode and the process is repeated with the sampling of air-travel demand.

The performance of the RL-model is improved by tuning of the (hyper)parameters of the Neural Network and environment. The efficiency of the reward generation is tuned through investigating the relative error by increasing the MIP-gap of the FPM and FAM. And the optimal features are selected through a Genetic Algorithm.

4. What is the performance of the reinforcement learning model and how does it perform against conventional fleet planning methods.

To evaluate the performance of the model-free reinforcement learning approach, a case-study con-

ducted on a domestic US 10 airport network, with 2 types of aircraft, for 5 periods (each consisting of 2 years). The reinforcement learning model is trained run for 5000 episodes and the DQN-model trained on 320,000 experiences. In the case study, two networks are trained to create two policies: a Stochastic Static (SS) and a Stochastic Dynamic (SD) policy. The difference between the latter is that the SS network does observe the market demand in the state. As a result, the policy is blind to the evolution of demand and the fleet policy 'static'. Three demand scenarios are devised with different parameters for the variance of the long-term mean growth and shock terms. The Average Demand (AD), the Dominant Market Demand (DMD), and Dominant Network Demand (DND) scenario. All scenarios differ in sampled stochasticity of growth and long-term mean growth and have therefore different effects on the results of the RL model. A validation set of 25 air-travel demand trajectories is generated to assess the training performance and an evaluation-set of 50 samples to test the best network. To assess the performance of two deterministic policies are developed, the Deterministic Static (DS) and Deterministic Dynamic (DD) to assess respectively the SS and SD policy.

The dynamic policy outperformed the static policy for every demand scenario. This indicates that the SD neural network learns generalizations and abstractions from the air-travel demand. Hence, it was proven that re-evaluation of the current air-travel demand increases the predictability of the optimal fleet decision and thus performance fleet plan.

The AD scenario had a relatively low smoothing of the shock term, and lower variance of the long-term mean growth, compared to the DND and DMD. As a result, the stochasticity was more visible in the sampled market and network demand. The SS and SD network performed near-optimal in the AD scenario. The dynamic policies showed only an average performance decrease of 3% during validation compared to the deterministic policies. During testing the SS neural network outperformed the DS counterpart; and, the SD network performed near-optimal to the DD approach, and both the SS and SD's variances were higher.

The divergence of the long-term mean growth has a large influence on the performance of both the dynamic and stochastic policies. The deterministic policies always assume the demand will follow the historical mean growth and therefore perform poorly on high network dispersions of demand modelled in DND. The stochastic policies, on the other hand, can learn the dispersion of mean growth trends better than the static counterparts. Especially the SD outperforms the DD both in variance and score.

The DND network shows much lower divergence and stochasticity than the other scenario's. This is attributed to the fact that the high stochastic and diverging network demand growth is mitigated in the market demand sampling. The resulting cumulative demand shows less variance, stochasticity, and resembles the Ornstein-Uhlenbeck sampling process employed by Sa et al. (2019). Both the deterministic and stochastic trained policies show high scores with low variance. This was related since the FAM optimisation for various market demand evolution redistributes the fleet easily.

Given the sub-questions and the findings above, it can be concluded that the model-free reinforcement learning model can learn the fleet planning problem under demand uncertainty. Through a case study, it was proven that the stochastic policy was able to generate fleet planning policies with similar score and variance to the deterministic benchmark for various demand scenarios. And, the inclusion of the demand parameters in the state space resulted in a dynamic policy which outperformed the static policy consistently.

Finally, a few recommendations for future work are devised. To begin, it was observed that increases stochasticity of air-travel demand deteriorates the performance of the stochastic trained policies to their deterministic counterparts. Overall, these training of validation of the stochastic models suffers from high variance. This is attributed to the assumption that the process is a fully

observable MDP. It can be argued that the current state of the air-travel demand is not predictively sufficient for future observation of air-travel demand. The action of a state is evaluated after two revelations of growth in air-travel demand, and can abruptly change due to the shock term of the OU sampling process. If the state is not sufficient statical of the future, it should be assumed that the fleet planning problem is a Partially Observable Markov Decision Process (POMDP). In a POMDP, the state is not fully observable but a probability distribution over a set of possible states. As a result, the agent's decisions make fleet decisions under the uncertainty of demand values or states. In future work, it should be investigated on a more detailed level if POMDP representation of the fleet planning problem is a viable solution and could potentially improve fleet predictions.

