UNIVERSIDADE D
COIMBRA

David Joaquim Marques Antunes

INTEGRATED SCHEDULE RECOVERY: A MULTI-AGENT APPROACH

Ph.D. Thesis in the Doctoral Program in Transport Systems supervised by Professor António Pais Antunes and Vikrant Vaze presented to the Department of Civil Engineering of the Faculty of Sciences and Technology of the University of Coimbra.

February 2019

INTEGRATED SCHEDULE RECOVERY: A MULTI-AGENT APPROACH

David Joaquim Marques Antunes

Ph.D. Thesis in the Doctoral Program in Transport Systems supervised by Professor António Pais Antunes and Vikrant Vaze presented to the Department of Civil Engineering of the Faculty of Sciences and Technology of the University of Coimbra.

February 2019

UNIVERSIDADE D
COIMBRA

[Page Intentionally Left Blank]

FINANCIAL SUPPORT

This research work was conducted under the MIT-Portugal Program and financed by "Fundação para a Ciência e a Tecnologia" (FCT) through the Ph.D. scholarship with reference SFRH/BD/52360/2013

MIT Portugal

FCT Fundação
para a Ciência
e a Tecnologia

[Page Intentionally Left Blank]

ACKNOWLEDGMENTS

I would like to express my deepest appreciation to Professor Antonio Pais Antunes and Professor Vikrant Vaze. Professor Antunes is a true role model. Professor Vaze's priceless support, selfless time and support kept me going and contributed greatly to this work.

I would like to thank to Professor Gonçalo Correia that was my supervisor during the thesis project, who was fundamental framing the project and who's insights and hard questions made me focus on what is important.

I would also thank to my colleagues, Rodrigo Lopes, Joana Carreira, and Francisco Antunes for all the comments and support.

I would thank the Fundação para a Ciência e a Tecnologia (FCT) and the MIT-Portugal Program granting me the full Ph.D. Scholarship which allowed me to pursue my studies in Portugal and the United States. Special reference must be made to the University of Coimbra and Massachusetts Institute of Technology, where I developed my research, for all the logistical and institutional support.

I would thank to my parents and brother whose unconditional love and support helped me overcome the difficult times during this period. Thank you for always believing in me.

Last but not least, would like to express my gratitude to my family and friends for all the patience, support and care

[Page Intentionally Left Blank]

TABLE OF CONTENTS

[Page Intentionally Left Blank]

LIST OF FIGURES

LIST OF TABLES

ABSTRACT

Delays and disruptions in airline operations annually result in billions of dollars of additional costs to airlines, passengers, and the economy. Airlines strive to mitigate these costs by creating schedules that are less likely to get disrupted or schedules that are easier to repair upon disruptions. New types of aircraft, larger fleets, expanding networks and increasing constraints on crew scheduling from regulators, crew collective bargains, and decreased fares, force airlines to make their operations more efficient. The presence of considerable uncertainty makes this already complex problem faced by the airlines even harder to solve. The air transportation industry has long tradition of using operations research techniques to solve their problems. To cope with their problems the airline industry has used optimization models in one form or another. Hence schedules are trimmed to eliminate buffers and slacks, but this can lead to greater propagation of delays, even due to minor incidents. These intricate problems are hard to understand and all its interactions are virtually impossible to grasp in a purely closed-form mathematical approach.

We present in this thesis several methodological approaches to help researchers and industry practitioners dealing with uncertainty in airline scheduling.

First, we present a robust optimization model for the crew pairing problem, which generates crew schedules that are less likely to get disrupted. Our model allows adding robustness without requiring detailed knowledge of the underlying delay distributions. Moreover, our model allows to capture in detail the delay propagation through crew connections and the complex cost structure of the pay-and-credit crew salary scheme, thus enabling us to find a good tradeoff between the planned costs on one hand and the expected delay and disruption costs on the other hand. Our robust crew pairing model is based on a deterministic crew pairing model formulated as a mixed-integer linear program. The robust version that we propose retains the linearity of the constraints and objective function, and thus can be handled by commercial solvers, which facilitates its implementation in practice. We propose and implement a new solution algorithm for solving our model to optimality. Several optimal solutions with varying robustness levels are compared for the network of a moderate-sized airline in the United States. We test the model's solutions in a simulation environment using real-world delay data. Our simulation results show that the robust crew pairing solutions lead to lower delays and fewer instances of operational infeasibilities, thus requiring fewer recovery actions to address them. We conclude that, with the inclusion of robustness, it is possible to generate crew pairing solutions that significantly reduce the delay and disruption costs with only a small increase in planned costs.

Second, we present a new stochastic simulation platform that is able to simulate the daily operations of an airline allowing industry analysts, practitioners and researchers to evaluate the behavior of an airline network under uncertainty. This simulation platform was developed specifically to be user-friendly and require moderate input needs. We demonstrate the adequacy of an agent-based approach to model a complex system such as an airline. Additionally, we demonstrate that an accurate representation of an airline's operation needs to explicitly consider the airline's response to uncertainty and disruptions, by accurately modeling the airline as a decision-maker in face of uncertainty. This is demonstrated through a detailed case study where we compare two recovery models that, due to their different objectives, lead to different behaviors of the simulated airline. We demonstrate and compare the tradeoffs between cancelations and flight delays that are made by these models to reach to their objectives.

Third, we present a machine learning metamodel approach where we leverage information extracted from the historic data, from previous schedule recovery problems solved by optimization models, to generate immediate solutions for the schedule recovery problem. In our proof of concept case study, we use an artificial neural network that, using historic data of solutions and problems, is able to extract the solution pattern and return the problem solution without needing to re-run the optimization model. We show that it is feasible to use machine learning metamodels to predict solutions of optimization models based on historic data. With an acceptable level of accuracy, the metamodel was able to predict the solution of a recovery optimization model.

RESUMO

Atrasos e perturbações nas operações resultam anualmente em milhares de milhões de dólares de custos adicionais para as companhias aéreas, para os seus passageiros e para a economia em geral. As companhias aéreas tentam mitigar esses custos criando horários com menos probabilidade de serem perturbados. Novos desafios devidos a novos tipos de aeronaves, à expansão de redes, ao aumento das restrições aos horários das tripulações por parte de reguladores e contratos coletivos e à diminuição de tarifas, forçam as companhias aéreas a tornarem as suas operações mais eficientes. A presença de incerteza torna a gestão das operações de companhias aéreas ainda mais complexa. O setor do transporte aéreo tem uma longa tradição de usar técnicas de investigação operacional para resolver os seus problemas operacionais. Neste sentido, os horários são otimizados para reduzir margens, o que pode levar a uma maior propagação de atrasos. São problemas complexos e de difícil compreensão difíceis de traduzir adequadamente por modelos matemáticos.

Apresentamos várias abordagens para ajudar os investigadores e profissionais da indústria a lidar com a incerteza na gestão operacional de companhias aéreas. Primeiro, apresentamos um modelo de otimização robusto para o problema do emparelhamento de tripulações, que gera horários para as tripulações com menos probabilidade de serem afetados por incerteza. O modelo permite adicionar robustez sem exigir um conhecimento detalhado das distribuições subjacentes aos atrasos. Além disso, o modelo permite capturar pormenorizadamente a propagação dos atrasos decorrentes das conexões da tripulação e a complexa estrutura de custos dos salários dos membros das tripulações, permitindo-nos encontrar um bom equilíbrio entre os custos planeados e os custos adicionais relativos a atrasos e perturbações das operações. O modelo robusto de emparelhamento de tripulação proposto é baseado num determinístico formulado como um problema inteiro misto. A versão robusta que propomos mantém a linearidade das restrições e da função objetivo e, portanto, pode ser tratada por softwares comerciais, o que facilita a sua implementação prática. Propomos e implementamos um novo algoritmo para obter soluções ótimas globais para o modelo. Soluções ótimas correspondentes a vários níveis de robustez são comparadas para uma companhia aérea de tamanho moderado que opera nos Estados Unidos. Testamos as soluções do modelo em um ambiente de simulação usando dados de atrasos reais. Os resultados da simulação mostram que as soluções robustas de emparelhamento de tripulações levam a atrasos menores e menos ocorrências de inviabilidade operacional, exigindo menos ações de recuperação para as resolver. Verificámos que, com a inclusão da robustez, é possível gerar soluções de emparelhamento de tripulações que reduzem significativamente os custos com atrasos e perturbações das operações com apenas um pequeno aumento nos custos planeados.

Em segundo lugar, apresentamos uma nova plataforma de simulação estocástica que é capaz de simular as operações diárias de uma companhia aérea, permitindo que analistas, profissionais e pesquisadores do setor avaliem o comportamento da rede de uma companhia aérea sob condições de incerteza. Demonstramos a adequação de uma abordagem baseada em agentes para modelar um sistema complexo como uma companhia aérea. Além disso, demonstramos que uma representação adequada da operação de uma companhia aérea carece da consideração explícita da respetiva resposta a incertezas e perturbações. O funcionamento da plataforma é exemplificado por meio de um estudo de caso detalhado em que comparamos dois modelos de recuperação de horários de companhias aéreas que, devido terem objetivos distintos, levam a diferentes comportamentos da companhia aérea simulada. Determinamos e comparamos os trade-offs entre cancelamentos e atrasos de voo que são feitos por esses modelos em função dos objetivos da companhia.

Em terceiro lugar, apresentamos uma meta-modelo “machine learning” onde aproveitamos informações extraídas dos dados históricos de problemas de recuperação de horários de companhias aéreas resolvidos por modelos de otimização para gerar soluções imediatas. No estudo de caso que desenvolvemos como prova de conceito, usamos uma rede neuronal artificial que, a partir de dados históricos de soluções e problemas, é capaz de extrair o padrão da solução e gerar a solução do problema sem precisar de resolver novamente o modelo de otimização. Mostramos que é possível usar meta-modelos de “machine learning” para prever soluções de modelos de otimização baseados em dados históricos com um nível aceitável de precisão. O meta-modelo conseguiu prever a solução de um modelo de otimização de recuperação de horários.

Essentially, all models are wrong, but some are useful.

Box and Draper, 1987

[Page Intentionally Left Blank]

1 INTRODUCTION

The work of this thesis is focused on airline operations, scheduling, and proactive and reactive measures to uncertainty. First, we present a robust planning model for the crew scheduling problem. It is a model with a proactive approach to disruptions. Second, we present a simulation platform where we explain the need to properly represent all agents and the airlines decision making process to be able to simulate the operations of an airline. This stochastic simulation model underlying the platform is able to capture in detail the operation of an airline. Useful for researchers and industry practitioners alike, it allows testing schedules, scenarios, and effectiveness of reactive and proactive measures to tackle uncertainty. Finally, this thesis presents a machine learning metamodel that can learn from the historic data of an airline and generate solutions to recovery from disruptions, improving the process of solving the schedule recovery problem significantly in terms of efficiency.

1.1 Background

The airline industry in 2018 generated 800 billion US dollars in revenues (IATA, 2018). It is easy to understand that inefficiencies and uncertainties affecting the operations of an airline can lead to losses of millions of dollars every year. A comprehensive study analyzing data from 2007 sets the cost of delays in aviation for the US economy at 32.9 billion US dollars (Ball et al., 2010). Additionally, the increase in passengers has been exponential in the last fifty years; see Figure 1.1. In 2017, the total passengers carried by air worldwide rose to approximately 4 billion (IATA, 2018; World Bank, 2018).

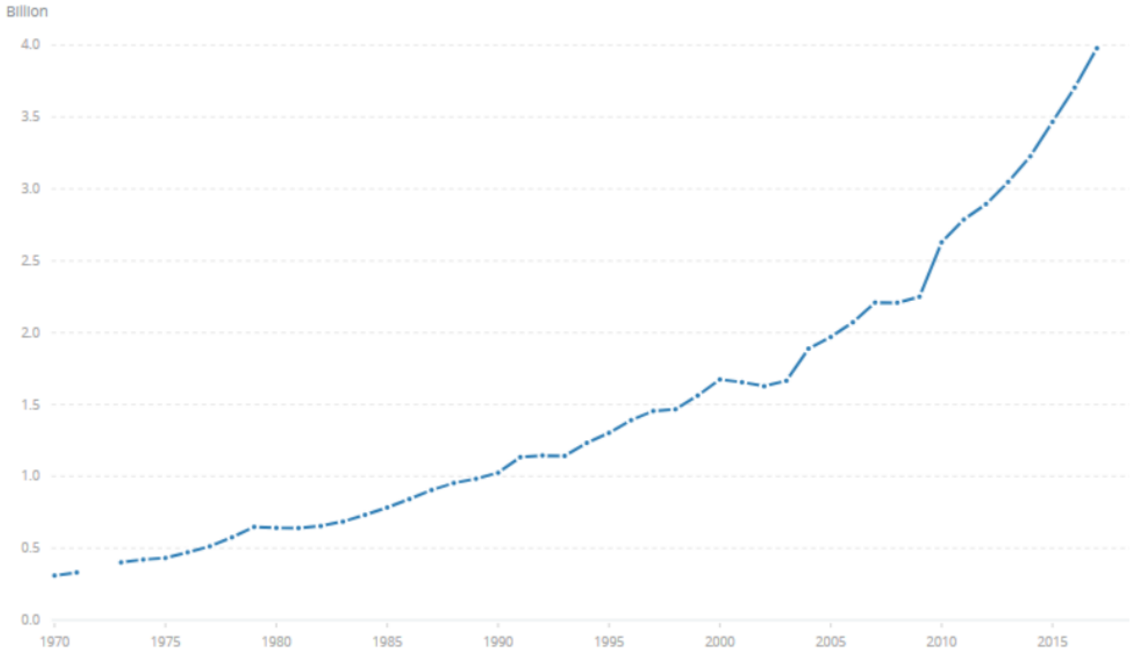


Figure 1.1. Passengers carried worldwide from 1970 to 2017 (World Bank, 2018)

This increase is not accompanied by the profits since there is a high level of competition. One factor of this is that the margin levels are moderate in comparison to other industries. Average profit margins were around five percent in 2017 (IATA, 2018). The mature markets, United States and Europe, are in consolidation (World Bank, 2017) and are characterized by a high level of competition. Increasing demand, high level of competition and relatively low margins lead to a strong need of efficient operations. Being the operations of an airline highly complex, the industry uses advanced models to support their decision processes and develop the schedules for all resources under their control. Efficient schedules naturally involve small buffers, which are brittle and easy to be disrupted. Hence, one of the biggest challenges that airlines face nowadays is how to operate in face of uncertainty.

There are several sources of uncertainty – for instance, fuel costs. These costs are highly uncertain, however they are outside the scope of this thesis, because fuel cost uncertainty can be considered a strategic problem, and we want to focus on operational uncertainty. By operational uncertainty we mean uncertainty that will directly affect the operations of an airline. i.e. uncertainty that will directly affect the schedules and create delays and or cancellations. According to the Bureau of Transportation Statistics (BTS), in the US there are five types of causes of delays considered in the Airline On-Time Time Performance database (BTS, 2016):

- **Air Carrier**: The cause of the cancellation or delay was due to circumstances within the airline's control (e.g. maintenance or crew problems, aircraft cleaning, baggage handling, fueling, etc.);

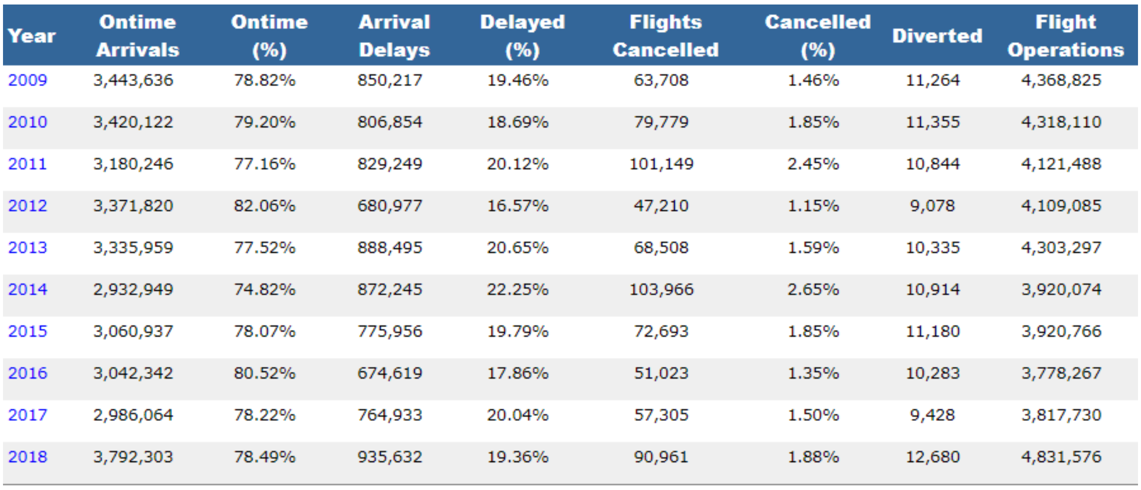
- **Extreme Weather**: Adverse meteorological conditions that, in the judgment of the carrier, delay or prevent the operation of a flight (e.g. tornado, blizzard, hurricane, etc.);

- **National Aviation System (NAS):** Delays and cancelations attributable to the NAS that refer to a broad set of conditions, such as non-extreme weather conditions, airport operations, heavy traffic volume, and air traffic control;

- **Late-arriving aircraft:** A previous flight with the same aircraft arrived late, causing present flight to depart late. (This is an extremely important point since it corresponds to propagated delays in the network. The caveat here is that some propagated delays are classified under the Air Carrier point, as we will explain further on in greater detail);

- **Security:** Delays or cancelations caused by the evacuation of a terminal or concourse, re-boarding of aircraft because of security breach, inoperative screening equipment, and/or long lines in excess of 29 minutes at screening areas.

If we observe the data of the Airline on-Time Performance database at a glance (Figure 1.2), we see that around eighty percent of the flights in the United States are considered on time and one to two percent of the flights are canceled. This is similar in the European market. According to EUROCONTROL (2018), around eighty percent of the flights are considered on time in 2017 in Europe.



Year	Ontime Arrivals	Ontime (%)	Arrival Delays	Delayed (%)	Flights Cancelled	Cancelled (%)	Diverted	Flight Operations
2009	3,443,636	78.82%	850,217	19.46%	63,708	1.46%	11,264	4,368,825
2010	3,420,122	79.20%	806,854	18.69%	79,779	1.85%	11,355	4,318,110
2011	3,180,246	77.16%	829,249	20.12%	101,149	2.45%	10,844	4,121,488
2012	3,371,820	82.06%	680,977	16.57%	47,210	1.15%	9,078	4,109,085
2013	3,335,959	77.52%	888,495	20.65%	68,508	1.59%	10,335	4,303,297
2014	2,932,949	74.82%	872,245	22.25%	103,966	2.65%	10,914	3,920,074
2015	3,060,937	78.07%	775,956	19.79%	72,693	1.85%	11,180	3,920,766
2016	3,042,342	80.52%	674,619	17.86%	51,023	1.35%	10,283	3,778,267
2017	2,986,064	78.22%	764,933	20.04%	57,305	1.50%	9,428	3,817,730
2018	3,792,303	78.49%	935,632	19.36%	90,961	1.88%	12,680	4,831,576

Figure 1.2. On-time performance flight delays at a glance (BTS, 2018)

As referred before, uncertainty will eventually affect an airline in one way or another. If an event, or set of events, are severe enough they will result in disruptions, that manifest typically in delays or cancelations. To handle these issues, airlines have a department commonly called Airline Operational Control Center (AOCC). The mandate of an AOCC can be resumed as: "guarantee smooth operations and solve any issue of the operations, proactively and reactively". According to Ball et al. (2006), an AOCC is comprised of :

- **Airline operation controllers:** At the helm and responsible for aircraft re-routing, flight cancelations, and delays decisions;
- **Crew planners:** Solve disruptions of crew schedules and coordinate with airline operations controllers so solutions remain feasible for crew;
- **Costumer service coordinators:** Solve disruptions for passengers and coordinate with airline operations controllers to evaluate the impact on passengers of disruption solutions;
- **Dispatchers:** communicate directly with the pilots providing flight plans; and,
- **Air traffic control groups:** collect and provide information related to the ATC systems and share it with the airline operations controllers.

Of course, this differs greatly from airline to airline, depending on the size, regulatory framework, region, tools and structure of the airline.

Most times the schedule recovery problems are solved sequentially; see Rosenberger et al. (2002). This is a relict coming from the planning phase where problems are also tackled in a sequential manner. The airline planning and airline recovery procedures are commonly performed as presented in Figure 1.3.

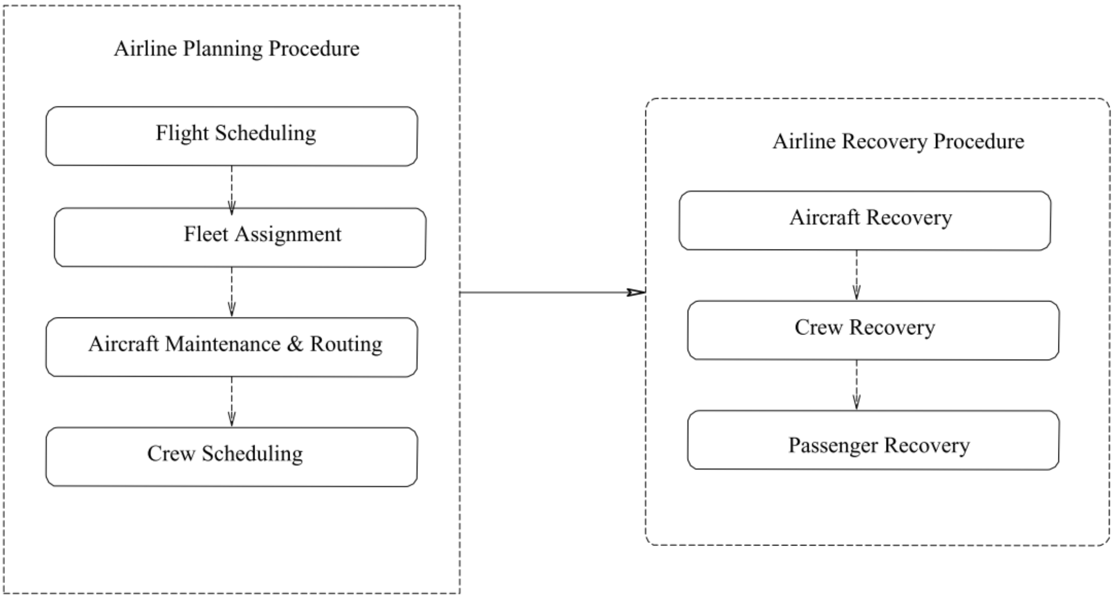


Figure 1.3. Airline decision process (Kasirzadeh et al., 2014)

First, the more expensive and scarce resources are scheduled: the aircraft. This step is generally called as Flight Scheduling or Schedule Generation. Here all flights to be flown are scheduled considering the different markets.

The next step is Fleet Assignment. In this phase, the available aircraft types are assigned to flights. These flights can have several stops at different airports. Each connection between a pair of airports is called a flight leg. The objective is, with the different groups of aircraft existing in the airline, to maximize the profit by serving markets. The revenue from a flight leg depends on the market for that leg and the type of aircraft that is used for flying it. Besides the aircraft type, several other constraints are considered in this phase. For instance: legal aspects and connecting flights.

After the allocation of the fleet type, the Tail Assignment or Aircraft Routing is made. In this step, individual aircraft are allocated. Since the individual identification number of an aircraft is written on the tail, this process is known as Tail Assignment. Here the main constraint is the maintenance that the aircraft must undergo, which vary for each aircraft type, age and flown hours.

Being everything related to the aircraft already handled, the next resources to be planned are the crews. Crews are very important for airlines and, second to fuel, they are usually the largest single cost of an airline (Kohl & Karisch 2002; Ball et al. 2010; Bierlaire & Gallay 2010). The scheduling of the crews in an airline is generally divided in two sub-problems: Crew Pairing and Crew Assignment. First the crew paring is solved in which each flight is exactly covered by one, and only one, pairing that produces the minimum cost. These pairing include rest periods, training time and vacations and must then adhere to several constraints, for instance: legal restrictions on flight hours and minimum rest among others that may be imposed by the labor organizations. The airline itself can impose some constraints to attend to details of the network or add robustness to the schedules. The second step, Crew Assignment, is the process of attributing a schedule for each individual member taking in account seniority, equality in hours flown, and personal preferences of crew members.

One can divide the crews in two types: the cockpit crew and the cabin crew. Cockpit crews are the ones that fly the aircraft and the cabin crews are there to assist the passengers. Each cockpit crew is qualified to fly a specific kind of aircraft type or fleet family. The problem is divided in various sub problems as a function of the fleet type and crew qualifications.

As referred before, commercial air transport heavily relies on optimization techniques to solve these allocation/scheduling problems, but this results in a very fragile schedule, almost without any slacks, that is very sensitive to any perturbation. Even a minor perturbation – because there are very few buffers – can lead to system wide disturbances of the carrier's schedule.

This problem has been addressed by the literature in two different ways. One is schedule recovery, where the airline, once a delay or disturbance appears, tries to recover the normal schedule in an efficient and effective way. The other is robust schedule design where the schedules are generated with a given level of robustness to face uncertainty. These two ways – schedule recovery and robust design – are intrinsically related. A highly robust schedule, designed for the worst-case scenario, has a low probability to be disrupted, hence a low schedule recovery cost. However, it is probably too conservative and costly. On the other hand, a highly optimized schedule with no slack or other kind of buffers will be prone to disruption and these disruptions translate into higher schedule recovery costs for the airlines.

Besides the combination of robustness and recovery, recovery should be attended in a broad perspective, in an integrated manner. It is not good practice to recover a disrupted schedule only by trying to alter the resources of a flight that is disrupted. For instance, if a flight is delayed and the crew that is onboard has a connecting flight, one could solve that disruption by using another crew that is on that airport. But the problem could also be solved by delaying the second flight enough time so that the crew can make it on time. With this simple example, it is possible to see that schedule recovery tends to be a practice where the whole airline and all its components must be considered: aircraft, crews, and passengers.

Regardless of the planning efforts made by airlines, the recovery problem will always be present in their operations. Even the most robust, worst-case design will fail at some point, and there will be a need of a highly efficient and effective schedule recovery method.

1.2 Goal, Objectives, and Tasks of work

The main goal of this work can be summarized as:

- **Improve the scheduling practices of airlines using proactive and reactive approaches.**

In concordance with this goal we devised three objectives. We also present the key tasks that where performed to achieve these objectives:

- <u>Develop a proactive approach to improve the airline scheduling process.</u>

    - Review the literature on airline planning, airline scheduling, uncertainty, proactive responses to uncertainty (robust scheduling);

    - Understand the uncertainty and how it affects the operations of an airline;

    - Develop a robust crew scheduling model that generate crew pairings that induce robustness in face of propagated delays through the network of an airline;

- <u>Develop an approach to be able to test and evaluate reactive and proactive models used to tackle the airline scheduling problem.</u>

    - Review the literature on simulation, airline operations and scheduling;

    - Develop a stochastic simulation platform to be able to benchmark scheduling solutions in face of uncertainty;

    - Understand the implications of different schedule recovery models in an airline's operations.

- <u>Develop a reactive approach to improve the airline scheduling recovery process.</u>

    - Review the literature on airline scheduling recovery, machine learning and metamodels.

    - Understand if a machine learning is the right representation of a schedule recovery model and can be used to mimic decision making made by an airline;

    - Create a machine-learning tool able to generate immediate solutions for the schedule recovery problem leveraging existing historic data for this purpose.

To be noted, that some of the tasks helped with more than just one objective. For instance, literature review done to answer the first objective also helped with the second and third objective.

1.3 Structure of Thesis

The thesis is organized into five chapters. Chapter 1, the current one, is the introduction where the theme of the thesis, background and scope are explained. It also explains the objectives of the work and lists them in detail.

Chapter 2 contains the proactive approach to uncertainty in airline scheduling. Since the crews have more complex rules and are important operational costs for airlines, second only by fuel costs, we created a robust crew scheduling model, specifically a robust crew pairing. It captures the complex legal framework of the industry and collective bargaining agreement (CBA). The resulting model is a robust crew pairing model that is formulated in linear form as a Mixed Integer Problem (MIP). The model is applied to an US carrier, Virgin America, in a case study where several levels of robustness are compared to each other using a stochastic simulation model to evaluate the solutions.

Chapter 3 contains the detailed characterization of a simulation platform. The platform relies on a stochastic simulation model using an agent-based modeling approach that represents in detail all daily operations of an airline. It allows to compare proactive and reactive schedules, robust schedules and schedule recovery solutions, and test them in face of uncertainty. We further elaborate on the idea that agent-based representation is a suitable way to represent an airline. We developed a case study where we use data from two airlines, Virgin America and Alaska Airlines, and apply two different schedule recovery approaches.

Chapter 4 contains a machine learning (ML) metamodeling approach to the airline schedule recovery problem. This approach leverages the historic data of schedule recovery problems and solutions to train a ML metamodel, specifically, an Artificial Neural Network. It is able to capture the patterns in the data and with this information generates immediate solutions. This chapter also explains the methodological approach to feed the historic data in a structured way. In a proof of concept case study, using data generated by the case study presented in Chapter 3, we verify that these patterns exist and can be captured using the ML metamodel.

Finally, in Chapter 5 we present the final conclusions of the whole thesis and research questions that this thesis opened, and further work that should be carried out to answer them.

1.4 Publications and presentations

The work presented in this thesis was published partially and/or presented on various events. Chapter 2 was submitted and has been accepted for publication:

- Antunes, D., Vaze, V., and Antunes, A. (2019). A Robust Pairing Model for Airline Crew Scheduling. Transportation Science. In press.

Events where part of this work was presented orally or as poster are listed below:

- IO2018, XIX Congress of the Portuguese Association of Operational Research, September 2018, Aveiro, Portugal. Title: Simulating Airline Daily Operations: Schedule Recovery and Airline Behavior.

- 21th ATRS World Conference, July 2017, Antwerp, Belgium. Title: Robust Airline Scheduling.

- CITTA 9th Annual Conference, November 2016, Coimbra, Portugal. Title: Airline Schedule Recovery: A Simulation Framework.

- 2016 MIT Portugal Conference, June 2016, Braga, Portugal. Title: Robust Airline Scheduling (poster). It received the “Best Poster Award”.

- 13th GET conference (Transport Study Group), January 2016, Figueira da Foz, Portugal. Title: Robust Airline Scheduling

[Page Intentionally Left Blank]

2 A ROBUST PAIRING MODEL FOR AIRLINE CREW SCHEDULING

2.1 Introduction

The airline schedule planning process consists of making a set of decisions regarding flight, aircraft and crew schedules. Traditionally, airline schedule planning is done sequentially, and the typical objective of each stage in the sequence is the minimization of the operating costs associated with the decisions being made in that stage if the operations are to be carried out exactly as planned. The actual costs can be considerably different from their planned values though, because airline operations often get disrupted due to various unforeseen events, including adverse weather, airport congestion, mechanical problems, security delays, etc. Therefore, recent literature has emphasized the need of accounting for these delays and disruptions while planning the schedules so that the true costs can be minimized.

This paper focuses on the crew scheduling stage in the airline schedule planning process and develops a new mathematical model and solution algorithm for generating crew schedules that strive to minimize the total cost, which includes the deterministic component of the planned crew costs as well as the costs of disruptions to planned crew schedules. For expositional convenience, from here on, we will refer to the deterministic component of the planned costs simply as “planned costs”.

The crew scheduling stage itself consists of two sequential sub-stages. In the first sub-stage, sequences of flights are generated to be operated by a single crew. This sub-stage, called crew pairing, creates flight sequences that are typically 1 to 4 days long, but does not assign these flight sequences to any specific crew members. The second sub-stage, called crew rostering, then combines several such sequences into monthly crew schedules and assigns these monthly schedules to individual crew members while accounting for other considerations such as training and vacation periods. This paper focuses on the first sub-stage, that is, crew pairing.

This chapter is organized as follows. In the remainder of this section we provide some background information on the crew pairing problem and on optimization approaches under uncertainty. The existing literature on crew pairing problems under uncertainty is reviewed in Section 2.2. In Section 2.3, we present an overview of our robust crew pairing model and, in Section 2.4, we provide the mathematical formulation of two variants of the model. Next, in Section 2.5, we describe the solution algorithm designed to solve the model and, in Section 6, we prove some of its key theoretical properties. The two subsequent sections are dedicated to presenting a case study involving a real-world airline network (Section 2.7) and to discussing the results that we have obtained through its application (Section 2.8). Finally, in Section 2.9, we summarize the main contributions of this paper and identify some important directions for future research.

2.1.1 Crew Pairing Problem

The crew pairing problem is a complex mathematical problem faced by airlines. A crew pairing is defined as a schedule for a crew member that covers a sequence of flights while adhering to several restrictions, such as starting and ending at a crew base, and satisfying various work rules imposed by the collective bargaining agreements (CBAs) and government regulations. These pairings must cover each flight scheduled by the airline exactly once.

A crew pairing problem is a special case of the set partitioning problem (SPP) where each flight must be covered by exactly one crew pairing and the objective is to minimize the total cost of all chosen pairings. Note that the production versions of the crew pairing problem that are implemented by an airline in practice may sometimes include some side constraints related to crew grouping, workload balance, etc. However, to keep our discussion general, like most of the previous literature on the crew pairing problem under uncertainty, we focus our attention on the set partitioning formulation of this problem. The mathematical model representing the SPP is as follows:

$$\min \sum_{p \in \mathcal{P}} c_p \, x_p \tag{2.1}$$

Subject to:

$$\sum_{p \in \mathcal{P}} k_{fp} \cdot x_p = 1 \qquad \forall f \in \mathcal{F} \tag{2.2}$$

$$x_p \in \{0,1\} \qquad \forall p \in \mathcal{P} \tag{2.3}$$

where:

$\mathcal{F}$: Set of all flights;

$\mathcal{P}$: Set of all feasible pairings;

$c_p \in \mathbb{R}^+$: Cost of pairing p;

$k_{fp} \in \{0,1\}$: Binary parameter that indicates if flight f is included in pairing p;

$x_p \in \{0,1\}$: Binary decision variable that indicates if pairing p is chosen.

Solving even the linear programming relaxation of this model in a naïve way requires explicitly enumerating all possible pairings, and even for a small airline's network this might result in several billions of possible pairings. So instead of enumerating all possible pairings, the model is solved using a method known as column generation. It is an iterative method where a restricted version of the set partitioning problem, known as a restricted master problem (RMP), is solved first using only a small subset of all feasible pairings. Then a second problem, called the pricing problem (PP), is solved to generate additional pairings to be added to the RMP such that these new pairings decrease the optimal value of the RMP objective function.

The PP itself is challenging to solve due to the commonly used non-linear crew pay scheme (known as pay-and-credit scheme) and due to the several complex legal and administrative restrictions on crew pairings. A crew pairing consists of a sequence of crew duties with consecutive duties separated by crew rest times, where a crew duty is defined as a sequence of flights scheduled to be operated by a crew during a work-day. Typically, at least for the North American airlines, the planned crew pay for a pairing equals the maximum of two terms: 1) sum of costs of the duties that constitute the pairing, and 2) a fraction (α) of the total elapsed time of the pairing. Total elapsed time of a pairing is sometimes called the time away from base (TAFB). Thus, the planned crew pay for a pairing is given by:

$$c_p = max\left(\sum_{d \in D_p} c_d \, , \alpha \cdot T_p\right) \tag{2.4}$$

where D_p is the set of duties in pairing p, c_d is the cost of duty d, and T_p is the TAFB of pairing p. The duty cost (c_d) itself is given as the maximum of three terms, as follows:

$$c_d = \max(b_d, \beta \cdot e_d, \theta) \tag{2.5}$$

where b_d is the sum of block times of all flights in duty d, β is a constant parameter, e_d is the elapsed time of duty d and θ is the minimum guaranteed crew pay per duty. The block time is the length of the time interval between an aircraft's departure time from the gate at its origin airport and the aircraft's arrival time at the gate at its destination airport. This interval includes the flying time, takeoff time, landing time, taxi-out time and taxi-in time. The parameters α, β and θ are typically negotiated between the crew union and the airline management through a CBA.

Government regulations and CBAs also impose several constraints on what constitutes a permissible pairing. Typical constraints include an upper limit on the elapsed time of a pairing, an upper limit on the elapsed time in each duty, an upper limit on the flying time in each duty, a lower limit on the time between flights within the same duty (called sit time), and a lower limit on the time between two consecutive duties (called rest time).

2.1.2 Optimization Approaches under Uncertainty

Airlines rely heavily on optimization models to solve various resource allocation and scheduling problems like the crew pairing problem. However, these models traditionally have focused on minimizing planned costs alone. This may result in highly fragile schedules, which have very little or no buffers, and thus are very vulnerable to disruptions. Consequently, even a minor perturbation can lead to system-wide disturbances in the airline's operations. There are several approaches for capturing the effects of such uncertainty in optimization models. Three major approaches are chance-constrained optimization (sometimes called as stochastic optimization), stochastic programming and robust optimization.

Chance-constrained optimization models ensure that the probability of satisfying a certain constraint is greater than or equal to a certain level. They restrict the feasible region of the model to ensure that the probability of a solution remaining feasible under uncertainty is high. These models assume the input data to be random and require a priori knowledge of the probability distribution functions associated with the data.

One drawback of the chance-constrained optimization approach is that it is often difficult (or even impossible) to obtain the underlying distributions of the uncertain input parameters. Sim (2004) states that, even in the cases where we know the distributions, it is still computationally challenging to evaluate the chance constraints, let alone solving the optimization model. Moreover, the chance constraints can destroy model's convexity properties and increase the complexity of the model, thus further increasing the computational effort required to solve it. Hence applying this approach to solve the crew pairing problem, whose deterministic version itself has a heavy computational burden, seems prohibitively challenging. Indeed, we are not aware of any existing study that attempts to apply the chance-constrained optimization approach to solve the crew pairing problem under uncertainty.

Stochastic programming models assume some of the input parameters (coefficients in the objective function and/or in the constraints) to be uncertain. Uncertainty is usually characterized by a probability distribution function of the input parameters. Typically, the problem is formulated as a two-stage optimization model where some or all parameter values get realized after the first-stage decisions are made, and therefore the model is formulated as one of minimizing the expected costs of the consequences of the first-stage decisions. Note that the second-stage decisions under each realization of the parameter values also affect the consequences of first-stage decisions, and thus the accuracy of this approach is affected by the accuracy in the characterization and solution of the second-stage model. Another challenge is the complete enumeration of all possible parameter realization scenarios and the corresponding solution of the second-stage model in each scenario. Therefore, an approximate solution is often sought by evaluating only a small number of scenarios. Indeed, the computational burden of applying a stochastic programming approach to the crew pairing problem is enormous. Yen and Birge (2006) is an example of a study from past literature that attempts to do so. However, it focuses on a flight network of small size, assumes a simplified crew pay structure, considers a simplified version of the second-stage problem (called the recovery problem), and uses a sample of only 100 scenarios of parameter realizations.

Robust optimization models involve arguably less severe computational challenges when applied to the crew pairing problem, but there is no prior literature attempting to do so, except for a recent study by Lu and Gzara (2015) which is discussed in Section 2.2. The application of a robust optimization approach to the crew pairing problem is far from being straightforward, as discussed in Section 3, which could be one of the reasons for the lack of literature in this area. This approach has been shown to perform well in many other application areas. Ye et al. (2014) compared a stochastic programming approach and a robust optimization approach for a production scheduling problem and concluded that their performance is similar. Also, Chen et al. (2007) compared the same two approaches, and noted that, compared to stochastic programming, robust optimization relaxes the requirement from the users to know the probability distribution functions, and instead only needs modest assumptions about them. They also noted that the size of the robust linear models is polynomial, rather than exponential, in problem dimensions, making it possible to use (state-of-the-art) commercial solvers to handle them. A comparison made by Hasani et al. (2012), applied to closed-loop supply networks, showed that lack of knowledge about the best fit parameter distributions, as well as the associated modeling and computational complexities, limit the application of stochastic programming, and therefore a robust optimization approach is more preferable. In summary, robust optimization models neither need detailed information about the probability distribution functions, nor suffer from tractability issues as severe as those faced by stochastic programming models.

Although robust versions of a tractable deterministic optimization model might not always be tractable, tractable robust versions of many known optimization problems have been formulated. In this paper, we develop a tractable version of the crew pairing problem under uncertainty based on the concept of robust optimization presented by Bertsimas and Sim (2004). We solve the model to optimality, with an algorithm that iteratively uses commercial solvers, within a reasonable amount of time for moderate-sized real-world airline networks. We implement and solve our model to compute and compare several solutions with varying robustness parameter levels for the network of a moderate-sized low-cost airline in the United States. The model's solutions are thoroughly tested in a simulation environment using real-world delay data. The simulation results show that our robust crew pairing solutions lead to lower delays and fewer instances of operational infeasibilities, thus requiring fewer recovery actions to address them. We find that with the inclusion of robustness, it is possible to generate crew pairing solutions that significantly reduce the delay and disruption costs with only a small increase in planned costs.

2.2 Literature Review

A number of prior studies have dealt with the crew pairing problem under uncertainty. In addition to the planned crew costs, they have focused, directly or indirectly, on capturing the additional operating costs due to disruptions. All prior studies can be broadly categorized into two classes based on their objective: 1) studies that attempt to minimize delay propagation through crew connections by judicious allocation of buffers in crew connection times, and 2) studies that attempt to create crew schedules that are easier to recover once a disruption happens. Note that typical crew schedule recovery actions in response to a disruption include delaying flight departures, crew swaps, use of reserve crews, flight cancellations and crew deadheads.

Yen and Birge (2006) present a model that aims to reduce the propagated delay by minimizing the expected value of the total costs through a stochastic programming approach. In addition to the planned costs in the form of crew salaries, they include an expected delay cost in the objective function. The only possible recovery actions in the model consist of delaying flight departures, and it does not capture the effects of violations of crew duty rules such as upper limits on duty flying times and duty elapsed times. Moreover, the crew pay structures are much simpler compared with the complex structure of crew pay in North American airlines as described in Section 2.2.1.1.

A bi-criteria optimization approach is proposed by Ehrgott and Ryan (2002) to include robustness into the crew pairing model. Their idea of a robust solution is based on decreasing the number of aircraft changes by the crew when there is insufficient ground time between two successive flights. While this is an important aspect of robustness, there are many other aspects that may lead to significant delays and infeasibilities in the operation of a crew pairing. The authors acknowledge that the model has several computational difficulties and a large amount of customization is necessary. Several parameters need to be set, making expert knowledge of the model necessary to be able to implement it.

Tam et al. (2011) compare stochastic programming and bi-criteria optimization approaches. They conclude that bi-criteria optimization leads to better solutions for cases with moderate delay costs and that stochastic programming might be preferred for cases with more substantial delay costs.

Recognizing that crew swaps are an important type of crew recovery action, Shebalov and Klabjan (2006) attempt to maximize the crew swapping opportunities at the expense of a small increase in the planned crew costs. They solve a somewhat simplified daily (rather than multi-day) crew pairing problem. They acknowledge that the underlying mixed-integer linear programming formulation is very challenging to solve using commercial solvers, and instead solve it using a customized algorithm combining delayed column generation and Lagrangian relaxation techniques. They conclude that, to seek a good tradeoff between planned crew costs and crew recovery potential, the planned crew costs should be within 1% of the minimum planned crew costs value. They note that this 1%

bound could include several alternative crew schedules, and that the exact choice of such bound, as well as the actual crew schedule, needs to be further examined.

Schafer et al. (2005) propose an approach that modifies the cost of each crew pairing by including, in addition to the planned costs, a linear penalty function that penalizes short sit times between flights within a duty, short rest times between duties, long flying times within a duty, and long elapsed times for a duty. Using an iterative approach and a discrete events simulation model to evaluate the resulting crew schedules, the authors fine-tune the coefficients of the penalty function to best approximate the expected operational costs. This study ignores crew recovery actions and also assumes that there is no aircraft-based propagation of delays. Moreover, the success of this approach is contingent on the accuracy of the delay distributional assumptions underlying the discrete events simulation model.

A model for integrated fleet assignment and crew scheduling is presented in Gao et al. (2009) where the cost of crew scheduling is not considered in an explicit manner. Instead, the model minimizes certain indicators of the cost of crew scheduling under uncertainty. Specifically, it attempts to restrict the number of different fleet families and the number of different crew bases serving each airport in the network, to create more aircraft and crew swapping opportunities. This study is similar, in spirit, to Shebalov and Klabjan (2006) in that both focus specifically on improving crew swapping opportunities, but not on other issues such as crew delay propagation and potential infeasibilities in crew schedules.

AhmadBeygi et al. (2010) formulate a model which proposes minor changes to flight schedules while maintaining all the crew connections obtained by minimizing the planned cost. This is achieved by adjusting the placement of existing slack to improve operational performance by reducing propagated delays. The model re-times some flights such that more slack is provided in places where it is needed the most to absorb propagated delays, while reducing the slack in places where it is less useful. Rather than adjusting crew pairings to make them more resilient to disruptions, this approach adjusts the flight timings while holding crew pairings unchanged.

The integrated aircraft routing and crew pairing model presented in Dunbar et al. (2012) relies on a simplified model of crew costs based only on flying hours. It is a significant simplification in comparison to the pay-and-credit scheme (described in Section 2.1.1) that many airlines use, especially in North America. Their computational results are obtained for a daily network consisting of 54 flights.

Weide et al. (2010) also develop an integrated aircraft routing and crew pairing model which tries to ensure that the aircraft and crew stay together as much as possible. Since the two individual models by themselves are already hard to solve, combining them would increase the computational burden significantly. To avoid this issue, the authors solve the two individual models in an iterative manner. It is a highly customized modeling approach where several parameters, weights, and a non-robustness measure must be set a priori, and it also does not guarantee optimality.

Finally, Lu and Gzara (2015) propose a robust optimization model that attempts to add robustness to a highly simplified version of the crew pairing problem, wherein crew costs are approximated by the TAFB (time away from base) component alone. This simplified cost of a pairing is the main metric of planned performance for the whole model, and robustness is induced by considering uncertainty in this metric. Despite this highly simplified cost structure, the approach still leads to a non-linear optimization model that is solved through Lagrangian relaxation. The computational results are based on an airline network with 46 daily flights spanning across a four-day time horizon.

In summary, due to the complexity of the problem, most existing approaches to crew pairing under uncertainty make strong simplifying assumptions about the crew pay rules and delay costs, and most end up developing highly customized algorithms. Furthermore, the literature on applying the robust optimization approach to the crew pairing problem is rather scarce despite the success of this approach as evidenced in other applications. Our paper addresses this gap by developing a robust crew pairing model and an associated solution algorithm that can be implemented by directly using commercial solvers in an iterative manner.

2.3 Model Overview

Before presenting the full mathematical formulation of the model proposed in Section 4, we provide here the motivation behind our modeling approach by highlighting the complexity associated with incorporating robustness into the crew pairing problem in a comprehensive manner. As mentioned in Section 2, all previous models for the crew pairing problem under uncertainty either assume a highly simplified crew pay structure or rely on a very limited interpretation of the idea of crew schedule robustness, or both. Our model is much more comprehensive. Specifically, we account for the full complexity of the pay-and-credit crew salary structure, and we also account for various dimensions of crew schedule robustness by considering multiple complex implications of crew schedule disruptions on an airline’s overall network. Next, we describe these various implications of crew schedule disruptions.

When a flight is delayed for any reason, such as weather, mechanical failure, security, etc., its delayed arrival may leave insufficient time until the departure of the next flight scheduled to be operated by the same crew, thus delaying the departure of the next flight as well. We call this phenomenon as delay propagation through crew. The delay to the next flight is categorized as propagated delay, to be contrasted with the delay to the original flight, which is called the root delay. Propagated delays have two major types of operational impacts on an airline’s network. First, propagated delays generate additional flight delay costs to the airlines through additional aircraft operating costs, passenger misconnection costs, loss of passenger goodwill, etc. Second, propagated delays can increase the elapsed time of a crew duty and/or the elapsed time of a crew pairing, which in turn has two effects – it might result in additional crew costs, and the extra duty elapsed time can also make some pairings infeasible to operate.

In this paper, we develop two variants of a robust crew pairing model, called Robust Model 1 (RM1) and Robust Model 2 (RM2). RM1 captures the full pay-and-credit crew salary structure as well as the flight delay costs due to propagated delays but does not capture the effects of the potential increases in duty and pairing elapsed times. RM2 captures all three, i.e., the full pay-and-credit crew salary structure, the flight delay costs due to propagated delays, and the effects of the potential increases in duty and pairing elapsed times. While flight delays can also increase crew flying time leading to additional crew costs, this is usually a small component of the overall delay costs because most delays are taken on ground waiting at the gate. Modeling the exact fraction of flight delays taken on ground versus in air is beyond the scope of this paper. Instead, we use two alternative approaches to handle this cost as reflected by our two robust model variants: RM1 includes crew delay costs within the flight delay cost calculation, while RM2 calculates crew delay costs separately as functions of only the increases in duty elapsed times and pairing elapsed times.

Our robust crew pairing model is based on the concept of robustness first developed by Bertsimas and Sim (2004) wherein the desired level of robustness or conservatism for the i^{th} constraint in the model is controlled by a robustness budget (Γ_i). Note that this approach also allows adding robustness to the objective function coefficients by simply defining a new decision variable, setting it equal to the original objective function (and thus moving the original objective function to the set of constraints) and then setting this new decision variable as the new objective function. The central idea is to ensure that the solutions produced by the model are protected against (i.e., are feasible under) all uncertainty scenarios where at most $\lfloor \Gamma_i \rfloor$ coefficients in the i^{th} constraint take their worst-case values, one other coefficient takes a value which is at most $(\Gamma_i - \lfloor \Gamma_i \rfloor)$ times as bad as the worst-case value, and all other coefficients remain at their nominal values. The rationale behind this is based on the observation that it is very unlikely that all coefficients of a constraint will take their worst-case values at the same time, but a subset of them sometimes will do so. Bertsimas and Sim (2004) also show that the robust version of a linear optimization model can itself be reformulated as a linear optimization model, and the robust version of an integer linear optimization model can itself be reformulated as an integer linear optimization model. Moreover, they also provide bounds on the constraint violation probabilities.

There is an obvious challenge in applying this concept directly to the set partitioning formulation of the crew pairing model given by (1)-(3): the uncertainty comes from flight delays which affect, in a non-linear way, both the feasibility and the true cost of covering a flight by a pairing. As explained in Section 2.1.1, real-world crew pairing problems are characterized by a prohibitively large number of feasible pairings, making explicit enumeration impossible. Instead, column generation techniques are typically used to generate and add additional columns to the RMP (restricted master problem) based on their ability to reduce the RMP objective function value. Column generation involves solving a problem called as pricing problem (PP). Therefore, to add robustness to the crew pairing model, we need to especially focus our attention on the pricing problem under uncertainty.

Past studies have developed and used different methods to solve the PP. A multi-label shortest path algorithm is often used (Desaulniers et al., 2005; Vance et al., 1997), which relies heavily on the model structure and requires significant implementation effort due to its highly customized nature. AhmadBeygi et al. (2009) developed an integer programming model as an alternative approach for solving the PP for moderate-sized crew pairing problem instances using a commercial solver. Their approach is especially suitable for our purposes. It allows us to modify their original PP by adding a variety of new features to account for robustness considerations.

2.4 Mathematical Formulation

We now describe the mathematical formulation for our two variants of the robust crew pairing model. Section 4.1 presents the robust version of the restricted master problem (RMP), while Section 4.2 presents the robust version of the pricing problem (PP). We denote the original PP formulation presented by AhmadBeygi et al. (2009) as the *base formulation*. To keep our discussion focused, we present in Section 4.2 only the new features that transform the base formulation into its robust version and relegate the discussion of the remaining parts of the base formulation to the appendix.

2.4.1 Robust Crew Pairing Model

We will use the following notation in the RMP formulation.

2.4.1.1 Sets

$\mathcal{F}$: Set of all flights;

$\mathcal{P}$: Set of all feasible pairings in the absence of uncertainty;

D_p: Set of all duties in pairing p;

L_p^s: Set of all ordered flight pairs such that the second flight in the pair immediately follows the first flight within the same duty in pairing p;

L_p^r: Set of all ordered flight pairs such that the second flight in the pair immediately follows the first flight, but the two are in consecutive duties in the same pairing p.

2.4.1.2 Parameters

$\Gamma \in [0, |\mathcal{F}|]$: Robustness budget;

$k_{fp} \in \{0,1\}$: Binary parameter that indicates if flight f is included in pairing p;

$\gamma \in \mathbb{R}^+$: Parameter that represents the ratio of the cost of one hour of flight delay to the crew pay for one hour of flying time;

$\upsilon_p \in \mathbb{N}^+$: Number of flights in pairing p;

$\alpha, \beta, \theta \in \mathbb{R}^+$: Constant parameters in the pay-and-credit crew salary scheme as described in (4) and (5);

$b_{dp} \in \mathbb{R}^+$: Sum of planned block times (ignoring delays) of all flights in duty d in pairing p;

$\Delta_f \in \mathbb{R}^+$: Root delay of flight f;

$e_{dp} \in \mathbb{R}^+$: Planned elapsed time (ignoring delays) of duty d in pairing p;

T_p: Time away from base (TAFB) of pairing p;

$s_{f_1 f_2} \in \mathbb{R}^+$: Planned sit time (ignoring delays) between flights f_1 and f_2;

$r_{f_1 f_2} \in \mathbb{R}^+$: Planned rest time (ignoring delays) between flights f_1 and f_2;

$\overline{f_d}$: Last flight in duty d;

$\overline{\overline{f_p}}$: Last flight in pairing p;

$\delta \in \{0,1\}$: Binary parameter that equals 0 for RM1 and 1 for RM2;

$\tau \in \mathbb{R}^+$: Minimum sit time;

$\rho \in \mathbb{R}^+$: Minimum rest time;

$\lambda \in \mathbb{R}^+$: Maximum duty elapsed time.

2.4.2 Decision Variables

$x_{f_1 f_2 d} \in \{0,1\}$: Binary decision variables from the base formulation indicating if flight f_1 is immediately followed by flight f_2 within the d^{th} duty;

$y_{f_1 f_2 d} \in \{0,1\}$: Binary decision variables from the base formulation indicating if flight f_1 ends the d^{th} duty and is followed by flight f_2 starting the $(d+1)^{th}$ duty;

$w_{fd} \in \{0,1\}$: Binary decision variables from the base formulation indicating if flight f in duty d is the last flight in the pairing;

$\chi_p \in \{0,1\}$: Binary decision variable that indicates if pairing p is chosen;

$\zeta \in \mathbb{R}^+$: Auxiliary delay variable;

$q_f \in \mathbb{R}^+$: Another auxiliary delay variable, specific to each flight $f \in F$;

$\zeta_p \in \mathbb{R}^+$: Another auxiliary delay variable, specific to each pairing $p \in \mathcal{P}$;

$q_{fp} \in \mathbb{R}^+$: Another auxiliary delay variable, specific to each combination of flight $f \in F$ and pairing $p \in \mathcal{P}$;

$\eta_p \in \mathbb{R}^+$: Robust version of the cost of pairing p (equals 0 if the pairing is not selected);

$\psi_{dp} \in \mathbb{R}^+$: Robust version of the cost of duty d in pairing p (equals 0 if the pairing is not selected);

$p_{f_1 f_2 p} \in \mathbb{R}^+$: Propagated delay from flight $f_1$to flight f_2 within pairing p (equals 0 if the pairing is not selected).

Using this notation, the RMP formulation for the robust crew pairing problem is given by (2.6)-(2.20). There are three main differences between this formulation and the deterministic crew pairing formulation (2.1)-(2.3). First, the cost of pairing p has been changed from $c_p \cdot \chi_p$ to its robust version η_p. Second, there is an extra additive term in the objective function, $\gamma \cdot \left(\Gamma \cdot \zeta + \sum_{f\in\mathcal{F}} q_f\right)$, which represents the flight delay costs due to propagated delays. Third, the set of feasible pairings is now further restricted by constraints (2.19). Constraints (2.9)-(2.18) are necessary to ensure that the modified objective function (2.6) and constraints (2.19) represent the robust version of the crew pairing model.

$$\min \sum_{p\in\mathcal{P}} \eta_p + \gamma \cdot \left(\Gamma \cdot \zeta + \sum_{f\in\mathcal{F}} q_f \right) \tag{2.6}$$

$$\sum_{p\in\mathcal{P}} k_{fp} \cdot \chi_p = 1 \qquad \forall f \in \mathcal{F} \tag{2.7}$$

$$\chi_p \in \{0,1\} \qquad \forall p \in \mathcal{P} \tag{2.8}$$

$$\zeta = \sum_{p \in \mathcal{P}} \zeta_p \qquad (2.9)$$

$$q_f = \textstyle\sum_{p \in \mathcal{P}} q_{f,p} \qquad \forall f \in \mathcal{F} \quad (2.10)$$

$$\zeta_p + q_{fp} \geq \sum_{f' \in \mathcal{F}} p_{f'fp} + \Delta_f \cdot k_{fp} \cdot \chi_p \qquad \forall f \in \mathcal{F}, \forall p \in \mathcal{P} \quad (2.11)$$

$$p_{f_1 f_2 p} \geq \sum_{f' \in \mathcal{F}} p_{f' f_1 p} + \Delta_{f_1} \cdot \chi_p - \left(s_{f_1 f_2} - \tau\right) \quad \forall (f_1, f_2) \in L_p^s \,, \forall p \in \mathcal{P} \quad (2.12)$$

$$p_{f_1 f_2 p} \geq \sum_{f' \in \mathcal{F}} p_{f' f_1 p} + \Delta_{f_1} \cdot \chi_p - \left(r_{f_1 f_2} - \rho\right) \quad \forall (f_1, f_2) \in L_p^r \,, \forall p \in \mathcal{P} \quad (2.13)$$

$$\eta_p \geq \textstyle\sum_{d \in D_p} \psi_{dp} \qquad \forall p \in \mathcal{P} \quad (2.14)$$

$$\eta_p \geq \alpha \cdot \left(T_p \cdot \chi_p + \left(\zeta_p + q_{\overline{\overline{f_p}}p}\right) \cdot \delta\right) \qquad \forall p \in \mathcal{P} \quad (2.15)$$

$$\psi_{dp} \geq b_{dp} \cdot \chi_p \qquad \forall d \in \mathcal{D} \,, \forall p \in \mathcal{P} \quad (2.16)$$

$$\psi_{dp} \geq \beta \cdot \left(e_{dp} \cdot \chi_p + \left(\zeta_p + q_{\overline{f_d}p}\right) \cdot \delta\right) \forall d \in \mathcal{D} \,, \forall p \in \mathcal{P} \quad (2.17)$$

$$\psi_{dp} \geq \theta \cdot \chi_p \qquad \forall d \in \mathcal{D} \,, \forall p \in \mathcal{P} \quad (2.18)$$

$$\lambda \geq e_{dp} \cdot \chi_p + \left(\zeta_p + q_{\overline{f_d}p}\right) \cdot \delta \qquad \forall d \in \mathcal{D} \quad (2.19)$$

$$\begin{gathered} \zeta \in \mathbb{R}^+;\ q_f \in \mathbb{R}^+\ \forall f \in F;\ \zeta_p, \eta_p \in \mathbb{R}^+\ \forall p \in \mathcal{P};\ q_{fp}, \psi_{dp} \in \mathbb{R}^+\ \forall f \in F, p \\ \in \mathcal{P};\ p_{f_1 f_2 p} \in \mathbb{R}^+\ \forall f_1, f_2 \in F, p \in \mathcal{P} \end{gathered} \qquad (2.20)$$

The first term in the objective function (2.6) represents the total cost of crew pairings while accounting for the pay-and-credit crew salary structure and the additional crew costs (from increased duty and pairing elapsed times) due to delays. The second term in the objective function, $\gamma \cdot \left(\Gamma \cdot \zeta + \sum_{f \in \mathcal{F}} q_f\right)$, representing the rest of the flight delay costs, is similar in spirit to the expression $(\Gamma \cdot p + \sum_i q_i)$ in the objective function of the robust portfolio management model (13) presented in Bertsimas and Thiele (2006). Note that, since the unit of crew costs is an hour of crew flying time, we use a conversion factor γ to relate the unit delay costs for a given aircraft family with the crew pay per flying hour.

Constraints (2.7) and (2.8) are constraints (2.2) and (2.3) from the original set partitioning problem, respectively, the only difference being that variables x_p are now replaced by χ_p. Constraints (2.9) and (2.10) relate the auxiliary variables ζ and q_f to their pairing-specific counterparts, ζ_p and q_{fp}, respectively. Constraints (2.11) require that if a pairing p covering flight f is selected, then our formulation will protect flight f against a total arrival delay (root delay plus propagated delay) equal to $\zeta_p + q_{fp}$. Constraints (2.12) and (2.13) make sure that the delay propagated between every pair of consecutive flights within every pairing is calculated correctly. Constraints (2.14)-(2.18) compute the cost of pairing p similarly to equations (2.4) and (2.5). However, in addition to crew salaries calculated under the pay-and-credit scheme, they now also include the extra crew pay due to delay-induced increases in duty and pairing elapsed times. Constraints (2.20) are the variable value constraints for all the new decision variables.

Note that the formulation (2.6)-(2.20) is an integer linear optimization problem which can be solved for various values of the robustness parameter $\Gamma \in [0, |\mathcal{F}|]$ and for two different values of $\delta \in \{0,1\}$. Setting $\Gamma = 0$ is equivalent to solving the deterministic version of the problem which does not protect against any delays, while setting $\Gamma = |\mathcal{F}|$ will protect against the possibility of all flights in all pairings simultaneously having the worst-case delays. For either of these two extreme cases, or for any of the infinite number of cases in between, setting $\delta = 0$ results in RM1 which makes the objective function robust against propagated delay costs, but does not protect the pairings against delay-induced increases in duty and pairing elapsed times. Setting $\delta = 1$ results in RM2 which additionally protects the pairings against such delay-induced increases in duty and pairing elapsed times.

Except for the main auxiliary variables (ζ and q_f), all other decision variables (namely, χ_p, ζ_p, q_{fp}, η_p, ψ_{dp}, and $p_{f_1 f_2 p}$) in this problem are defined at the level of each pairing and hence are composite variables. Due to this structure, it is advisable to attempt to convert this into a standard set partitioning problem to enable its solution using column generation techniques. We successfully do so by defining a new pairing cost coefficient, ${c'}_p$, as follows:

$$ {c'}_p = {c''}_p + \gamma \cdot \left(\frac{\upsilon}{|\mathcal{F}|} \cdot \Gamma \cdot \zeta_p + \sum_{f \in \mathcal{F}} q_{fp} \right) \qquad \forall p \in \mathcal{P} \quad (2.21) $$

where,

$$ {c''}_p = \max\left(\sum_{d \in D_p} c_{dp} \, , \alpha \cdot \left(T_p + \left(\zeta_p + q_{\overline{\overline{f}_p}p} \right) \cdot \delta \right) \right) \qquad p \in \mathcal{P} \quad (2.22) $$

$$ c_{dp} = \max\left(b_{dp}, \beta \cdot \left(e_{dp} + \left(\zeta_p + q_{\overline{f}_d p} \right) \cdot \delta \right), \theta \right) \qquad \forall d \in \mathcal{D} \, , \forall p \in \mathcal{P} \quad (2.23) $$

Then the reformulated RMP is given as follows:

$$ \min \sum_{p \in \mathcal{P}} {c'}_p \cdot \chi_p \quad (2.24) $$

$$ \sum_{p \in \Psi} k_{fp} \cdot \chi_p = 1 \qquad \forall f \in \mathcal{F} \quad (2.25) $$

$$ \chi_p \in \{0,1\} \qquad \forall p \in \Psi \quad (2.26) $$

Here, $\Psi \subseteq \mathcal{P}$ is defined as the subset of feasible pairings that satisfy constraints (2.19). Note that, a set partitioning problem requires the objective function to be completely represented by a per-pairing cost. In order to distribute the second term in (2.6) across the pairings, the expression $\Gamma \cdot \zeta$ needs to be somehow distributed across the pairings to avoid double-counting. Note that we could choose any distribution scheme as long as the sum across all pairings in any feasible pairing solution equals exactly $\Gamma \cdot \zeta$. For simplicity, we distribute it proportionally to the number of flights covered by each selected crew pairing. Therefore, a pairing containing υ (out of the total of $|\mathcal{F}|$) flights will correspond to $\frac{\upsilon}{|\mathcal{F}|} \cdot \Gamma \cdot \zeta$ of the total cost corresponding to the term $\Gamma \cdot \zeta$.

2.4.3 Pricing Problem

When presenting the pricing problem, we will drop the pairing subscript p to avoid redundancy. The PP objective function is as follows:

$$min\ c'' + \gamma \cdot \left(\frac{v}{|\mathcal{F}|} \cdot \Gamma \cdot \zeta + \sum_{f \in \mathcal{F}} q_f \right) - \sum_{f \in \mathcal{F}} \sum_{d \in \mathcal{D}} \pi_f \cdot z_{fd} \tag{2.27}$$

π_f are the dual variable values for constraints (2.25) in the RMP. z_{fd} are binary decision variables indicating if flight f will be covered in duty d of the selected pairing. Note that v is not known a priori when finding the minimum reduced cost pairing. If we let v be a decision variable, then the PP will lose its linearity and hence will become even harder to solve. Instead, we solve a separate pricing problem for each possible v value iteratively. The solution algorithm is described in Section 2.5. The constraints presented in (2.28)-(2.32) are newly added to the base formulation of the PP to capture the delays.

$$\zeta + q_f \geq \sum_{f' \in \mathcal{F}} p_{f'f} + \Delta_f \cdot \sum_{d \in \mathcal{D}} z_{fd} \qquad \forall f\ \in\ \mathcal{F} \tag{2.28}$$

$$p_{f_1 f_2} \geq \sum_{f' \in \mathcal{F}} p_{f' f_1} + \Delta_{f_1} - \left(s_{f_1 f_2} - \tau\right) - M\left(1 - \sum_{d \in \mathcal{D}} x_{f_1 f_2 d}\right) \ \forall f_1, f_2 \\ \in\ \mathcal{F}, f_1 \neq f_2 \tag{2.29}$$

$$p_{f_1 f_2} \geq \sum_{f' \in \mathcal{F}} p_{f' f_1} + \Delta_{f_1} - \left(r_{f_1 f_2} - \rho\right) - M\left(1 - \sum_{d \in \mathcal{D}\prime} y_{f_1 f_2 d}\right) \ \forall f_1, f_2 \\ \in\ \mathcal{F}, f_1 \neq f_2 \tag{2.30}$$

$$q_{\bar{\bar{f}}}\ \geq q_f - M \cdot \left(1 -\ w_{fd}\right) \qquad \forall d\ \in \mathcal{D}, \forall f \in \mathcal{F} \tag{2.31}$$

$$q_{\overline{f_d}}\ \geq q_{f_1} - M \cdot \left(1 -\ y_{f_1 f_2 d} -\ w_{f_1 d}\right) \forall d\ \in \mathcal{D}, f_1, f_2 \in \mathcal{F}, f_1 \neq f_2 \tag{2.32}$$

$\mathcal{D}' = \{1, \dots, D-1\}$ is the set of all duty indices except the last duty. Parameters $w_{fd} \in \{0,1\}$ equal 1 if f is the last flight in the pairing and the pairing is d duties long. Constraints (2.28) are similar in spirit to the second set of constraints in the formulation of the portfolio management model presented in Bertsimas and Thiele (2006). Right-hand side of each constraint (2.28) is the total arrival delay of a flight f and the left-hand side consists purely of the two auxiliary delay variables, namely, ζ and q_f. Together ζ and q_f must serve as an upper bound on flight arrival delay, but the balance between ζ and q_f relates to the solution's robustness level. Constraints (2.28) and the way we compute the cost of a pairing play a very important role in setting the appropriate level of robustness consistent with the robustness parameter Γ. For a small value of Γ (which can be thought of as the penalty on ζ), ζ takes a relatively large value to ensure that constraints (2.28) are satisfied even with low (or even zero) values of q_f. For smaller values of Γ, and therefore for larger values of ζ, more propagated delays are allowed due to the smaller penalty. So, setting Γ to low values allows solving the crew pairing model with more emphasis on the planned crew pairing costs and less on the delay costs. On the other hand, as Γ progressively increases from 0 to $|F|$, it makes lower values of $\sum_{f' \in \mathcal{F}} p_{f'f}$ increasingly more attractive and thus makes the solution increasingly more robust.

Constraints (2.29) and (2.30) allow the model to calculate the delays propagated within a duty and between duties, respectively. The big M ensures that if the sum of $x_{f_1 f_2 d}$ or $y_{f_1 f_2 d}$ is 0, then the respective constraints are rendered redundant. Otherwise, these constraints (along with non-negativity constraints on all $p_{f_1 f_2}$ variables) ensure that propagated delay is the non-negative part of the difference between the total delay to the previous flight (which itself is the sum of previous flight's propagated and root delays) and the slack (which is the difference between the planned sit/rest time given by $s_{f_1 f_2}$ and $r_{f_1 f_2}$, and the minimum sit/rest time given by parameters τ and ρ, respectively). The model captures propagated delays affecting a flight in such a way that, if the flight has enough slack in the sit or rest time, then it will absorb all the propagated delay; else it will pass that delay (or part of it) to the next flight operated by the same crew. Finally, note that constraints (2.28) ensure that if a flight f is not covered in the chosen (minimum reduced cost) pairing, then the corresponding q_f equals 0. Constraints (2.31) and (2.32) correctly set the q_f values for the last flight in the pairing and the last flight in each duty, respectively.

Here lies another important difference between our work and the study by Lu and Gzara (2015). We compute in detail the delays, giving explicit attention to how they propagate through the crew connections across the airline network and to how they influence the final cost. Lu and Gzara (2015) use a simplification where they assume the cost of a pairing to be proportional to only the TAFB, and then they add uncertainty to that simplified cost. In other words, while we are adding uncertainty to the root delay, which is the commonly used measure of representing uncertainty in airline operations, Lu and Gzara (2015) directly consider uncertainty in the cost of a pairing. While their model entails a higher level of abstraction, ours treats planned crew costs as well as delay costs in much greater detail.

In addition to adding constraints (2.28)-(2.32) to the PP base formulation, a few other modifications to the base formulation are necessary in order to make it robust.

$$T' = \sum_{f\in\mathcal{F}}\sum_{d\in\mathcal{D}} b_f \cdot z_{fd} + \sum_{(f_1,f_2)\in\mathcal{L}}\sum_{d\in\mathcal{D}} s_{f_1f_2} \cdot x_{f_1f_2d} + \sum_{(f_1,f_2)\in\mathcal{O}}\sum_{d\in\mathcal{D}\prime} r_{f_1f_2} \cdot y_{f_1f_2d} + \left(\zeta + q_{\overset{\equiv}{f}}\right) \cdot \delta \qquad (2.33)$$

In the base formulation, TAFB is computed as the sum of the elapsed times of the duties in constraint 2.33 the pairing (the first two terms on the right-hand side) plus the sum of the rest periods (the third term in the right-hand side). In constraint (2.33), we modify the base formulation renaming the variable T to T' and adding $\left(\zeta + q_{\overset{\equiv}{f}}\right) \cdot \delta$ to the right-hand side. This extra term equals 0 for RM1. But for RM2 it represents, through the auxiliary variables, the total arrival delay to the last flight in the pairing and hence contributes to the total elapsed time of the pairing. This in turn affects the pairing cost.

$$e'_d = \sum_{f\in\mathcal{F}} b_f \cdot z_{fd} + \sum_{(f,f_1)\in\mathcal{L}} s_{ff_1} \cdot x_{ff_1d} + \left(\zeta + q_{\overline{fd}}\right) \cdot \delta \qquad \forall d \in \mathcal{D} \qquad (2.34)$$

In the base formulation, constraints (2.34) define the elapsed time of the d^{th} duty (e_d) as the sum of all flying times and all sit times in it. In constraints (2.34), we modify the base formulation renaming the variable e_d to e'_d and adding $\left(\zeta + q_{\overline{f_d}}\right) \cdot \delta$ to the right-hand side. This equals 0 for RM1. But for RM2 it equals the arrival delay to the last flight in the duty, through the auxiliary variables. This influences the duty cost and also affects what duties can be considered feasible given the upper limit on the duty elapsed time.

In this section, we have only presented the new and modified PP constraints when compared to the base formulation. The remaining constraints, namely, the cost constraints, pairing constraints, balance constraints, and auxiliary constraints which remain unchanged from the base formulation, are listed in the appendix. The appendix also presents some other new and modified constraints which are not essential for understanding the robustness considerations.

2.4.4 Solution Algorithm

In this section, we propose a new solution algorithm to solve the robust crew pairing model using repeated runs of an off-the-shelf commercial optimization solver. Let $v_{max} \in Z^{+}$ be the maximum number of possible flights in a crew pairing. Then we may need to solve at most v_{max} different PPs. We start with $v = 1$ and solve the corresponding PP. If the solution yields a negative reduced cost, then we add it to the RMP, solve the RMP and again solve the PP for the same v value. When the PP does not produce a negative reduced cost column for that v value, we solve the PP for the next larger integer value of v. When we reach the v_{max} value, we cycle back to $v = 1$ and continue the process. Interestingly, observe that finding v_{max}, which is the maximum number of flights in a crew pairing, is itself a PP that we address by solving the base formulation after setting $c'' = 0, \gamma = 0$ and $\pi_f = -1\ \forall f \in F$. The overall solution process is formally described by Algorithm 1.

For example, consider a 100-flight network, and assume $\Gamma = 50$. Let us also assume that $v_{max} = 20$. Then, instead of solving a single PP we would solve up to 20 different PPs each with a different v value. Since $|F|$ and v are constant parameters of the PP, it retains its linearity property.

```
Algorithm 1
 1: solve RMP;
 2: solve PP with $c'' = 0, \gamma = 0$ and $\pi_f = -1 \; \forall f \in F$ to get $v_{max}$;
 3: set counter = 0;
 4: set $v = 1$;
 5: solve PP for current $v$;
 6: if (the optimal solution of PP gives negative reduced cost)
 7:         add PP solution to RMP;
 8:         solve RMP;
 9:         set counter = 0
10:         goto 5;
11: else
12:         increment counter by 1;
13:         if (counter = $v_{max}$)
14:                 stop - optimum found;
15:         end-if
16:         if ($v < v_{max}$)
17:                 increment $v$ by 1;
18:                 goto 5;
19:         else
20:                 goto 4;
21:         end-if
22: end-if
```

2.5 Theoretical Properties

We present in this section some key theoretical properties of our model, after providing insights into how the robustness budget parameter Γ works for its two variants (RM1 and RM2).

Let us first consider a particular crew pairing $\tilde{p}$ in RM1 that consists of v flights. The PP model ensures that all q_f values for flights other than those covered in pairing $\tilde{p}$ will be automatically set to zero. For a given pairing, the values of c'' and $\sum_{f \in \mathcal{F}} \sum_{d \in \mathcal{D}} \pi_f \cdot z_{fd}$ are constants. Also, let us define $\Gamma' = \frac{v}{|F|} \cdot \Gamma$. Thus, once we fix the pairing this leaves us with a PP objective function given by (2.35). This objective function is comparable to the last two terms in the objective function of the portfolio management model presented in Bertsimas and Thiele (2006). However, since their model involves the maximization of returns, the signs of these two terms are negative. For a given crew pairing $\tilde{p}$, the robust version of our pricing formulation for RM1 equals the following:

$$\min \quad \Gamma' \cdot \zeta + \sum_{f \in \tilde{p}} q_f \tag{2.35}$$

Subject to: $$\zeta + q_f \geq p_f + \Delta_f \qquad \forall f \in \tilde{p} \tag{2.36}$$

$$\zeta \geq 0, q_f \geq 0 \qquad \forall f \in \tilde{p} \tag{2.37}$$

Here, p_f is defined as the propagated delay to flight f in pairing $\tilde{p}$, which is a constant for a given pairing.

We start the analysis of model (2.35)-(2.37) by considering a simple illustrative example of a pairing $\tilde{p}$ consisting of 4 flights $\{f_1, f_2, f_3, f_4\}$ and their total arrival delays given by $p_f + \Delta_f$ as follows.

$$p_1 + \Delta_1 = 1;\ p_2 + \Delta_2 = 2;\ p_3 + \Delta_3 = 3;\ p_4 + \Delta_4 = 4. \tag{2.38}$$

We vary Γ' (thus varying the robustness budget) from 0 to 4. Note that we are limiting this discussion to integer Γ' values ($\Gamma' \in \{0,1,2,3,4\}$) for illustration purposes only. In general, our formulation allows Γ' to take any fractional values as well. The formulation (2.35)-(2.37) then becomes:

$$\min \Gamma' \cdot \zeta + \textstyle\sum_{f \in \{1,2,3,4\}} q_f, \tag{2.39}$$

Subject to: $$\zeta + q_1 \geq 1; \zeta + q_2 \geq 2; \zeta + q_3 \geq 3; \zeta + q_4 \geq 4 \tag{2.40}$$

$$\zeta, q_1, q_2, q_3, q_4 \geq 0 \tag{2.41}$$

Table 1 lists the optimal objective function values for each Γ' scenario. As expected, the value of the objective function increases with increasing Γ'. The corresponding values of decision variables $(\zeta, q_1, q_2, q_3, q_4)$ are also presented.

Table 2.1. Solution to the Illustrative Example

	$\Gamma' = 0$	$\Gamma' = 1$	$\Gamma' = 2$	$\Gamma' = 3$	$\Gamma' = 4$
ζ	4	4	3	2	1
Obj. function value	0	4	7	9	10
q_1 $(\Delta_1 = 1)$	0	0	0	0	0
q_2 $(\Delta_2 = 2)$	0	0	0	0	1
q_3 $(\Delta_3 = 3)$	0	0	0	1	2
q_4 $(\Delta_4 = 4)$	0	0	1	2	3

As shown in Table 1, for this illustrative example, the objective function always equals the sum of the Γ' worst flight arrival delays.

- For Γ' = 1, the optimal value of the objective function is the largest of the four flight arrival delays (4).

- For Γ' = 2, the optimal value of the objective function is the sum of the two largest flight arrival delays (4+3=7).

- For Γ' = 3, the optimal value of the objective function is the sum of the three largest flight arrival delays (4+3+2=9).

- For Γ' = 4, the optimal value of the objective function is the sum of all four flight arrival delays (4+3+2+1=10).

In the following, we generalize these observations. For this, let us define $c^0 = c'' - \sum_{f\in\mathcal{F}}\sum_{d\in\mathcal{D}}\pi_f \cdot z_{fd}$ which has the same form as the objective function of the deterministic version of PP. Also, without loss of generality, let us renumber the flights in pairing $\tilde{p}$ in non-decreasing order of total arrival delays within pairing $\tilde{p}$. Then, we have the following result, which helps in understanding how the proposed formulation works.

Theorem 1: For RM1, the robust version of PP is equivalent to minimizing $c^0 + \kappa\left[\sum_{f\in\{|\tilde{p}|-\lfloor\Gamma'\rfloor+1,\dots,|\tilde{p}|\}}\left(p_f + \Delta_f\right) + (\Gamma' - \lfloor\Gamma'\rfloor)\cdot q_{|\tilde{p}|-\lfloor\Gamma'\rfloor}\right]$ where the term in the square brackets equals the sum of the $\lfloor \Gamma' \rfloor$ largest flight arrival delays plus a fraction $(\Gamma' - \lfloor \Gamma' \rfloor)$ of the $(\lfloor \Gamma' \rfloor + 1)^{\text{th}}$ largest flight arrival delay.

Proof: To prove this result, we need to show that for any given pairing $\tilde{p}$, the optimization model (2.35)-(2.37) sets $\Gamma' \cdot \zeta + \sum_{f \in \tilde{p}} q_f$ equal to $\sum_{f \in \{|\tilde{p}| - \lfloor \Gamma' \rfloor + 1, \dots, |\tilde{p}|\}} \left(p_f + \Delta_f\right) + (\Gamma' - \lfloor \Gamma' \rfloor) * q_{|\tilde{p}| - \lfloor \Gamma' \rfloor}$ in the optimal solution. We can re-write the objective function (2.35) as follows:

$$\Gamma' \cdot \zeta + \sum_{f \in \tilde{p}} q_f$$

$$= (\Gamma' - \lfloor \Gamma' \rfloor) \cdot \zeta + \sum_{f \in \{1, \dots, |\tilde{p}| - \lfloor \Gamma' \rfloor\}} q_f + \sum_{f \in \{|\tilde{p}| - \lfloor \Gamma' \rfloor + 1, \dots, |\tilde{p}|\}} \left(\zeta + q_f\right)$$

$$\geq (\Gamma' - \lfloor \Gamma' \rfloor) \cdot \zeta + \sum_{f \in \{|\tilde{p}| - \lfloor \Gamma' \rfloor + 1, \dots, |\tilde{p}|\}} \left(\zeta + q_f\right)$$

$$\geq (\Gamma' - \lfloor \Gamma' \rfloor) \cdot \zeta + \sum_{f \in \{|\tilde{p}| - \lfloor \Gamma' \rfloor + 1, \dots, |\tilde{p}|\}} \left(p_f + \Delta_f\right).$$

The equality is nothing but a rearrangement of the terms. The first inequality holds because $q_f \geq 0 \; \forall f \in \tilde{p}$ and the second inequality holds due to constraints (2.36).

The last term $\sum_{f \in \{|\tilde{p}| - \lfloor \Gamma' \rfloor + 1, \dots, |\tilde{p}|\}} \left(p_f + \Delta_f\right)$ is the sum of the $\lfloor \Gamma' \rfloor$ largest flight arrival delays. Now we just need to find a feasible solution that has an objective function value of $(\Gamma' - \lfloor \Gamma' \rfloor) * \zeta + \sum_{f \in \{|\tilde{p}| - \lfloor \Gamma' \rfloor + 1, \dots, |\tilde{p}|\}} \left(p_f + \Delta_f\right)$. Such feasible solution is easy to find by first setting ζ to the $(\lfloor \Gamma' \rfloor + 1)^{th}$ largest flight arrival delay in the pairing and then adjusting the last $\lfloor \Gamma' \rfloor$ of the q_f values to make sure the last $\lfloor \Gamma' \rfloor$ of the inequality constraints (2.36) are satisfied as equalities. In other words, it is easy to verify that $\zeta = p_{|\tilde{p}| - \lfloor \Gamma' \rfloor + 1} + \Delta_{|\tilde{p}| - \lfloor \Gamma' \rfloor + 1}$, $q_1 = q_2 = \cdots = q_{|\tilde{p}| - \lfloor \Gamma' \rfloor} = 0$, and $q_{\mathrm{f}} = p_f + \Delta_f - \zeta \; \forall \mathrm{f} \in \{|\tilde{p}| - \lfloor \Gamma' \rfloor + 1, \dots, |\tilde{p}|\}$ is a feasible solution to constraints (36). So, this solution is optimal. Thus, we have proved Theorem 1.■

A corollary of Theorem 1 for integer values of Γ' is given by:

Corollary 1: For RM1, and for integer values of Γ', the robust version of PP is equivalent to minimizing $c^0 + \kappa\left(\sum_{f\in\{|\tilde{p}|-\Gamma'+1,\ldots,|\tilde{p}|\}}(p_f + \Delta_f)\right)$ where the term in the parentheses equals the sum of the Γ' largest flight arrival delays.

It is straightforward to prove this corollary by simply substituting $\Gamma' = \lfloor\Gamma'\rfloor$ in Theorem 1.■

We now illustrate with the help of an example how the model will select more robust pairings at higher values of Γ' and less robust ones at lower Γ' values. Let us consider two pairings each consisting of four flights: one with arrival flight delays of 1, 2, 3, 4 and the deterministic component of the pricing objective function ($c^0 = c_p - \sum_{f\in\mathcal{F}}\sum_{d\in\mathcal{D}}\pi_f z_{fd}$) equal to 50; and the other with flight arrival delays 2, 4, 6, 8 and $c^0 = 45$. For $\Gamma' < 1\frac{1}{3}$, the second pairing is preferred by the PP, while for $\Gamma' > 1\frac{1}{3}$, the first pairing is preferred. At $\Gamma' = 1\frac{1}{3}$, they both are equally preferred. Note that the second solution is better for the deterministic version of the PP, but the first one is more robust and hence is preferred at higher Γ' values.

From this discussion, we note several similarities as well as differences between RM1 and the original concept of a robustness budget by Bertsimas and Sim (2004). Our model, like theirs, allows the parameter Γ' to dictate the level of robustness where a larger value corresponds to more protection against uncertainty. Like the original Bertsimas and Sim (2004) model, RM1 also produces a solution identical to the deterministic model when we set the robustness budget to 0; and produces the most robust solution when the robustness budget is set to be equal to the total number of flights in the pairing.

However, as mentioned in the introduction, our deterministic model is considerably more challenging than the general linear programming model presented in Bertsimas and Sim (2004). Moreover, the way uncertainty affects our model is quite different and more complex than just a change in the constraint coefficients and/or objective function coefficients. Indeed, uncertainty in a delay propagation setting is measured at two different levels: 1) root delays of the flights, and 2) total arrival delays of the flights, with the latter being the sum of the root delays and the propagated delays. Therefore, our treatment of uncertainty is somewhat different than that of Bertsimas and Sim (2004). They define uncertainty sets as the parameter value intervals given by $\left[\bar{a}_{ij} - \hat{a}_{ij}, \bar{a}_{ij} + \hat{a}_{ij}\right]$ and their model finds the optimal solution assuming that at most $\lfloor \Gamma_i \rfloor$ of the coefficients in the i^{th} constraint take their worst-case values (i.e., $\bar{a}_{ij} + \hat{a}_{ij}$), one other coefficient in the i^{th} constraint takes a value which is at most $(\Gamma_i - \lfloor \Gamma_i \rfloor)$ times as bad as the worst-case value, and all other coefficients remain at their nominal values (i.e. at $\bar{a}_{ij}$). In contrast, in RM1, we define the uncertainty set for the root delay of flight f as $\left[0, \Delta_f\right]$ and we find an optimal solution assuming that, while all root delays can take their worst-case values (Δ_f), the delay cost of a pairing only accounts for the $\lfloor \Gamma' \rfloor$ largest flight arrival delays and a fraction $(\Gamma' - \lfloor \Gamma' \rfloor)$ of the $(\lfloor \Gamma' \rfloor + 1)^{\text{th}}$ largest flight arrival delay.

Recall that RM1 ignores the fact that sometimes the actual elapsed time of a crew duty and/or the actual TAFB are larger compared to their scheduled values due to arrival delays to the last flights in a duty. Theorem 1 and its Corollary 1 do not necessarily hold for RM2. However, even for RM2, Theorem 1 and Corollary 1 do hold in certain cases as stated by Theorem 2 below.

Theorem 2: For RM2, Theorem 1 and Corollary 1 hold if, for every feasible pairing, the last flights in all duties are among the $\lfloor \Gamma' \rfloor$ flights with the largest arrival delays.