Secondly, the optimisation of the optimal fleet policy and optimal profit in the FPM proved to be the bottleneck in terms of computational time. To improve the computational efficiency of the training process, the default gap of the optimality gap is increased to speed up the optimisation of the FPM and FAM. As a result, the estimations of the profits and consequently the reward of the fleet decisions start to deviate from the true value, introducing additional noise. Parallel to this, the computational complexity of the FPM increases exponential with larger networks, more aircraft types, or periods. Moreover, the relative influence of a fleet decision on the profit becomes smaller with a growing airport network, and thus requires high precision to generate a meaningful reward. This discrepancy will prove a problem and future work should investigate methods to evaluate the performance of fleet decisions without the usage of optimisation algorithms.

The original Ornstein-Uhlenbeck demand forecaster showed initially good results on single market simulations. Although proved to have a converging effect on the cumulative demand, with a loss in network stochasticity. The proposed adaptations showed a much better representation of the air-travel demand both on market and network level. However, the standard deviation of the long-term mean growth was estimated; and changes of the standard deviation of the shock terms showed very different demand trajectories. Moreover, the performance of the fleet policies was very sensitive to the demand scenario. It raised the question of how the networks, trained on a scenario, would perform on a different scenario. Future research should investigate the influence of the standard deviation parameters and trained networks.

This research focused on the air-travel demand as the only uncertain parameter in the fleet planning problem. In reality, more parameters exist which influence the fleet planning problem and optimal fleet decision (e.g. Fuel price, competition, aircraft failure etc.). Further research should investigate the implementation of other uncertain parameters, or an ensemble of uncertain parameters, in the fleet planning problem and the performance of the model-free reinforcement learning method.

In previous research, Requeno and Santos (2018) developed a model-based adaptive dynamic program to solve the fleet planning problem successfully. However, a less sophisticated demand sampling model and fleet planning optimisation was employed, a comparison of the model-free versus the model-based approach is missing. Future work should perform a quantitative and qualitative comparison of the two approaches in a fleet planning environment under uncertainty.

Bibliography

O. Anschel, N. Baram, and N. Shimkin. Averaged-dqn: Variance reduction and stabilization for deep reinforcement learning. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 176–185. JMLR. org, 2017.

T. E. Bartlett. An algorithm for the minimum number of transport units to maintain a fixed schedule. *Naval Research Logistics Quarterly*, 4(2):139–149, jun 1957. ISSN 00281441. doi: 10.1002/nav.3800040205. URL http://doi.wiley.com/10.1002/nav.3800040205.

M. Bazargan and J. Hartman. Aircraft replacement strategy: Model and analysis. *Journal of Air Transport Management*, 25:26–29, 2012. URL http://www-01.ibm.com/software/integration/optimization/cplex-optimizer/.

R. Bellman. The theory of dynamic programming. *Bulletin of the American Mathematical Society*, 60(6):503–515, 1954.

P. Belobaba, A. Odoni, and C. Barnhart. *The global airline industry*. John Wiley & Sons, 2015.

A. Ben-Tal, L. El Ghaoui, and A. Nemirovski. *Robust optimization*, volume 28. Princeton University Press, 2009.

J. F. Benders. Partitioning procedures for solving mixed-variables programming problems*. Technical report, 1962. URL https://link.springer.com/content/pdf/10.1007/bf01386316.pdf.

Y. Bengio, A. Lodi, and A. Prouvost. Machine learning for combinatorial optimization: a methodological tour d'horizon. *arXiv preprint arXiv:1811.06128*, 2018.

C. Berner, G. Brockman, B. Chan, V. Cheung, P. Dębiak, C. Dennison, D. Farhi, Q. Fischer, S. Hashme, C. Hesse, et al. Dota 2 with large scale deep reinforcement learning. *arXiv preprint arXiv:1912.06680*, 2019.

S. Bouarfa, H. Blom, R. Curran, and K. Hindriks. A study into modeling coordination in disruption management by airline operations control. *AIAA AVIATION 2014 -14th AIAA Aviation Technology, Integration, and Operations Conference*, 06 2014. doi: 10.2514/6.2014-3146.

L. Buşoniu, B. De Schutter, and R. Babuška. Approximate dynamic programming and reinforcement learning. In *Interactive collaborative information systems*, pages 3–44. Springer, 2010.

N. Casas. Deep deterministic policy gradient for urban traffic light control. *arXiv preprint arXiv:1703.09035*, 2017.

P. Clarck. *Buying the Big Jets*. 2007. ISBN 9780754670902.

R. J. Dakin. A tree-search algorithm for mixed integer programming problems. *The Computer Journal*, 8(3):250–255, mar 1965. doi: 10.1093/comjnl/8.3.250. URL https://academic.oup.com/comjnl/article-lookup/doi/10.1093/comjnl/8.3.250.