Proof: Consider the same solution that we considered in the proof of Theorem 1: $\zeta = p_{|\tilde{p}|-\lfloor \Gamma' \rfloor+1} + \Delta_{|\tilde{p}|-\lfloor \Gamma' \rfloor+1}$, $q_1 = q_2 = \cdots = q_{|\tilde{p}|-\lfloor \Gamma' \rfloor} = 0$, and $q_{\mathrm{f}} = p_f + \Delta_f - \zeta \ \forall f \in \{|\tilde{p}| - \lfloor \Gamma' \rfloor + 1, \dots, |\tilde{p}|\}$. Then, for the last flight f of every duty in pairing $\tilde{p}$, $\zeta + q_f = p_f + \Delta_f$. Note that this is the minimum value of $\zeta + q_f$ allowable by constraints (2.28). Thus, this gives us the minimum possible value of T' in pairing $\tilde{p}$ (as per constraint (2.33)) and the minimum possible value of e'_d in pairing $\tilde{p}$ (as per constraints (2.34)). So, if this solution is not feasible for the PP, then no other solution with this pairing $\tilde{p}$ is feasible. Also, T' and e'_d both have a non-decreasing effect on c^0, and therefore these minimum values of T' and e'_d minimize c^0 for the given pairing $\tilde{p}$. Moreover, we have already proved that this solution minimizes the expression $\Gamma' \cdot \zeta + \sum_{f \in \tilde{p}} q_f$ for a given pairing $\tilde{p}$. Thus, this solution minimizes both parts of objective function (2.27) for a given pairing $\tilde{p}$, and hence minimizes objective function (2.27) for a given pairing $\tilde{p}$. Thus, we have proved that Theorem 1 holds for RM2. Corollary 1 follows by substituting $\lfloor \Gamma' \rfloor = \Gamma'$. ■

Thus, Theorem 1 and Corollary 1 are, in general, more likely to hold for higher, rather than lower, Γ' values. Note that the condition stated in Theorem 2 (that for every feasible pairing, the last flights in all duties are among the $\lfloor \Gamma' \rfloor$ flights with the largest arrival delays), is sufficient but not necessary for Theorem 1 and Corollary 1 to hold for RM2. For example, it is easy to see that for problem instances with $\alpha = \beta = 0$ and large values of λ, Theorem 1 and Corollary 1 will hold for RM2 even without requiring the conditions in Theorem 2. In other cases, however, constraints (2.33) and (2.34) could force a lower value of ζ and higher values for some of the q_f variables than those in the solution considered in the proof of Theorem 1. This *may* result in a larger optimal value of the $\Gamma' \cdot \zeta + \sum_{f \in \tilde{p}} q_f$ part of the objective function for RM2 compared with that for RM1. In other words, RM2 is equivalent to minimizing c^0 plus a component that may be greater than the sum of the $\lfloor \Gamma' \rfloor$ largest flight arrival delays plus a fraction $(\Gamma' - \lfloor \Gamma' \rfloor)$ of the $(\lfloor \Gamma' \rfloor + 1)^{\text{th}}$ largest flight arrival delay, and this effect is prominent at small but nonzero Γ' values. While RM2 lacks the monotonicity properties (that is, Theorem 1 and Corollary 1) enjoyed by RM1, the RM2 is a more comprehensive characterization of the various mechanisms through which delays and disruptions propagate through crew connections. Hence, in the case study described in the following sections, we use both variants of our model, and compare the results of both variants with each other and with those of the deterministic crew pairing model.

2.6 Case Study

In the case study, we used real-world data on U.S. domestic flight delays from the Airline On-Time Performance (AOTP) database (Bureau of Transportation Statistics, 2016). For all our experiments, we used the flight network of Virgin America (VX) for calendar year 2014. We chose VX because its network stretches across continental U.S. and thus limits possible regional biases in our results. Besides, unlike many other airline carriers with nationwide networks, VX has a single fleet family, because of which we do not need to subdivide the network into parts to apply our model. Thus, our computational experiments are based on the entire VX network providing a more rigorous test of our modeling and solution approach. If we use the datasets from other moderate-sized airlines, we would need to focus on smaller-sized sub-networks corresponding to individual fleet families. We include in our network (see Figure 1) all VX flights that were flown daily. Our network consists of 94 daily flights connecting 14 continental U.S. airports. Figure 1 also provides a detailed view of the VX network within the northeastern part of U.S., since several airports in this network are located in this region which would be difficult to identify without zooming-in.

Before performing computational experiments using the VX network, two important types of parameters need to be set. Section 2.7.1 discusses the choice of the uncertainty sets, represented by the Δ_f parameters, while Section 2.7.2 discusses the choice of the unit cost of delays represented by the parameter γ.

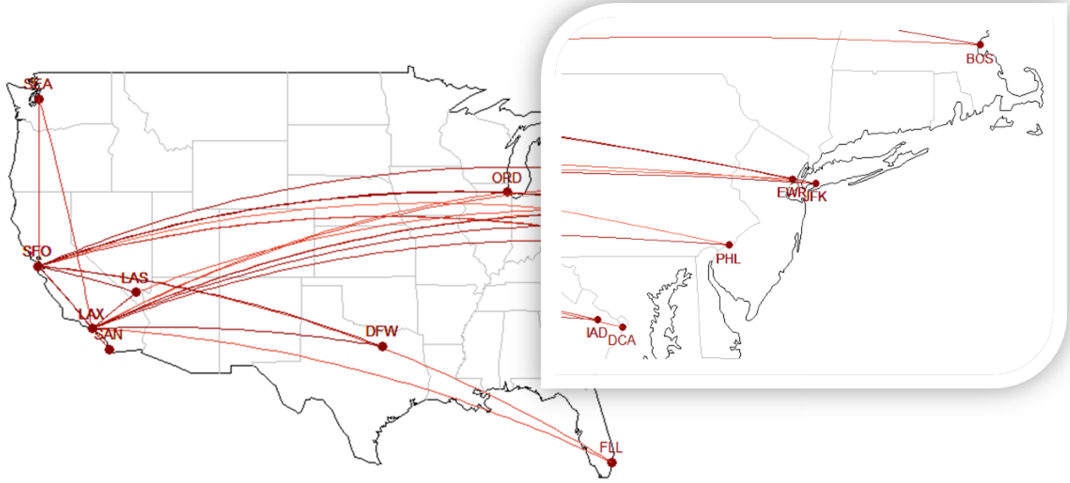


Figure 2.1. Route map of the VX network considered for the case study

2.6.1 Uncertainty Sets

As mentioned in Section 2.6, our model differs from the model in Bertsimas and Sim (2004) with respect to the treatment of the uncertainty sets. We define uncertainty in flight delays at two different levels, namely, root delays and total arrival delays. We define the uncertainty set for root delays as $[0, \Delta_f]$. While our model sets all root delays to their worst-case values, Δ_f, the delay cost of a pairing only accounts for the $\lfloor \Gamma' \rfloor$ largest flight arrival delays and a fraction $(\Gamma' - \lfloor \Gamma' \rfloor)$ of the $(\lfloor \Gamma' \rfloor + 1)^{\text{th}}$ largest flight arrival delay. We need to select the Δ_f values carefully because they directly impact our model's performance.

We used the flight delays data from the entire year 2014 from the AOTP database (Bureau of Transportation Statistics, 2016) to calculate the Δ_f values. The AOTP database consists of the scheduled and actual arrival times for all flights in our network. Additionally, for each flight with an actual arrival time that is later than the scheduled arrival time by 15 minutes or more, it also provides the source of the delay. The possible sources are: air carrier delay; extreme weather delay; national aviation system delay; security delay; and late aircraft delay. To calculate the root delays, we would like to eliminate the delays due to propagation through aircraft connections and through crew connections. The late aircraft delay category represents the delays due to propagation through aircraft connections. The air carrier delay category includes the delays due to propagation through crew connections, as well as non-propagated delays due to a variety of other causes including aircraft maintenance, cleaning, baggage loading, fueling, etc. Therefore, the exact value of delays propagating through crew connections is not publicly available. Furthermore, when the actual arrival time is earlier than 15 minutes past the scheduled arrival time, the AOTP database does not provide a delay cause. Due to these two data limitations, we are not able to eliminate all the propagated delays. Instead, we approximate root delay to be equal to the actual flight arrival time minus scheduled flight arrival time minus late aircraft delay, which results in slightly over-counting the root delays due to the two aforementioned reasons. Note that this is a limitation of the publicly available data and not a limitation of our methodology. An airline interested in using our methodology will have, available at its disposal, proprietary data that does not need to rely on any of these approximations. Note that our definition of root delay allows it to be negative.

We analyze these root delays for the full calendar year to identify reasonable values for the Δ_f parameters. We divided all root delay data by the associated nonstop flight segments. There are 47 (nonstop) segments in VX daily network. Figure 2 displays box-plots of root delays per segment. Each data point represents the root delay to a flight on that segment. Each box has a solid black line in the middle representing the median and the two ends of the box represent the 1st and 3rd quartiles. We define IQR (Inter-Quartile Range) as the difference between the 1st and 3rd quartiles. The lines extending in either direction from the box stretch between the smallest data point that is at least equal to 1st quartile minus 3 IQRs, and the largest data point that is at most equal to 3rd quartile plus 3 IQRs. Additionally, each data point that is at most equal to the 1st quartile minus 1.5*IQR and each data point that is at least equal to the 3rd quartile plus 1.5*IQR is shown in the figure by an individual red dot. Figure 2.2 indicates that the root delay distribution is highly skewed with a long tail and several outliers on the right side. Inclusion of all outliers while defining Δ_f values would create overly conservative schedules given that such large delays very rarely occur. On the other hand, ignoring all outliers would amount to ignoring a large part of the distribution. Therefore, to identify a good tradeoff between these two extremes, we tested various strategies for outlier removal before selecting the strategy of setting Δ_f equal to the largest data point that is at most equal to the 3rd quartile plus 6*IQR.

2.6.2 Unit Cost of Delay

The unit cost of delay is represented by γ. To set this value, we need to find the total cost of an hour of flight delay (excluding the cost due to additional crew pay) and the crew pay per hour of flying time. We based our calculations on two sources:

MIT Airline Data Project – From the Global Airline Industry Program (Massachusetts Institute of Technology, 2014), we obtained the crew pay per hour of flying time. For VX, we found this value to be $745.55 per hour.

From the Airlines for America's estimate (Airlines for America 2015) of unit delay cost ($77.81 per minute), we subtracted the per-hour cost of crew flying time ($745.55 per hour) to estimate the delay cost excluding that of additional crew pay to be approximately $3,923.05 per hour.

Thus, the ratio of delay costs (excluding crew delay costs) to crew pay is around 5.3, and we set $\gamma = 5.3$ for the RM2. For RM1 however, crew pay costs do not include the additional costs due to crew delays. Instead we included crew delay costs also as part of the total delay costs by using a value of 6.3 for γ instead of 5.3.

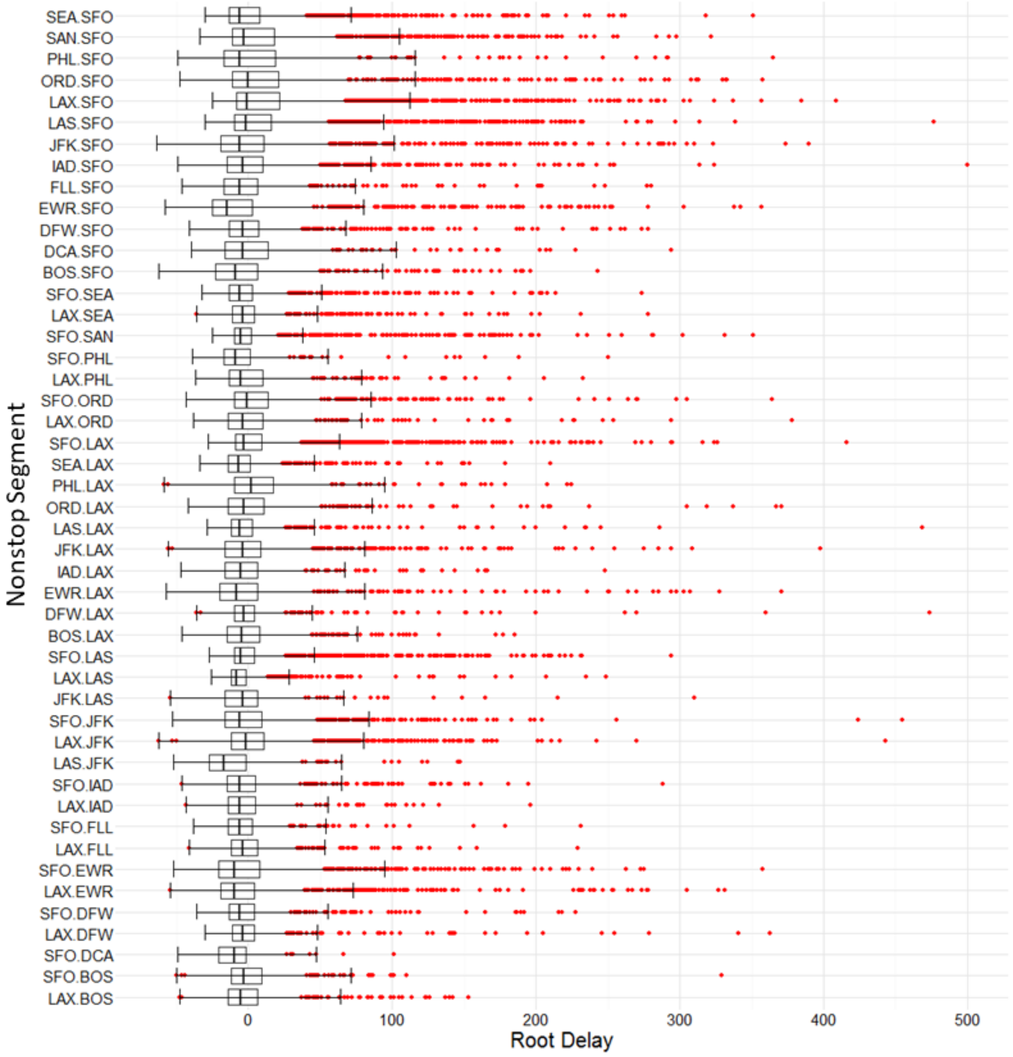


Figure 2.2. Boxplots of root delays (minutes) per nonstop segment

2.7 Case Study Results

For both variants of our model, RM1 and RM2, we ran 11 scenarios each with a different Γ value in {0,10,20,30,40,50,60,70,80,90,94}. IBM ILOG CPLEX 12.6 was used to solve all optimization instances, accessing the solver through ILOG Concert Technology with the code written in the *Java* language. All experiments were performed on a computer with an Intel® Core™ i7-920 (8 core) 2.66 GHz CPU, with 8 GB RAM and Windows 7 Professional 64-bit operating system. Each run took approximately 4 hours. Figure 3 shows the variation in the optimal value of the objective function (expression (2.6)) against changes in Γ values.

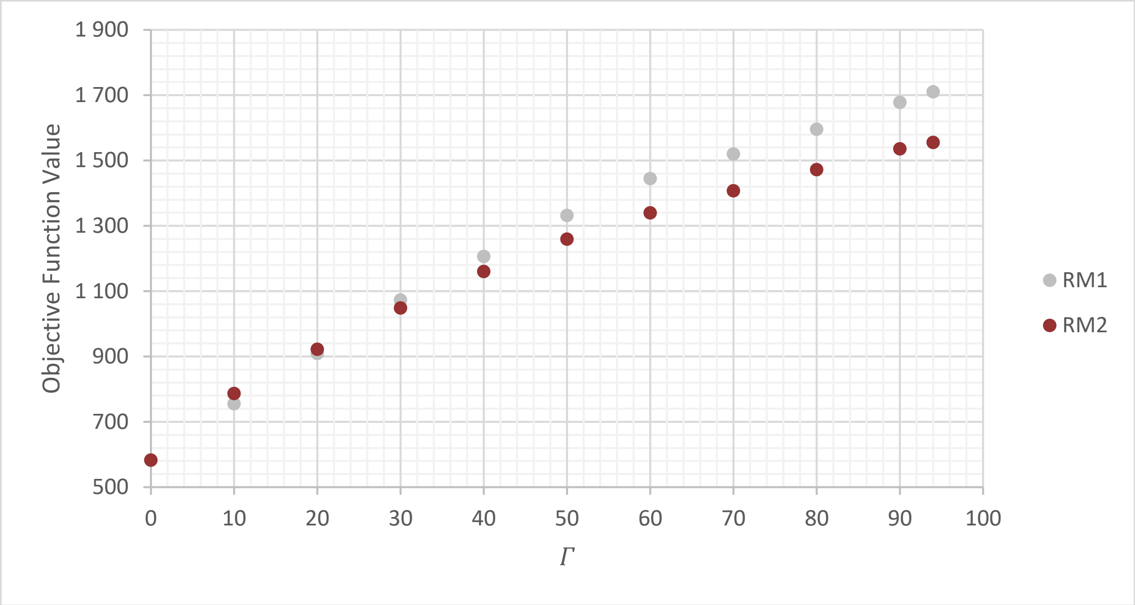


Figure 2.3. Change in optimal objective function value with Γ

As expected, with increasing Γ values, both model variants show a monotonic increase in the optimal value of the objective function, highlighting the fact that the objective function (2.6) protects against increasingly larger delay values. Recall that, unlike RM2 which incurs additional crew delay costs only when the last flight in a duty has a delayed arrival, RM1 incurs additional crew delay costs for every delayed flight arrival, as reflected through the different γ values used in the two model variants (6.3 for RM1 and 5.3 for RM2). As a result, RM1 is generally found to have higher optimal values of the objective function than RM2 (especially for larger Γ) despite having less strict versions of constraints (2.33) and (2.34).

Next, in Figure 2.4, we present the percentage change in planned crew pairing costs as a function of Γ. The percentage change is calculated by first subtracting the optimal value of the objective function for $\Gamma = 0$ from the planned crew cost under the optimal solution for a given Γ value, and then dividing the difference by the optimal value of the objective function for $\Gamma = 0$. As explained in Section 6, for RM1 the crew pairing cost component of the objective function expression equals the planned crew costs. However, for RM2 the crew pairing cost component of the objective function may exceed the planned crew costs. We find that, with the increase in Γ values, planned crew costs show a generally increasing but non-monotonic trend for both RM1 and RM2. Across the two model variants and across the 11 Γ values, the percentage increase in planned crew costs does not exceed 4% in all but one of the 22 combinations. In fact, for all 12 tested scenarios with $\Gamma \leq 50$, the percentage increase is under 3% for RM2 and under 2% for RM1. This demonstrates the ability of our model to add robustness as the expense of a very small increase in planned crew costs.

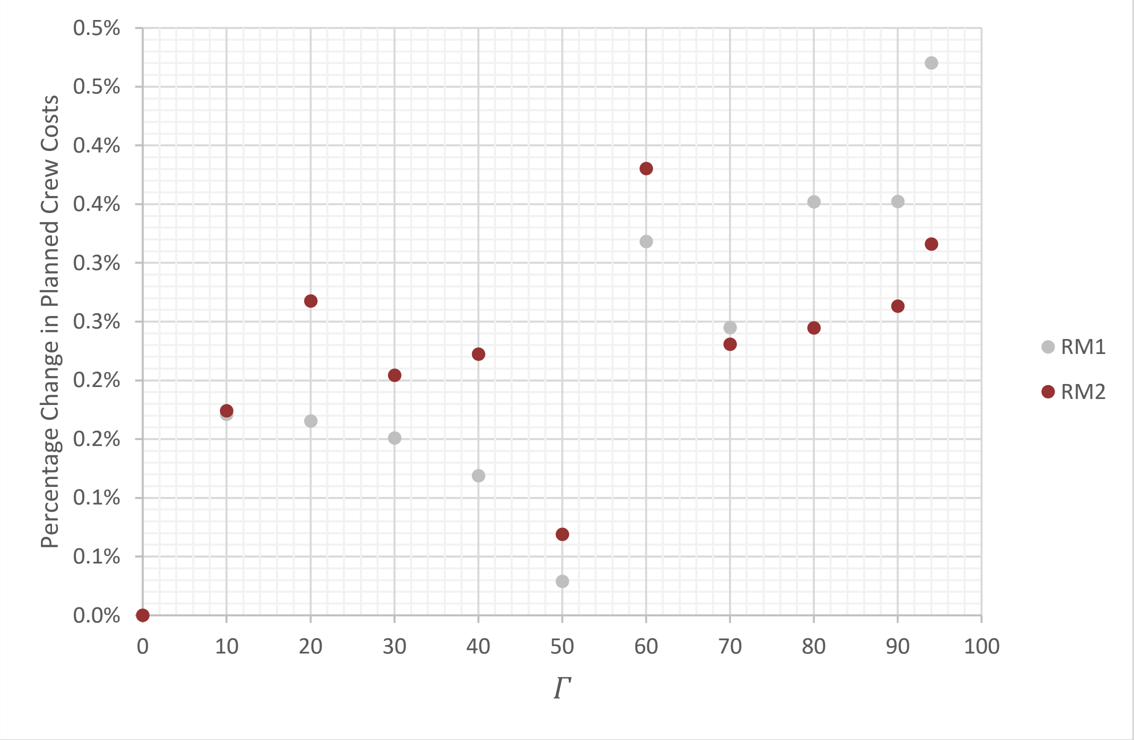


Figure 2.4. Percentage change in planned crew costs with Γ

To assess the quality of our robust solutions, we developed a detailed simulation approach. It uses the pairings generated by our robust crew pairing model as inputs, and outputs the probability distributions of delays propagated through crew connections as well as crew disruptions. We use two alternative ways to generate the root delays to be used as simulation inputs. First, we use a non-parametric kernel function for probability density estimation of root delays using root delay data for each flight in our network during calendar year 2014. Then, for each Γ value, we use those estimated density functions for generating 10,000 delay samples for every flight. Let us denote this as *Simulation Approach 1*. Note that this approach implicitly assumes that the root delays of different flights are statistically independent of each other. Our second approach, which we call *Simulation Approach 2*, does not require independence. In this approach, we simply use the actual root delays for every day in year 2014, thus generating 365 root delay samples. Thus, this second approach captures the underlying root delay correlations and allows us to test how that affects the performance of our robust solutions, although using much fewer root delay samples than the first approach.

In Table 2.2 we show the upper percentiles of the total propagated delay (in hours) across all flights in the network under Simulation Approach 1. For simplicity, we show only the upper percentiles (from 90th to 100th) and only a subset of the 11 tested Γ values.

Table 2.2. Upper percentiles of the total propagated delay using Simulation Approach 1

RM	Γ	90th	91th	92th	93th	94th	95th	96th	97th	98th	99th	100th
1	0	29.6	34.9	40.9	49.1	60.4	79.1	97.1	132.0	187.0	282.8	526.4
	20	29.8	33.7	38.6	45.0	51.9	62.6	72.9	90.0	117.6	167.4	411.1
	50	26.4	30.0	34.3	40.0	45.9	55.5	65.4	83.1	113.0	167.9	461.3
	80	30.3	34.2	39.1	45.9	55.6	70.4	85.1	110.0	152.2	216.7	533.2
	94	28.0	21.6	35.7	41.1	47.2	56.7	66.2	82.4	110.9	159.7	396.2
2	0	29.6	34.9	40.9	49.2	60.4	79.1	97.1	132.0	187.0	282.8	526.4
	20	25.9	29.6	34.2	40.6	48.4	61.3	74.8	96.8	131.7	189.5	582.6
	50	26.1	29.7	34.1	40.2	46.7	57.6	68.8	87.1	130.8	205.0	661.4
	80	27.1	30.4	34.6	40.0	46.2	55.7	65.7	83.4	113.7	165.6	550.6
	94	28.8	32.5	37.5	43.7	50.3	60.9	70.8	87.3	116.2	166.8	423.7

We see that, for both RM1 and RM2, most of these upper percentiles of the propagated delays are the highest for $\Gamma = 0$ and the lowest for $\Gamma = 94$. Comparing the two model variants we see that their performances are similar, though RM1 performs slightly better overall. For each model variant, the average improvement in these percentiles is more than 20% compared to the deterministic model. In terms of the 96th and higher percentiles, the benefits of our robust model are even more impressive with RM1 and RM2 resulting in more than 28% and 25% average improvement respectively.

In addition to the planned crew costs and the distributions of propagated delays, a comprehensive comparison of the results should involve comparing the likelihood of infeasibilities in the crew schedules. As explained in Section 2.2, robust schedules can be thought of as those which do not allow delays and disruptions to propagate easily and/or those which are easy to recover once disrupted. Since the ability to be easily recovered is beyond the scope of this paper, we focus on the likelihood of delay propagation and disruptions. Aside from delay propagation, flight delays may lead to violations of the maximum duty elapsed time rule. In Table 2.3 we show the upper percentiles of the total number of infeasibilities (i.e., the total number of crew rule violations) under Simulation Approach 1.

Once again we see that, in general, for both models, most percentiles of the number of crew infeasibilities are the highest for $\Gamma = 0$ and the lowest for $\Gamma = 94$. The performances of the models are again comparable, but here RM2 performs slightly better overall (as evidenced by fewer infeasibilities than RM1 in most cases). This is consistent with the fact that RM2 explicitly accounts for these potential infeasibilities through constraints (2.34) and tries to protect against them.

If we compare the 96th and higher percentiles (since most values lower than the 96th percentile are zeros anyway), RM1 and RM2 yield substantial average improvements (of more than 37% and 44% respectively) compared to the deterministic model. The results in Table 2.2 and Table 2.3 in conjunction with Figure 4 highlight the fact that both variants of our model bring significant improvements in the upper percentiles of delays and disruptions in return for a much smaller percentage increase in planned crew costs.

Table 2.3. Upper percentiles of the number of crew infeasibilities using Sim. Approach 1

RM	Γ	90th	91th	92th	93th	94th	95th	96th	97th	98th	99th	100th
1	0	0	0	0	0	0	0	2	4	8	11	48
	20	0	0	0	0	0	0	0	3	4	9	43
	50	1	1	1	1	1	1	1	2	3	8	42
	80	0	0	0	0	0	1	1	2	5	10	44
	94	0	0	0	0	0	0	0	2	6	11	42
2	0	0	0	0	0	0	0	2	4	8	11	48
	20	0	0	0	0	0	0	0	2	5	8	44
	50	0	0	0	0	0	0	0	2	5	11	44
	80	0	0	0	0	0	0	0	2	4	11	51
	94	0	0	0	0	0	0	0	1	3	8	45

Using Simulation Approach 2, we repeated these computations of the percentiles of total propagated delays and the number of crew infeasibilities, for both RM1 and RM2. The upper percentiles for a representative set of Γ values are presented in Tables 2.4 and 2.5, which are equivalent to Tables 2.2 and 2.3 for Simulation Approach 1.

Table 2.4. Upper percentiles of the total propagated delay using Sim. Approach 2

RM	Γ	90th	91th	92th	93th	94th	95th	96th	97th	98th	99th	100th
1	0	22.2	24.1	27.0	31.4	35.0	36.5	45.6	55.1	61.5	82.1	128.7
	20	24.6	26.6	29.7	34.3	37.6	41.3	44.5	51.2	65.1	70.3	113.9
	50	18.6	20.8	23.3	28.2	29.7	32.8	36.9	46.5	56.7	67.7	121.4
	80	21.0	23.4	23.8	28.9	30.9	33.2	39.0	43.9	58.3	65.0	105.1
	94	18.1	19.9	22.9	24.8	27.2	30.0	34.4	40.9	48.5	64.5	99.5
2	0	22.2	24.1	27.0	31.4	35.0	36.5	45.6	55.1	61.5	82.1	128.7
	20	18.9	21.3	24.7	28.1	32.7	33.6	35.6	52.3	63.6	68.1	128.1
	50	17.7	22.2	24.7	28.6	31.3	32.5	41.8	52.2	66.5	71.9	132.3
	80	23.5	26.3	28.5	33.5	36.1	40.0	44.6	50.3	65.1	74.5	108.1
	94	20.5	22.1	23.8	27.5	29.0	31.7	35.2	43.5	50.9	62.0	101.7

Similar to the results of Simulation Approach 1, we observe that, in most cases, the upper percentiles of total propagated delays and number of crew infeasibilities are found to be the highest (or at least close to the highest) for $\Gamma = 0$ and the lowest for $\Gamma = 94$. However, unlike the results of Simulation Approach 1, here we observe that the performances of the two models are not comparable. Instead, we find that in terms of propagated delay distributions RM1 consistently outperforms RM2, while in terms of the distributions of the number of crew infeasibilities RM2 outperforms RM1. If we compare the 96th and higher percentiles, RM1 and RM2 yield an average improvement of 15% and 10%, respectively, in terms of the propagated delays. Similar comparison based on the number of crew infeasibilities shows no significant improvement for RM1, but a 13% average improvement for RM2. Both these observations are consistent with how the models are structured. RM1 has a higher γ value which further reduces the propagated delays compared with RM2. On the other hand, RM2 is more conservative in constraints (2.6) thus resulting in fewer infeasibilities than RM1. The results in Table 2.4 and Table 2.5 when compared to those in Tables 2.2 and 2.3 show that the overall improvements in the upper percentiles of delays and disruptions are lower, but still significant, under Simulation Approach 2 than those under Simulation Approach 1.

In addition to the upper percentiles, we also calculated the expected values and standard deviations of propagated delays and number of infeasibilities. They are summarized in Tables 2.6 and 2.7 for Simulation Approach 1 and Simulation Approach 2, respectively. All expected values are followed, in parentheses, by the standard errors (that is, the ratio of the standard deviation of the simulation result divided by the square root of the number of simulation runs) to provide a measure of their statistical significance. Also, in order to allow for an easy comparison of the benefits and costs of each of the robust solutions under each simulation approach, the increases in planned crew costs are also provided in these tables. These increases are obtained by subtracting the planned crew cost of the deterministic solution (i.e., that obtained for $\Gamma = 0$) and are reported in the units of crew flying hour costs.

Table 2.5. Upper percentiles of the number of crew infeasibilities using Sim. Approach 2

RM	Γ	90th	91th	92th	93th	94th	95th	96th	97th	98th	99th	100th
1	0	0	0	0	0	0	0	1	1	2	2	8
	20	0	0	0	0	0	0	1	1	2	2	7
	50	1	1	1	1	1	1	1	1	2	2	7
	80	0	0	0	0	0	0	1	1	2	2	7
	94	1	1	1	1	1	1	1	2	2	3	5
2	0	0	0	0	0	0	0	1	1	2	2	8
	20	0	0	0	0	0	0	1	1	2	2	7
	50	0	0	0	0	0	0	1	1	2	2	7
	80	0	0	0	0	1	1	1	1	1	2	5
	94	0	0	0	0	0	0	1	1	1	1	4

In order to allow meaningful comparisons across Tables 2.6 and 2.7, let us recall that the cost of one hour of delay has been estimated to be approximately equal to the cost of 6.3 hours of crew flying time. Several key insights can be drawn from the results in these two tables. First, we observe that, similar to the upper percentile results, these results do not exhibit monotonic trends in expected values of delays and infeasibilities. However, in all cases, the solutions for $\Gamma = 0$ show the highest or near-highest expected values while solutions for $\Gamma = 94$ show the lowest or near-lowest expected values. Standard errors are stable across Γ values.

Table 2.6. Increases in the planned crew costs and expected values (and standard errors) of total propagated delays and number of infeasibilities for Sim. Approach 1

Γ	Increase in Planned Crew Costs (in Crew Flying Hours)		Total Propagated Delays (in Hours)		Number of Infeasibilities	
	RM1	RM2	RM1	RM2	RM1	RM2
0	0	0	15.12 (0.40)	15.12 (0.40)	0.39 (0.03)	0.39 (0.03)
20	9.65	15.61	9.90 (0.32)	11.33 (0.37)	0.24 (0.02)	0.27 (0.02)
50	1.69	4.02	10.40 (0.32)	11.51 (0.40)	0.69 (0.02)	0.31 (0.02)
80	14.29	13.46	12.79 (0.40)	14.93 (0.33)	0.34 (0.02)	0.33 (0.03)
94	27.44	18.44	10.34 (0.30)	10.60 (0.31)	0.27 (0.02)	0.25 (0.02)

Table 2.7. Increases in the planned crew costs and expected values (and standard errors) of total propagated delays and number of infeasibilities for Sim. Approach 2

Γ	Increase in Planned Crew Costs (in Crew Flying Hours)		Total Propagated Delays (in Hours)		Number of Infeasibilities	
	RM1	RM2	RM1	RM2	RM1	RM2
0	0	0	9.37 (0.80)	9.37 (0.80)	0.14 (0.03)	0.14 (0.03)
20	9.65	15.61	9.12 (0.70)	9.13 (0.73)	0.13 (0.03)	0.13 (0.03)
50	1.69	4.02	9.10 (0.69)	9.10 (0.76)	0.19 (0.03)	0.13 (0.03)
80	14.29	13.46	9.10 (0.67)	9.12 (0.70)	0.13 (0.03)	0.12 (0.02)
94	27.44	18.44	9.08 (0.59)	9.08 (0.64)	0.19 (0.03)	0.08 (0.02)

For Simulation Approach 1, all robust solutions (i.e., those with $\Gamma > 0$) provide a significant reduction in expected delay costs, which more than compensates (after accounting for the aforementioned factor of 6.3) for the increase in planned crew costs. Note that this analysis does not include the benefits to passengers due to reduction in delay costs, which are estimated (Ball et al. 2010; Barnhart et al. 2014) to be twice as large as those to the airlines. Moreover, Table 2.6 shows that, in most cases, the number of infeasibilities under robust solutions also shows a statistically significant reduction compared with the deterministic solution. While the benefits of these are difficult to quantify in a precise manner, they correspond to a considerable additional reduction in disruption costs to airlines and passengers. Finally, the undesirability of delay and disruption costs goes well beyond what can be captured purely by their expected values, because of the unpredictability associated with them, compared with equal amount of the planned crew costs. Indeed, the upper percentile results for robust solutions as presented earlier in this section point to even larger benefits of robust solutions than those indicated by the reductions in expected values. In summary, Simulation Approach 1 demonstrates in a comprehensive manner the substantial overall superiority of our robust crew pairing solutions compared to the deterministic solutions and makes a compelling argument for their implementation in practice.

For Simulation Approach 2, all robust solutions result in lower expected values of total propagated delays. In fact, barring a couple of exceptions, expected values of both total propagated delays and number of infeasibilities show a monotonically decreasing trend with increases in Γ values. However, the differences are not statistically significant in many cases, and in many cases they, by themselves, do not fully compensate for the corresponding increases in planned crew costs (after accounting for the aforementioned factor of 6.3). The lack of statistical significance is likely due to the significantly fewer (~0.3%) simulation runs used for Simulation Approach 2 (365) than for Simulation Approach 1 (10,000). Recall that Simulation Approach 2 strives to capture the root delay correlations across flights, and hence faces the challenge of using the limited number of available real-world realizations from the joint distribution of the large number of root delay variables.

The somewhat lower propagated delay reductions for the robust solutions under Simulation Approach 2 are consistent with the fact that our robust crew pairing model does not explicitly account for root delay correlations. However, as mentioned earlier in this section, the reduction in expected delay costs to the airlines is only a small subset of the overall benefits of robust models. When combined with passenger delay reduction benefits, reductions in the number of infeasibilities, and the benefits due to less uncertainty in delays and disruptions, the robust solutions are expected to more than compensate for the corresponding small increases in planned crew costs. Indeed, even under Simulation

Approach 2, the relative improvements in upper percentiles of total propagated delays (as shown in Tables 2.4 and 2.5) are much larger than those in their expected values shown in Table 7 clearly demonstrating the value of our robust solutions.

2.8 Conclusion

In sum, this paper makes four major contributions.

- First, we develop a new mixed-integer linear programming model for the robust crew pairing problem. Unlike most existing studies on crew pairing problems under uncertainty, our model captures the highly non-linear pay-and-credit crew salary scheme, a variety of complex real-world constraints on crew pairing feasibility and legality, and a variety of complex effects on the airline network due to the propagation of delays and disruptions through crew connections.

- Second, we develop a new solution algorithm that iteratively uses the commercial optimization solvers to solve the robust crew pairing problem to optimality. Unlike most existing studies on crew pairing problems, our approach does not require extensive customization of this new solution algorithm which facilitates its implementation in practice.

- Third, we motivate the effectiveness of our approach analytically for a small example and demonstrate it computationally on a moderate-sized real-world airline network. Our analytical results provide a series of intuitive insights into the functioning of the highly complex robust crew pairing model and highlight a number of mathematical properties of the two model variants. Our computational experiments underline the ease of implementation and fast execution of our model making it especially suitable for being used by the practitioners.

- Finally, we test our robust solutions in a detailed simulation environment using real-world delay data to demonstrate their effectiveness in reducing both the averages and the variability of delay and disruption costs to airlines and passengers. We

> demonstrate that our model leads to significant total benefits (lower values of averages and upper percentiles of flight delays, fewer crew infeasibilities, and lower passenger delays and disruptions) at the expense of a small increase in planned crew costs that is often below 3% and never exceeding 5%.

Hundreds of airlines around the world have similar-sized or smaller networks and fleets than our case study network. These airlines can benefit directly from this research. Examples of well-known such airlines include Transavia (42 aircraft), Air Canada Rouge (51 aircraft), China United Airlines (42 aircraft), Frontier Airlines (77 aircraft), CEBU Pacific Air (62 aircraft) and Thomas Cook Airlines (35 aircraft). Additionally, several other, larger airlines use multiple fleet families where the crew pairing decisions are often made separately for each fleet family's sub-network. Our model and solution algorithm can be directly applied to several such sub-networks within well-known large airlines across the world. Most studies in the previous literature on airline schedule optimization under uncertainty, and especially those focused on crew pairing, have typically used small- to moderate-sized networks, and have tackled instances no larger than the ones we used in this paper. Yet, given the ability of our model and solution algorithm to result in an optimal solution within a 4-hour time period for these moderate-sized networks, there are clear opportunities to test larger-sized instances directly using our model and solution algorithm. Moreover, we anticipate that our model and solution algorithm, when combined with additional exact and heuristic acceleration techniques, may enable solving the instances corresponding to the mega airlines that have recently emerged from the airline mergers across the world. This is a very promising direction for extending this research in the future.

On the modeling side, another promising direction for future research would be to explicitly incorporate the root delay correlations within our robust optimization framework for crew scheduling. The findings of recent studies in other areas, such as robust aircraft routing (e.g., Yan and Kung 2016), show that explicitly accounting for delay correlations in the definitions of delay uncertainty sets increases the effectiveness of the robust optimization approach. Thus, explicit handling of delay correlations in the definitions of uncertainty sets would be a logical next step in this research.

To fully assess the capabilities and performance of the pairings generated by our approach, future research may focus on developing a simulation model that includes an operations recovery model so that the cost of the infeasibilities can be accurately assessed. Furthermore, a passenger rebookings simulator should be integrated into the recovery-based simulation model to allow for full evaluation of passenger delay and disruption costs. Finally, integrated treatment of aircraft and crew schedules in robust scheduling and recovery models is another promising direction for future research.

3 AN AGENT-BASED SIMULATION FRAMEWORK FOR AIRLINE DAILY OPERATIONS AND ITS APPLICATION ON THE ASSESSMENT OF SCHEDULE RECOVERY MODELS

3.1 Introduction

Planning for uncertainty is one of the most challenging tasks that airlines face. Airline operations are subjected to several sources of randomness, and airlines need to consider several dimensions, components and interactions within their networks. Thus, the consequences of disruptions in their operations are difficult to comprehend. One way that can help tackle this challenge is a stochastic simulation framework that replicates the operations of an airline. Therefore, we propose a framework to simulate in detail the daily operations of an airline in a stochastic way and understand how decisions at an operational level, i.e. schedule recovery actions, will affect the overall operations.

When simulating an airline's operations, it is essential to accurately capture the response of the airline to the uncertainty and disruptions during operations. The actions of an airline during a major disruption are a significant determinant of the eventual outcomes and hence need to be modeled with especial care. Airlines call these major disruption situations as irregular operations, and the airline's actions to mitigate the problem are known as schedule recovery actions. When simulating airline operations one can represent this schedule recovery actions with a schedule recovery model that closely mimics the behavior of the airline being simulated. Interestingly, as will be shown later in this chapter, a simulation framework can also help in identifying the best recovery model across multiple candidate models to serve as representation of all recovery actions.

3.1.1 Airline Schedule Recovery Models

As stated earlier, the need for a recovery model arises because even the best plan with considerable buffers to absorb the predictable disturbances can sometimes get disrupted. Therefore, some remediation measures, known as schedule recovery actions in the airline industry, must be taken to mitigate the costs that arise from such disruptions. Several models for schedule recovery have been developed over the years. Some early models were developed by academic researchers (Argüello et al., 1997; Jarrah et al., 1993; Mathaisel, 1996; Teodorović and Stojković, 1990). With the maturation of the field of recovery optimization, a few of them made it into the decision support systems of the airlines.

Currently, there is little research that can comprehensively compare different recovery models to identify the ones that perform better for a given category of problem instance. Each model was typically developed while considering one or a few particular objectives. Some common objectives are: minimizing total expected delays, minimizing expected recovery costs, minimizing the time until complete recovery of schedules, minimizing disruption impacts on the most valuable customers, minimizing the number of customer complaints, optimizing the repositioning of the resources for future use, minimizing the adverse effects on the worst-affected parts of the system, etc. In rare cases the recovery models are integrated, they do not attend to the whole airline’s operations simultaneously. Consequently, each recovery model generates its own solutions that diverge from those of the other models, and most only consider a sub-problem that the airlines face.

In summary, choosing the right model to generate recovery solutions is a difficult task leaving a lot to guess-work. Airlines are typically forced to rely on their experience and existing decision support systems. However, the airlines are not able to easily compare the models they use with other possible models. Here we identify the need for a way to benchmark several recovery models against each other. This is where the use of a stochastic simulation framework can be advantageous. Indeed, a framework able to capture uncertainty and replicate in detail the daily operations of an airline would give the possibility of comparing recovery models, evaluating the robustness of the recovery solutions and assessing the tradeoffs one recovery model makes in comparison to another one. Such a framework would also help the work of researchers and industry professionals since it would be easier to compare models, develop prototypes, extend, and enhance existing ones. Each new iteration of a recovery model could be tested directly in a timely fashion, easily verifying the consequences of the new features or modifications that were made.

3.1.2 Contributions and Outline

With these considerations in mind, we developed a stochastic simulation framework based on an agent-based model using a limited amount of input data, and implemented it on a computer platform. With this framework, it is possible to simulate the daily operations of an airline considering all its main agents: aircraft, crews, and passengers. Using the Application Programming Interface (API) of an off-the-shelf commercial optimization suite, optimization models for recovery purposes can be used within the computer platform implementing the proposed framework in a straightforward manner. Constructed with a user-friendly interface, and with a projection on a Geographic Information System (GIS), this platform helps the user to verify each detail during the operations and identify peculiarities of the network, airline schedules, and recovery solutions.

The remainder of the chapter is structured in the following way. In Section 3.2 we describe the background concepts. Section 3.3 provides a review of the existing simulation platforms related to airline operations. In Section 3.4 we detail our airline simulation's framework. This is followed by Section 3.5 which presents a case study where we analyze different schedule recovery models and show how our simulation framework can be used to test these models. Finally, in Sections 3.6 and 3.7 we present the conclusions and future work, respectively.

3.2 Background Concepts

In this section, we provide a brief review of the background concepts relevant to this research. We will not review simulation theory, such as system dynamics or queueing theory. Rather our focus is on understanding how the literature tackles the problem of simulating airline operations, capturing the reality well enough to provide a good representation of the real world while balancing level of details against computational efficiency.

Developing and applying a highly detailed simulation framework would typically lead to a computationally complex and expensive endeavor. On the other hand, excessive simplifications would lead to a framework that is easy to apply but its outputs would be useless due to their large divergence from reality. Thus, any conclusions made on these outputs would be of limited practical value. In addition, to properly simulate an airline, its behavioral response to uncertainty and disruptions should be simulated. We need to model the actions that an airline will take throughout the day that will in turn influence the simulation of all other operations performed on that same day.

Airline operations can be divided into two phases – the planning phase and the execution phase. The execution phase includes reactive actions that are necessitated due the uncertainty that impacts the original schedules decided in the planning phase. Plans are an important input for our simulation framework. These are schedules for all resources that an airline utilizes, e.g., aircraft and crews, as well as the schedules of passengers. However, uncertainty can disrupt these schedules and render them unfeasible or sub-optimal after a disruption event. The airline needs to act to resolve infeasibilities or at least attenuate their downstream effects. These actions are called as airline schedule recovery. Both planning and recovery are well-studied topics and are addressed frequently with the use of optimization models. The models that are used for schedule planning are not the focus of this paper. For more information on this subject we refer the reader to Ball et al. (2006) and Barnhart et al. (2014).

A schedule recovery model, however rudimentary, is essential within any stochastic simulation framework able to adequately represent airline operations. Nonetheless, recovery models that are too rudimentary, such as a rule-based recovery model, are unable to resolve large-scale disruptions optimally or even near-optimally; in fact, they are often unable to even maintain feasibility of operations under large-scale disruptions. More sophisticated recovery models are not only able to resolve operational infeasibilities but are also a better depiction of real-world airline operations.

The scheduling recovery problems raised by operational uncertainty have been addressed in the literature in two different ways. One is through schedule recovery models where the airline tries to recover the normal schedule after a delay or disturbance appears. The other has led to proactive robust schedule planning models that intelligently anticipate potential operational problems and generate schedules that are more resistant to delays and disruptions. A highly robust schedule, designed for the worst-case scenario, has a low probability of being disrupted, but it can be too conservative, hence too costly. On the other hand, a highly optimized schedule with no slacks or buffers will be prone to disruptions, which in turn translate to higher costs for the airlines. Thus, adding the right degree of robustness is essential to minimize true costs to the airlines. An example of such a robust scheduling model was presented in the previous chapter, Chapter 2.

Besides the combination of robustness and recovery, recovery itself should be attended in a broad perspective. Schedule recovery should be made in an integrated manner. It is not good practice to recover a disrupted schedule only by trying to alter the resources that are disrupted. Schedule recovery tends to be an operation where the whole airline should be considered in a integrated manner.

Airline operations will always have to deal with scheduling recovery problems. Even the most robust, worst-case design will fail at some point and there will be a need of a highly effective and efficient schedule recovery. Indeed, in practice, an airline will always be obliged to act during the operations phase to circumvent unforeseen events. Thus, in the simulation of airline operations, airlines must be active agents because they will always act to solve issues during operations that will turn previous plans infeasible.

3.2.1 Airline decision making

To mimic the daily operations of an airline, one must be able to understand how the decision processes take place in such companies, how each piece connects and/or constrains other pieces and what are the possible decisions that the airline can take. One possible approach is to use agent-based models. Jennings (2000) stated that this type of models can significantly enhance our ability to model, design and build complex, distributed systems and that it would become mainstream in the future. Bond, Gasser, Jennings and Wooldridge (Bond and Gasser (1988) and Jennings and Wooldridge (1998b) in Wooldridge, 2008) point out some characteristics of a system for which it is advantageous to apply agent-based models:

- The environment is open, or at least highly dynamic, uncertain, or complex;
- Agents are a natural metaphor;
- Data, control, or expertise are distributed; and
- Existence of legacy systems.

In the case of airlines, the environment of its operations is clearly highly dynamic. Each day of operations can be heavily affected by uncertainty, turning each instance of the airline schedule recovery problem unique. It is a complex environment since there are numerous agents, stakeholders, resources, and restrictions that must be considered. The agents are clearly a natural metaphor, crew members, passengers, and aircraft can be easily understood as agents.

Airlines have in their structure one main entity responsible for decisions on the operational level known as Airline Operational Control Center (AOCC). However, the architecture of the AOCC distributes the responsibilities for each resource between internal sections and as such, each section has different types of data available and the expertise is specialized in each section for a given resource. Hence the AOCC centralizes the decision making, expertise and data but due the internal architecture of it, the decisions, data and expertise are distributed.

The final point is existence of legacy systems. Airlines use legacy systems, for instance Decision Support System (DSS), to tackle their operational problems. To create an agent-based model that to represent the reality of airline operations one must attend to how the existing legacy systems interact with each other so one can properly represent the airlines decision making.

In our interpretation, legacy systems are, besides the DSS, the human decision makers and their expertise that intervenes in the decision-making process. To a similar conclusion came also Devore et al. (2006). The authors speak about expanding the system boundaries so that the human is part of the decision support system. For instance, there is no fully automatic DSS that generates solution for the daily operations of an airline that is robust enough to be able to tackle all possible problems that an airline faces at operational level. There will always be a human in the loop to supervise and correct solutions for extreme cases that the DSS cannot handle by itself. Resuming to simulate an entity like an airline the decision making, i.e. behavior, of it must be captured. The behavior is depending on various factors: models used, strategy of the airline, experience and skill of the staff, legacy systems, etc. This behavior can greatly vary from one airline to another and here an agent-based approach is most helpful. Power and Sharda (2007) defined the issue of agent based modeling in an eloquent way:

> *"People have been thinking in terms of agent-based modeling for many years but just didn't have the computing power to actually make it useful until recently. With agent-based modeling, you describe a system from the bottom up, from the point of view of its constituent units, as opposed to a top-down description, where you look at properties at the aggregate level without worrying about the system's constituent elements."*

This is the process of assessing and understanding of a system and in a second phase designing it. Wooldridge (2008) clams that this is rather tentative and abstract in the beginning, but continually gets more concrete, detailed and complex.

Bonabeau (2002) had a similar perspective. He claims that agent-based modeling is rather a mindset than a technology. It contrasts with classical modeling approaches, since it is a bottom up approach where we try to define its agents/pieces. He describes the benefits of an agent-based model in three points:

- Captures emergent behavior;
- Natural descriptions of the system; and

- It a flexible modeling approach.

An emergent phenomenon is hard to model except by using an agent-based model, especially when there are phenomena that are counter-intuitive and due to the complexity of airline operations is hard to understand all the emergent phenomenon, counter-intuitive or otherwise. One can easily describe the operations of an airline by its main components: aircraft, crews, passenger, and the airline’s operational decision making, hence the agent-based modeling approach is a natural description of the system. By airline operational decision making we mean all decisions that an airline takes during operations, notably the actions to respond to unforeseen events as delays.

The final point of flexibility is was makes our proposed simulation framework user-friendly. Since it allows with easy to simulate any airline and extend the model with relative ease by, for instance, adding additional agents. In his paper, Bonabeau also identifies the shortcomings of agent-based models. The most important one relates to the purpose. There is no general simulation that works for all possible circumstances. This perspective is not new though – just recall the famous quote from Box and Draper (1987):

> *"Essentially, all models are wrong, but some are useful".*

A simulation framework should be designed with a well-defined purpose, as is our case. Furthermore, it should have well-defined users. The potential users of our framework can be divided into two groups. The first group consist of researchers or industry practitioners that want to test scheduling or recovery models for an airline. The second group are researchers that want to simulate an airline to see how it reacts to different circumstances or scenarios. This second group is the one that needs a fully fleshed out AOCC. They need a simulation that captures the behavior of an airline, the decision making of the airline needs to be adequately represented. During operations the decisions are taken inside the sub-departments that constitute the AOCC.

Our scope is not a system wide model but one that is easy to use, small to moderate data requirements, and that focus on one airline. Most simulation models available are system wide models that render them not adequate, or hard to use at least, for our purpose. A set of existing simulation models can be found in Section 3.3.3 where we analyze them in detail.

One other issue pointed out by Bonabeau (2002) is the fact that humans can have irrational behavior. This issue is more prominent in the case for social models. Our model is an operational model and as such modeling each agent as rational is not a major sidestep compared to the reality. Lastly is that an agent-based model is in general more computationally demanding than a simple top down rule-based model. This has also been taken into consideration in our case, when we balanced the computational effort versus the level of detail captured by the model.

3.3 Existing Simulation Platforms

One of the earliest attempts to build a simulation platform for the aviation industry is presented in Nordin (1980). This simulation platform was only focused on the operation of an airport and not an airline. Hence this makes it unfit for our purpose besides the fact that it is an already dated model.

The ground work for today’s airline operational simulation platforms was laid down by Rosenberger et al. (2002). They created SIMAIR, a discrete event simulation platform which was aimed to explicitly incorporate uncertainty into the planning process. This platform can be used to analyze the sensitivity to disruptions and then evaluate recovery policies.

SIMAIR was later enhanced by Lee et al. (2003) The newest version of the SIMAIR platform, version 2.2, dates back to 2004 (National University of Singapore and Georgia Institute of Technology, 2013). The approach behind SIMAIR is systems dynamics rather than agent-based simulation. In Figure 3.1 we show a high-level representation of the platform.

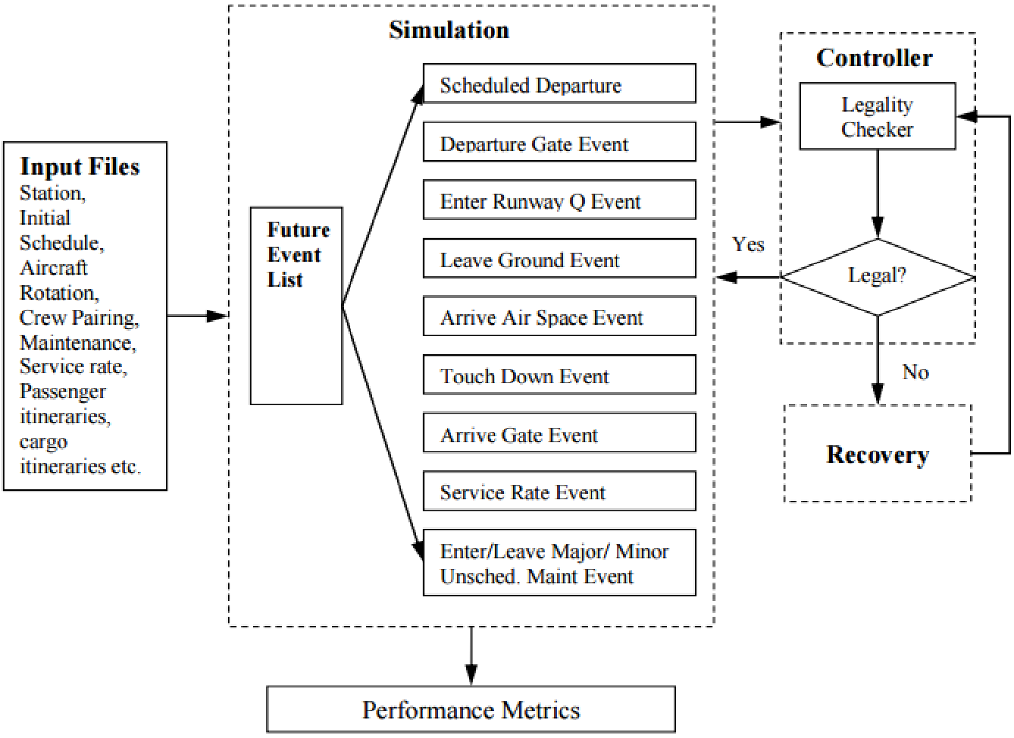


Figure 3.1. The SIMAIR platform (Lee et al., 2003)

SIMAIR consists of 3 modules: Simulation, Controller and Recovery. It can simulate Aircraft, Crew, Passengers and Cargo. SIMAIR is a complex platform with high level of detail, as we can see in Figure 3.2. This platform is difficult to feed with data due to the large amount of details it requires.

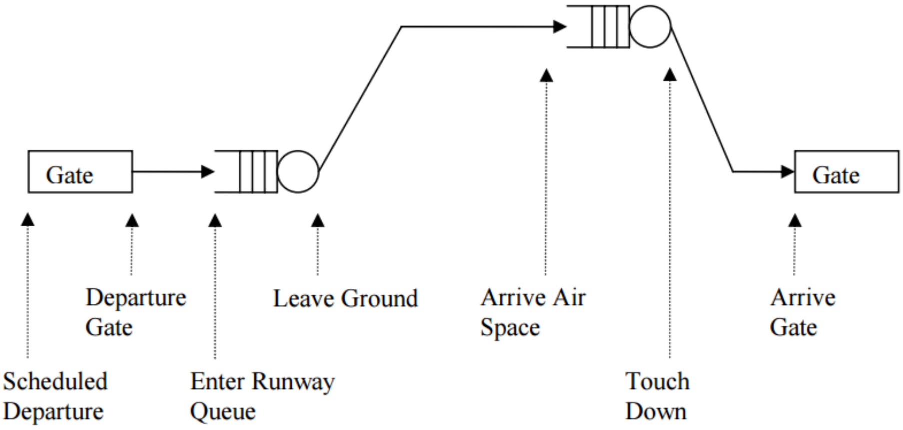


Figure 3.2. Representation of a flight leg inside SIMAIR (Lee et al., 2003)

The scheduling recovery model in SIMAIR is a “black box”. It not designed to obtain optimal solutions but rather directed to grant feasibility in a myopic way, and it does not consider swapping of aircrafts or crews. The initial implementation assumes a computer science background to be able to run it. If it is necessary to adapt it, an even deeper knowledge in computer science is needed.

Another noteworthy simulation platform is MIT Extensible Air Network Simulation (MEANS), described in Clarke (2004). It is also based on a discrete event simulation model that tracks aircraft and passengers over the network. The uncertainty is captured through re-runs varying some input data with a given distribution. This platform is focused on the large-scale simulation of a whole region. It could be used for simulating the daily operations of just one airline, however this is not the type of application it is originally intended for. If used for daily operations, the quality of the outcomes would suffer greatly. In particular, outputs would be characterized by an excessively high level of aggregation.

A similar conclusion was reached by the FAA when they compiled the “Model Use Shortfalls” of the existing simulation platforms in 2006 (FAA, 2006). They evaluated several existing systemwide platforms (NASPAC, Systemwide Modeler, ACES, FACET, LMINNET and AwSim/AERALIB) and identified several shortfalls in issues like simplifications of airport operations, en-route traffic and Airline Operational Control Center (AOCC). This last point is a major shortfall. Indeed, the AOCC is one of the most important features to be simulated as it “takes” most, if not all, decisions at an operational level. Usability was also evaluated in this context. The conclusion was that these platforms were hard to set up and lacked the ability to make Monte Carlo replications due to their size, i.e. computation time. Absent graphical visualization was another issue. Finally, they lacked a proper mechanism to validate the results. In order to overcome these shortfalls, FAA developed the System Wide Analysis Capability (SWAC) platform, later replaced by the National Airspace System Performance Analysis Capability (NASPAC) platform. This is a state-of-the-art platform that incorporates several additional features like fuel burn and delays (FAA, 2014). Nevertheless, once again, it is a system-wide simulation platform that is not compatible with our intended users and scope.

NASA's FACET is one of the platforms analyzed by the FAA as well, but it lacks airline details, and therefore does not fit our purpose. Moreover NASA is trying to license it (NASA, 2002). Systemwide Modeler (Baden et al., 2011) is a platform developed by MITRE. Yet, as the name indicates, it is also a system-wide platform designed to handle hundreds of thousands of flights and not designed to replicate the daily operations of an airline.

Barely none of the referred platforms are available as open source or free version. Making the assessment of the potential and drawbacks of each platform cumbersome. Under these circumstance, we decided to develop from scratch a user-friendly platform based on an agent-based simulation model.

3.4 Proposed Simulation Platform

We propose an integrated platform able to simulate the whole operation of an airline using an agent base modeling approach, the Airline operational Simulation Platform (AOSP). Since the simulation component inside the AOSP is stochastic it also contains a schedule recovery model able to fix schedules that are perturbated during the simulation. The main reason behind the decision of including a recovery model was to be able capture the decision-making process of the airline. The secondary reason was a practical one. Due the stochastic nature of the simulation this could lead to infeasibilities during the running of the AOSP that could lead to catastrophic computation errors.

Another concern while developing the AOSP was usability. We developed it always with the objective of being user-friendly. Hence significant effort was taken while developing the graphical user interface (GUI). This GUI helps to have a better understanding of all processes that are happening inside the AOSP.

AOSP was developed using the AnyLogic software (Anylogic, 2018). It allows to develop high quality models having the appropriated tools for it and is flexible enough to accommodate any kind of customizations as deemed necessary by the user. Every aspect of it can be tailored by the user through *Java* language. Also, AOSP is incorporates a Geographical Information System (GIS) allowing to simulate any airline network in any part of the globe. This GIS also enhances the GUI since the AOSP accurately projects the flights and airports worldwide. The framework is tailored to be used to simulate one day of operations but allows to simulate longer time window, i.e. weeks or months, only needing a minor effort to adapt the AOSP to this purpose.

AOSP's simulation model is an agent-based model containing three types of agents: aircraft, crews and passengers. Each agent receives as input a fixed schedule. As soon the simulation starts running, the agents are put in the starting positions and will try to operate the flights as the input schedules dictate. However, the AOSP is capable to add uncertainty to the simulation run by altering the flight arrival time and these delays will propagate through the network causing additional perturbations. These will propagate through flights that share resources and are time dependent. Here, the AOSP simulation model is designed in a user-friendly fashion allowing the user to tailor the delay characteristic according to his needs. For more information on the delay modeling please refer to sub-section 3.5.1. The following figure, Figure 3.3, depicts AOSP and the different modules it integrates.

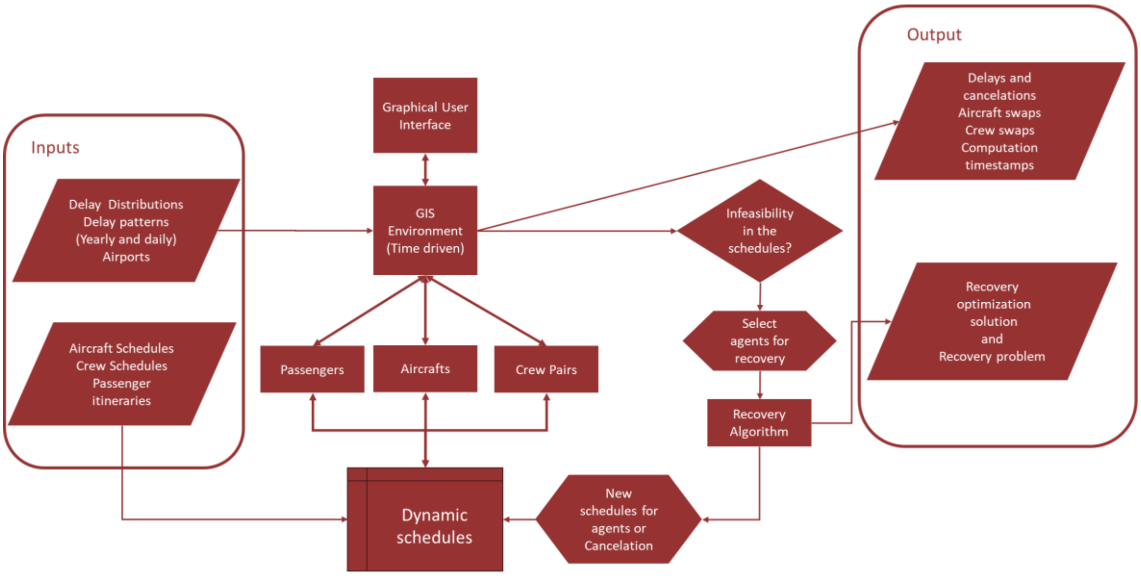


Figure 3.3. AOSP high-level representation

In the flowchart of Figure 3.3, we observe that there are two main input sources. One is referring to delay distributions and patterns that feed directly into the main environment of the AOSP. The second source are the schedules for crews and aircraft, and the passenger itineraries. These are feed into dynamic schedules for each agent. These schedules are dynamic since the recovery model can modify them. The AOSP is constructed so that it can use the format of the Airline On-Time Performance database form the Bureau of Transportation Statistics, BTS (2016), with only minimal data transformation and cleaning. The recovery model is called when a signal from the environment is sent informing that there is an infeasibility for one of the agent's schedules. New schedules will be defined with additional delays, resources swapping or flight cancelations to counter this infeasibility. These new schedules will be saved into the "dynamic schedules" and the simulation continues to run. The recovery algorithm will save all important data, namely recovery problem and solution, to the output files. This is one component of the output being many other data saved during the simulation run.

The current version of the platform, used in the case study, Section 3.5, has two recovery models included besides the default "delay until feasible" model. Before AOSP starts, the user must choose which recovery model will be active. For more on the recovery models in the current version of AOSP please refer to sub-section 3.5.1. The recovery module, depending on the chosen recovery model, will generate a solution that will change the "dynamic schedules" and, if necessary, generate also new agents (that represent the move up crews and or new aircrafts), delay flights or even cancel them. Once these updates are made, the simulation resumes from the same instant. To be noted, the whole simulation stops, and all computational power is diverted to the recovery optimization model and only after the solution is generated and dynamic schedules updated the simulation resumes. This means that, from the simulation point of view, the recovery optimization model is solved instantaneously. It is as if the AOCC could analyze the problem, solve it, and send the new schedules and cancelations information immediately.

All those operations are timestamped and send to the final output database, and computation times of each operation, optimization, output writing, can be analyzed afterwards. The framework encompasses an uncertainty module that generates delays. These delays are stochastic in nature. However, they are generated with a unique seed for each run. Hence, each uncertainty scenario can be re-run if needed since the seed is also stored in the output database.

3.4.1 Input and Output

To keep the model as simple as possible we tried to limit the input needs of the AOSP to a minimum. Of course, complex models typically need a large amount of data, otherwise their output is not significant enough. Yet, we tried to strike a balance that limits the need for input data and maintains an easy to use concept without reducing the quality of the results. The input data is divided in three types of parameters: cost parameters specific of the airline being simulated; schedules for aircraft, crew and passengers; and delay distribution parameters. The cost parameters mimic airline's cost structure. There are used to simulate the company's costs. These parameters are, for instance. cost of 1-min of delay and crew block hour cost.

The schedules of all agents need to be input in the simulation platform. Since we want this step to be as easy as possible, we had to decide on a data structure that would also be according to this principle. We chose to use a data structure like the one used by the biggest open airline delay database – the Airline On-Time Performance of the Bureau of Transportation Statistics, BTS (2016). The data of this database is copied directly and used as input file for the AOSP, only needing minimal modifications and complementary information to be added (specifically, airport coordinates and time zones).

The delay parameters are structured in the same way as in Tu et al. (2008). AOSP simulates the delays using three components: yearly component, daily component and random component. To be noted, it is possible to use the platform without any of these delay parameters turning AOSP into a deterministic simulation platform.

Each application of the platform will create an output database saving the most important information, like flight times, real-time timestamps for each activity, and performance metrics for each agent. This enables the user to analyze the simulation run in further depth than he could just from looking at the graphical user interface. These output databases can be analyzed to compare several runs with each other.

3.4.2 Agent Based Logic

Agent based simulation is a modeling logic. Using a few rules, it allows to simulate complex systems, like an airline. Probably its most interesting feature is the emergence phenomenon. It consists in behavior that was not explicitly modeled but emerges as the various agents interact with each other.

Agent-based modeling is the core modeling principle behind the proposed simulation platform. These agents' interactions allow to capture the reality of the daily operations of an airline since all need to be available for a flight to depart, but each agent has its own schedule and can suffer disruptions. Additionally, we are also capturing the agency of the airline, as the recovery model makes the main decisions of the airline. Hence, we can understand the recovery model as the airline's AOCC. The agents do interact with each other but on a lower level. When they encounter a problem, an infeasibility to operate their schedule, the problem is passed to a higher level to the recovery algorithm, i.e. simulated AOCC.

Interestingly there are lots of platforms currently available to develop agent-based simulation models, however this is not the scope of this work. For more information on this issue we refer the reader, for instance, to the work of Abar et al. (2017). In this research, the authors assembled a rich list of modeling tools and describe their characteristics, strengths, scalabilities, use scopes and development efforts.

The simulation and modeling of a real and complex system is inherently a simplification. Having that in mind, we need to trim our model and chose our simplifications wisely, so it fulfills our objectives. Our objective was designing a model that has the ability without needing major customization to simulate the daily operations of any airline. Also, we want to capture uncertainty in a simple manner and in a form that the data is available for a large set of flights and airlines, so it is easily implemented by researchers or industry practitioners. Data quantity, and quality, is a major issue and maybe one of the main reasons why simulation models are not disseminated more since most of the existing simulation models are custom models, needing custom made input data. Additionally, it is common for simulation models to include a wide array of parameters that are not easy to obtain directly, or derive, in most cases.

As mentioned before, the proposed platform involves the simulation of three agent types, each with its own representation as agent. Also, our agents are rather shallow since most of these decisions are of the *if-then* kind. The most complex decisions are made by the simulated AOCC, proxied by the recovery algorithm that is active in a given simulation run.

3.4.2.1 Aircraft

The aircrafts are modeled as individual agents, the most complex one. With the arrival and departure of an aircraft most of the interactions with the other agents, i.e. crews and passengers, are triggered. In Figure 3.4 we see the state chart that represents and controls the behavior of the aircraft:

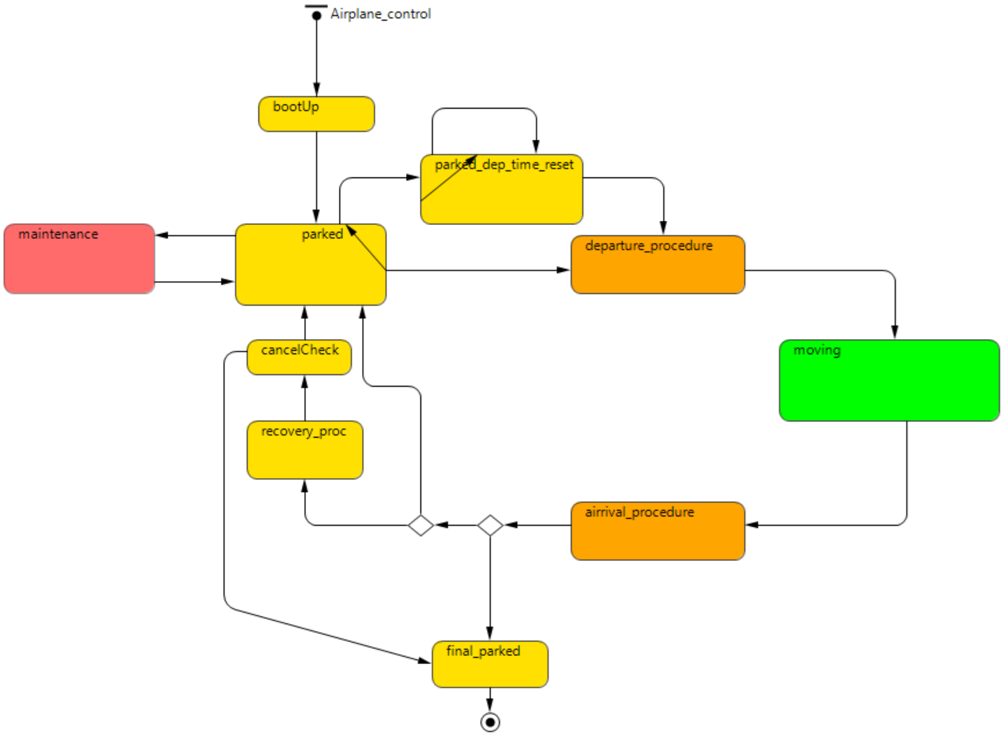


Figure 3.4. AOSM Aircraft agent state chart representation

We can see that the central point of the chart is the *parked* state, connecting all other states in a direct or indirect form. This state is where the aircraft agent rests. Once the departure time of the next scheduled flight is reached, the aircraft changes to the next state, *departure_procedure*. Here the aircraft communicates with the crew and passengers scheduled to be on this flight, informing that it is ready to depart. If those agents are also ready to depart the simulation will clear it for start and move this agent to the next state, *moving*. As the name suggest in this state the aircraft starts to fly. Additionally, the simulation will stochastically generate the arrival delay for this flight. This does not correspond to the reality since we do not know beforehand at what time an aircraft will reach the gate at the destination airport. Immediately after the delays are defined, the agents start moving from the departure airport to the arrival airport. This can be accompanied in the GIS visualization as depicted in Figure 3.5.

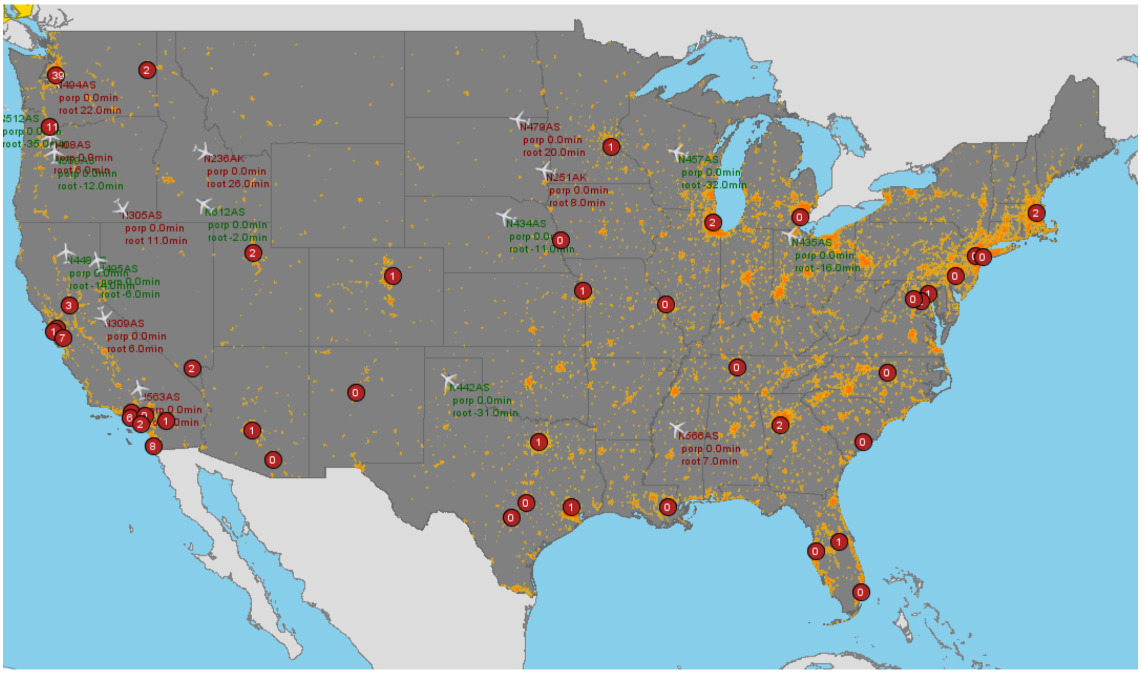


Figure 3.5. GIS visualization inside AOSM snapshot of Alaska Airlines operation

When an aircraft finishes the flight, it will enter the *arrival_procedure* state. To be noted, the model represents the gate to gate time as flying. This means in the visualization the aircraft will be shown flying from gate-to-gate, including taxi-in and taxi-out times. This period is known as block time in the airline industry. Once entered the *arrival_procedure* state, the agents are liberated from this flight. The next step in this loop is an if-statement that checks if this was the last flight scheduled for this aircraft or not. If it was the last scheduled flight, the aircraft is sent to the *final_parked* state that retires it from the simulation. If it was not the final flight, the next if-statement in the flowchart is reached by this agent. Here the simulation checks if there is enough time for the turnaround and if this will not compromise the next flight or if there are other disruptions that may impede the normal flight departure, creating propagated delay for this flight late arrival. If there is no problem, the aircraft moves to the *parked* state where it waits for the next flight in the schedule. In the case of not enough turnaround time the *recovery_proc* state is activated. Here it “moves” the simulation to a higher level and the recovery algorithm will start. As mentioned before, while this happens, the simulation stops and will only resume after the recovery model has given a solution and this solution is implemented. At this point we also depart from reality since it means that the AOCC would instantly have come up with a solution and, in the same instant, implemented it.

After a solution has been found, the aircraft moves to the next state: *cancelCheck*. As the name suggests we check if the flight that the aircraft was supposed to fly has been canceled. If it has been canceled the aircraft is retired from the simulation and send to the final parked position. This means that if a flight is canceled all following flights are canceled too. This is how it is implemented and needs to be considered when implementing a custom recovery model with the ability to cancel flights.

There are two more states that are present though they play a minor role in the agent’s “life”. The *bootUp* state is where the agents are brought to life and are given all their characteristics like: type of aircraft, tail number, seats, scheduled flight list, starting position, etc. The *parked_dep_time* state was created to solve a computational issue where the simulation would throw an error while changing schedules of a parked aircraft. In other words, it is a shadow state that was necessary to create due to computation and synchronization issues.

To be noted, there is a *maintenance* state but in the current version (the version used in the case study, see section 3.6) it is a mere placeholder. It is supposed to be called each time the aircraft must be subjected to maintenance, however since the simulation was only intended to be run on 24-hour time windows, this was a low priority state and was not fully fleshed out. In case it is mandatory to consider maintenance in the scenario to be tested, this must be programmed in the *maintenance* state.

3.4.2.2 Crew

The next agent corresponds to the crew, more exactly, to crew pairings since crews are modeled as a team, not as individual persons. Hence, we do not distinguish between cockpit and cabin crew in our simulation. In practice we assume that crews stay together until the end of their parings, the set of duties and rest times during a day. Similarly, as with the aircraft, here we also have a state chart that controls the behavior of the crew pairings, as illustrated in Figure 3.6.

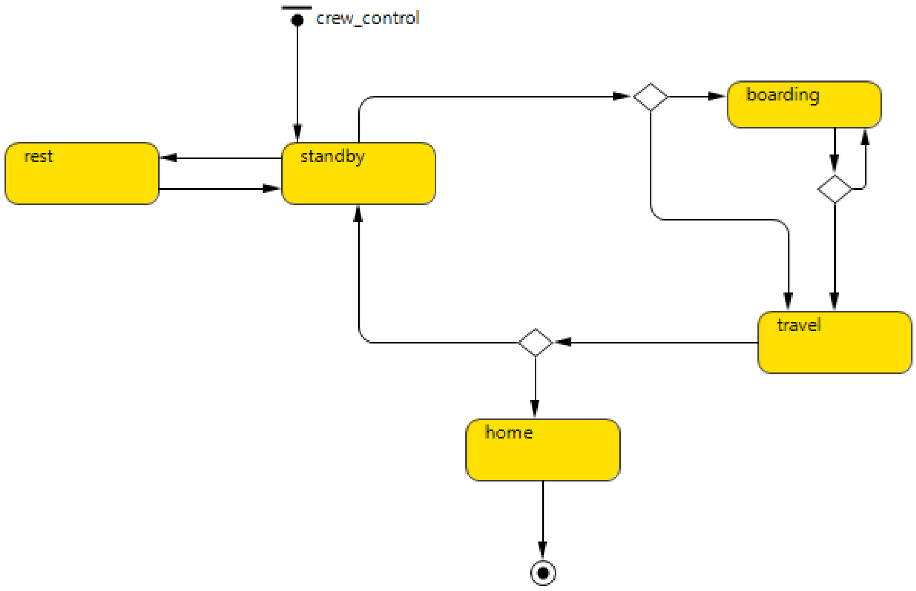


Figure 3.6. AOSM Crew agent state chart representation

The state chart for the crew is simpler than the one for aircraft agents. They start at the crew base being set into the *standby* state. Once a crew receives the "go" signal from the aircraft (send by the aircraft agent scheduled on the same flight as the crew pairing in the *departure_procedure*), it will launch the change of state of this agent moving it forward. If this agent and the passengers are ready, it moves directly to the *travel* state. If passengers are delayed, then the crew will wait for the passengers to be ready. This happens in the *boarding* state. Once all passengers are ready the crew agent receives the signal and moves into the *travel* state. Only here the crew sends a message back to the aircraft and the aircraft starts its *moving* state. This connection between the three agents is what creates the emergence behavior. Each agent will wait for each other to start the flight, if any of the agents (aircrafts, crew pairing or passengers) is missing the flight will be delayed. To be noted, the flight will wait for all the passengers. No passengers will stay on the ground and miss the flight in the current implementation. After landing, the simulation checks if the crew has any more flights to fly. If there are not, the agent is retired. In case it has more flights to fly it goes again to the *standby* state and waits for the next departure time. Also, there is a *rest* state that indicates when the crew is at rest. Similarly, as it happened with the *maintenance* state with the aircraft agents, this is not fully fleshed out in the current version of the model (this will change in future versions).

3.4.2.3 Passengers

Finally, we have the representation of the passengers, where each agent represents a group of passengers with the same origin, destination, and route to fly. These agents have the simplest state chart of all agents included in the agent-based model as we can see in Figure 3.7.

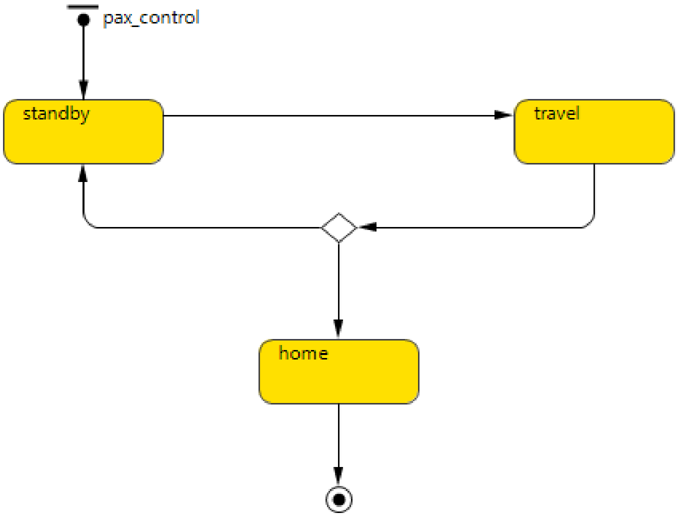


Figure 3.7. AOSM Passenger agent state chart representation

The passenger agent has only three states. The first state, *standby*, is when passengers are at the airport waiting for their flight. It will transition into the *travel* state by a message sent by the crew agent scheduled on the same flight as this passenger agent. This message is sent as soon as the crew agent moves out of its *standby* state. The *travel* state is activated when the flight starts, and changes to the *home* state once the flight arrives at its final destination. If there is a multi-leg flight, the agent just returns to the *standby* state waiting for its next flight to depart.

3.5 Case Study

In this case study we make an evaluation of the usefulness of the AOSP and evaluate of the performance of two different schedule recovery models where a default recovery procedure will be the baseline case. First, we describe these recovery models, the input data, the selected airlines, and scenarios that we considered. Also, we describe in detail the modeling of the delays in our simulation. With this case study, we want to further demonstrate the advantages of simulation as a tool for researchers and industry practitioners. In our case study two recovery models, are the proxies for the airlines AOCC, i.e. they represent the airlines decision making. Due their different objective functions each recovery model will take different decisions when faced with the same schedule recovery problem. Hence, the solutions will vary depending on the recovery model, for each problem and an emergent phenomenon will appear at the end of the day of operations. Finally, we demonstrate that the tradeoffs in delay and cancelations, due their different objective functions, would be hard to understand without a simulation tool.

3.5.1 Scheduling Recovery Models

Out of the literature, we decided to include two recovery models into the simulation framework: DSTAR and PDM. These models were respectively taken from the following articles:

Abdelghany, K. F., Abdelghany, A. F., & Ekollu, G. (2008). An integrated decision support tool for airlines schedule recovery during irregular operations. European Journal of Operations Research, 185, 825–848.

Bratu, S., & Barnhart, C. (2006). Flight operations recovery: New approaches considering passenger recovery. Journal of Scheduling, 9 (3), 279-298.

The main reason that led us to consider these two models is their dissimilar philosophy: DSTAR is a partially integrated recovery model whereas PDM is a passenger focused recovery model, each with its own advantages and disadvantages. We are aware that there is a large set of recovery models in the literature and that a few of those might outperform them. However, choosing the best optimization model was not the objective of this case study. We did want to test different types of recovery models in depth, and these two allow us to show that they lead to different outcomes i.e., to different behaviors of an airline. The aim was to test two models with diverse points of view in terms of the objective, and these two models were adequate for this purpose.

As referred before, the main criterion was the different underlying objectives, nonetheless we had a second criterion in consideration: the implementation effort required to incorporate them into the simulation. These two models have similar input needs and, in addition, are designed to be called at similar times during the operations, hence it is possible to integrate them in a modular way into the simulation framework. This modularity assures also that the comparison does not suffer from the fact that the simulation framework has be customized for each recovery model. The result is a modular structure so that each recovery action is an independent module of the framework and there is no custom-made input or output streams for the interaction outside this module. Finally, although needing a larger programming effort, modularity allows to consider user-defined recovery actions. The module is made in a way that the input and output connections of the actual recovery model are inside the module. Each recovery model must be defined with its own input and output functions. To construct a recovery model from scratch the user must define what input it needs and the output structure to update schedules.

For an in depth understanding of these recovery models, we refer the reader to the appendix. There, a detailed description of the DSTAR and DPM recovery models is provided (respectively in II.a.i and II.a.ii).

3.5.2 Airlines

We selected two airlines with different networks and aircraft manufacturers. In this case study we decided to use moderate sized airlines so that we could grasp completely the emergence phenomena if they occurred. We used airlines within the US since there are data readily available for the model inputs, namely from the Airline On-Time Performance database of the Bureau of Transportation Statistics (BTS, 2016). To compute the cost parameters we used the MIT Airline Industry Program (Massachusetts Institute of Technology, 2014) that includes a large set of parameters for several US airlines.

The smaller of the two chosen airlines, in size of the fleet and number of daily flights, is Virgin America (VX). This is a moderated sized airline operating in the continental United States of America. We derived a daily network (i.e., a network that repeats every day) from the database. In Figure 3.8, we see a representation of the daily network of VX that we used in this case study.

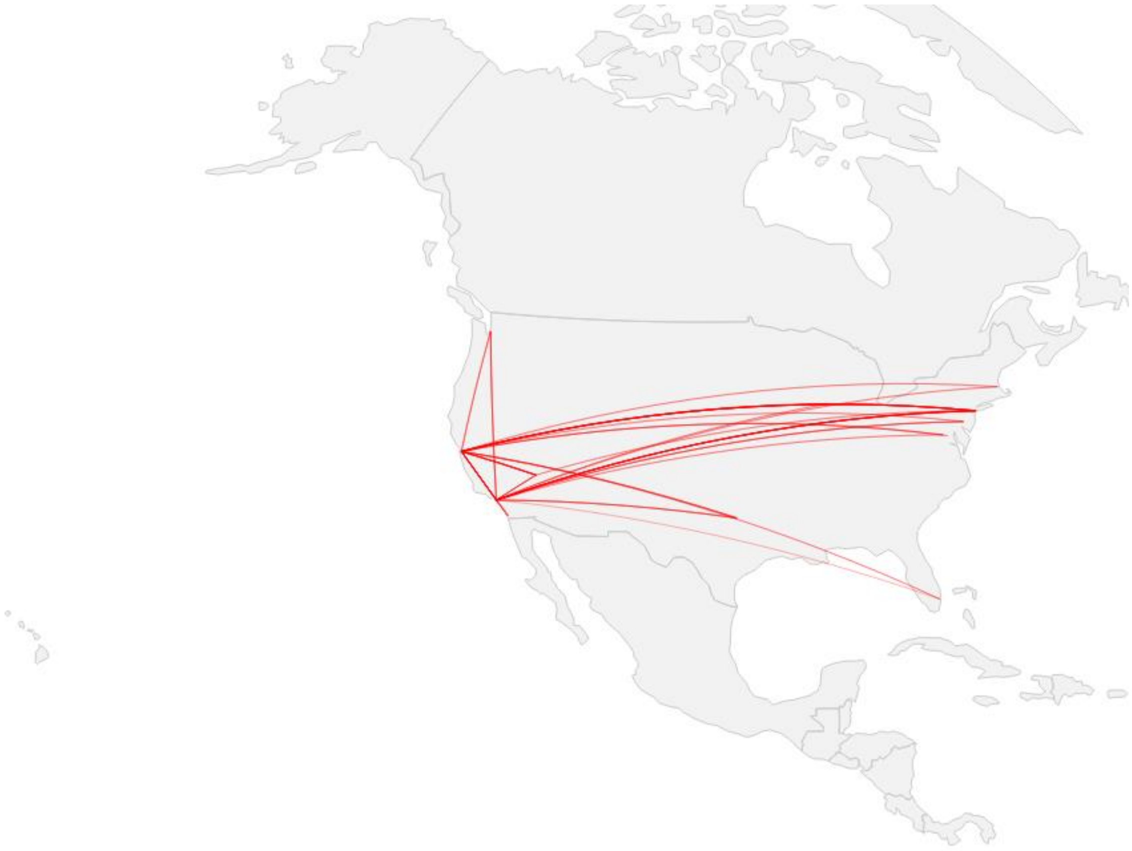


Figure 3.8. Virgin America daily flight network

Virgin America has a network that stretches across the continental United States of America, limiting possible regional biases in our results. Since it uses a single fleet family, of Airbus 320 aircraft, we do not need to subdivide the network to apply our models. Different aircraft types could be an issue for the recovery actions, since it would add another dimension to the simulation if there were incompatibilities between some aircraft and the crews that operate them. The daily network consists of 94 daily flights that connect 14 continental United States of America airports.

The second airline is Alaska Airlines (AS). Similarly to Virgin America, the network is also derived from the Airline On-Time Performance database. In Figure 3.9, we can see the representation of this network.