G. B. Dantzig and D. R. Fulkerson. Minimizing the number of tankers to meet a fixed schedule. *Naval Research Logistics Quarterly*, 1(3):217–222, sep 1954. URL http://doi.wiley.com/10.1002/nav.3800010309.

G. B. Dantzig and P. Wolfe. Decomposition Principle for Linear Programs. *Operations Research*, 8 (1):101–111, feb 1960. ISSN 0030-364X. doi: 10.1287/opre.8.1.101. URL http://pubsonline.informs.org/doi/abs/10.1287/opre.8.1.101.

L. Davis. Handbook of genetic algorithms. 1991.

Santos B. de Koning, M.C.T.C. AE4020: Literature Study: Fleet Planning under Uncertainty. *Delft University of Technology: Aerospace Engineering*, 2019.

D. Erhan, Y. Bengio, A. Courville, and P. Vincent. Visualizing higher-layer features of a deep network. *University of Montreal*, 1341(3):1, 2009.

Y. Ermoliev. On the method of generalized stochastic gradients and quasi-féjer sequences. *Cybernetics*, 5(2):208–220, Mar 1969. ISSN 1573-8337. doi: 10.1007/BF01071091. URL https://doi.org/10.1007/BF01071091.

V. François-Lavet, P. Henderson, R. Islam, M. G. Bellemare, J. Pineau, et al. An introduction to deep reinforcement learning. *Foundations and Trends® in Machine Learning*, 11(3-4):219–354, 2018.

R. E. Gomory. An algorithm for integer solutions to linear programs. *Recent advances in mathematical programming*, 64(260-302):14, 1963.

I. Goodfellow, Y. Bengio, and A. Courville. *Deep Learning*. MIT Press, 2016. http://www.deeplearningbook.org.

R. D. Gritta, E. Lippman, and G. Chow. The impact of the capitalization of leases on airline financial analysis: An issue revisited. *LOGISTICS AND TRANSPORTATION REVIEW*, 30(2), jun 1994. URL https://trid.trb.org/view/408366.

M. Hessel, J. Modayil, H. Van Hasselt, T. Schaul, G. Ostrovski, W. Dabney, D. Horgan, B. Piot, M. Azar, and D. Silver. Rainbow: Combining improvements in deep reinforcement learning. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

J. L. Higle and S. Sen. *Stochastic Decomposition : a Statistical Method for Large Scale Stochastic Linear Programming*. 1996. ISBN 9781461368458.

C.I. Hsu, H.C. Li, S.M. Liu, and C.C. Chao. Aircraft replacement scheduling: a dynamic programming approach. *Transportation research part E: logistics and transportation review*, 47(1):41–60, 2011.

P. Kall and S. W. Wallace. *Stochastic Programming Second Edition*. 1994. URL https://www.stoprog.org/sites/default/files/files/manujw.pdf.

H. L. Khoo and L. E. Teoh. An optimal aircraft fleet management decision model under uncertainty. *Journal of Advanced Transportation*, 48(7):798–820, nov 2014. URL http://doi.wiley.com/10.1002/atr.1228.

D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

D. Kirby. Is Your Fleet the Right Size? *Journal of the Operational Research Society*, 10(4):252–252, dec 1959. ISSN 0160-5682. doi: 10.1057/jors.1959.25. URL https://www.tandfonline.com/doi/full/10.1057/jors.1959.25.

W. Kool, H. van Hoof, and M. Welling. Attention, Learn to Solve Routing Problems! mar 2018. URL http://arxiv.org/abs/1803.08475.

S. Lam, L. Lee, and L. Tang. An approximate dynamic programming approach for the empty container allocation problem. *Transportation Research Part C: Emerging Technologies*, 15(4):265–277, aug 2007. ISSN 0968090X. doi: 10.1016/j.trc.2007.04.005. URL https://linkinghub.elsevier.com/retrieve/pii/S0968090X07000277.

F. Li, J. Johnson, and S. Yeung. Convolutional neural networks for visual recognition, lecture 4: Backpropagation and neural networks. http://cs231n.stanford.edu/slides/2018/cs231n_2018_lecture04.pdf. [Online; accessed 14-October-2019].

G.F. List, B. Wood, L. K. Nozick, M. A. Turnquist, D. A. Jones, E. A. Kjeldgaard, and C. R. Lawton. Robust optimization for fleet planning under uncertainty. *Transportation Research Part E: Logistics and Transportation Review*, 39(3):209–227, 2003.

O. Listes and R. Dekker. A scenario aggregation based approach for determining a robust airline fleet composition. Technical report, 2002. URL https://core.ac.uk/download/pdf/6499148.pdf.