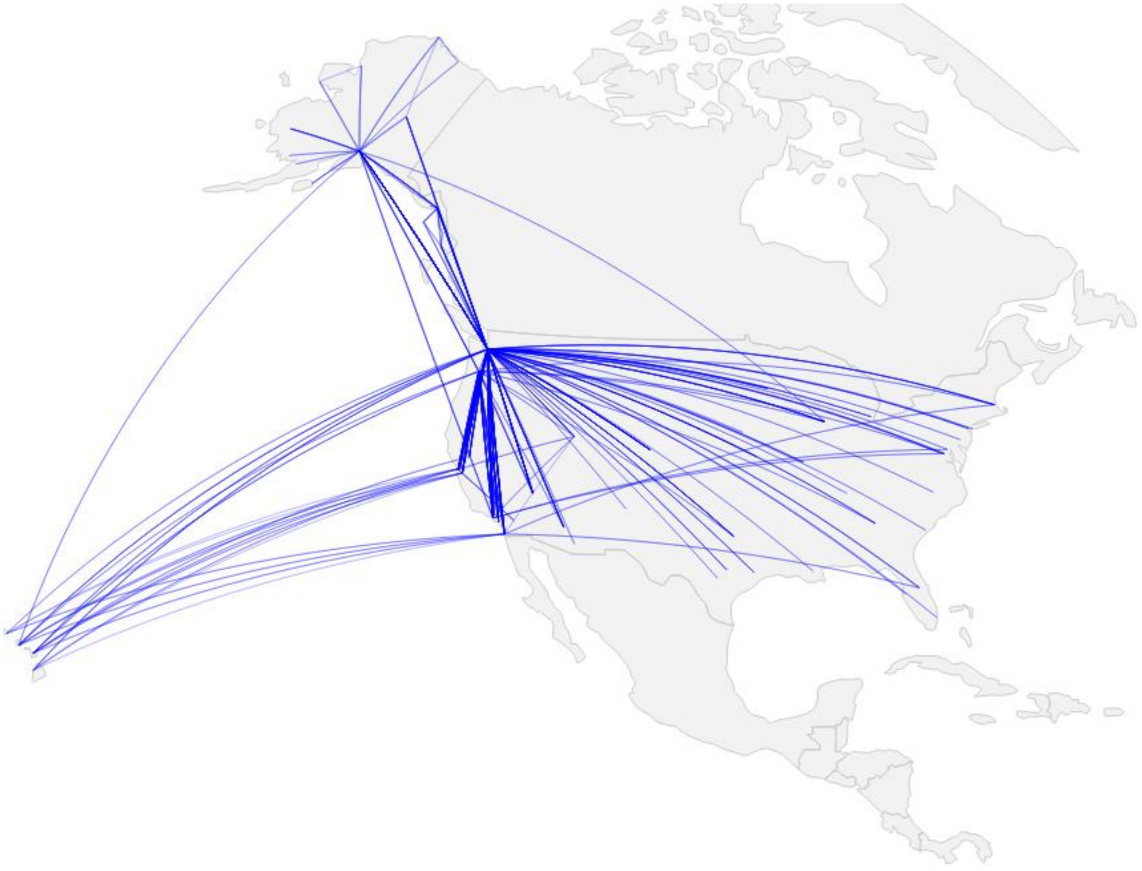


Figure 3.9. Virgin America daily flight network

It is a network of 67 airports that stretches out over the continental United States of America, Alaska and Hawaii archipelago. It offers 544 daily flights and the fleet consist in

165 Boeing 737 aircraft, 25 of them of the Classic Line type (CL) and the other 140 are of the Next Generation type (NG). The CL and NG aircraft have different type ratings and pilots and co-pilots need two type ratings to fly all aircraft. For our case study, we assumed that all pilots and co-pilots are trained in both. This allowed us to avoid subdividing the problem due to incompatibilities between aircraft and crews.

3.5.3 Cost parameters for DSTAR and DPM

Our next step was to define the cost parameters for the recovery models (DSTAR and DPM). These cost parameters are based, for AS and VX, on the values shown in Table 3.1.

Table 3.1. Base values needed to obtain cost parameters for AS and VX airlines

	AS	VX
Average value of passenger's time ($/hour) Source :http://airlines.org/ (Airlines of America, 2015)	48.5	48.5
Average Load Factor (%) Source: airlinedataproject.mit.edu (Massachusetts Institute of Technology, 2014)	84.9	83.5
Cost of block hour ($) Source: airlinedataproject.mit.edu (Massachusetts Institute of Technology, 2014)	2958	3049
Average Airline Arrival delay-delayed flights (min) Source: *https://transtats.bts.gov* (BTS, 2016)	47.0	45.7
Total cost of block time + passenger time value ($) Source: airlinedataproject.mit.edu (Massachusetts Institute of Technology, 2014)	9711.0	8880.6

For each of the recovery model, three parameters need to be computed:

DSTAR

- cd_f - cost of a one-minute delay of flight f;
- c_{rf} - cost of assigning resource r to flight f;
- cc_f - cost of canceling flight f.

DPM

- s_p - cost per disrupted passenger;
- o_{tf} - cost of delaying flight f, t time-steps;
- e_{cm} - cost of calling a reserve crew to operate the m-th flight leg of the schedule of crew c.

3.5.3.1 $\boldsymbol{cd_f}$ - Cost of 1-minute delay of flight f (\$/min)

The Cost of 1-minute delay is computed similarly as in the original paper. After reviewing the data for the seat capacity of the aircraft a significant variance was noted, hence seat capacity was used to compute this parameter for each flight. Sources used were the FAA Registry (FAA, 2017) and the website of the airlines (Alaska Airline, 2017; Virgin America, 2016):

- VX: 119 to 149 seats;
- AS: 124 to 181 seats.

To calculate the passenger value of time per flight we multiplied the average load factor, for each airline a, by the capacity of the aircraft times and by the average value of passengers' time. Then, we divided by sixty to obtain a per minute value as we can see in equation (3.1). This value was used as input to calculate cd_f as shown below in equation (3.2) for both airlines as a function of the number of seats offered in each flight.

$$ValPaxTime_a = AvrgLoadFactor_a \times seats_{af} \times AvrgValPaxTime/60 \qquad (3.1)$$

$$cd_f = \frac{BlockHourCost_a}{60} + ValPaxTime_a = \begin{cases} AS = 49.30 + 0.686 \cdot seats_f \\ VX = 50.81 + 0.675 \cdot seats_f \end{cases} \qquad (3.2)$$

3.5.3.2 c_{rf} - Cost of assigning resource r to flight f ($)

For the c_{rf} parameter, as in the original paper, we only allow swaps if the delay to be avoided is 20 minutes or more. In other words, the parameter is equal to 20 minutes of delay of flight f:

$$c_{rf} = 20 \times cd_f = \begin{cases} AS = 20 \times (49.30 + 0.686 \cdot seats_f) \\ VX = 20 \times (50.81 + 0.675 \cdot seats_f) \end{cases} \tag{3.3}$$

3.5.3.3 cc_f - Cost of canceling flight f ($)

The value for canceling a flight was retrieved directly for the original paper where the DSTAR model is described, i.e., $400,000.

$$cc_f = \begin{cases} AS = 400{,}000 \\ VX = 400{,}000 \end{cases} \tag{3.4}$$

3.5.3.4 o_{tf} - Cost of delaying flight f, t time-steps ($)

Similar as with the delay cost of the DSTAR model, we computed this cost based on the block hour cost for the airlines, AS and VX. The only difference is that in our case study we have 15-minute time steps, and as such we need to multiply it with t time steps as shown in equation (3.5).

$$o_{tf} = cd_f \times 15 \times t = \begin{cases} AS = 15 \times t \times (49.30 + 0.686 \cdot seats) \\ VX = 15 \times t \times (50.81 + 0.675 \cdot seats) \end{cases} \tag{3.5}$$

3.5.3.5 s_p - Estimated cost per disrupted passenger ($/passenger)

To calculate this parameter, we used a similar method as described in the original paper. Specifically, we estimated the average passenger delay by multiplying the average arrival delay of the airline by the value of passenger’s time, as depicted in equation (3.6).

$$s_p = ValPaxTime \times \frac{AvrgArrDelay_a}{60min} = \begin{cases} AS = 38.35 \\ VX = 46.10 \end{cases} \tag{3.6}$$

3.5.3.6 e_{cm} - Cost of calling a reserve crew to operate the m-th flight leg of crew c's schedule ($)

Here we also followed the original paper. It is the cost of a block hour times the block hour flown by a given crew pairing, as in equation (3.7).

$$e_{cm} = \begin{cases} AS = 1033.95 \times blocktime \\ VX = \ 741.74 \times blocktime \end{cases} \tag{3.7}$$

3.5.4 Delay estimation and curve fitting

As referred before the AOSM is a stochastic model. The stochasticity is introduced assuming independent delays for each flight. This is a simplification for instance, delays for two given flights in a given region due to weather are not independent since a whole region will be affected. However, the delays of these two flights would be considered independent in our model. The AOSP does consider dependency of delays of the model since late arrival of a flight can propagate the delays of the next flight.

Next, we are going to explain the methodology used to estimate and fit the delays, sub-section 3.5.4.1 and the evaluate the goodness of fit in sub-section 3.5.4.2 of those curve fittings.

3.5.4.1 Delay fitting methodology

To model the delays we aggregated all uncertainty in a per flight basis as done in:

Tu, Y., Ball, M.O. & Jank, W.S., 2008. Estimating Flight Departure Delay Distributions A Statistical Approach With Long-Term Trend and Short-Term Pattern. Journal of the American Statistical Association, 103(481), 112–125. (Tu et al., 2008)

The idea is to define splines that capture structural delays, i.e. the non-stochastic component of the delays. These splines were calibrated for each airport served by the airline. The two components are:

- Yearly component;
- Daily component.

Delay is divided into two parts as we can see in equation (3.8):

$$D = d_p + \ d_r \tag{3.8}$$

Where:

D - Total delay;

d_p - Propagated delay;

d_r - Root delay.

The yearly pattern is fitted to the average daily departure offset. We are talking about offsets and not delays since negative values are possible. We used three years (2013-2015) to calculate the average offset. Root delay is itself modeled considering three components, as in equation (3.9).

$$d_r = p_y + \ p_d + \varepsilon \tag{3.9}$$

Where:

p_y - yearly pattern;

p_d- daily pattern;

ε - Error component (stochastic)

The daily pattern is fitted to the average departure offset discretized in intervals of fifteen minutes for the whole day. We also used three years, from 2013 to 2015, to calculate the average offset for each fifteen-minute interval. Since offsets distributions tend to have very long tails, we made a transformation of the data using the log-modulus transformation. A logarithmic transformation could not be used since there are offsets smaller or equal to zero. Equation (3.10) provides the formulation of our transformation.

$$f(x) = \begin{cases} ln(|x|), & x < 0 \\ 0, & x = 0 \\ ln(x), & x > 0 \end{cases} \tag{3.10}$$

The error component (ε) was modeled as a finite mixture model that consists in a combination of several normal distributions that are scaled and skewed so that the combination of them represents the error as accurately as possible, i.e. has the best fit.

3.5.4.2 Goodness of fit

For the goodness of fit tests for the error component (ε), we have used the *Box-Pierce* and *Ljung–Box* methods to test the following hypotheses:

- ***H0:*** The data are independently distributed they do not exhibit serial correlation
- ***H1:*** The data are not independently distributed; they exhibit serial correlation.

Virgin America

First, we have analyzed the case of Virgin America starting with the yearly pattern. Since we are doing several fittings at the same time, a Bonferroni correction was applied. The family for the correction was the number of airports. Hence, for our chosen α of 0.01 we obtained the value computed in equation (3.11):

$$\alpha_b = \frac{\alpha}{m} = \frac{0{,}01}{14} = 0.0007 \tag{3.11}$$

p_y– Yearly pattern

The in the following table, Table 3.2, we can see the results of the *Box-Pierce* test until a lag equal to 10.

Table 3.2. Result of the Box-Pierce test for the yearly pattern of VX airline

Lag	SFO	LAX	LAS	SAN	SEA	DFW	ORD	FLL	BOS	EWR	JFK	PHL	IAD	DCA
1	0.0788	0.2715	0.0920	0.0524	0.1072	0.2976	0.1262	0.0156	0.6159	0.0186	0.0093	0.4909	0.0076	0.0670
2	0.0492	0.0884	0.2208	0.1211	0.0143	0.3038	0.1833	0.0226	0.8993	0.0254	0.0328	0.0403	0.0236	0.0335
3	0.0980	0.0194	0.0443	0.0557	0.0043	0.4943	0.0337	0.0496	0.1799	0.0466	0.0757	0.0594	0.0494	0.0606
4	0.1432	0.0157	0.0353	0.1052	0.0089	0.2940	0.0216	0.0117	0.2273	0.0919	0.0964	0.0075	0.0336	0.0531
5	0.2054	0.0305	0.0039	0.1613	0.0186	0.3562	0.0350	0.0234	0.1950	0.0438	0.1467	0.0151	0.0142	0.0757
6	0.2681	0.0594	0.0124	0.2055	0.0269	0.2514	0.0392	0.0195	0.0390	0.0373	0.0762	0.0111	0.0269	0.0408
7	0.3452	0.0903	0.0226	0.2628	0.0463	0.3085	0.0093	0.0297	0.0644	0.0131	0.0592	0.0203	0.0457	0.0685
8	0.4330	0.0984	0.0375	0.3475	0.0356	0.2292	0.0117	0.0417	0.0699	0.0150	0.0932	0.0304	0.0617	0.1044
9	0.4738	0.1439	0.0595	0.4384	0.0515	0.2869	0.0015	0.0636	0.0953	0.0254	0.0828	0.0304	0.0617	0.1353
10	0.5680	0.1699	0.0815	0.5332	0.0779	0.3666	0.0029	0.0766	0.0831	0.0323	0.1105	0.0220	0.0873	0.1617

We can see that with the chosen α we have no problem with the yearly pattern, and we can assume that the errors after fitting the yearly pattern are independent. The Ljung-Box test gives us a similar image, view Table 3.3.

Table 3.3. Result of the Ljung-Box test for the yearly pattern (VX)

Lag	SFO	LAX	LAS	SAN	SEA	DFW	ORD	FLL	BOS	EWR	JFK	PHL	IAD	DCA
1	0.0775	0.2696	0.0906	0.0514	0.1058	0.2956	0.1246	0.0148	0.6144	0.0182	0.0091	0.4891	0.0074	0.0659
2	0.0478	0.0862	0.2180	0.1190	0.0137	0.3053	0.1775	0.0021	0.8572	0.0246	0.0318	0.0390	0.0229	0.0325
3	0.0954	0.0180	0.0422	0.0537	0.0040	0.4899	0.0321	0.0047	0.1749	0.0450	0.0738	0.0530	0.0484	0.0586
4	0.1364	0.0147	0.0334	0.1018	0.0083	0.2931	0.0019	0.0111	0.2208	0.0892	0.0936	0.0069	0.0320	0.0508
5	0.2000	0.0287	0.0090	0.1564	0.0174	0.3478	0.0045	0.0222	0.1876	0.0414	0.1426	0.0140	0.0132	0.0723
6	0.2612	0.0520	0.0112	0.1990	0.0252	0.2417	0.0084	0.0182	0.0356	0.0066	0.0722	0.0101	0.0251	0.0380
7	0.3369	0.0856	0.0206	0.2547	0.0437	0.2969	0.0083	0.0278	0.0592	0.0099	0.0551	0.0185	0.0428	0.0642
8	0.4256	0.0927	0.0344	0.3380	0.0328	0.2167	0.0104	0.0389	0.0639	0.0134	0.0873	0.0274	0.0648	0.0983
9	0.4628	0.1363	0.0549	0.4279	0.0476	0.2722	0.0013	0.0596	0.0873	0.0228	0.0765	0.0271	0.0571	0.1276
10	0.5564	0.1606	0.0782	0.5223	0.0724	0.3498	0.0024	0.0715	0.0750	0.0290	0.1023	0.0193	0.0811	0.1522

p_d– Daily pattern

Having set the yearly pattern, we can now focus on the daily pattern and see if the tests also pass the goodness of fit tests. Table 3.4 contains the results of the Box-Pierce test up to a lag of 10 and Table 3.5 for the Ljung-Box test, also up to a lag of 10:

Table 3.4. Results of the Box-Pierce test for the daily pattern (VX)

Lag	SFO	LAX	LAS	SAN	SEA	DFW	ORD	FLL	BOS	EWR	JFK	PHL	IAD	DCA
1	0.3635	0.0781	0.8311	0.4412	0.8832	0.6459	0.0726	0.7713	0.5922	0.3182	0.0363	0.0147	0.1936	0.1231
2	0.5673	0.0303	0.1018	0.5514	0.9864	0.6541	0.1572	0.0988	0.1266	0.5523	0.0237	0.0495	0.1234	0.3023
3	0.6485	0.0081	0.0984	0.5502	0.9959	0.9878	0.2897	0.1192	0.2938	0.6458	0.0290	0.1034	0.1464	0.2417
4	0.7999	0.0230	0.1747	0.7141	0.9977	0.8983	0.0370	0.0684	0.5444	0.7943	0.0585	0.1529	0.1750	0.2096
5	0.8821	0.0390	0.2732	0.8309	0.8423	0.2986	0.4406	0.0923	0.0762	0.8199	0.7369	0.2113	0.2566	0.3184
6	0.8907	0.0134	0.3651	0.9064	0.8440	0.5995	0.5653	0.0565	0.0744	0.8920	0.0689	0.1397	0.3644	0.4037
7	0.9145	0.0258	0.4753	0.8428	0.8609	0.8008	0.6106	0.0857	0.0383	0.9210	0.0683	0.0996	0.4257	0.4546
8	0.4766	0.0536	0.3371	0.9046	0.8380	0.4021	0.4917	0.1172	0.1624	0.7560	0.0678	0.0987	0.5071	0.5063
9	0.5781	0.0083	0.4062	0.9279	0.7684	0.3548	0.5393	0.1288	0.7247	0.7899	0.1031	0.1386	0.6082	0.4342
10	0.5397	0.0085	0.4413	0.9500	0.7872	0.5218	0.6792	0.1683	0.6551	0.8563	0.0954	0.1826	0.6945	0.4621

Table 3.5. Results of the Ljung-Box test for the daily pattern (VX)

Lag	SFO	LAX	LAS	SAN	SEA	DFW	ORD	FLL	BOS	EWR	JFK	PHL	IAD	DCA
1	0.3593	0.0753	0.8384	0.4288	0.8815	0.8815	0.0644	0.7642	0.4418	0.2681	0.0048	0.0095	0.1791	0.0982
2	0.5612	0.0022	0.1019	0.5303	0.9860	0.9860	0.1442	0.0805	0.2569	0.4769	0.0188	0.0334	0.1037	0.2526
3	0.6403	0.0060	0.0902	0.5174	0.9860	0.9860	0.2694	0.0939	0.6266	0.5393	0.0217	0.0719	0.1185	0.1683
4	0.79325	0.0930	0.1618	0.6835	0.9974	0.9975	0.3425	0.0467	0.4200	0.6989	0.4476	0.1041	0.1373	0.1202
5	0.87675	0.0470	0.2557	0.8065	0.8213	0.8213	0.4055	0.0617	0.6341	0.6790	0.0544	0.1420	0.2053	0.1971
6	0.8836	0.0107	0.3439	0.8891	0.8198	0.8198	0.5283	0.0312	0.5305	0.7720	0.0417	0.0636	0.3011	0.2537
7	0.9073	0.0220	0.4519	0.7980	0.8345	0.8345	0.5658	0.0516	0.7688	0.7858	0.0419	0.0295	0.3495	0.2714
8	0.5388	0.0044	0.3548	0.8712	0.8029	0.8070	0.4206	0.0670	0.5174	0.2334	0.0377	0.0229	0.4214	0.2883
9	0.4943	0.0060	0.4249	0.8963	0.7146	0.7146	0.4596	0.0696	0.5049	0.2146	0.0600	0.0344	0.5215	0.1613
10	0.4943	0.0067	0.3984	0.9231	0.7303	0.7303	0.5494	0.0929	0.4650	0.2865	0.0463	0.0473	0.6105	0.1496

As we can see in these tables, the daily pattern does not display any problem too, and we can assume that the yearly and daily pattern splines represent well our delays.

ε – Error component

The only step missing is to check the goodness of fit of the finite mixture model. We ran our model with a range of values for k (k represents the number of normal distributions used fitting the error component with a finite mixture model). The log-likelihoods obtained for the Virgin America models are in Table 3.6.

Table 3.6. Log-likelihood for the finite mixture models (VX)

	k			
	2	3	4	5
SFO	-207848,23	-207043,30	-206830,80	-206774,20
LAX	-173793,91	-173190,40	-173062,75	-173032,2
LAS	-50369,53	-50104,2	-50046,73	-50039,41
SAN	-23438,05	-23369,38	-23348,44	-23343,31
SEA	-33461,48	-33319,25	-33288,89	-33278,47
DFW	-14842,05	-14795,74	-14785,48	-14790,46
ORD	-20985,14	-20895,79	-20887,46	-20885,09
FLL	-16897,34	-16866,84	-16864,97	-16852,76
BOS	-22155,27	-22102,49	-22101,82	-22080,57
EWR	-25357,34	-25292,47	-25268,65	-25265,63
JFK	-51460,65	-51297,76	-51279,68	-51279,07
IAD	-21816,21	-21815,05	-21739,62	-21749,95
PHL	-8574,102	-8554,04	-8552,17	-8550,86
DCA	-10093,56	-10062,86	-10059,08	-10058,70

Observing the log likelihood figures for each k value we observe that biggest increase was with a $k = 3$ Increasing it further to $k = 4$, adding a forth normal distribution to fit the error component, would only lead to a small increase in loglikelihood. Being the increase in loglikelihood with k = 5, ie. five normal distributions, marginal compared to lower values of k. Based on these results, we decided that the best representation for our model was a finite mixture model of three distributions ($k = 3$). It offers the best balance of accuracy versus complexity. The gains of adding another distribution ($k = 4$) were not significant in our understanding.

Alaska Airlines

Similar as what we did for VX, for Alaska Airlines we also used a Bonferroni correction. However, since there are more airports in this case (67), the value is different as shown in equation (3.12).

$$\alpha_b = \frac{\alpha}{m} = \frac{0{,}01}{67} = 0{,}00015 \tag{3.12}$$

As before, we applied the *Box-Pierce* and *Ljung-Box* tests on the residuals of the yearly pattern and daily pattern. The error component was also modeled with a finite mixture model. and the results were resumed in similar tables as before, Table 3.4 and Table 3.5. Due the extensive size of the tables for Alaska Airlines, due to the number of airports, they are moved to the appendix II.b ,tables II.1 to II.5. In this appendix, there are also some further details on the results of the goodness of fit tests.

3.5.5 Scenarios

We have created several combinations of scenarios where we combined different airlines, schedule recovery models and types of underlying delays. The scenarios are combinations based on three key points:

- Airlines: Alaska Airlines vs Virgin America;

- Underlying delays: One day simulated 1000 times, denoted as *Simulation Delay Approach* 1 from now on vs. one whole year simulated 15 times, denoted as *Simulation Delay Approach* 2.

- Recovery models: Default, DSTAR and DPM.

To be noted, we did not consider the delays that propagate to the next day. We considered the cost of said delays, but delays of a previous day were not propagated to the next day, i.e. simulation run. This means that, at the beginning of the simulation, the aircraft were on their start positions at the time to fly their first flight. The default recovery model is a model that delays flights until all agents (aircraft, crew and passengers) are available to make the flight.

3.5.6 Results and Discussion

As stated before, we made several runs that correspond to millions of simulated flights. This has led to dozens of gigabytes of output data. Hence, we decided to focus our results on two main metrics, the propagated delay and the cost. We analyzed how the recovery models will solve the schedule recovery problems and the emerging behavior arising from the interactions of the recovery model with the other components of our simulation platform. The baseline case of our analysis was the default recovery model. We compared the two other recovery models, DSTAR and DPM with the base case and with each other. The default recovery model is a simple set of rules that delay all agents (aircraft, crew and passengers) until they are all ready to fly. Multiplying the airlines and simulation approaches by the number of recovery models (Default, DSTAR and DPM), we were facing 12 scenarios. In total, the number of simulated flights was approximately 12,4 million as shown in Table 3.7. Table 3.8, shows the delays divided into delay ranges.

Table 3.7. Total simulated flights

	Simulation Delay Approach 1			Simulation Delay Approach 2		
	Default	DSTAR	DPM	Default	DSTAR	DPM
VX	94 000	94 000	94000	514 650	514 650	514 650
AS	544 000	544 000	544 000	2 978 400	2 978 400	2 978 400

From Table 3.8, the first conclusion we can draw is that most flights are on time or suffer from just a few minutes of propagated delay. Around eighty-five percent of all flights have less than a one-minute delay. We have focused our attention on the heavier delayed flights.

These are the flights for which the most differences between recovery actions have been observed.

With Simulation Delay Approach 1, we see that DPM mainly reduced the delays in the range of 15-60 minutes, around 1.5 percent for Virgin America and 1.0 percent for Alaska Airlines. DSTAR also reduced the delays in the same range. However, it also reduced the delays in the 60-300 range. DPM cancels around three percent of the flights for both airlines. DSTAR does not cancel any flight. For the default, no cancelations were expected, as this model does not consider flight cancelations (all flights must be flown). In contrast, DSTAR is able to cancel flights , but this never happened because this would not lead to the best possible solution. The reason is probably due to the cost of cancelation of $400,000 per flight, we will explore this idea in detail further on.

Table 3.8. Percent of flights per Propagated delays range

		Sim. Delay Approach 1			Sim. Delay Approach 2		
	Prop. delay range (min)	Default (%)	DSTAR (%)	DPM (%)	Default (%)	DSTAR (%)	DPM (%)
VX	<1	84.94	85.34	86.25	87.43	87.97	88.59
	1-15	9.90	10.59	9.58	7.99	8.50	7.51
	15-60	3.10	1.71	0.21	2.68	1.41	0.17
	60-300	2.04	2.26	0.29	1.88	2.05	0.28
	>300	0.02	0.09	0.48	0.03	0.08	0.48
	Canceled	0.00	0.00	3.20	0.00	0.00	2.97
AS	<1	89.03	88.92	90.66	90.50	90.42	91.76
	1-15	5.59	6.72	5.44	4.80	5.72	4.57
	15-60	3.82	2.68	0.21	3.28	2.29	0.17
	60-300	1.54	1.66	0.07	1.39	1.56	0.06
	>300	0.03	0.01	0.00	0.03	0.02	0.00
	Canceled	0.00	0.00	3.63	0.00	0.00	3.44

DPM on the other hand does not have a cancelation cost but uses cancelations as last resort solution. The cancelation option in the DPM model can be understood as a feasibility sink; that is, if DPM could not cancel it would return an infeasible solution. Thus, it can be said that cancelations take place when there is no other solution. Cancelations might be sub-optimal since they are integrated in a myopic way. Meaning, DPM was not able to predict future delays. With the knowledge of future delays DPM might have been able to avoid some cancelations. However not knowing future delays fits the reality since during operations the airline is not aware of future delays in most cases. To be noted, both models, DPM and DSTAR, cannot predict future delays.

For the next set of analyses of delays, flight cancelations were not considered possible. If we were to consider them, it would be necessary to attribute an arbitrary delay value to those flights. The canceled flights would then be offset by that arbitrary value of our choice. This would destroy the comparability since we are going to analyze the distributions of delays and the canceled delays would all be concentrated on one value, the arbitrary value. Figure 3.10 shows the boxplots of the differences of median propagated delays between DSTAR and Default for each flight. The boxplots are ordered, color-coded, by median value. Bottom boxplot is the flight with smallest median and top boxplot is the fight with largest median value.

As mentioned before Figure 3.10 shows the differences between the delays obtained for the base case and through the DSTAR model, by flight and ordered by the median value. We can see that most of the box-plots are on the left-hand side of the figure, i.e. the negative side. This means that the DSTAR model in general performed better, most of the flights have smaller propagated delay in comparison with the default model.

Interesting is to see that there are a set of flights that constantly perform worse than the Default model, the top box plots in Figure 3.10. These flights have in general more propagated delay when the simulation was running using DSTAR model than the default model. This happens mostly to flights scheduled close to the end of the day. Since we simulated only one day, the recovery model could not use next day flights to help recover those end-of-the-day flights being obliged to delay them. Finally, we can also observe that the delay differences are zero or negligible for about half of the flights. Hence, we can conclude that the DSTAR model reduces delays in comparison with the default model but does not dominate it in every flight.

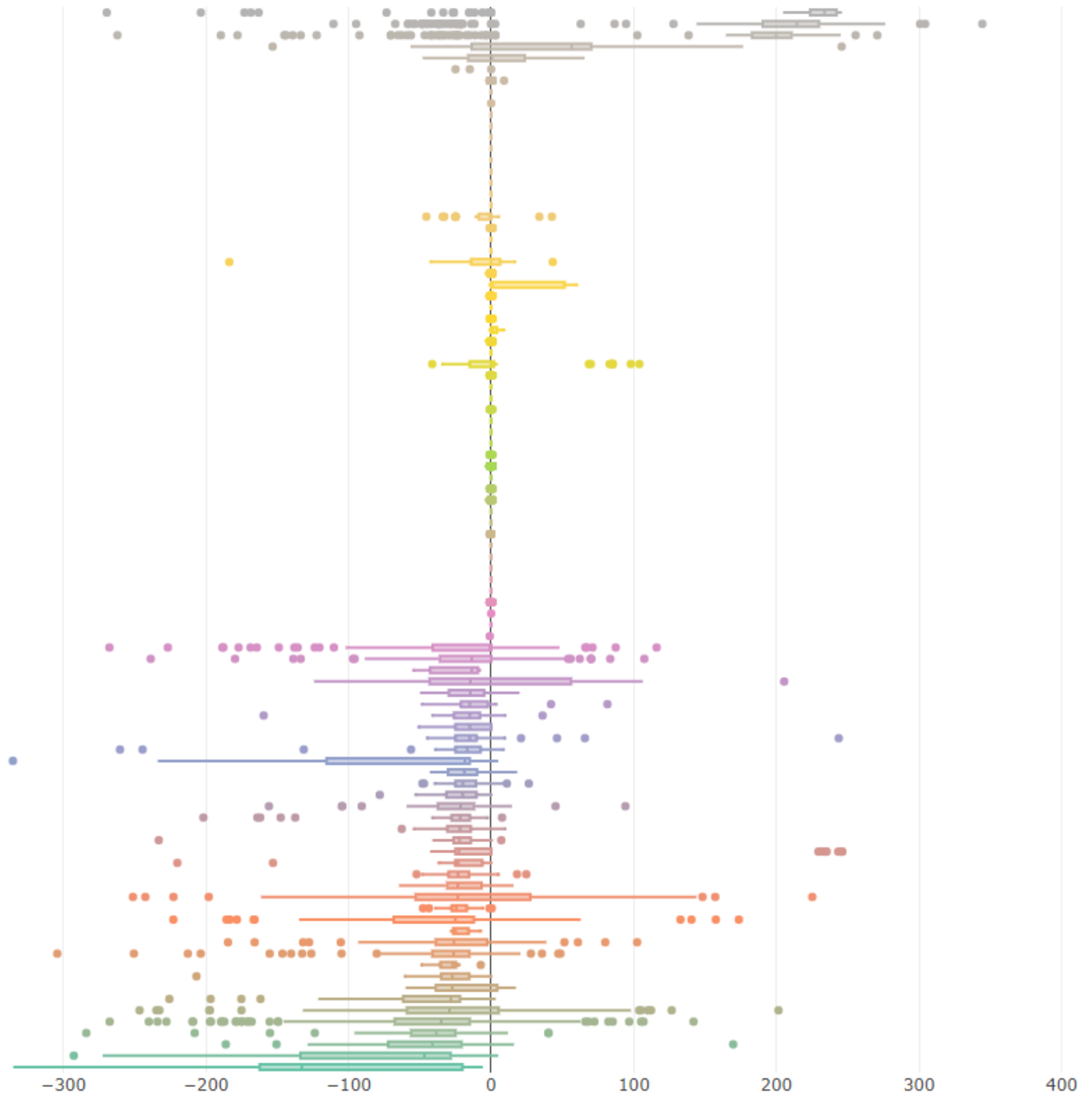


Figure 3.10. Boxplots of differences of median propagated delays for each flight between DSTAR and Default for Virgin America in minutes

Figure 3.11 shows how the DPM model performed in comparison with the default recovery model. As with the previous figure the boxplots are ordered, color-coded by median value. We see that, as before with the DSTAR vs. Default model, the DPM model often outperformed the default model. However, there were some cases where the opposite happened, but this is also due to the one-day-only simulation.

The most important, and interesting, comparison is between DSTAR and DPM, see Figure 3.12. In this figure, it is possible to see that most of the medians are on the positive side of the graph. These indicates that the DPM performed better than the DSTAR model. The caveat here is the filters that were applied, i.e., flight cancelations were not possible. Recall that the DSTAR model did not cancel any flights and DPM canceled around three percent. We can conclude that this a tradeoff between these models. DSTAR prefers to delay a flight more and DPM chooses to cancel this same flight.

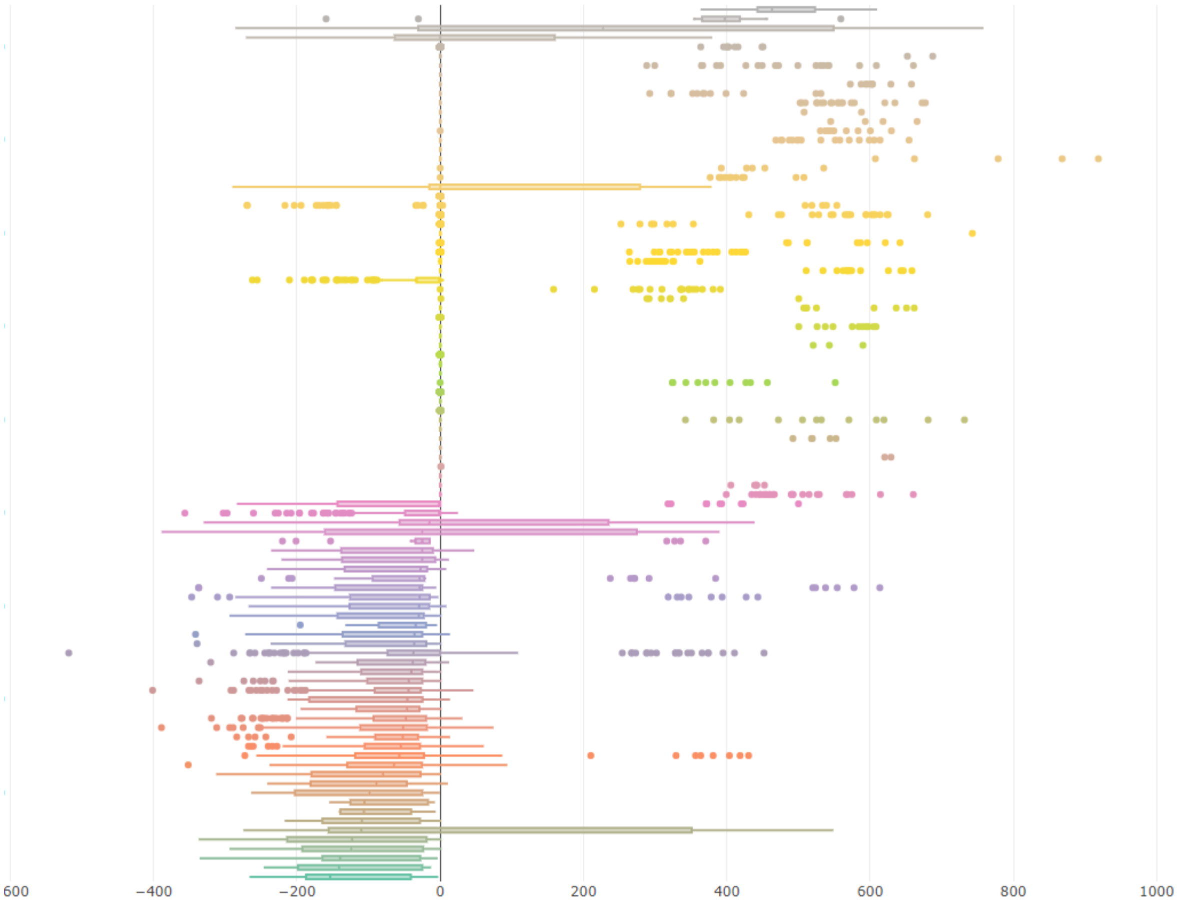


Figure 3.11. Boxplots of differences of median propagated delays for each flight between DPM and Default of Virgin America in minutes

We conduct the same type of analysis for Alaska Airline, to see if changing the airline has any influence on the performance of the recovery models. The results we have obtained are described in Figures 3.13 to 3.15. Since there are around five times more flights for AS than for VX, we are not showing boxplots but only the median values (red curve) and quartile ranges (gray area). Using boxplots as before in Figure 3.1, 3.11 and 3.12 would lead to unreadable figures.

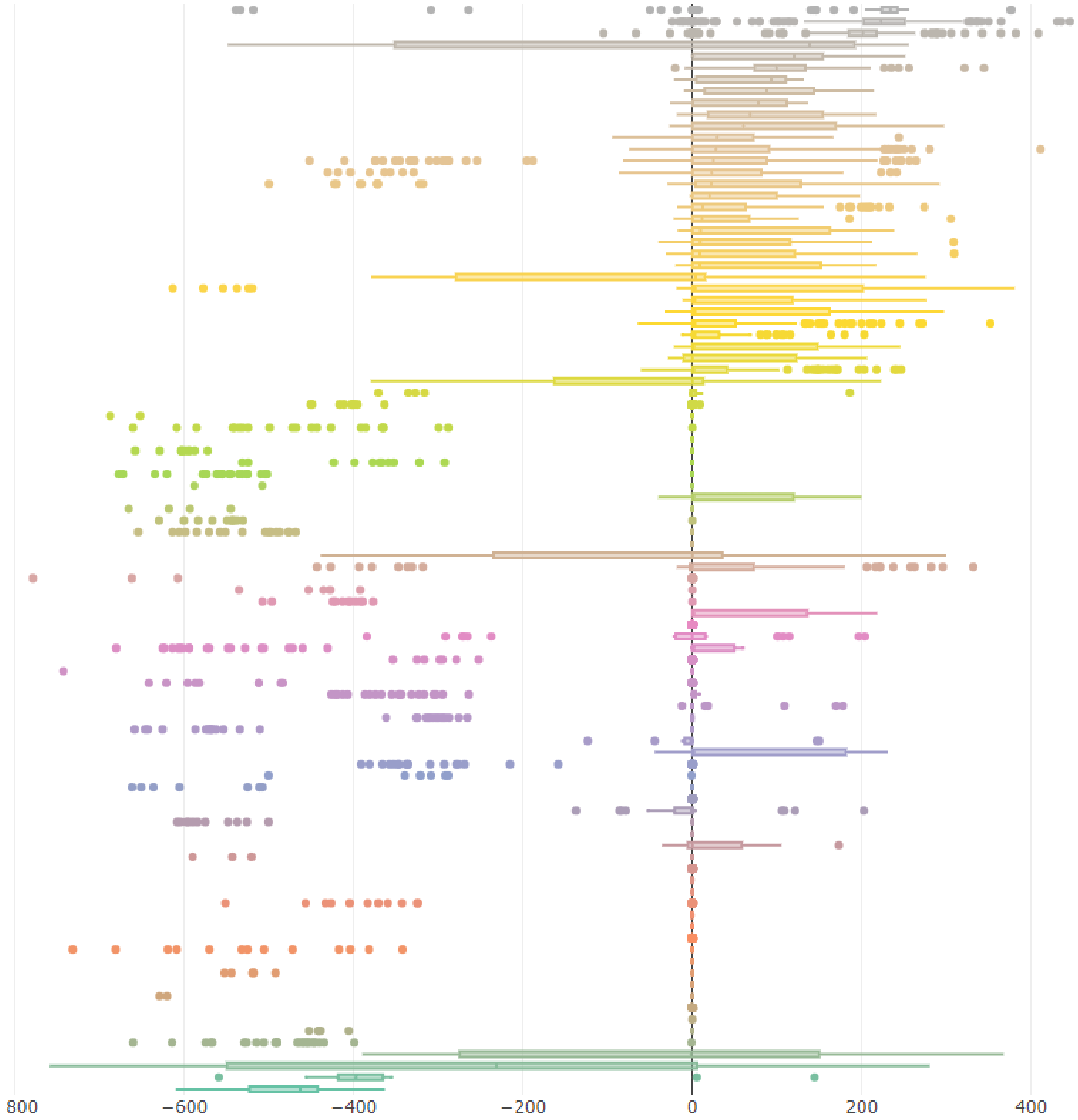


Figure 3.12. Boxplots of differences of median propagated delays for each flight between DSTAR and DPM Virgin America in minutes

In this figure, Figure 3.13, most flights are represented on the left side of the chart, meaning that the DSTAR model performed better than the default model. Indeed, approximately half of the flights have their median and quartile limits on the negative side, and for around one third of the flights the difference is negligible. Only a small proportion of the flights have a positive median of the differences. This is similar to what happened with Virgin America.

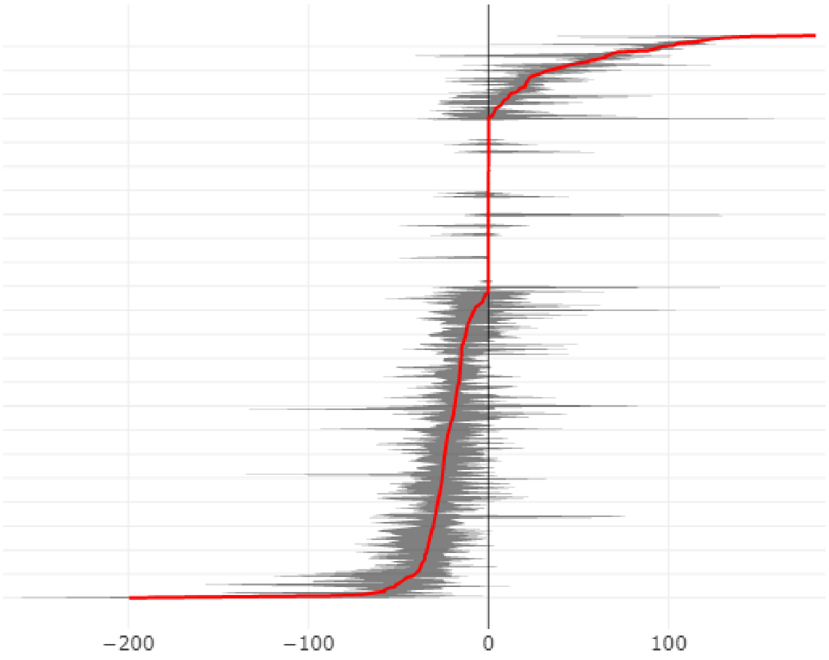


Figure 3.13. Ordered median value of differences of propagated delay between DSTAR and Default of Alaska Airlines in minutes

Next, we describe the performance of the DPM model in comparison with the base case, see Figure 3.14. In this case, as in the VX case, the differences are negative or negligible except for a very small number of flights. This indicated that the results shown before are not due to peculiarities of the VX network but rather to the recovery models.

Finally, in Figure 3.15, we have the comparison between the two recovery models and, as it happened for Virgin America, the DPM model outperformed the DSTAR model. However, also here we must attend to the filters. Canceling or not canceling a flight is an important decision, being hard to assess all the costs, including externalities, that an airline incurs when it cancels a flight. We see that with these simulation runs we can analyze the difference in performance of two recovery models and this would be extremely important information to have for an airline.