V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, S. Petersen, C. Beattie, A. Sadik, I. Antonoglou, H. King, D. Kumaran, D. Wierstra, S. Legg, and D. Hassabis. Human-level control through deep reinforcement learning. *Nature*, 518, 2015. URL https://www.nature.com/articles/nature14236.pdf.

J.C. Nash. The (Dantzig) simplex method for linear programming. *Computing in Science & Engineering*, 2(1):29–31, 2000. ISSN 15219615. doi: 10.1109/5992.814654. URL http://ieeexplore.ieee.org/document/814654/.

M. Naumann and L. Suhl. How does fuel price uncertainty affect strategic airline planning? *Operational Research*, 13(3):343–362, oct 2013. URL http://link.springer.com/10.1007/s12351-012-0131-0.

C. C. New. Transport Fleet Planning For Multi-Period Operations. *Oper Res Soc*, 26(1):51–66, 1975. URL www.jstor.org.

C. Novoa and R. Storer. An approximate dynamic programming approach for the vehicle routing problem with stochastic demands. *European Journal of Operational Research*, 196(2):509–515, jul 2009. ISSN 03772217. doi: 10.1016/j.ejor.2008.03.023. URL https://linkinghub.elsevier.com/retrieve/pii/S0377221708003172.

P. Odonkor and K. Lewis. Control of shared energy storage assets within building clusters using reinforcement learning. In *ASME 2018 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference*. American Society of Mechanical Engineers Digital Collection, 2018.

C. Olah, A. Mordvintsev, and L. Schubert. Feature visualization. *Distill*, 2(11):e7, 2017.

T. H. Oum, A. Zhang, and Y. Zhang. Optimal demand for operating lease of aircraft. *Transportation Research Part B: Methodological*, 34(1):17–29, jan 2000. URL http://linkinghub.elsevier.com/retrieve/pii/S0191261599000107.

B. Pearce. The shape of air travel markets over the next 20 years. https://www.iata.org/whatwedo/Documents/economics/20yearsForecast-GAD2014-Athens-Nov2014-BP.pdf, 2014. [Online; accessed 4-October-2019].

M. Ponsen, M. E. Taylor, and K. Tuyls. Abstraction and generalization in reinforcement learning: A summary and framework. In *International Workshop on Adaptive and Learning Agents*, pages 1–32. Springer, 2009.

W. B. Powell. Approximate Dynamic Programming-I: Modeling. Technical report, 2009. URL https://castlelab.princeton.edu/html/Papers/EORMS-ADP{_}Modeling{_}Dec72009.pdf.

W. B. Powell. *Approximate dynamic programming : solving the curses of dimensionality*. Wiley, 2011. ISBN 9780470604458. URL https://www.wiley.com/en-nl/Approximate+Dynamic+Programming:+Solving+the+Curses+of+Dimensionality,+2nd+Edition-p-9780470604458.

Pytorch. Reinforcement learning (dqn) tutorial. https://pytorch.org/tutorials/intermediate/reinforcement_q_learning.html, 2020. [Online; accessed 12-02-2020].

A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, and I. Sutskever. Language models are unsupervised multitask learners. *OpenAI Blog*, 1(8), 2019.

M. G.J. Repko and B. F. Santos. Scenario tree airline fleet planning for demand uncertainty. *Journal of Air Transport Management*, 65:198–208, oct 2017. URL https://www-sciencedirect-com.tudelft.idm.oclc.org/science/article/pii/S0969699717302806.

L. Requeno and B. Santos. Multi-period adaptive airline fleet planning problem. *Submitted to: Transportation Science*, 2018.

G. A. Rummery and M. Niranjan. On-line Q-learning using connectionist systems. Technical report, 1994. URL http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.17.2539{&}rep=rep1{&}type=pdf.

C. A. A. Sa, B. F. Santos, and J. B. Clarke. Portfolio-based airline fleet planning under stochastic demand. *Omega*, page 102101, 2019.

N. V. Sahinidis. Optimization under uncertainty: state-of-the-art and opportunities. *Computers & Chemical Engineering*, 28(6-7):971–983, jun 2004. URL https://www.sciencedirect.com/science/article/pii/S0098135403002369?via{%}3Dihub.

B. Santos. Lecture Notes: Network and Fleet Planning, [Accessed on: 2019/10/15]. Technical report, Delft University of Technology, Faculty of Aerospace Engineering, 2017a. URL https://www.brightspace.tudelft.nl.

B. Santos. Lecture Notes: Airline Planning and Optimization, [Accessed on: 2019/10/15]. Technical report, Delft University of Technology, Faculty of Aerospace Engineering, 2017b. URL https://www.brightspace.tudelft.nl.