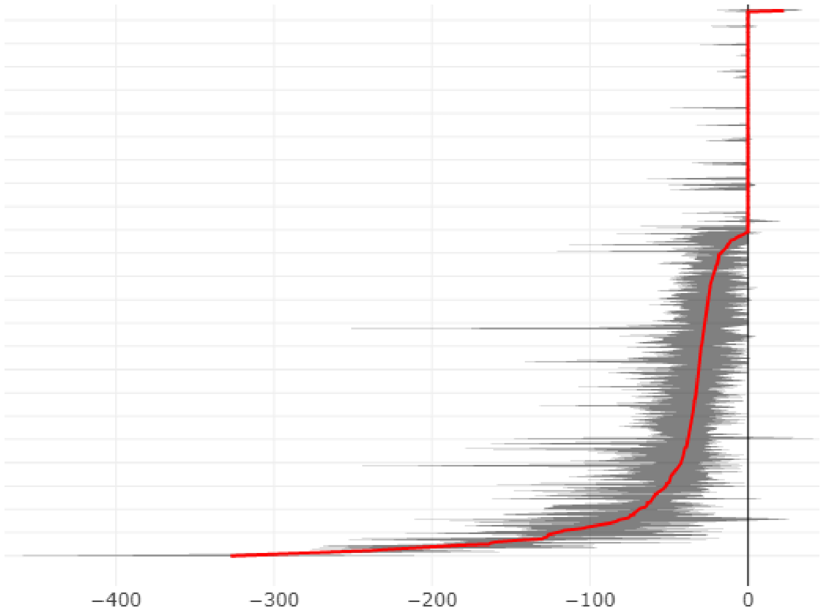


Figure 3.14. Ordered median value of differences of propagated delay between DPM and Default of Alaska Airlines in minutes

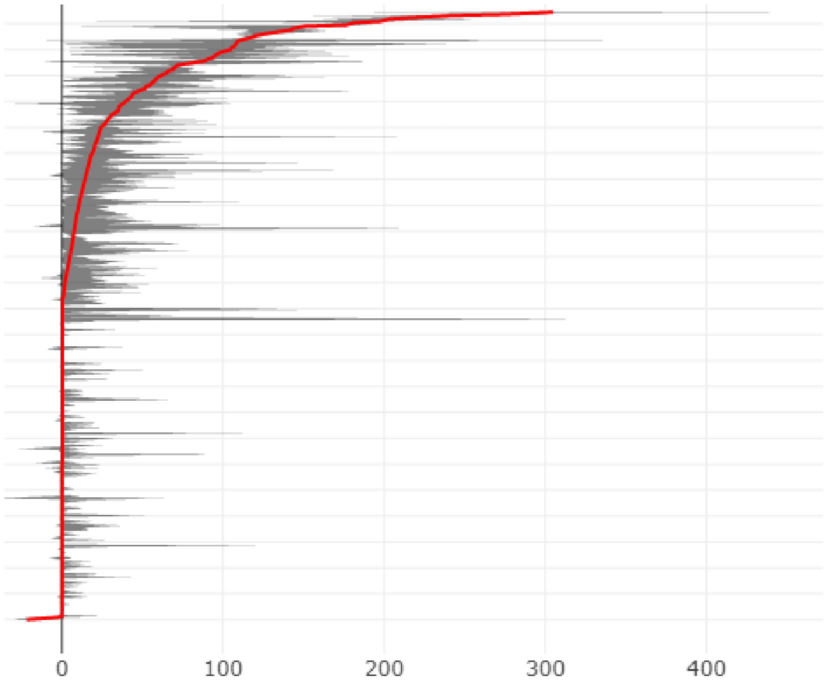


Figure 3.15. Ordered median value of differences of propagated delay between DSTAR and DPM of Alaska Airlines in minutes

We now describe the results we have obtained in our Simulation Delay Approach 2, where we simulated a whole year for each airline (repeated fifteen times, i.e. fifteen years). Table 3.9 summarizes the median differences of propagate delay for the DSTAR and DPM models versus the default model for both airlines summing the total days with negative delays on one hand and positive values on the other hand, on a per day and per flight basis. In the table, we also show the case where we only analyzed flights with propagated delays over fifteen minutes, which we will discuss later.

As the table shows, Table 3.9, in the case of median differences of propagated delays on a daily basis, there were no significant differences between both models and both airlines. The more interesting results are on a per flight basis. In this respect, we observed that the two models in total reduce more delays than they add for both airlines in a given scenario. This can also be verified by observing Figure 3.16. This figure shows the median differences per flight for DSTAR vs default and DPM vs default for Virgin America. Since around 85% of the flights have less than one minute of delay, and this fact could mask the results of the recovery models, we decided to analyze only “delayed” cases in the base case. This means we only consider the flights that, for the default model, had a propagated delay over 15 minutes, and then analyzed the performance of the DSTAR and DPM models for these flights. We found that the median differences were mostly negative for the propagated delays provided by the DSTAR model. That is, the DSTAR model clearly outperformed the Default model, see Figure 3.17. We can deduct from these results that, by considering all flights, where over 85% are (practically) on time, the advantages get masked by those flights with low delays. If we focus our attention only on the significantly delayed flights, there is a clear advantage of using a better recovery action, i.e., a “smarter” AOCC.

For the DSTAR case we see that in comparison to the base case most flights benefited, negative differences, when using DSTAR only a handful of flights had worse performance in comparison with the base case. In total there is a net positive effect since adding all delays (benefited ones) outweigh the sum of positive differences (the negative affected ones). DPM has no negative affected flight to account for, hence there is a clear benefit of using DPM versus the Default case. see Figure 3.17 and Table 3.9.

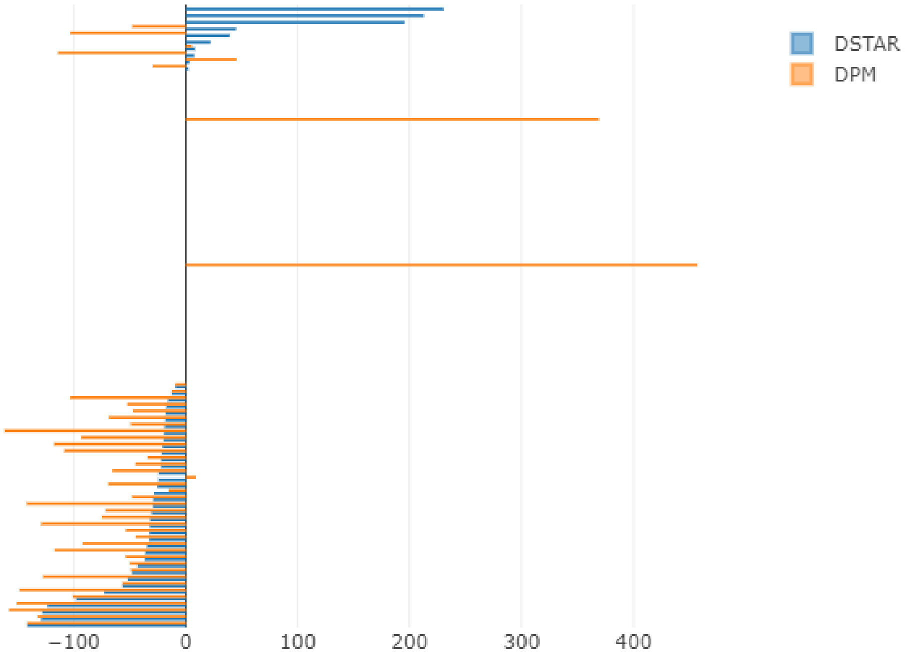


Figure 3.16. Median differences of propagated delays DSTAR and DPM vs Default, per flight, Virgin America in minutes

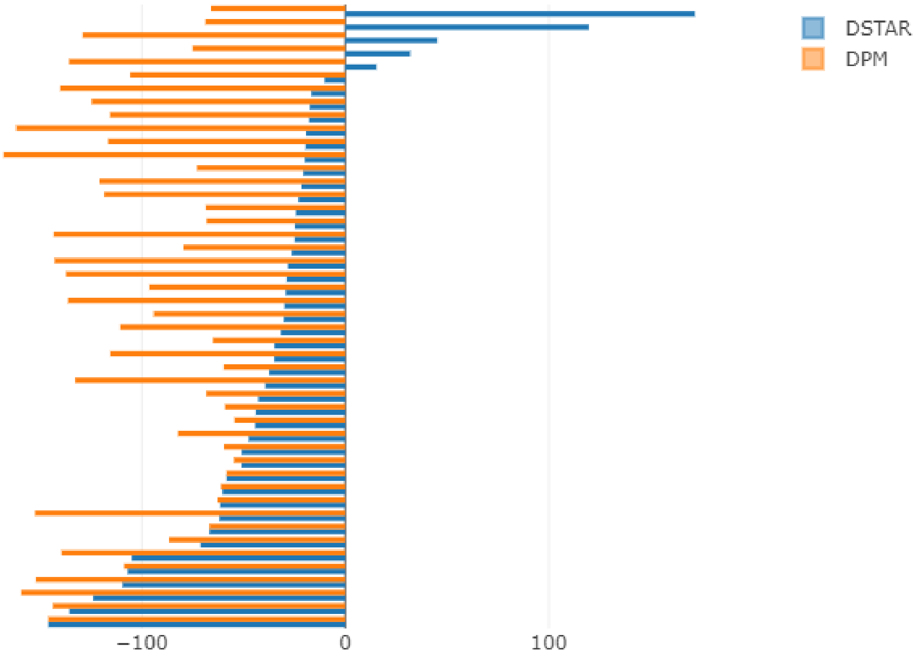


Figure 3.17. Median differences of propagated delays DSTAR vs Default and DPM vs Default (prop delay>15 min), per flight, Virgin America in minutes

A similar propagated delay analysis was conducted for Alaska Airlines. The results are shown in Figures 3.18 to 3.21. Here, also on a daily basis, the median differences were almost negligible, as shown in Table 3.9. As in the VX case, the real advantages are masked by the high percentage of flights that are on time or have small propagated delays. In this case, the DPM model performed better than the default model. The differences between the DSTAR and DPM are much smaller. The DPM model still performed better in almost every day. The differences between the two recovery models are around 10 to 15 minutes. For Alaska Airlines, we also looked at the data on a per flight basis. Here we see the similar S-shape as before, showing that the DSTAR and DPM model performed better than the default model.

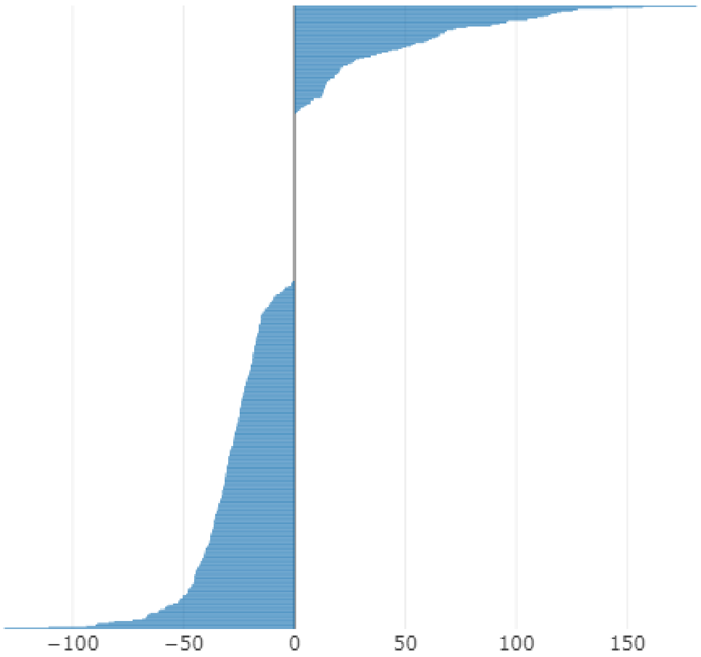


Figure 3.18. Differences of propagated delays DSTAR vs Default, per flight, Alaska Airlines in minutes

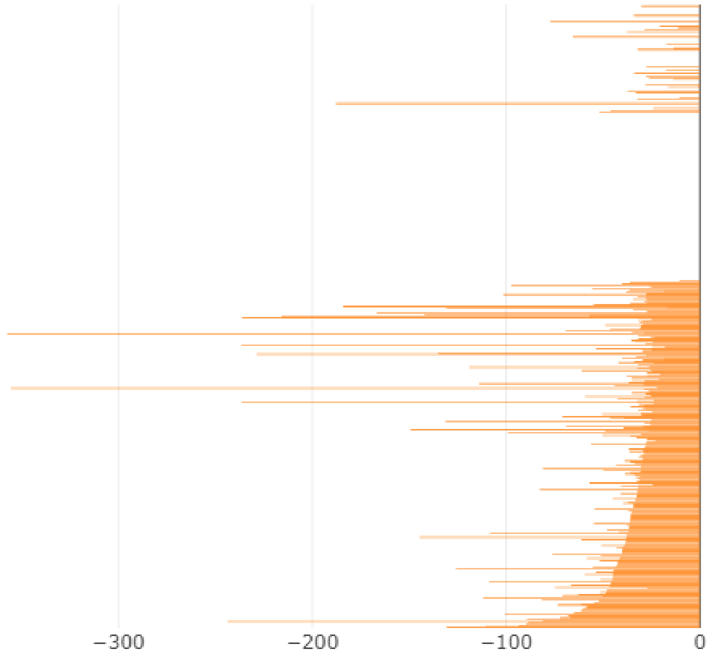


Figure 3.19. Differences of propagated delays DPM vs Default, per flight, Alaska Airlines in minutes

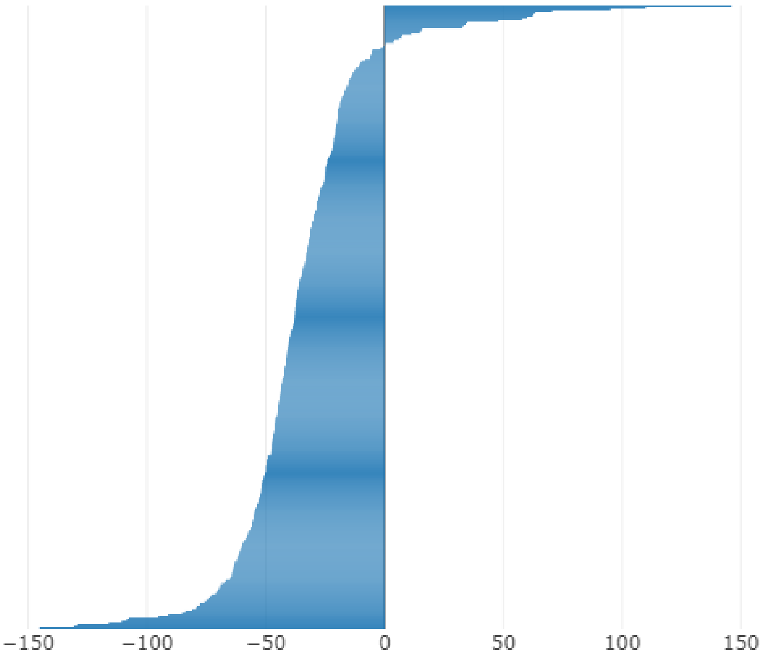


Figure 3.20.- Differences of propagated delays (>15) DSTAR vs Default, per flight, Alaska Airlines in minutes

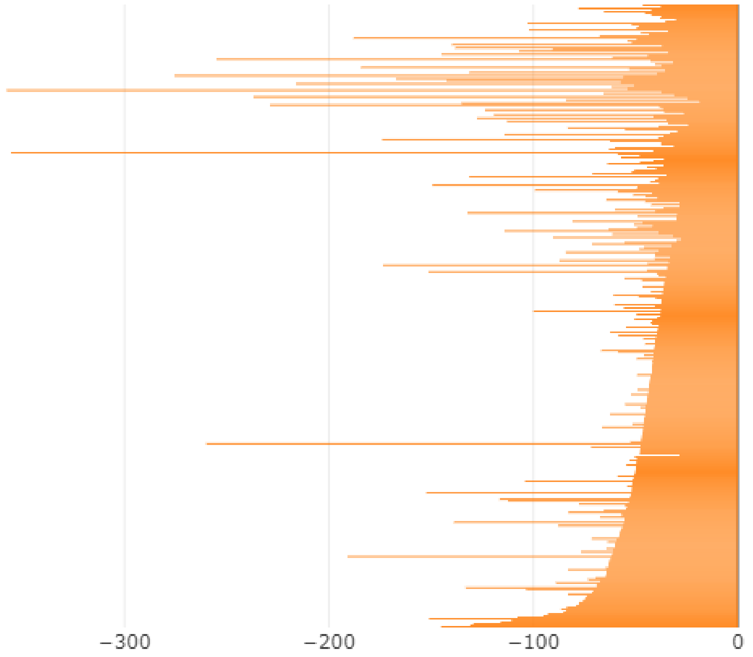


Figure 3.21.- Differences of propagated delays (>15) DPM vs Default, per flight, AS in minutes

Table 3.9.- Sum of differences of propagated delays per day, Simulation Approach 2

		VX		AS	
		Negative	Positive	Negative	Positive
per day	DSTAR	0,0	0,0	-1,7	0,0
	DPM	0,0	0,0	-360,0	0,0
per flight	DSTAR	-1550,9	766,3	-9250,8	4254,2
	DPM	-3283,6	885,5	-16215,5	383,8
per flight with propagated delay >15	DSTAR	-1997,7	383,8	-14627,2	995,0
	DPM	-4884,7	0,0	-24376,5	0,0

We have also examined the cost implications of using the different recovery models. As expected, the major implications relate to cancelations. First, we focused on the Simulation Delay Approach 1 to see how expected daily costs vary with the recovery model. In Figure 3.22 and Figure 3.23, we depict the boxplots of those costs on a logarithmic scale for VX and AS, respectively.

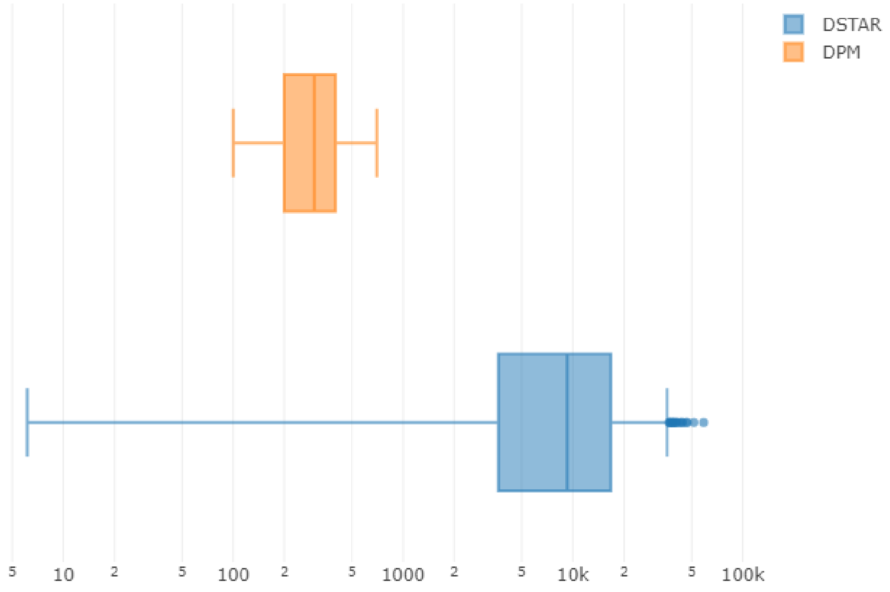


Figure 3.22. Cost Boxplots of Simulation Delay Approach 1 (Virgin America)

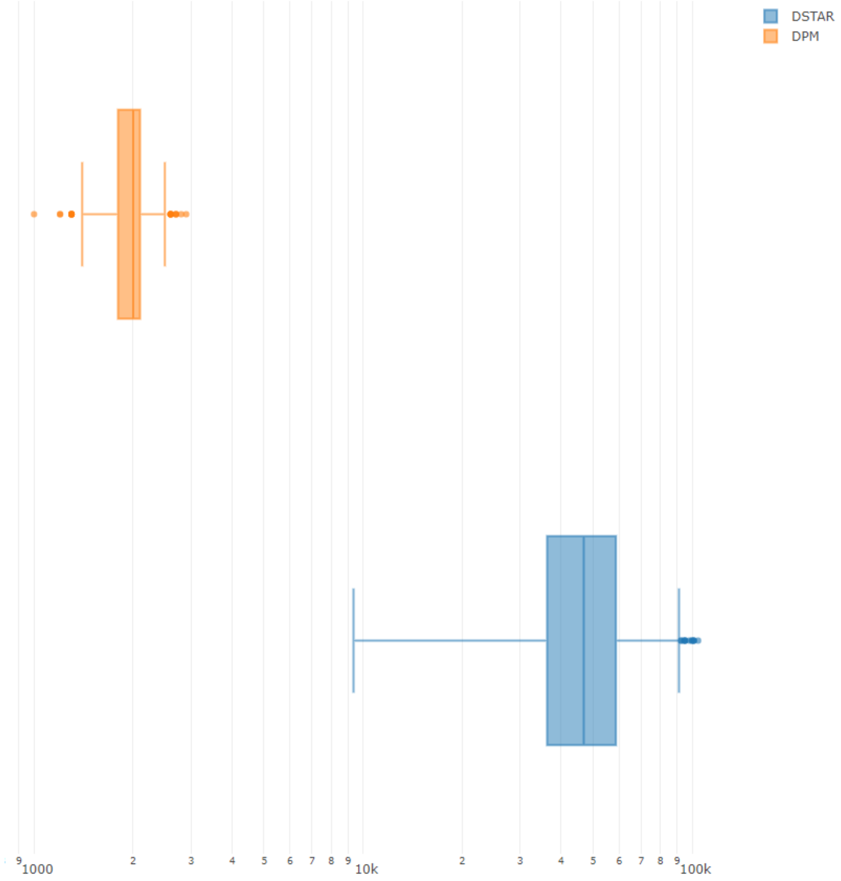


Figure 3.23. Cost Boxplots of Simulation Delay Approach 1 (Alaska Airlines)

As we can see, the median cost values provided by the DPM recovery model are significantly lower than those provided by the DSTAR model. This is in line with the perception that we got from the propagated delay analysis. We suspect that the main cause of those differences are the cancelations, since DPM does not account for cancelation costs. Since we considered a cancelation cost of $400,000 in the DSTAR model, we added $400,000 for each cancelation in the DPM model’s results and redraw the previous boxplots, see Figure 3.24 and Figure 3.25.

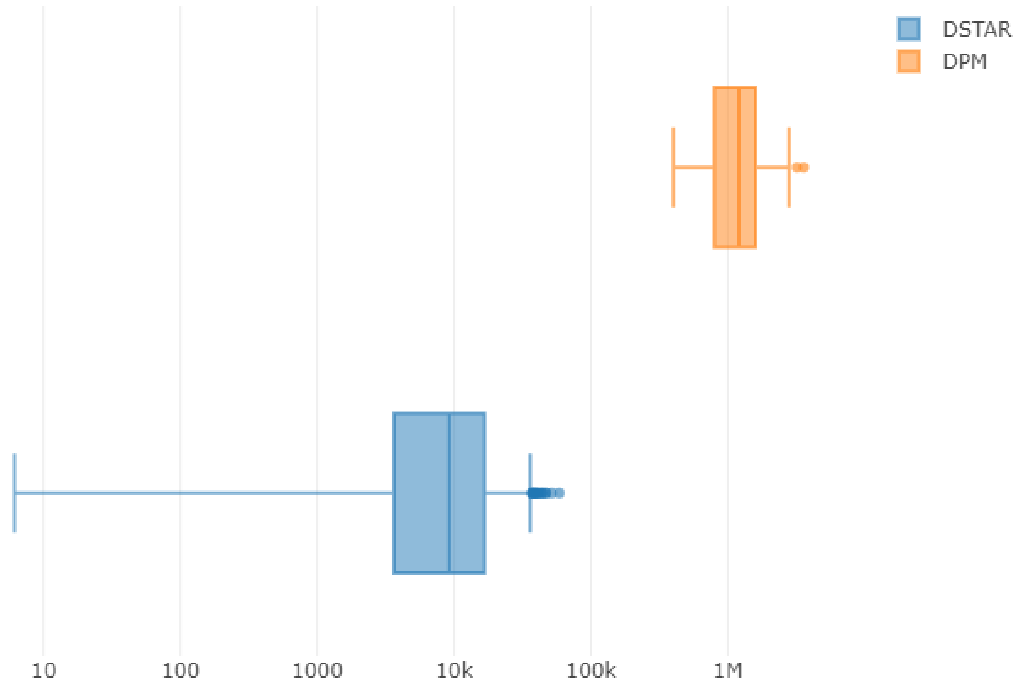


Figure 3.24. Corrected Cost Boxplots of Simulation Delay Approach 1 (Virgin America)

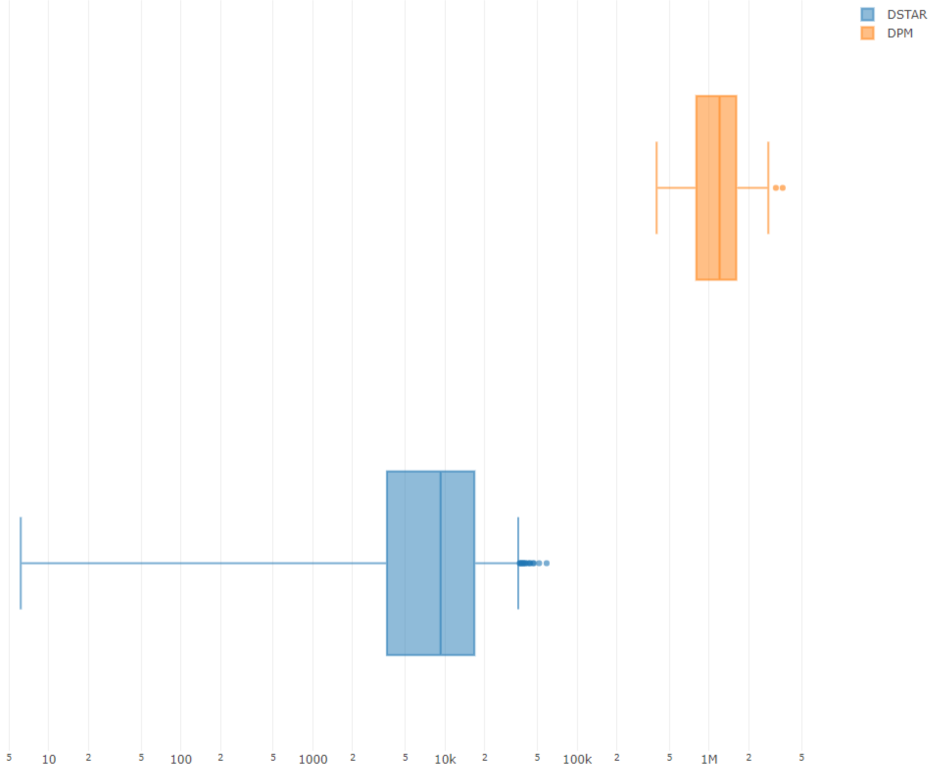


Figure 3.25. Corrected Cost Boxplots of Simulation Delay Approach 1 (Alaska Airlines)

The figures make clear that, when flight cancelation costs are taken into account, the DSTAR model outperformed the DPM model in every case by over one order of magnitude for both airlines.

In Figure 3.26 and Figure 3.27, we show the boxplots we have obtain for Simulation Delay Approach 2. We first computed the median value for each cost for each recovery model and then used those values to render the final boxplots. In this scenario we also have the same problem as before in the Simulation Delay Approach 1. Since the cancelations where an issue as also here the resorted to the same corrective measure as before. We obtain the following boxplots for each airline with corrected cost figures, Figure 3.26 and Figure 3.27:

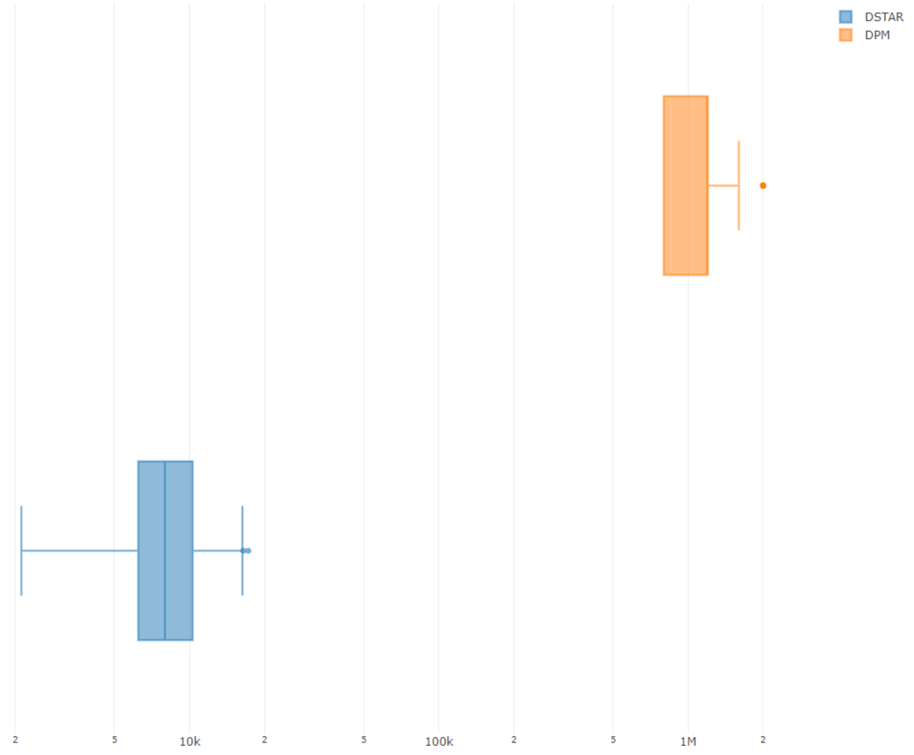


Figure 3.26. Corrected Cost Boxplots of Simulation Delay Approach 2, Virgin America

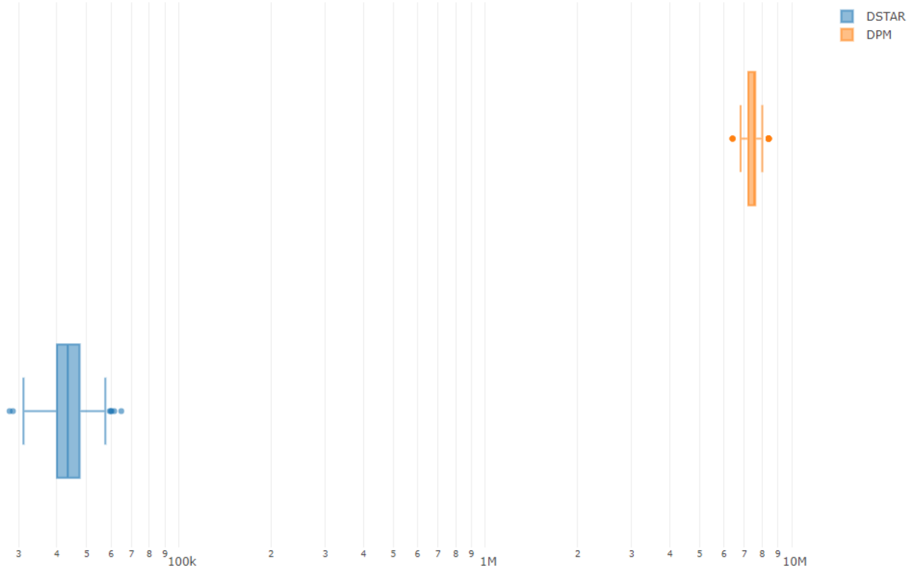


Figure 3.27. Corrected Cost Boxplots Simulation Delay Approach 2, Alaska Airlines

As happened for Simulation Delay Approach 1, the costs given by the DPM model are at least one order of magnitude higher than those given by the DSTAR model.

We cannot say that any one of the models clearly dominates one other, but we can say that the cancelation being a feasibility sink in the DPM model will lead to an increased cost of this model that is not really captured. If anyone uses this model in a real scenario this issue could lead to a pitfall. DPM was able outperform the DSTAR model when only considering propagated delays but at the cost of a higher level of cancelations. Hence there are significant Tradeoffs between DSTAR prefers to delay flights, not canceling any flights and DPM prefers to cancel flights and in consequence have fewer and smaller delays.

3.6 Conclusion

We demonstrated that it is possible to design a user-friendly simulation model with high quality details of an airlines operation that is flexible enough to accommodate most researchers needs and able to factually represent the daily operation of an airline. Through the analysis of the outputs of the AOSM in the case study it is clear that without a simulation model it would be hard to compare recovery models and identify where these recovery models have their strengths weaknesses and identify the trade-offs of the solutions.

Further, we demonstrate that for the full comparison of an airline operation a simulation needs to include enough detail and capture its uncertainty. While each recovery model in fact obtains its optimum, with the simulation model developed by us it is possible to better understand what the implementation of a specific recovery model’s solution entails. Not saying that this model is a full representation of the reality, but it captures the realty in such a way that its useful for most users. For researches, airline industry practitioners and even for teaching purposes at an advanced level.

We saw that the high dimensionality of airline operations is complex and to simple reduce it to key performance indicators might be misleading. The comparison of just two schedule recovery models allowed us to explore such complexity. This make the case that without a simulation model to explore the results of a schedule recovery model we would face great challenges to gauge the actual performance of each schedule recovery model. The biggest problem is to compare different optimization models with each other. Each has its own objective function; hence each has a different objective. In the here presented case study we saw that with a detailed exploration of the schedule recovery models' solutions it was possible to better understand how two recovery optimization models work, what the differences in their performance are and where are the trade-offs. These trade-offs that one schedule recovery model is willing to make in comparison with the other, represents different type of decision-making profile that each airline would have with DSTAR or DPM.

Subsequently we conclude its not enough to just include all aircrafts, flown routes, schedules and personnel to simulate an airline. The decision-making profile, i.e. agency, of the airline needs to be represented to properly capture the reality. We show that to have an adequate representation of an airline in a simulation one needs to include its decision making-profile. In our case we used the recovery models as proxy of the AOCC and saw that just different philosophies of the recovery models (since in our case the Recovery models are a proxy of the AOCC it also means the airline would have different decision-making profiles) will lead to significantly different outcomes. DSTAR delayed more and did not cancel and DPM model ended up cancelling more of the flights but had smaller and fever delays. This is even more important if the objective of the simulation model is a system wide simulation. Meaning that each airline's decision-making profile needs to be captured so it properly mimics the actions it takes in real life. Assuming that every airline has the same behavior in such a systemwide simulations might lead to a misrepresentation of the reality. As demonstrated by our simulation where we only simulate one airline, but with different recovery models that lead to different decision making profiles, i.e. behaviors.

3.7 Future work

One of the main priorities is to incorporate the non-independent distributions. For instance, introduce weather information and based on that information create the delay profiles. One other interesting addition to the simulation model would be several type ratings for crews so mixed fleets can be used and this information is used during the recovery process. Also dividing between cockpit crews and cabin crews to capture better the reality of the operations the airlines face on a daily basis. One other issue that needs to be addressed is to include the maintenance so realistic multi-day simulation can be made and capture better the reality of an airline operation. Sensibility analyses on the cost parameters would also be interesting to understand how the models would react to varying costs, what are the implications on the recovery solutions, and gauge their robustness to these costs. Understand which other actions on airlines takes, i.e. the agency of the AOCC at operational level, better reflects, and captures the reality. Finally simulate proactive agency profiles of airlines. Meaning simulate a proactive behavior of the airline and see what ate the benefits of these actions. Also compare various proactive solutions with each other.

4 AIRLINE SCHEDULE RECOVERY: USING METAMODEL TO PREDICT OPTIMAL RECOVERY ACTIONS

4.1 Introduction

Irregular operations are one of the most significant problems that the airlines regularly face. The airline department mandated to handle this issue is known as the Airline Operational Control Center (AOCC). It supervises and controls the operations on a daily basis with reactive and proactive actions. Depending on size and complexity of the airlines, AOCC´s characteristic can largely vary. It might consist of a small team or, at the other extreme, multiple sub-departments each staffed with dozens of people. When the AOCC is subdivided into separate units, a structure more common in larger airlines, each unit handles one specific problem type. For instance, one unit each is responsible for aircraft, crews, or passengers, respectively. Each tries to solve its own problem having at its disposal its own tools to solve their problems. These tools can be called as schedule recovery models. They try to recover from irregular operations, so that the schedules can be operated normally again. These tools must be run in short time windows since the problem at hand must be solved real-time during operations. This means that there is a short time window of minutes to solve the problem. We must be able to run the actual model in seconds, since the solution must be communicated and negotiated with other departments inside the AOCC and these actions take time too. There are already some integrated solutions developed in research, but actual full implementations of them are still rare; there is no production grade tool that can substitute the whole AOCC.

Most of the models used in each sub-department focus only on their own objectives. Each solves issues related to only aircraft, only crews, or only passengers, etc. These models are run several times a day leading to hundreds, or even thousands, of optimization runs during a year. These runs generate solutions for several individual instances of the problem.

However, this data, consisting of problems and their corresponding solutions, are typically not reused. Hence, even if a very similar problem reappears in future, the recovery model is often required to be run again instead of using the preexisting solution from the previous similar day. It might happen that the staff of the AOCC can predict the solution, due the recurrence of the problem, but there is no structured way to use this historic data on recovery problems and solutions.

A model that would be able to learn from these historic data could accelerate the decision process. A similar conclusion, that the database of historic solutions of the recovery problems is an underused asset, was made by Rashedi and Vaze (2019). One approach is the creation of a metamodel (a model is a representation of the reality, a metamodel is a representation of a model) that predicts the solutions of the optimization model based on the historical data on recovery problems with an acceptable degree of accuracy. Best candidates for such a metamodel are Machine Learning (ML) models. Interestingly, a machine learning model used as a metamodel, once trained, could potentially shed some light on the implicit logic inside such machine learning models which are often notorious for their “black box” nature. If the ML model to be used is calibrated and validated on the historical data consisting of optimal recovery solutions, the ML metamodel’s predictions are likely to be close to the optimal solution as well. On the other hand, one can argue that since recovery models must run in short time windows and often rely on heuristics to solve the recovery problem, their optimality is typically not guaranteed. In this sense, the machine learning model may generate solutions that are as good as or better than those obtained from the heuristics. This is another advantage of a machine learning metamodel. A metamodel framework as proposed in this work can be trained offline. Hence, we can decouple the run time of the recovery model from live operations, as depicted in Figure 4.1. For instance, instead using the heuristics used live inside the AOCC, we can afford to use an exact solution approach which might be slower, but would guarantee optimal results. Because of the offline nature of the process, these models are not constrained to produce solutions in a time window of a few seconds. This means that we can train the ML metamodel with historical problems and the corresponding exact optimal solutions, even if it takes hours to solve. Additionally, it is even possible to reduce the number of assumptions and/or simplifications underlying these models, since the computational time is not a major issue anymore. Finally, this would enable the use of fully integrated recovery models, which due to their burdensome computational times, cannot be used in real-time AOCC operations.

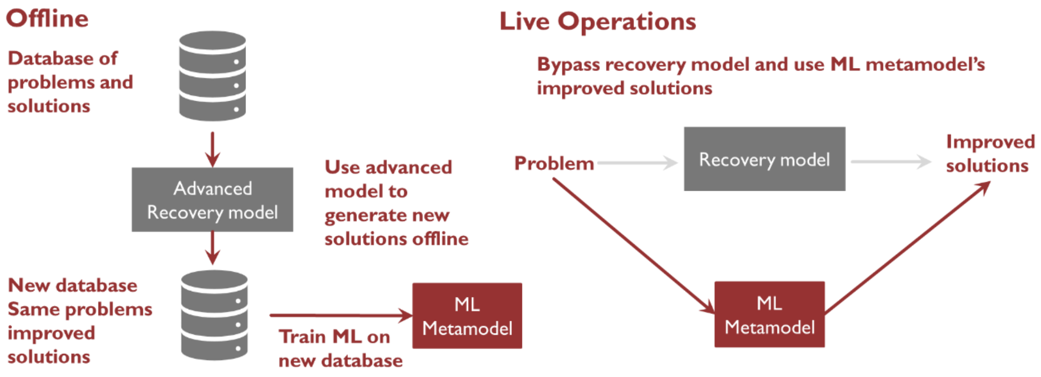

Offline
Database of problems and solutions
Advanced Recovery model
Use advanced model to generate new solutions offline
New database Same problems improved solutions
Train ML on new database
ML Metamodel
Live Operations
Bypass recovery model and use ML metamodel's improved solutions
Problem
Recovery model
Improved solutions
ML Metamodel

Figure 4.1. Illustration of using ML metamodel trained on optimal solutions to historical instances

Another important benefit of this approach is that it is possible to be implemented gradually, thus not requiring a large commitment of time and effort from the start. Initially, one can use the historical recovery solutions for each sub-department inside the AOCC and train ML metamodels based on them. Depending on the accuracy and general quality of these predictions, as a first step, the ML metamodel can be implemented partially, or its predictions can be used as indicators that have the benefit of being immediately available. This is so because, ML models, once trained, typically generate almost instantaneous predictions. This instantaneous feedback can help operational managers prepare for the likely optimal solutions. Moreover, it can also be used to accelerate the computational times of the full recovery model solution heuristics. Providing solution values of some variables, or dual values of some constraints, from the ML to the full recovery model, before the full recovery model's run starts, can potentially reduce the computational time of the full recovery model. After one has more confidence in the ML metamodel's prediction accuracy, one can gradually substitute each original recovery model by the corresponding ML metamodel. This step-by-step inclusion, as depicted in Figure 4.2, could eventually lead to a fully autonomous AOCC where human interactions would only be needed for surveillance purposes and to solve more extreme or rare events. Also, once fully implemented, this could be the stepping stone to having a fully integrated, cross-airline irregular operations management. Moreover, real-time operational negotiations (such as for ground delay programs or airspace flow programs) between multiple airlines would also be much more efficient if there was an autonomous AOCC's in each airline, assuming common communication protocols between them are in place. Airline driven ground holding was a thematic that was analyzed by Yan in his Ph.D. thesis Yan (2017).

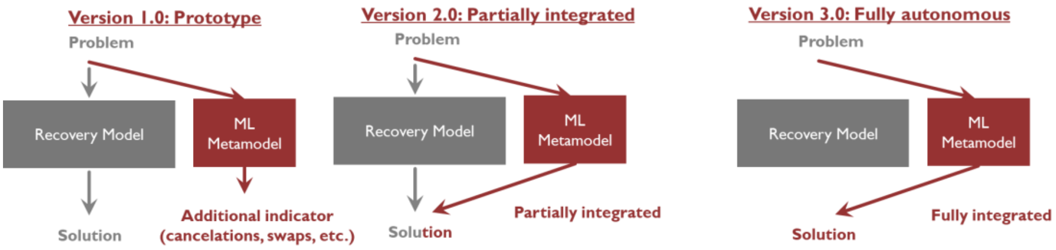


Figure 4.2. Illustration of a step-by-step inclusion of an ML metamodel into the AOCC systems

Another important feature of the ML metamodel framework as proposed here is that the ML metamodel could also learn from the staff working in the AOCC. Even the best recovery model has some assumptions or limitations because of which the staff working at the AOCC needs to manually alter the solutions eventually. Staff with years of experience in an AOCC have invaluable knowledge that would be lost if they leave the airline. The ML metamodel would be able to learn from this staff too, thus include the years of experience into its predictions. For instance, if we were to train the ML metamodel on the solutions not directly from the schedule recovery model but from these manually corrected solutions, we would have an ML metamodel that learned for the recovery model and the knowledge of the staff working in the AOCC (see Figure 4.3). Additionally, other data sources can be used to increase accuracy of the solutions. For instance, weather data that were used by the staff to correct the solutions can also be used to improve the performance of the ML metamodel.

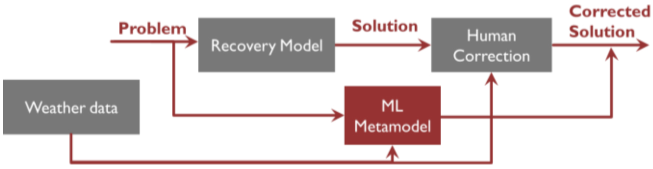


Figure 4.3. Illustration of an ML metamodel learning from human input and using information from weather data

In Section 4.2, we will briefly review the literature related to airline schedule recovery, and metamodels. In Section 4.3, we present our methodology. Section 4 contains our case study where we present our proof-of-concept. Discussion of the results is provided in Section 4.5. Finally, we present the conclusions in Section 4.6, and an overview of the promising directions for future work in Section 4.7.

4.2 Literature review

As mentioned in Section 4.1, even the most robust schedules can sometimes break down and hence need to be recovered from disruptions. This is known as the airline schedule recovery process. Every airline usually has an Airline Operational Control Center (AOCC) that manages those recovery measures. These AOCCs are responsible for the execution of the planned schedules. Many airlines rely on a rule-based approach heavily dependent on the staff expertise. To cope with the complexity, they approach this problem in a sequential manner. In other words, they use the same approach for recovery as for planning the original schedule (Clausen et al. 2010). Assessing the quality of the recovery solution is a challenging task. There are some Decision Support Systems that help in defining the recovery measures but there are no fully integrated airline schedule recovery systems in operation at any of the airlines (Clausen et al. 2010).

Schedule recovery is a research topic for a large number of existing studies. One of the first approaches in literature was developed by Teodorović and Stojković (1990). Since then, a large number of subsequent studies have focused on improving the problem formulations to make them increasingly realistic and have also focused on developing faster and more accurate solution methods to solve these models. The first resources to be attended in an airline schedule recovery process are usually the aircrafts, because of their relative scarcity and high cost in the airline business. The crew scheduling problem is also tackled numerous times in the literature since the crews are the second most important resources in an airline. However, the crew schedule recovery problem has been paid less attention compared to the more strategic planning problem of airline crew scheduling (Ball et al. 2010).

Passenger recovery is rarely addressed in the literature. One example is the research done by Bratu and Barnhart (2006). They developed an integrated model and paid particular attention to passenger recovery processes. Passenger disruptions rarely drive the schedule recovery, but arriving on time is the service characteristic most valued by passengers (Mitra ,2001, Bratu & Barnhart 2006)). Bratu and Barnhart (2006) claim that passenger centric schedule recovery can be made in an efficient way and that an efficient solution can be produced in less than three minutes. Barnhart et al. (2014) faced with the fact that there is limited data available in relation to air travelers, developed a multinomial logit model for estimating historical passenger travel so future researcher would have a baseline to construct future models on it.

Cargo has been the object of very little attention in the airline schedule recovery literature. Nevertheless, in the last couple of years, many airlines have been increasingly using their passenger flights to haul a considerable quantity of cargo. If one of those flights gets disrupted, it may also disrupt the cargo. While we are not aware of any previous studies focusing on schedule recovery of cargo transport operations by commercial passenger airlines, there is some literature that accounts for cargo in the scheduling of commercial passenger airlines. Due to the emergence of mixed airlines (passengers and cargo) and the use of free belly space of passenger aircraft for cargo, the airline schedule recovery process can also affect the cargo transportation, and hence it must be accounted for by the AOCCs of those airlines.

Derigs and Friedrichs (2013) attempted to solve the problem of integrated scheduling and pointed out some differences in passenger flights and cargo flights. One is that, in cargo, some flights are mandatory and some are optional. So, the cargo carriers establish their network to satisfy the most important (mandatory) flights and then they try to fit in the other flights (optional). This is not a common practice for passenger flights. One other difference is that cargo trips are usually one-way. That is, cargo usually moves from one point to another and does not return. However, the aircraft must return to its hub in most cases. Passengers normally take round trips and return to their point of origin eventually. Another distinctive feature of air cargo business is that the clients attribute a high value to schedule stability in order to be able to plan ahead. Derigs and Friedrichs (2013) described an integrated model to handle those differences. They proposed a mixed-integer mathematical model that considers flight strings and not flight legs.

Ball et al., (2006) provide an overall review of the various facets of the challenge of irregular operations in the airline industry. It covers the airports perspective, airspace perspective and also the airlines perspective. The last of these three is the focus of our research. They have identified four points for ensuring a proper schedule recovery. First, the recovery must ensure obeying the crew work rules and aircraft maintenance requirements. This requires the knowledge of recent history of both, as well as the knowledge of passenger itineraries. Second, recovery may use additional resources, i.e., standby crews and aircraft. Third, there are multiple objectives of the airline schedule recovery problem, and this multi-objective nature of the problem turns it into a tradeoff of various performance indicators that the airlines need to be aware of. Fourth, the problem must be solved in a time window of seconds. We refer the reader to this work for a deeper understanding of irregular operations including aircraft recovery, crew recovery and passenger recovery. The multi-objective and real-time nature of the problem, as highlighted by the last two points, makes it especially difficult to integrate the aircraft, crew and passenger recovery operations into a single mathematical model.

Existing studies on integrated schedule recovery have attempted varying degrees of integration. DESCARTES (DEcision Support for integrated Crew and AiRcrafT recovery) was one of the first attempts to solve the airline schedule recovery in an integrated way (Kohl et al., 2007). The authors note that the process of schedule recovery relies heavily on the staff that is responsible for it at each airline, i.e. staff of the AOCCs. Integrated schedule recovery is one of the major challenges that both academia and industry have been trying to tackle for years using a wide array of approaches. This has resulted in several related initiatives, notably the creation of the 2009 ROADEF challenge, a competition organized by the French Operational Research and Decision Analysis Society. Each year ROADEF announces an operations research related challenge proposed by the industry. In the 2009 version, AMADEUS SAS proposed a challenge where they made a call for algorithms to solve a disruption management problem for a commercial airline. This challenge did not include crews. Bisaillon et al. (2010) won the 2009 ROADEF Challenge. It was an important milestone in the recovery research because it compares the performance of several models.

As mentioned before, schedule recovery consists of repairing the schedule after a disruption. But as mentioned by Petersen et al. (2010), an optimization approach might not be possible due to the computational time necessary to solve any component of the problem. They also state that most carriers usually do not use mathematical programming to tackle the integrated problem. Most frequently the recovery is performed sequentially, and the passengers are the last components to be attended to. However, since regulations (in EU or US, for example) may require that compensation should be given to the passenger for large delays, Petersen et al. (2010) make a case for the companies to solve the recovery problem in an integrated, rather than a sequential, way. Petersen et al. propose such an integrated approach but reduce its scope, so it is solvable in an acceptable time frame. They also reduce the flights that are considered by the model. This model uses the concept of flight strings rather than flight trips.

Abdelghany et al. (2008) solve the recovery problem using a combination of simulation and optimization models. Similarly, to other authors they claim that most recovery models are not integrated. They mention that the type of violations/disruptions for crews and aircrafts are similar and he mentions that it is possible for crews to make a deadhead trip (flying as passenger) but does not mention the similar movement for aircrafts: flight of an aircraft without passengers. They also refer the typical practices to solve planning disruptions and claim that due to the complexity involved, this is made on a flight-to-flight basis rather than in an integrated way. These researchers state that schedule recovery is a time critical process that needs solutions in a very short time window. This work does not have in consideration the passengers.

Castro and Oliveira (2011) attempted to create a fully automated system (although always with a human in the loop) for the integrated schedule recovery problem. They constructed a multi-agent framework in which each of the agents represents one department of an AOCC (aircraft, crews, passengers). Each agent creates its own solution while focusing on their respective infeasibilities. Then each agent must negotiate the solution with the other agents to obtain a final integrated recovery plan. They assume a considerably high automation of the whole recovery process that might be difficult to implement since there are still very few companies that have a DSS to start with and thus might be very reluctant to change to a totally automated process. Castro and Oliveira were the first to tackle the problem with a multi-agent framework. In 2012 they made some improvements on this original paper (Castro et al., 2012) and tested it with real data from the Portuguese airline company TAP Air Portugal.

Clausen et al. (2010) claimed that the traditional sequential approach is outdated and inefficient. They divided past research studies into three categories: aircraft recovery models, crew scheduling models and integrated schedule recovery. This paper showed that the most used operations research methods were mixed-integer optimization models and several heuristics. For a more recent review of the schedule recovery in transportation, we refer the reader to Visentini et al. (2014). These authors make an extensive review of existing schedule recovery models for various transportation model, air included, present in the literature. And for a deeper understanding of Irregular operations we refer the reader to Barnhart and Vaze (2015)

Another topic that we briefly reviewed the literature on is metamodeling, which can be thought of as a heuristic approach. Metamodeling (also knowns as surrogate modeling) can be defined as the act of using a model of a given base model to expedite the modeling and solution process. This approach is widely used in various fields such as engineering design, design optimization, model-driven optimization, and simulation optimization, to name a few. The details and types of metamodeling approaches are not the focus of this work. For further detail on metamodeling and a literature review of the state-of–the-art on this topic we refer the reader to Sprinkle et al. (2014).

Metamodeling is a well-researched topic, as can be seen from the review Barton (1994) of the state of art of the previous century. Most common approach to metamodeling is the model-driven engineering approach in a context where expensive simulation models must be run several times to get key performance indicators. To accelerate this process, one creates a metamodel that substitutes the base simulation and can generate the same KPI's in a fraction of the time. Wang and Shan (2007) provide a comprehensive list of metamodeling techniques in support of engineering design optimization. Many studies have used Artificial Neural Networks (ANN) as a metamodel. Hornik et al. (1989) and Funahashi (1989) concluded that a Multilayer Perceptron, a basic type of ANN, was capable of approximating any function to a desired degree of precision. However, the most common use of ANN in metamodeling situations is in simulation metamodeling. There are plenty of examples of neural networks, being universal approximators, used as metamodels, for instance: Melo et al. (2014) and Pan et al. (2014). Here, a computationally expensive model (viz., a simulation model) is substituted by a metamodel to be able to quickly generate simulation outputs. Rauch et al. (2016) use high performance computing to tackle the challenge of training of the artificial neural networks. One example of a study that uses metamodels to partially substitute an optimization problem is Werbos et al. (2012).

The closest example to metamodeling techniques applied to airlines in the existing literature is by Cetinkaya and Camci (2014), which focused on creating a metamodel for airport simulation. In addition to our focus on a very different application, our metamodeling approach also differs from the one used in this study.

We are not aware of any other study, except Rashedi and Vaze (2019) , that uses Machine learning models for approximating an optimization model. Most metamodeling on complex models, like computationally intensive simulation models, involves prohibitively large computational times for the individual runs of the complex underlying models. In our case, we have moderate computational times, but have an extremely short time window to solve them. Therefore, we attempt to leverage an ML metamodel to generate immediate solutions. In the next section we present our methodology and explain in detail our conceptual approach.

4.3 Methodological Approach

We use as inputs the parameters of problems that are faced by an AOCC and the resulting solutions generated by the AOCC for those problems. We create a database of these problems and solutions and then train an ML metamodel on that database. Afterwards, we include the trained ML metamodel into the AOCC systems and predict solutions to new problem instances with it.

These problem-solution constitute the database that will be used to train the ML model that will predict the solution of the schedule recovery problem. Once the ML metamodel is trained and if the solutions are of acceptable accuracy, we can then, in a second step, plug the ML metamodel into the AOCC systems. When a new problem instance is presented to the AOCC, the ML metamodel will input that same problem and instantly predict the solution, (as illustrated in Figure 4.4).

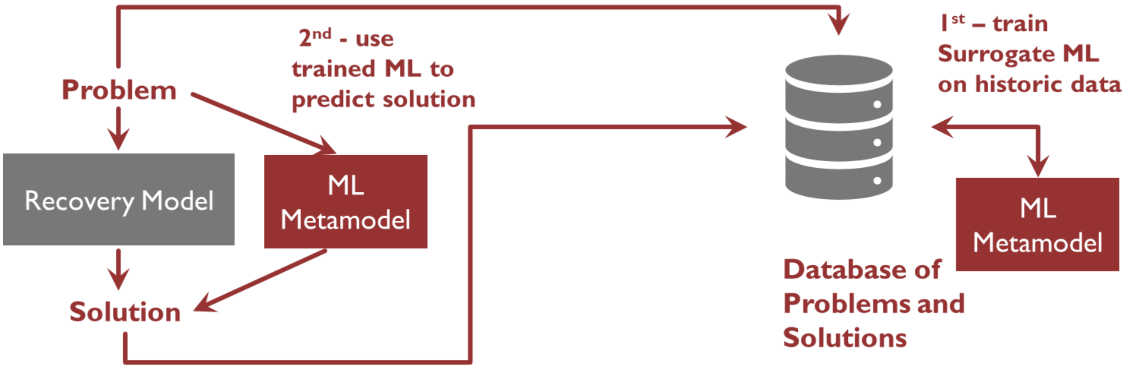

Problem
2nd - use trained ML to predict solution
Recovery Model
ML Metamodel
Solution
1st – train Surrogate ML on historic data
ML Metamodel
Database of Problems and Solutions

Figure 4.4. High level representation of the ML metamodel methodology

Using this approach, it is also possible to use more complex optimization models to solve the recovery problem. This is of special interest if we want to use an integrated schedule recovery model that typically has prohibitively long computational times, and hence cannot be used live in an AOCC. This approach decouples the running time of the models from the execution time during operations. In practice, we can run more advanced optimization models on the set of problems present in the historic database and save the high quality solutions. This new set of solutions, should outperform, at least in theory, the solutions from the original optimization model that was used live by the AOCC. These will then be used to train the ML metamodel. Once the ML metamodel is trained so it returns solutions with acceptable level of accuracy the ML metamodel can be implemented live into the AOCC systems to generate better solutions, at least in theory, than the original schedule recovery model.

The 1st part consist of training the ML model on historic data of optimization problems and solutions. In other words, using the problem parameter matrices, we estimate an ML model, so it accurately predicts the solutions.

A popular way to use machine learning for optimization problems is as heuristic. View seminal works applied to the Traveling Salesman problem like Kirkpatrick, (1984) using an simulated annealing and Geem and Kim, (2001) using an genetic algorithm for the same problem. In these cases the ML models are used as a heuristic approach to substitute the optimization approach without using any of the data generated by the optimization model. At best, the solutions of the optimization approach are used as benchmarks to gauge the quality of the solutions of the ML model. Although such a ML model does not guarantee optimal solutions, ML models have been gaining popularity.

An alternative, proposed here, is to train the ML model using as features the problem itself, in the forms of the problem's cost and constraints matrices, and as labels the solution vector computed by the optimization solver. This would combine the strengths of both models. A properly trained ML model would provide high quality solutions since the ML model would be trained using the true optimal solutions, and these solutions will be provided extremely fast.

In a matrix representation of a linear, integer or mixed integer optimization problem (the features), first row represents the objective function, while each of the other rows represents a constraint. Each column, except the last one, represents a decision variable, while the last column represents the right-hand side of the problem. Finally, the solution (the labels) is given as one value for each variable, which can be thought of as on extra row. However, the dimensions of feature matrix and the labels vector will differ depending on the problem. Also, the problem structure differs considerably from the image recognition problems that have been widely studied in the last several years where most advances have been made using deep artificial neural networks. These models will not work here, at least not directly, since the columns and rows can be interchanged in an optimization problem's matrix representation without affecting the problem itself. Any model that relies on neighborhood of the input data will have difficulties in producing meaningful results. This means that the first step is to create a structured, i.e. standardized, way of representing the problems, with varying sizes and interchangeable columns and rows, so that we are able to train an ML metamodel on them. To get started with this task we first briefly recall the canonical form of a linear problem:

$$Minimize\ \ c^T x \tag{4.1}$$

$s.t.$

$$Ax \leq b \tag{4.2}$$

$$x \geq 0 \tag{4.3}$$

Linear problems can be maximization problems or minimization problems depending on the objective. This example problem (denoted by (4.1)-(4.3)) is a minimization problem. x is the vector of decision variables, and c is a vector of cost coefficients. b is a vector of the *right hand side* (rhs) coefficients. Both, c and b are known vectors. Matrix A is a matrix of coefficients that are present in the constraints given by (4.2). A is also a known matrix. Finally (4.3) requires x variables to be greater or equal to zero. Now if we write this problem in the matrix form, we obtain the following:

$$minimize \quad \begin{bmatrix} c_1 & \cdots & c_n \end{bmatrix} \begin{bmatrix} x_1 \\ \vdots \\ x_n \end{bmatrix} \tag{4.4}$$

$s.t.$

$$\begin{bmatrix} a_{11} & \cdots & a_{1n} \\ \vdots & \ddots & \vdots \\ a_{m1} & \cdots & a_{mn} \end{bmatrix} \begin{bmatrix} x_1 \\ \vdots \\ x_n \end{bmatrix} \leq \begin{bmatrix} b_1 \\ \vdots \\ b_m \end{bmatrix} \tag{4.5}$$

$$\begin{bmatrix} x_1 \\ \vdots \\ x_n \end{bmatrix} \geq \begin{bmatrix} 0 \\ \vdots \\ 0 \end{bmatrix} \tag{4.6}$$

Here m is the number of constraints and n is the number of decision variables. This representation of the problem allows us to exploit its structure and train a ML metamodel on several optimization problems. The known coefficients will be used as input features, and the unknown as output labels. Consider now the set of all problems $P = \{p_1, \dots, p_k\}$, where k is the number of problems p_i in the database. Consequently, we have an additional dimension for the matrix form given by (4.4) to (4.6). Problem p_i in set P is given by (4.7) to (4.9).

$$minimize \quad \begin{bmatrix} c_{1p_i} & \cdots & c_{np_i} \end{bmatrix} \begin{bmatrix} x_{1p_i} \\ \vdots \\ x_{np_i} \end{bmatrix} \tag{4.7}$$

$s.t.$

$$\begin{bmatrix} a_{11p_i} & \cdots & a_{1np_i} \\ \vdots & \ddots & \vdots \\ a_{m1p_i} & \cdots & a_{mnp_i} \end{bmatrix} \begin{bmatrix} x_{1p_i} \\ \vdots \\ x_{np_i} \end{bmatrix} \leq \begin{bmatrix} b_{1p_i} \\ \vdots \\ b_{mp_i} \end{bmatrix} \tag{4.8}$$

$$\begin{bmatrix} x_{1p_i} \\ \vdots \\ x_{np_i} \end{bmatrix} \geq \begin{bmatrix} 0 \\ \vdots \\ 0 \end{bmatrix} \tag{4.9}$$

For any problem p_i in the historical database, all vectors and matrices, including the solution vector x_{p_i}, are known. Now the next step is to train the ML metamodel on this dataset and predict a new solution vector for an unseen problem $p*$ for which we are not given the values of the vector x_{p^*} . If we have several optimization problems we can think of the overall system of problems as several sheets of problems, all staked together, as illustrated by Figure 4.5.

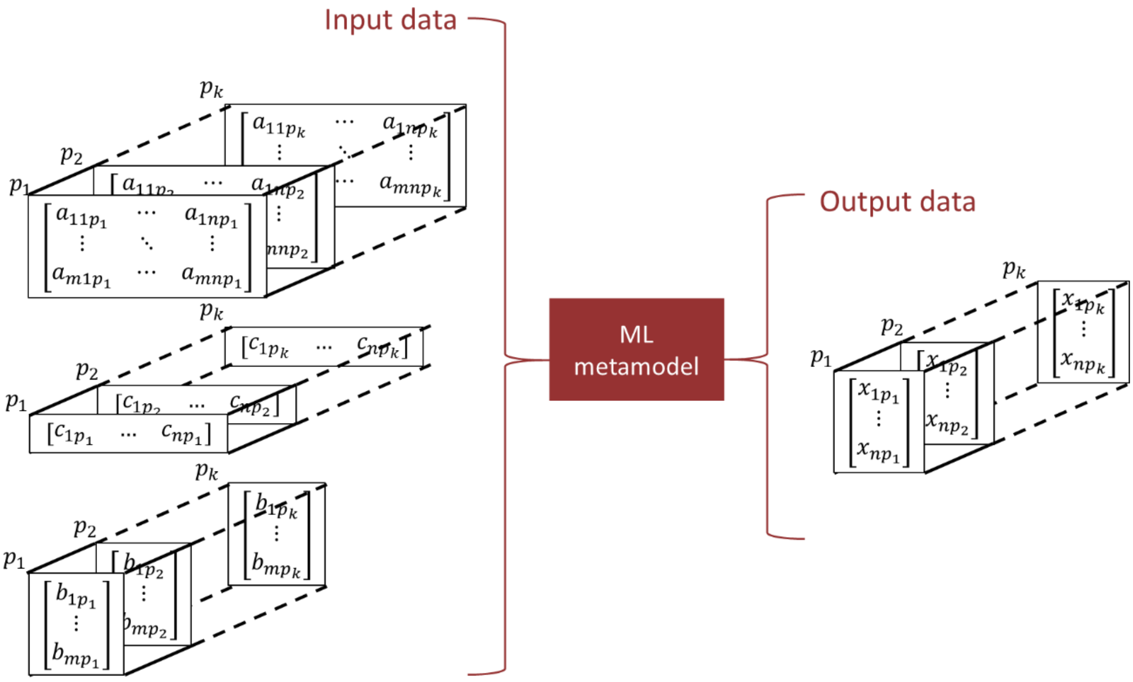


Figure 4.5. Input-output pairs for ML metamodel training

Figure 4.5 illustrates that the ML metamodel will be trained using matrix A and vectors b and c as inputs, and the solution vector x as output for each p_i. Each vector or matrix of a problem p_i can be understood as a sheet (layer) inside a cube with depth equal to k. Training the metamodel on this set of information will lead to the ML metamodel eventually "understand" the patters that exist in the data. For the optimization problem and the ML metamodel's perspective the inputs are known. The outputs are only known from the ML metamodels perspective for every $p_i \in P$. Yet the metamodel does not know beforehand the solution vector of the problem $p*$, the unseen problem and not previously solved by the optimization model. The objective of the second step is to predict the values of the solution vector x for problem $p*$, as depicted in Figure 4.6.

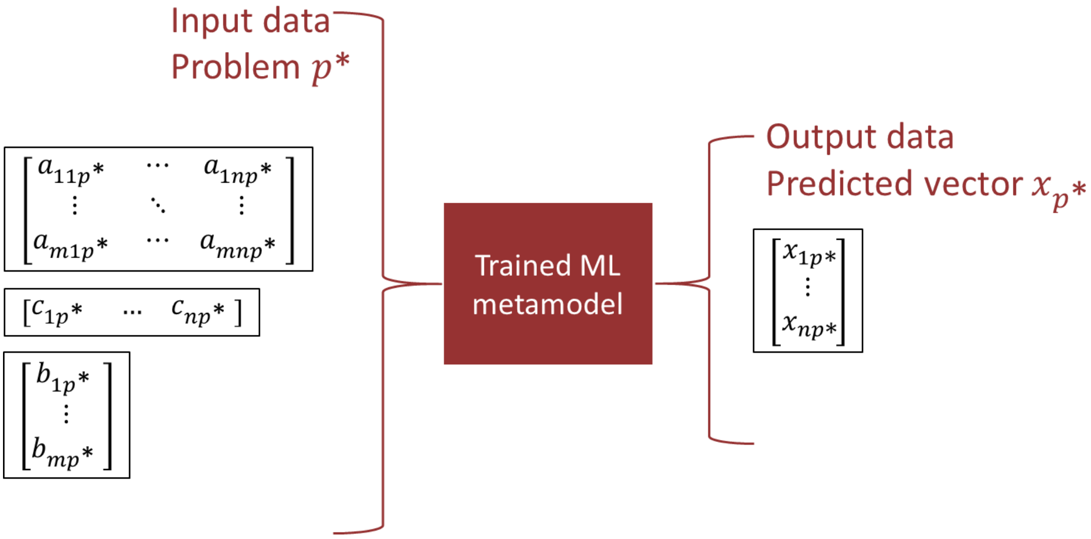

Input data
Problem $p*$
$\begin{bmatrix} a_{11p*} & \cdots & a_{1np*} \\ \vdots & \ddots & \vdots \\ a_{m1p*} & \cdots & a_{mnp*} \end{bmatrix}$
$[c_{1p*} \quad \cdots \quad c_{np*}]$
$\begin{bmatrix} b_{1p*} \\ \vdots \\ b_{mp*} \end{bmatrix}$
Trained ML metamodel
Output data
Predicted vector x_{p*}
$\begin{bmatrix} x_{1p*} \\ \vdots \\ x_{np*} \end{bmatrix}$

Figure 4.6. Predict solution vector x_{p^*} based on input data of problem p^*

This naïve would be directly applicable to a set of problems that have the exact same size as each other. But if either m or n (or both) vary across the problems within P, then this approach is not directly applicable. First one needs to find the maximum values of both m and n across all $p_i \in P$, and then add extra columns or rows filled with zeros to the dataset to make all problems artificially become the same size as that given by the maximum values of m and n. All problems p_i will be of the same size after this adjustment. However, it's not enough to simply add new columns and rows, because the types of constraints must be respected as well. Furthermore, special attention must be given if there is more than one type of variables. This implies that we must attend to the number of variables and constraints and the fact that they are interchangeable. Each combination of constraints and variables needs to have a fixed sized *window* that is equal or greater than the maximum size in both dimensions: number of rows and number of columns.

We will explain this concept further using an example. Consider an example problem p_i, with two types of variables, x and y and two types of constraints, $cntr1$ and $cntr2$. Consequently, matrix A has two rows, as seen in (4.11).

$$minimize \quad \begin{bmatrix} c_1 & c_2 \end{bmatrix} \begin{bmatrix} x_1 \\ y_1 \end{bmatrix} \tag{4.10}$$

$s.t.$:

$$\begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix} \begin{bmatrix} x_1 \\ y_1 \end{bmatrix} \leq \begin{bmatrix} b_1 \\ b_2 \end{bmatrix} \tag{4.11}$$

$$\begin{bmatrix} x_1 \\ y_1 \end{bmatrix} \geq \begin{bmatrix} 0 \\ 0 \end{bmatrix} \tag{4.12}$$

For this example problem we know that for the whole set of problems P, the maximum n for each variable type is equal to two. Meaning the largest problem has the following set of variables: x_1, x_2, y_1 and y_2. As said before each constraint is dependent on the size of variable x. Thus, each constrain type has two constrains in the largest problem (i.e. for the largest p_i, matrix A has in total four columns and four rows). We need to code this accordingly into the data so the ML metamodel will not have problems reading it. Thus, the input data for this example problem would be:

$$\begin{bmatrix} a_{11} & 0 & a_{12} & 0 \\ 0 & 0 & 0 & 0 \\ a_{21} & 0 & a_{22} & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}, \tag{4.13}$$

$$\begin{bmatrix} c_1 & 0 & c_2 & 0 \end{bmatrix}, \tag{4.14}$$

$$\begin{bmatrix} b_1 \\ 0 \\ b_2 \\ 0 \end{bmatrix} \tag{4.15}$$

And the output is given in (4.16):

$$\begin{bmatrix} x_1 \\ 0 \\ y_1 \\ 0 \end{bmatrix} \tag{4.16}$$

The optimization problems can vary in number of rows (constraints) and columns (variables). Also, the columns and rows are interchangeable, the position is not fixed. A structured way to represent the data in each instance (for each p_i) is needed. The way to achieve this is creating smaller window inside the existing data and divide the inputs into standardized sets. In this example we had two constraints, $cntr1$ and $cntr2$, and each constrain was dependent on size x. Thus, the largest A matrix is a matrix of 4×4. This size is the base of our standardized input format for the ML metamodel. We divide the inputs for matrix A into small windows: “constrain ↔ variables”. Resulting in four small *windows* ("$A_{cntr1} \leftrightarrow x$", "$A_{cntr2} \leftrightarrow x$"," $A_{cntr1} \leftrightarrow x$" and "$A_{cntr1} \leftrightarrow x$") for A matrix as depicted in figure 4.7. Similar proceeding is necessary to standardize the input format for the vector c and b.

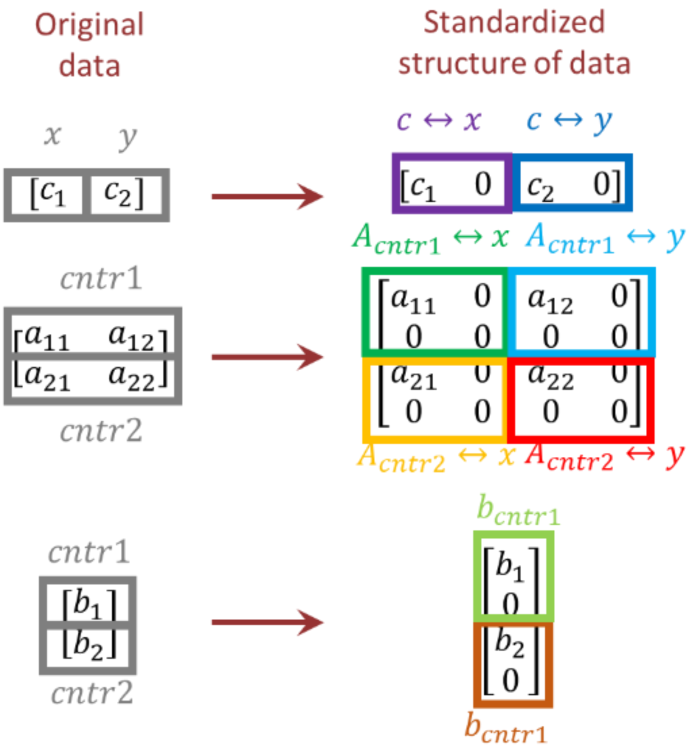

Original data
Standardized structure of data
x y
$[c_1 \; c_2]$
$c \leftrightarrow x$ $c \leftrightarrow y$
$[c_1 \; 0]$ $[c_2 \; 0]$
$A_{cntr1} \leftrightarrow x$ $A_{cntr1} \leftrightarrow y$
$cntr1$
$\begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix}$
$cntr2$
$\begin{bmatrix} a_{11} & 0 \\ 0 & 0 \end{bmatrix}$ $\begin{bmatrix} a_{12} & 0 \\ 0 & 0 \end{bmatrix}$
$\begin{bmatrix} a_{21} & 0 \\ 0 & 0 \end{bmatrix}$ $\begin{bmatrix} a_{22} & 0 \\ 0 & 0 \end{bmatrix}$
$A_{cntr2} \leftrightarrow x$ $A_{cntr2} \leftrightarrow y$
b_{cntr1}
$cntr1$
$\begin{bmatrix} b_1 \\ b_2 \end{bmatrix}$
$cntr2$
$\begin{bmatrix} b_1 \\ 0 \end{bmatrix}$
$\begin{bmatrix} b_2 \\ 0 \end{bmatrix}$
b_{cntr1}

Figure 4.7. Standardized input of data for the ML metamodel

Each *window* can now directly be connected to the ML metamodel having always the same size and structure for every $p_i \in P$ and unseen p^*. This allows to capture the data in a format so it's not affected by the interchangeability of the columns and rows. This standardization method also solves the problem of p_i with missing constraints. These windows are possible to connect directly to the ML metamodel and if there is data missing due to smaller size, missing constrains, missing variables, etc. the input values are just set to zero inside the corresponding *window*.

As said this coding of the information allows the metamodel to be trained on problems p_i that have different sizes, i.e., different number of constraints and/or variables. Also, this allows to only feed part of the input vectors or matrices if one wished to do so. For instance, for the previous example given by (4.13) to (4.16), let us assume that we only want to predict the values of the x variable and during training it was shown that only the first type of constraints ($cntr1$) is enough to accurately predict the value of the x vector for all p_i. Then, the input for this particular case for the A matrix we only would need as input windows: "$A_{cntr1} \leftrightarrow x$","$c \leftrightarrow x$" and "$b_{cntr1}$", given by (4.17) to (4.19):

$$\begin{bmatrix} a_{11} & 0 \\ 0 & 0 \end{bmatrix}, \tag{4.17}$$

$$[c_1 \quad 0], \tag{4.18}$$

$$\begin{bmatrix} b_1 \\ 0 \end{bmatrix} \tag{4.19}$$

And the output, the x vector for this particular p_i would be as given by (4.20):

$$\begin{bmatrix} x_1 \\ 0 \end{bmatrix} \tag{4.20}$$

In this example, we only show how to apply this methodology to a linear problem, but this methodology is also equally applicable to integer problems (IP) or mixed integer problems

(MIP). To be able to train an ML metamodel additional steps might be needed to guarantee integrality of the solutions though.

This framework is not limited to data from the optimization model. Additional data can be used as input to help improve the accuracy of the ML metamodel that is not present in the optimization problem but available for all p_i, or at least for most of them. Likewise, transformation of the data can also be used; for instance, normalization of the input data and/or output data can be explored for its predictive quality using this framework as well.

4.4 Proof of concept case study

To show the feasibility of the concept presented in the previous section, we designed a case study to serve as proof of concept. This is a feasibility study where we have chosen one type of ML model to be used as metamodel: Artificial Neural Networks (ANN). This is one of the most flexible machine learning model types that exists today. In the last few years, great advancements have been made in this very dynamic research topic. These ANNs are most prominently used in the field of image recognition, and are popular in academic research and the industry alike.

4.4.1 Historical Data

As explained in Section 4.2, ML models need large sets of data to be able to identify patterns in those data sets. This is especially the case with artificial neural networks which are known to be very data-intensive to be effective. The kind of data that is needed is not easy to obtain from a public database, nor do we have access to such data from an AOCC. One could try to infer such data from public datasets. However, this would be extremely challenging given the large number of missing data points. Given our need for a large dataset such endeavor would be also an onerous task.

A better approach would use a simulation model to simulate the operations of an airline performing schedule recovery. This recovery model would serve as proxy for the AOCC. For this purpose, we used the AOSP and simulated one day of operations for Alaska Airlines (AS). This is the simulation model platform by us and presented in Chapter 3 using data from the case study in that chapter, given in Section 3.5. It allows simulating any airline's operations across the world due to an integrated GIS environment (see Figure 3.5) that enables the user to have a detailed understanding for the simulation as it runs. The model also propagates delays from one flight to another via shared resources, such as, crews and aircraft. The ability to recover schedules is of utmost importance to us, and is provided by the AOSP.

We simulated Alaska Airlines' operations to create a database of problems and solutions. Figure 4.8 depicts the network operated by Alaska Airlines that was used in our simulation. We derived the daily operations of the airline based on the data extracted from the Airline on-Time performance database from the Bureau of Transportation Statistics, BTS (2016). This information from this database also allowed calibration of the delay distributions. For further information, please refer to the case study in Chapter 3, Section 3.5. In total, we repeated one day of operations 1000 times resulting in 544,000 total simulated flights.

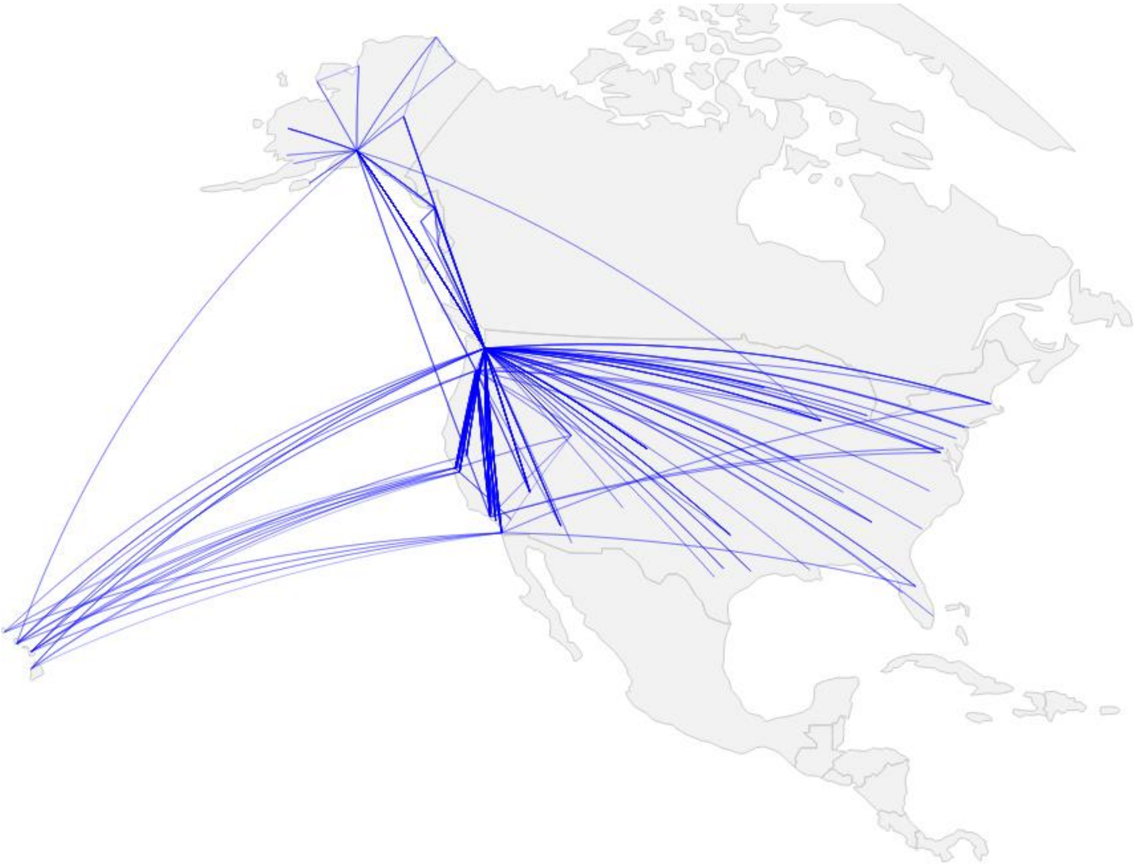


Figure 4.8. Network of Alaska Airlines used in the simulation

As proxy for our AOCC, we use a partially integrated schedule recovery model. This recovery model is an integrated part of the simulation framework and is able to automatically recover schedules if the need arises to do so. It is based on the DSTAR model developed by Abdelghany et al. (2008). This model has the capability of delaying and canceling flights, and swapping aircraft and crews between flights. Across the 1,000 repetitions, the recovery model was executed 1,900 times. Hence, our ML metamodel has a database of 1,900 problem-solution pairs to be trained on. The delay distribution is summarized in Table 4.1. This data is the same as what was presented earlier in Chapter 3, Table 3.8, but here we only show the data that is necessary for this case study. These are the propagated delays, and not the root delays, added by the simulation model stochastically. Propagated delays that occur when the time buffers are not able to absorb delays and result in propagation of delays trough the flights that are time depended.

Table 4.1. Propagated delays for simulated flights

Prop. delay range (min)	(%)
<1	88.92
1-15	6.72
15-60	2.68
60-300	1.66
>300	0.01
Canceled	0.00

The DSTAR model was run each time the AOSM generated a notification of infeasibility of a flight that could not be flown on schedule. This flight and all its resources together are considered as the “problem flight”. Additionally, DSTAR model’s pre-processing selects a set of flights according to predefined rules and adds those resources, aircraft and crews to the set of flights that are allowed to be modified to recover the schedule. These additional flights themselves do not have any issue with their schedules, but could be adjusted to try to minimize the overall costs by giving the recovery model more options to recover the schedules. Across the 1,900 instances, the recovery model added a maximum of 7 flights to the set of flights to be recovered. Since most of the non-problematic flights had small or no delays added to them, we decided to focus only on the 1st flight, the “problem flight”. This solves another potential issue associated with the ordering of the added flights to be modified by the schedule recovery model. Then the ordering could be different from instance to instance. The ML metamodel could have difficulties in the training phase if the orders are different in each instance. Figures 4.9 and 4.10 show the histograms of the delays added by the model to the problem flights and the cost of all 1,900 recovery problem solutions, respectively.

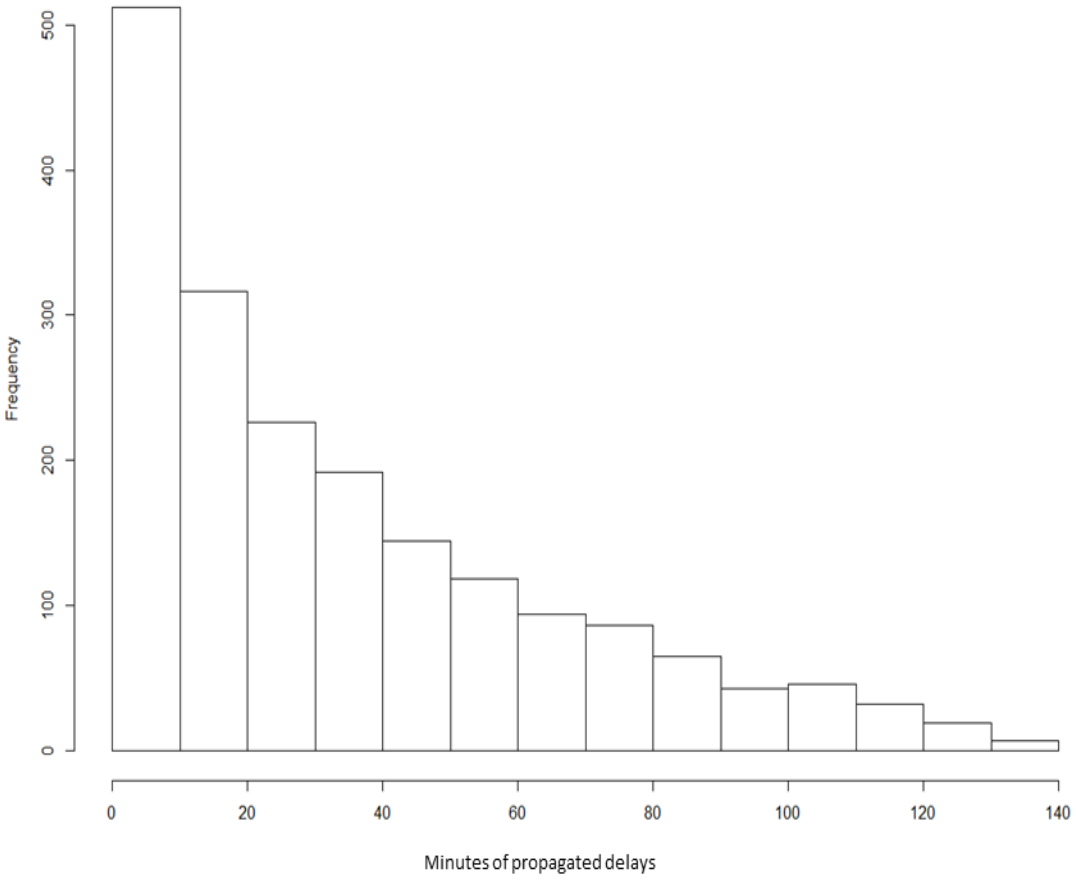


Figure 4.9. Histogram of delays of the problem flight

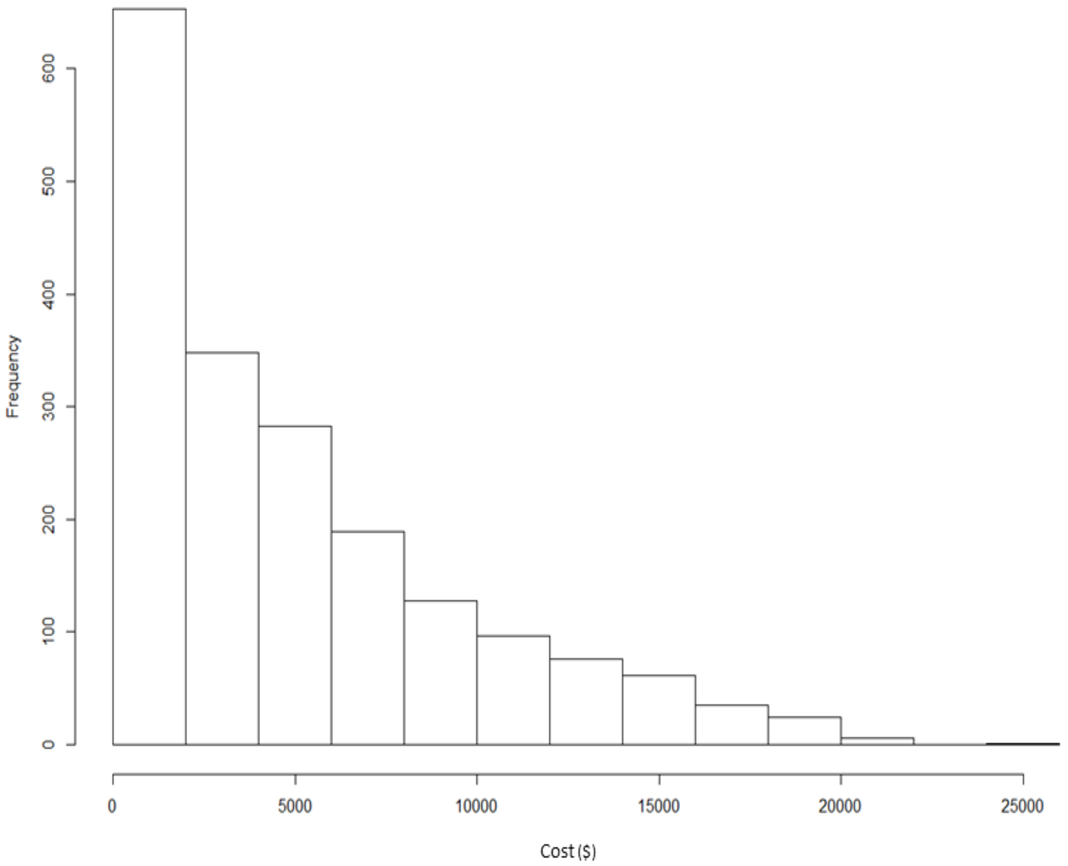


Figure 4.10. Histogram of costs of all 1900 instances

If we compare the histogram of the delay of the first flight with the histogram of the costs, we see that they are very similar. This happens because the cost is almost a linear transformation of the delays. The objective function of the DSTAR schedule recovery model is a sum of three independent costs: resource swapping, minutes of delays and cancelations. Since there weren't any cancelations and swapping of resources was rare (please refer to Chapter 3 to understand the reason behind the inexistence of cancelations with DSTAR), the delays are the main contributors, if not the only contributors, to the total cost. The only difference is that the cost also included the delays of the other flights that the recovery model delayed. However, problem flight was the most delayed one in most cases, subsequently dominated the contribution to the total cost. Since we observed a better performance of our approach when predicting the delays, likely due to the fact that delays are only from the problem flight and cost includes all flights considered by the recovery model, we only show the results of ML metamodels when trying to predict the delays added by DSTAR. Figure 4.11 shows the number of problem flights, grouped together by the planned departure time at the gate:

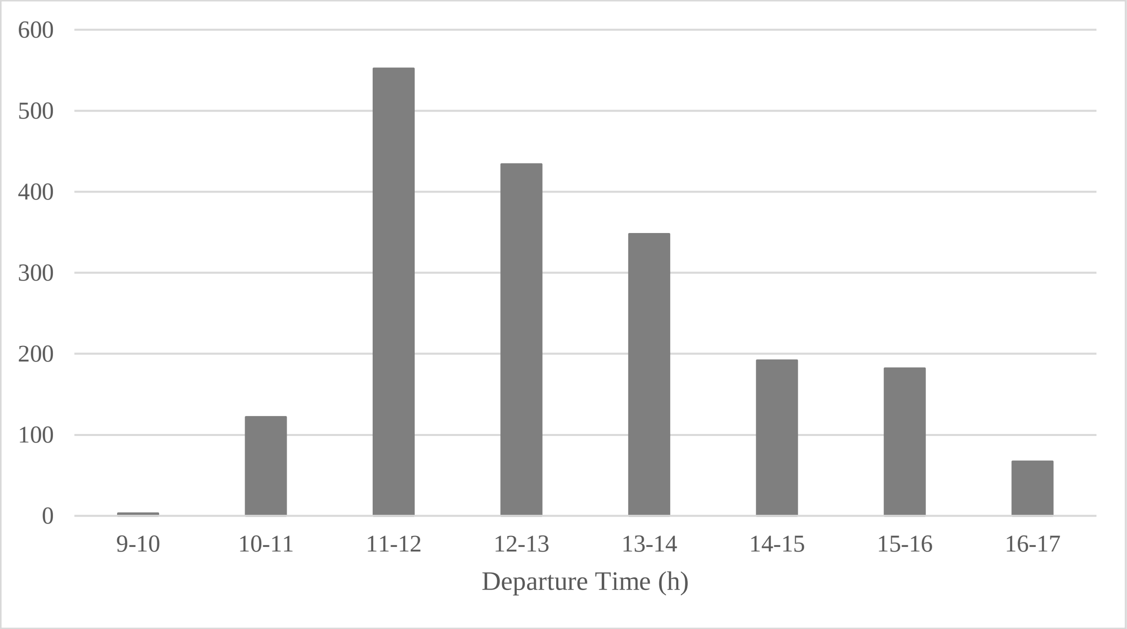


Figure 4.11. Number of problem instances per scheduled departure time

As seen from Figure 4.11, in the beginning of the day there are no major issues. Most problem flights appear between 11:00-12:00 hours. This number gradually reduces over the day. The next graph, Figure 4.12, a set of boxplots of delays, shows us the distribution of the delays added by the DSTAR model grouped together by departure hour. We see that the medians are relatively low, most between twenty and forty minutes. However, the distributions are skewed, presenting long tails stretching to a delay of over two hours in some cases. This information, i.e. the solution of the DSTAR model, is the ground truth that we will try to predict with the ML metamodels.