H. R. Sayarshad and K. Ghoseiri. A simulated annealing approach for the multi-periodic rail-car fleet sizing problem. *Computers & Operations Research*, 36:1789–1799, 2009. doi: 10.1016/j.cor.2008.05.004. URL https://ac.els-cdn.com/S0305054808000993/1-s2.0-S0305054808000993-main.pdf?{_}tid=fe999ce9-068c-479f-8bbf-1c5bfb2d18ca{&}acdnat=1551692050{_}b7084a02fde6d1897dce162e647d001a.

T. Schaul, J. Quan, I. Antonoglou, and D. Silver. Prioritized experience replay. *arXiv preprint arXiv:1511.05952*, 2016.

G. J. Schick and J. W. Stroup. Experience with a Multi-Year Fleet Planning Model. *Omega*, 9(4):389–396, 1981. URL https://ac.els-cdn.com/0305048381900839/1-s2.0-0305048381900839-main.pdf?{_}tid=0f8a9b0a-8f5c-432a-afaa-d41da3792935{&}acdnat=1550760022{_}20835925b51cc9a3bae666a39756cf6e.

A. Shapiro and A. Philpott. A Tutorial on Stochastic Programming. Technical report, 2007. URL https://www2.isye.gatech.edu/people/faculty/Alex{_}Shapiro/TutorialSP.pdf.

A. Shapiro, D. Dentcheva, and A. Ruszczynski. *Lectures on Stochastic Programming Modeling and Theory*. 2009. URL https://www2.isye.gatech.edu/people/faculty/Alex{_}Shapiro/SPbook.pdf.

D. P. Shube and J. W. Stroup. Fleet Planning Model. *Winter Simulation Conference Proceedings*, 1975. URL https://repod.lib.ncsu.edu/handle/1840.4/7294.

D. Silver. Ucl course on rl: Lecture 4 model-free prediction. http://www0.cs.ucl.ac.uk/staff/d.silver/web/Teaching_files/MC-TD.pdf, 2018. [Online; accessed 18-October-2019].

D. Silver, J. Schrittwieser, K. Simonyan, I. Antonoglou, A. Huang, A. Guez, T. Hubert, L. Baker, M. Lai, A. Bolton, et al. Mastering the game of go without human knowledge. *Nature*, 550(7676):354–359, 2017.

N. J. Smelser and P. B. Baltes. *International encyclopedia of the social & behavioral sciences*, volume 11. Elsevier Amsterdam, 2001.

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

G. Tesauro. Temporal difference learning and td-gammon. *Communications of the ACM*, 38(3):58–68, 1995.

D. Urieli and P. Stone. Model-Selection for Non-Parametric Function Approximation in Continuous Control Problems: A Case Study in a Smart Energy System. Technical report, 2013. URL https://www.cs.utexas.edu/{~}pstone/Papers/bib2html-links/ECML13-urieli.pdf.

H. van Hasselt, A. Guez, and D. Silver. Deep reinforcement learning with double q-learning. corr (2015). *arXiv preprint cs.LG/1509.06461*, 2015.

R. M. Van Slyke and R. Wets. L-shaped linear programs with applications to optimal control and stochastic programming. *SIAM Journal on Applied Mathematics*, 17(4):638–663, 1969.

Z. Wang, T. Schaul, M. Hessel, H. Van Hasselt, M. Lanctot, and N. De Freitas. Dueling network architectures for deep reinforcement learning. *arXiv preprint arXiv:1511.06581*, 2016.

C.J.C.H. Watkins. Learning from delayed rewards. 1989.

K. Weinberger. Machine learning for intelligent systems: Lecture 12 bias-variance tradeoff. https://www.cs.cornell.edu/courses/cs4780/2018fa/lectures/lecturenote12.html, 2018. [Online; accessed 14-01-2020].

J. K. Wyatt. Optimal Fleet Size. *Journal of the Operational Research Society*, 12(3):186–187, sep 1961. ISSN 0160-5682. doi: 10.1057/jors.1961.30. URL https://www.tandfonline.com/doi/full/10.1057/jors.1961.30.

F. Xie and Y. Huang. A multistage stochastic programming model for a multi-period strategic expansion of biofuel supply chain under evolving uncertainties. *Transportation Research Part E: Logistics and Transportation Review*, 111:130–148, 2018.

H. Xu. CS567: Stochastic Linear/Integer Programming: Algorithms for Two-Stage Linear Recourse Problem, 2008. URL `http://cgm.cs.mcgill.ca/{~}avis/courses/567/notes/stoch3.pdf`.