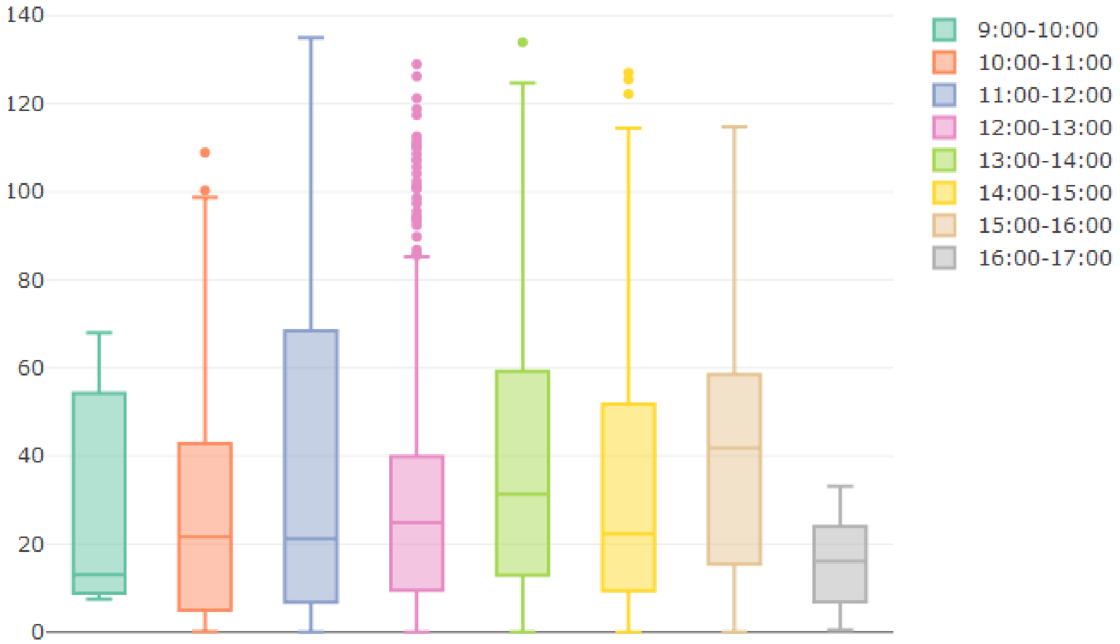


Figure 4.12. Boxplot of delays for each scheduled departure hour

4.4.2 Machine Learning models

As ML model, we decided to use Artificial Neural Networks (ANNs), and specifically a feed forward variant of ANNs. We developed various versions of two types of models: regressions to predict exact values and classifications to predict classes. We used Google's Tensorflow library (Google, 2018) to develop our ANNs. TensorFlow is an open-source software library for machine learning, developed by Google, capable of building and training neural networks to detect and decipher patterns and correlations. Most of the effort was spent on deciding the structure of the ML metamodel (defining hyperparameters like: number of layers, types of layers, types and number of connections, etc.) so they would generate a prediction with acceptable accuracy.

Two types of predictions were made. One is a value prediction, trying to predict the delays of a solution, similar to a regression. The other type is a classification where we predict the group to which the solution belongs, based on predetermined definitions of groups or classes. In the latter case, the ANN tries to predict the right class of the solution. In this case study, we present four models where we focus only on the delays added to the first flight as explained before.

ANN1: Classification - only coefficients from the objective function used as features;

ANN2: Classification - coefficients of objective function and constraints used as features;

ANN3: Regression - only coefficients from the objective function used as features;

ANN4: Regression - coefficients of objective function and constraints used as features;

All training was done with a validation set of corresponding to 20% of the data. This means that in the training phase, the set of 1,900 problems and solutions was divided into two groups where 80% (1,520 instances) were used for training, and 20% (380 instances) were used for validation. This is a standard feature of the TensorFlow library (Google, 2018).

4.5 Results and Discussion

In this subsection, we present the results of the case study, discuss them and compare the performance of the four ANNs.

4.5.1 ANN1

The first model, ANN1, is the simplest one of the four presented here. We divided the solutions into only two classes. To create the classes, we divided the data into two groups; with group 1 corresponding to delays less than ten minutes and group 2 corresponding to delays greater than or equal to ten minutes. The final topology, number of neurons, number of layers, and type of layers for ANN1 are presented in Figure 4.13. In Figure 4.13, we only show the dense layers, while not showing the concatenation and dropout layers, and other minor details. This makes it easier to understand the general topology of ANN1. As a relevant indicator of the size, we can use the number of trainable parameters. ANN1 had a total of 49,926 trainable parameters. Since the size of the input data can vary with problem size, the number of columns of the problem matrix of the optimization problem is dependent on the number of variables. Thus, we need to make sure that we can handle the largest size of input dimensions as described in Section 4.3. We analyzed the dataset of 1,900 runs to obtain the largest dimensions. When the actual problem size is smaller, we filled the remaining parts of the problem matrices with null or empty values. “Input OF X” and “Input OF m” refer to the objective function coefficients of variable X and m respectively, of the DSTAR problem matrix. The X values are essentially the swapping costs of resources and m parameters are the costs of 1-minute of delays. Please refer to Appendix I.a.i to better understand the structure (objective function, constraints, parameters and variables) of the DSTAR model, and also to understand how the cost parameters were computed.

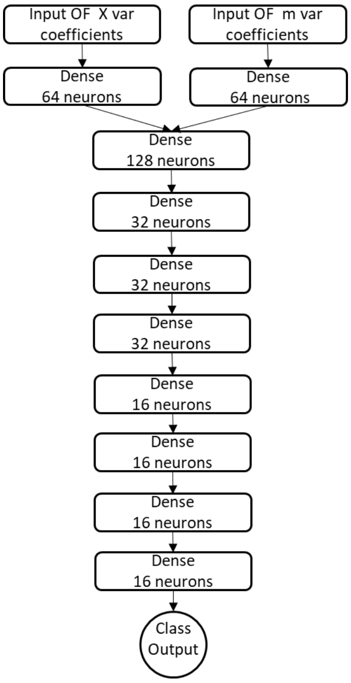


Figure 4.13. ANN1 Structure

We observe that ANN1 tends to overestimate, predicting more than the actual number of flights to have greater than or equal to ten minutes of delay (which corresponds to group 2). That was expected since the initial set of data was unbalanced in favor of group 2. Hence ANN1 has a higher tendency to return as solution group 2 labels (see Table 4.2). One can think that the information gained is limited. The total accuracy of the model is 76.21, Table 4.3. However, if we analyze the Receiver Operating Characteristic curve (ROC curve) we see a different picture, Figure 4.14. This plots the False Positive Rate (FPR) on the X axis and the True Positive Rate (TPR), also known as Sensitivity, on the Y axis. The FPR closely related to the concept of Specificity; FPR equals to one minus the Specificity.

Table 4.2. Confusion matrix of ANN1

		Prediction	
		Group 1	Group 2
Truth	Group 1	161	351
	Group 2	101	1287

Table 4.3. Accuracy of ANN1

	False	True
number	452	1448
%	23.79	76.21

The diagonal in this graph represents a random choice between the two classes. Hence any model with the ROC curve below the diagonal would represent a performance poorer than a coin toss. This diagonal represents a lower bound for any model we want to test. A perfect model that would always classifies correctly would have the first point of the curve at (0,1) continuing a horizontal line afterwards. One common indicator of fitness of a model is the Area Under Curve (AUC). A perfect model would return an AUC equal to 1 and random choice would return an AUC value of 0.5. As we can see from Figure 4.14, ANN1 has an AUC value of 0.7245 that is good performance already. Note that the curve is color coded. This color code is according to the scale on the right-hand side in Figure 4.14. It represents the value that the ANN returns. Meaning it represent the certainty that the model had that the correct label would be Group 2 (a value between 0 and 1) this indirectly also tells us the certainty that the model had that model had it would be Group 1, being the certainty for Group 1 equal to one minus the certainty for Group 2.

If we analyze a selected point on the curve, we can extract interesting information about the performance. Consider the three points in Table 4.4. Each point on the ROC curve represents 3 values: Model Certainty, FPR and TPR. Model certainty is the parameter that help us decide the best cutoff value. Lower values will consider more of the instances, returning a label for more instances but at the cost of accuracy. Higher cutoff values would lead to higher accuracy but with the cost of more instance without any label given by the model, i.e. more instances with inclusive results.

With this model certainty parameter (cutoff value), we can construct partial accuracy values. For instance, if we consider only values higher as 0.90 of model certainty (cutoff value of 0.90), only a fraction of 0.07 of the total cases are considered. The rest of the 0.93 instances would return as solution “inconclusive”. However, for the 0.07 fraction the partial accuracy, on the fraction, would be 97%. This is a high accuracy but for a very small fraction of the data. With a cutoff value of 0.80 the model would be useful for a fraction of 0.21 with a partial accuracy of 94%, that is a significant increase in fraction for a low cost in accuracy loss. The most interesting value is the certainty level of 0.70. here the model would consider a fraction 0.80 of the cases, only returning inconclusive for 0.20, with a partial accuracy of 83%. This is an encouraging performance for a relatively simple ANN topology attending we only feed data form the objective function to it.

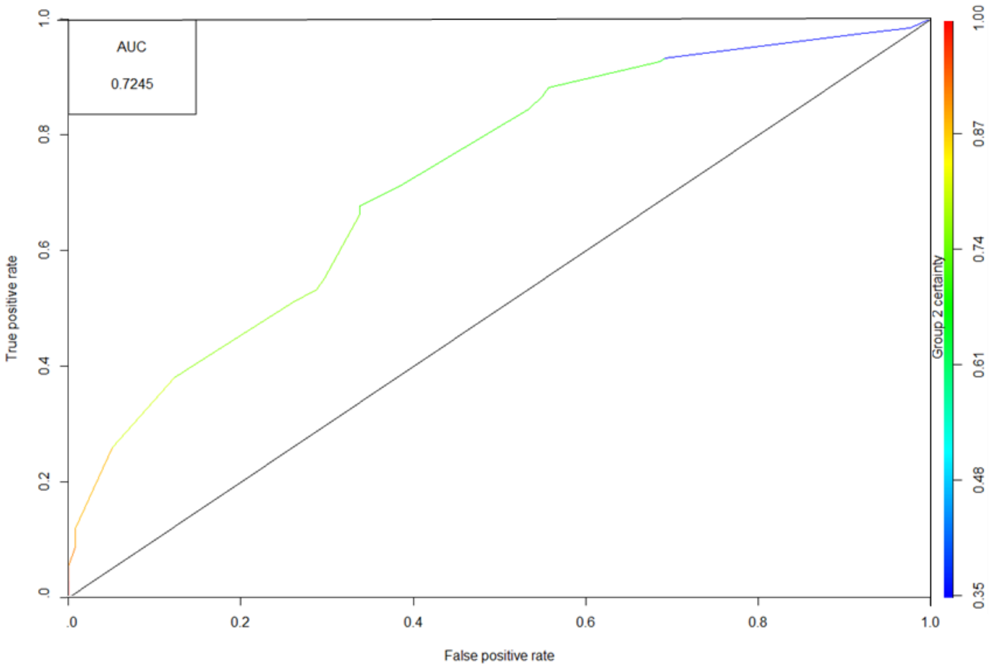


Figure 4.14. ROC curve of ANN1

Table 4.4. Selected ROC points of ANN1

ROC curve point				
Model Certainty	FPR	TPR	Fraction of cases	Partial accuracy
0.90	0.01	0.09	0.07	97%
0.80	0.05	0.26	0.21	94%
0.70	0.56	0.88	0.80	83%

4.5.2 ANN2

For the next model, ANN2, we used additional information present in the problem matrices to try to generate better predictions. This also meant that we needed to change the topology, number and size of layers and the connections between them. The resulting structure is presented in Figure 4.15. Similarly, as with the previous ANN, ANN1 (Figure 4.13), we only show the most important information, input layers, dense layers, the connections between them, and the output neuron, i.e. predicted class.

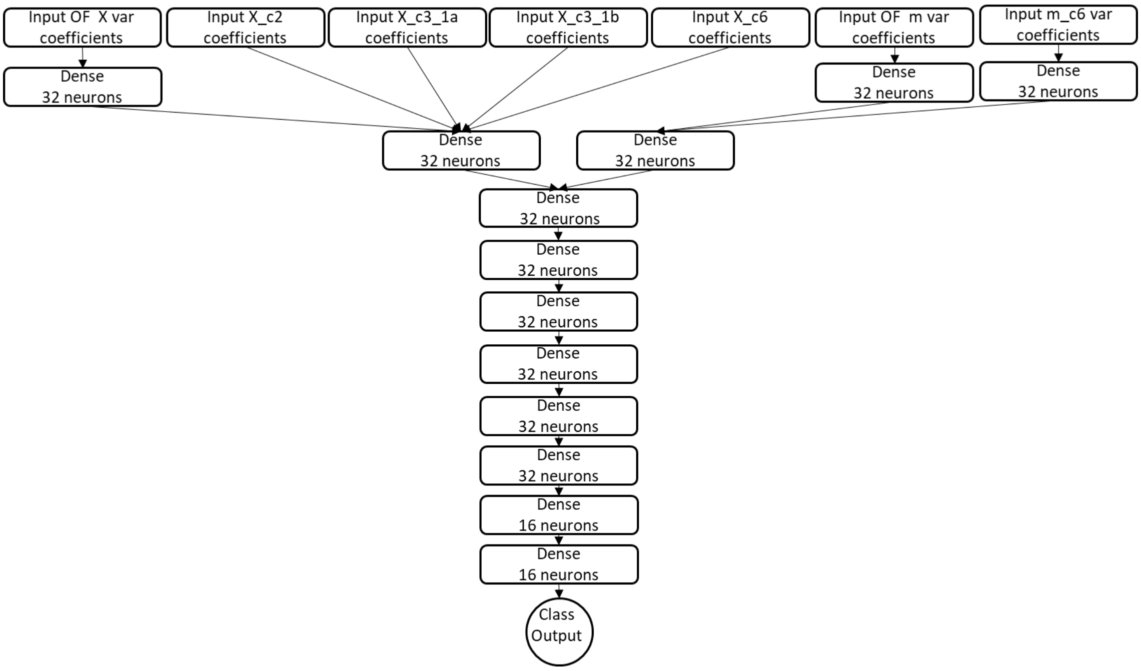


Figure 4.15. Structure of ANN2

Here, besides the objective function parameters of variables X and m, “Input OF X” var coefficients and “Input OF m” var coefficients respectively, we also used information from the constraints. On one side we included the coefficients multiplying the X variable in constraints 2, 3.1a, 3.2b and 6, input layers “X_c2”, “X_c3.1a”, “X_c31b”, and “X_c6” represent the standardized windows as defined as we referred in section 4.3. Note that constraint numbering is according to the original paper by Abdelghany, et al. (2008). For variable m, we also gather the constraint coefficient information for constraint 6 in input layer “m_c6” (i.e. window “m_c6”). We can see that X and m coefficients from the objective function and constrains are first joined together in one dense layer each and then those dense layers merger together into the main trunk of ANN2. Information from the objective function passes an additional dense layer before joining with the information from the constraints, for both variables. ANN2 has 2,200,294 trainable parameters making it much more computationally intensive to train than ANN1.

Table 4.5. Confusion matrix of ANN2

		Prediction	
		Group 1	Group 2
Truth	Group 1	148	364
	Group 2	81	1307

Table 4.6. Prediction accuracy of ANN2

	False	True
number	445	1455
%	23.42	76.58

The added data does not improve the accuracy significantly. We can see this by comparing Tables 4.2 and 4.3 with 4.5 and 4.6. The confusion matrix shows a small decrease in false labeling for both groups, but the total accuracy only increases slightly (less than one percent point increase in comparison with ANN1).

By analyzing the corresponding ROC curve, we have a more complete picture of the performance of ANN2 compared to that of ANN1. First indicator is the AUC that is equal to 0.734 for ANN2, which is a slight increase in comparison with ANN1. Similarly, as with Table 4.4 in case of ANN1, we present the same points of the ROC curve in Table 4.7 for ANN2. If the cutoff value of the certainty level is 0.90 ANN considers a fraction of 0.21 of the cases with a partial accuracy level of 94% this means this outperforms the point in the ROC curve with a certainty of 0.80 for ANN1. Having a similar accuracy level (94%) for a larger fraction, 0.21 for ANN2 and 0.07 for ANN1. However, when we analyze the certainty level of 0.70 the accuracy level of ANN2 is higher for the same level of ANN1 but the fraction of cases is smaller, 0.56 for ANN2 versus 0.80 for ANN1. So, we can say that in some case ANN2 is slightly better than ANN1, but does not consistently dominate

ANN1. Hence, we cannot claim that this added complexity is worth it since the increase in accuracy is limited.

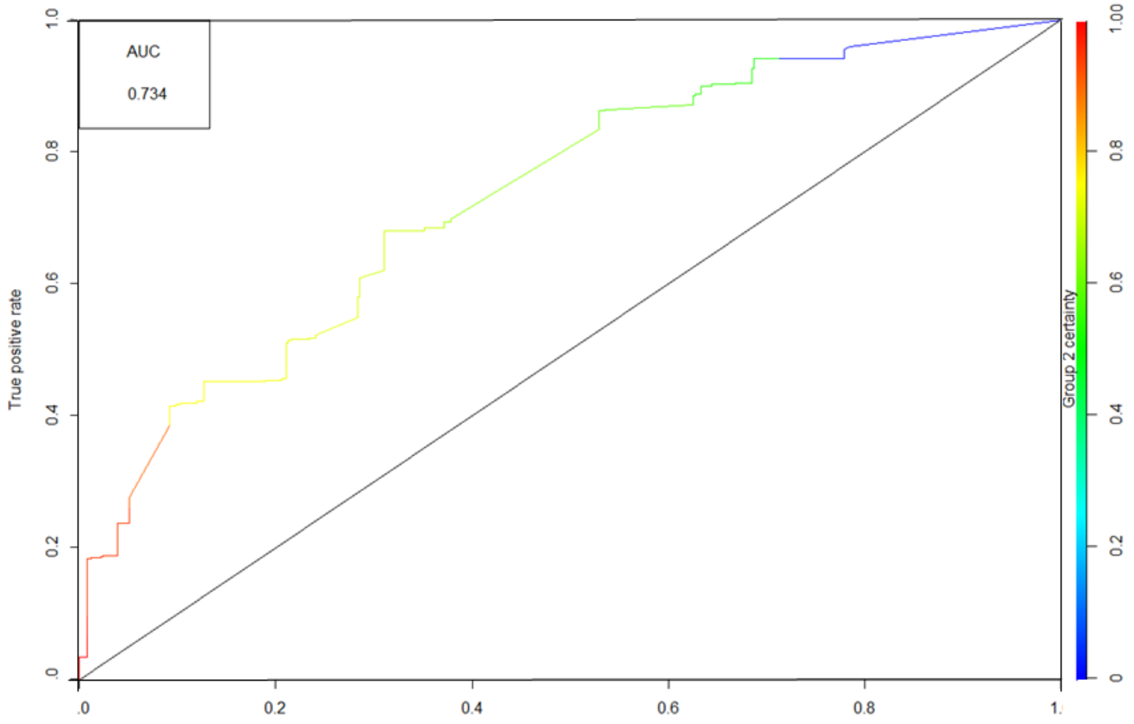


Figure 4.16. ROC curve of ANN2

Table 4.7. Selected ROC points of ANN2

ROC curve point				
Model Certainty	FPR	TPR	Fraction of cases	Partial accuracy
0.90	0.05	0.26	0.21	94%
0.80	0.09	0.39	0.32	93%
0.70	0.31	0.64	0.56	87%

4.5.3 ANN3

ANN3 is similar to ANN1 in terms of its topology. The only difference lies in the last neuron where instead of providing a class as the result, the provided result is a scalar value: ANN3 tries to predict the delay added by the DSTAR schedule recovery model. Figure 4.17 depicts the topology of ANN3.

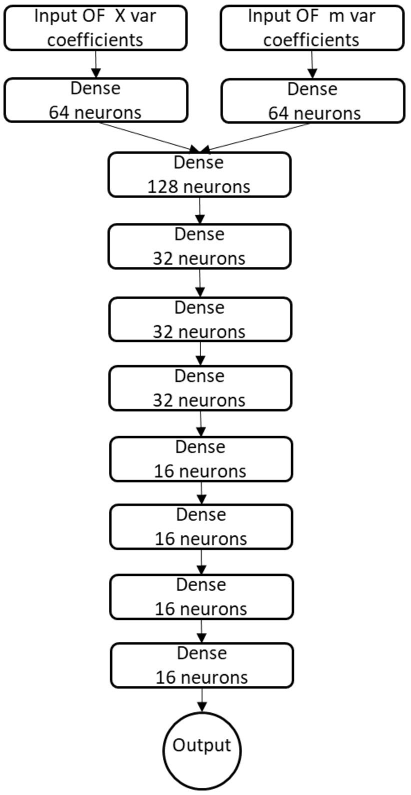


Figure 4.17. Structure of ANN3

Since this is a value prediction, i.e., similar to a regression, there is no model certainty returned by the ANN (unlike for ANN1 and ANN2). The only output is the predicted value of delay added by the DSTAR. Figure 4.18 shows us the violin graph of the absolute differences between the prediction and the actual values, on the Y axis. We can see that most of the predictions have a difference of 25 minutes or lower, with a large number of values being predicted with less than 10 minutes of differences from the ground truth, which is actual solution value of DSTAR.

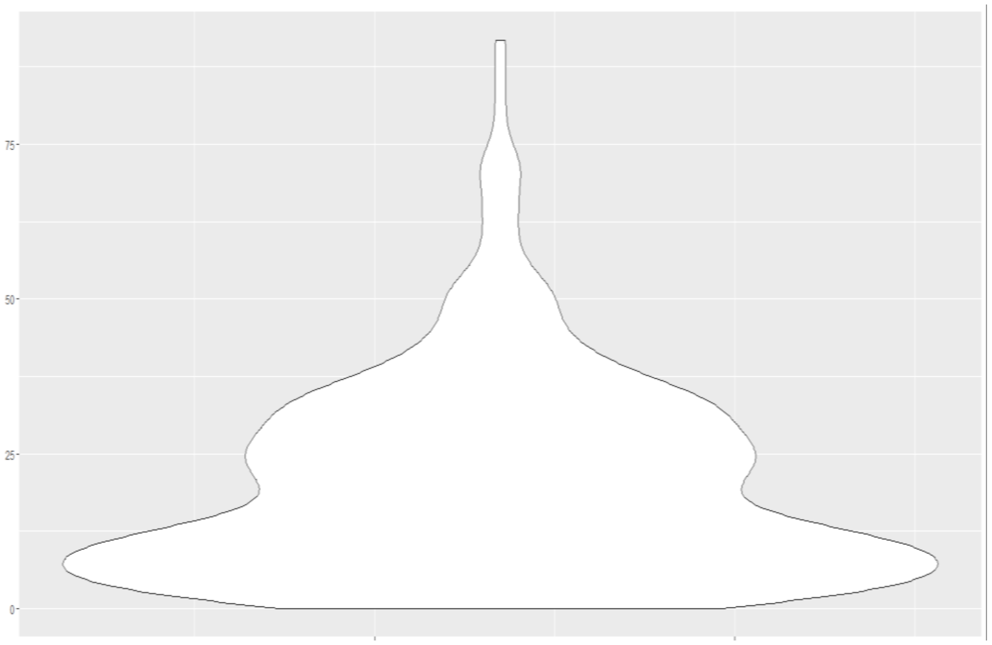


Figure 4.18. Violin plot of the absolute differences between truth and prediction of ANN3

Next, we analyze in more details the prediction depending on the value of the solution, i.e., in terms of the delay added by DSTAR. With this purpose in mind, we generated a boxplot graph of the DSTAR delay divided into bins with ranges of 10 minutes each (see Figure 4.19). We see that for added delays of around 30 minutes, ANN3 tends to overestimate the added delays. For values larger than that the model tends to underestimate the true values. This progression seems to be almost linear with an increase in the DSTAR delay value. This suggests that the model might need more flexibility or that there is a pattern in the data that was not shown to the ANN in the training phase.

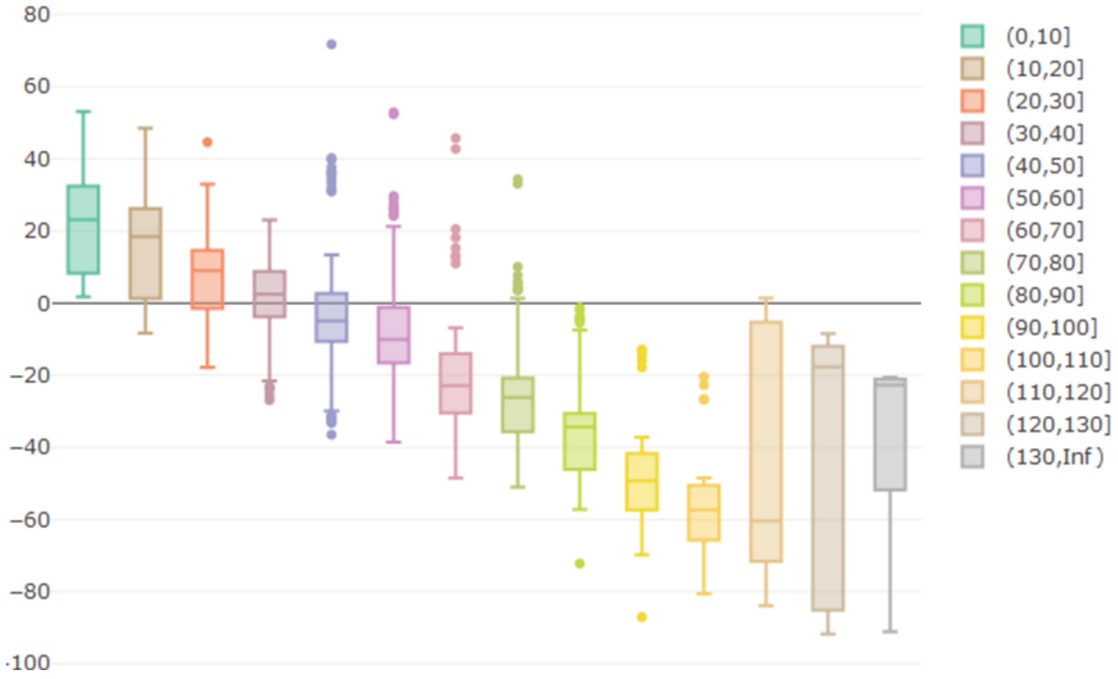


Figure 4.19. Boxplots of the differences between truth and prediction of ANN3, grouped by delay value

4.5.4 ANN4

As we can see from Figure 4.20, the structure of this ANN is similar to that of ANN2. However, after some additional testing, we found that a shallower ANN (one with fewer, but wider, layers with more neurons), had a slighter better performance. It was only marginally better, and hence we cannot say that a wider ANN would necessarily be preferable. Figure 4.21 provides the violin chart of the differences between the prediction versus the DSTAR solution, similar to that in Figure 4.18 for ANN3. We can see that the overall performance is better than ANN3 having more prediction values at a lower difference between them and the truth.

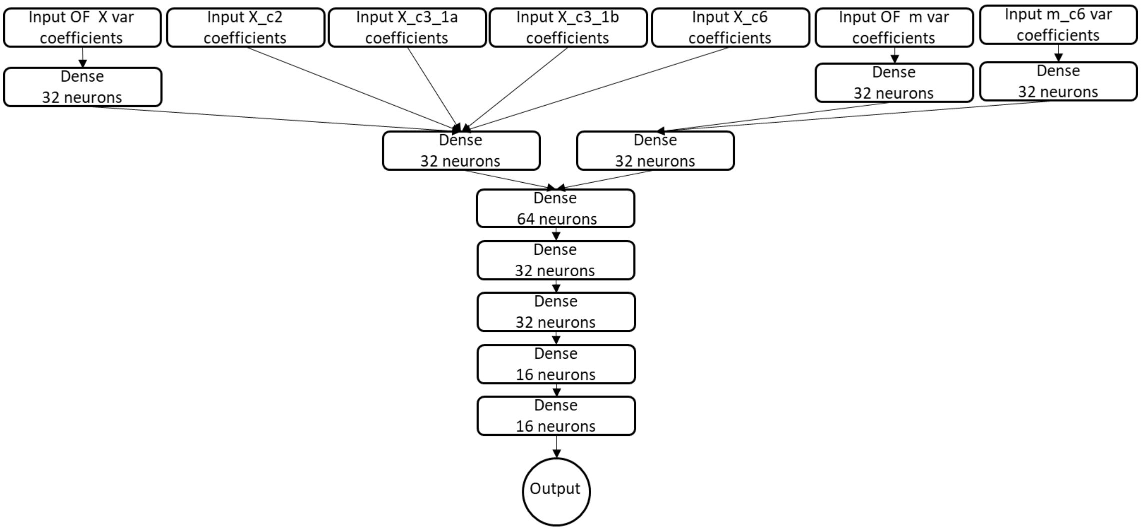


Figure 4.20. Structure of ANN4

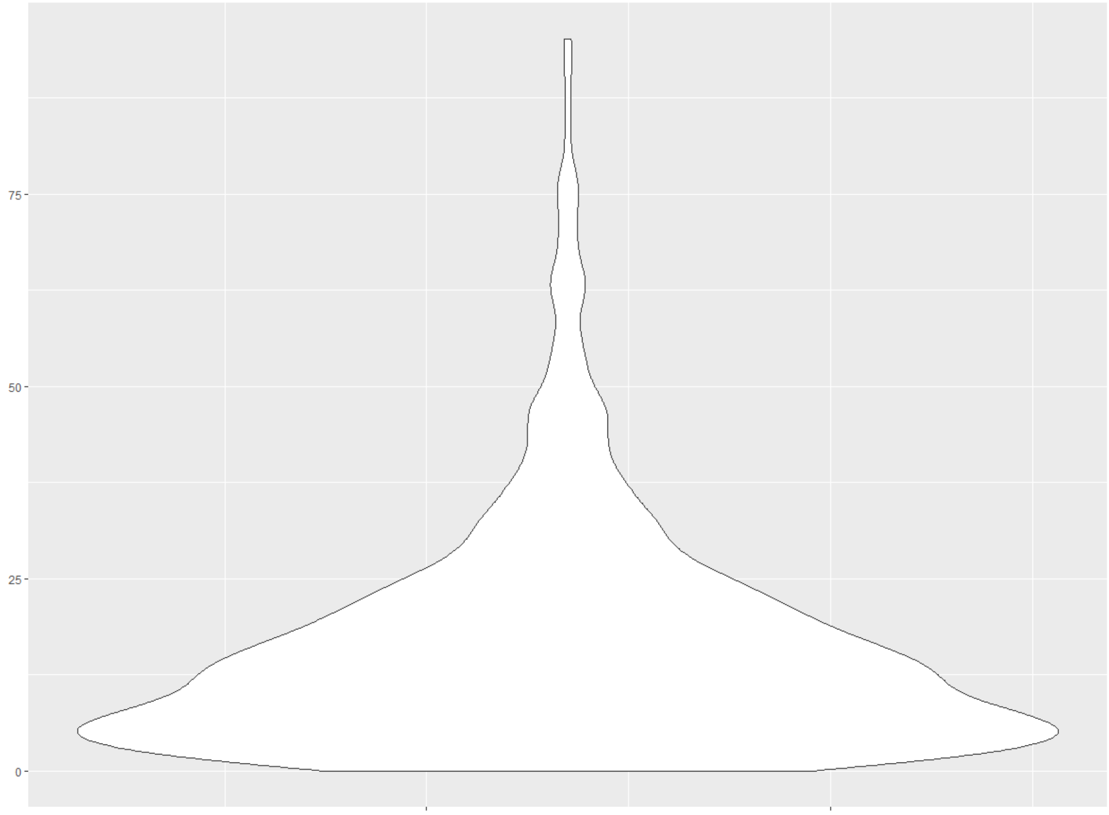


Figure 4.21. Violin plot of the absolute differences between truth and prediction of ANN4

We wanted to see if ANN4 was able to “understand” the aforementioned pattern identified while analyzing the comparative boxplots forANN3. With that purpose in mind, we created a similar graph for ANN4 (see Figure 4.22). Although there is still a small overestimation for small values of delays and underestimation for higher values, the linear trend that we saw with ANN3 (Figure 4.19) is a bit less clear. ANN4 seems to be flexible enough, or is fed the right data, to be able to "understand" that pattern. In this case, the more complex model ANN4 appears to perform better than the less complex one.

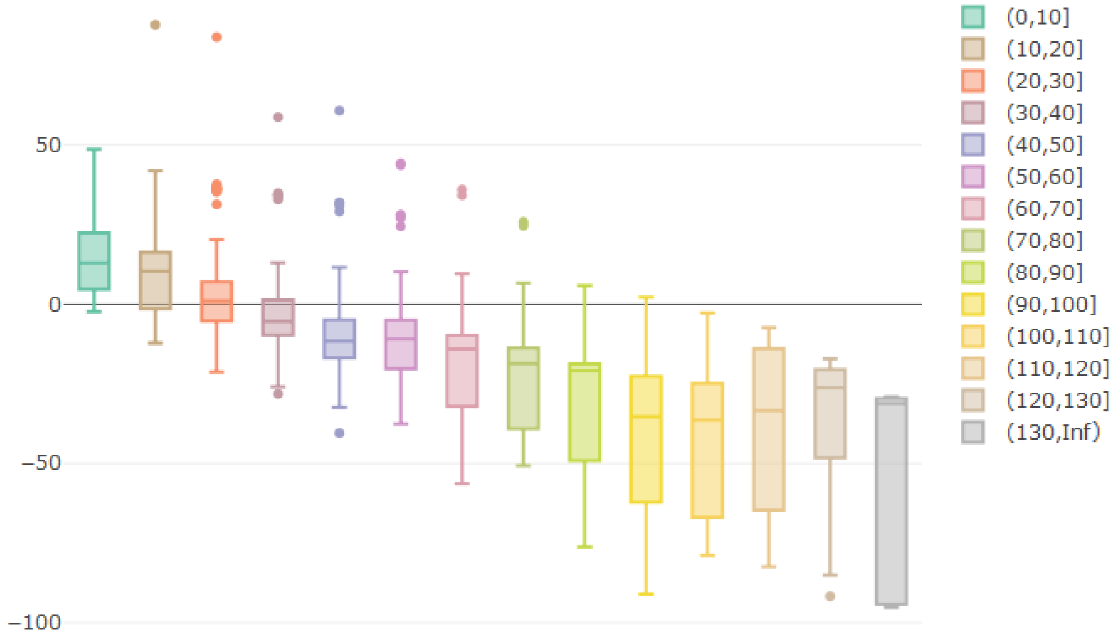


Figure 4.22. Boxplots of the differences between truth and prediction of ANN4, grouped by delay value

To further compare ANN3 and ANN4, we create a graph that depicts the empirical cumulative density function (ECDF) of the absolute differences between prediction and truth for ANN3 and ANN4 (see Figure 4.23). From this graph, we can see that ANN4 outperforms ANN3, by shifting the ECDF to the left side compared to that for ANN3. In this case, when comparing ANN4 vs ANN3, we can argue that the added complexity of feeding more data and having a wider artificial neural network topology is beneficial.

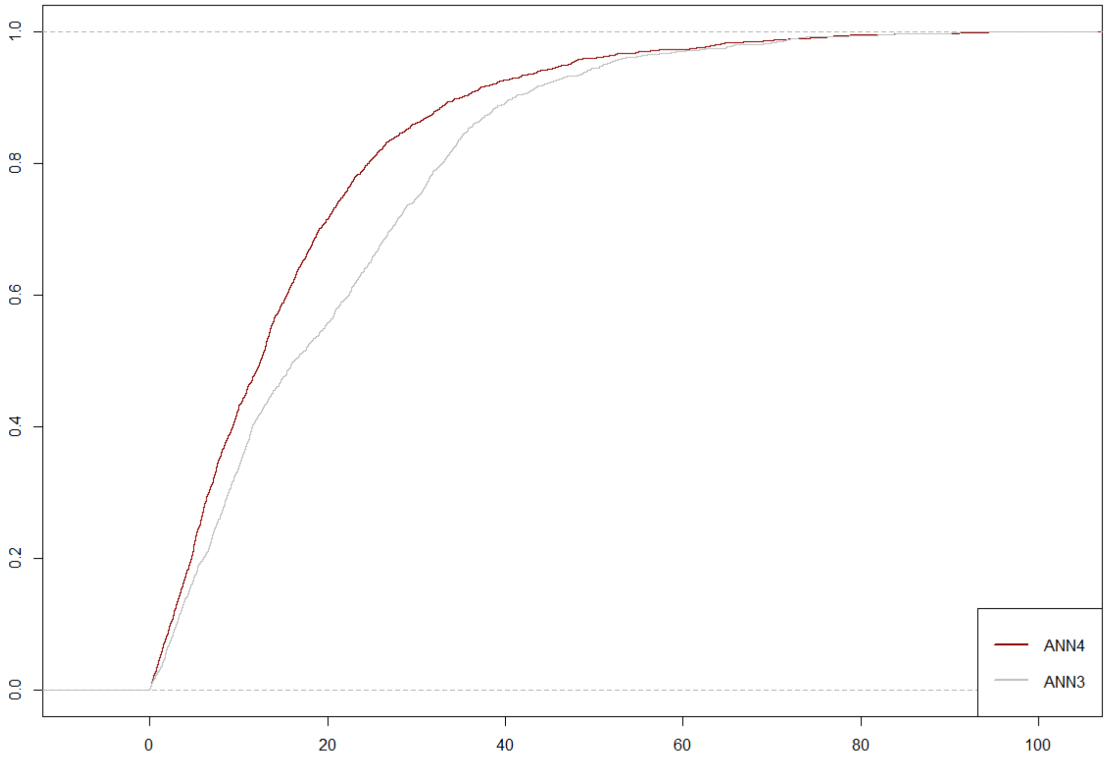


Figure 4.23. ECDF of the absolute differences between prediction and truth for ANN3 and ANN4

This difference can be further evidenced by analyzing common accuracy metrics compiled in Table 4.8.

Table 4.8. Accuracy metrics for ANN3 and ANN4

	ANN3	ANN4	Difference
R^2	0.32	0.51	0.19
R	0.10	0.26	0.16
MAE (min)	20.34	16.20	-4.14
MAPE (%)	7.90	6.10	-1.80
MSE (min)	681.80	487.53	-194.27
RMSE (min)	26.11	22.08	-4.03

When comparing the goodness-of-fit or R^2 (or its square-root – R) values, we see that the values are better for ANN4 than for ANN3. However, these metrics are not the most adequate to measure the accuracy of predictions of an ANN. Mean absolute error (MAE) given by (4.21) is more informative. For ANN3 the MAE is 20.34 minutes, while for ANN4 it improves by approximately 4 minutes to 16.20 minutes. To better understand this error in a relative sense compared to the truth, the Mean Absolute Percentage Error (MAPE) given by (4.22) was computed. Here, we see that ANN3 has an error of only 7.90%, whereas ANN4, which is generally the better preforming model, achieves an even better result with just 6.10% MAPE. Finally, we also compute the Mean Square Error (MSE), given by (4.23), and the Root Mean Squared Error (RMSE), given by (4.24). As expected, ANN4 has smaller values of both these errors than the corresponding values for ANN3. MSE allows comparing models to each other, but its actual value is difficult to use to judge model accuracy. RMSE is a better metric for this purpose. As with all other error measures, the RMSE for ANN4 is better than that for ANN3 demonstrating its superiority.

$$MAE = \frac{1}{N}\sum_{i=1}^{N} |\hat{Y}_i - Y_i| \tag{4.21}$$

$$MAPE = \frac{1}{N}\sum_{i=1}^{N} \frac{|\hat{Y}_i - Y_i|}{Y_i} \tag{4.22}$$

$$MSE = \frac{1}{N}\sum_{i=1}^{N} \left(\hat{Y}_i - Y_i\right)^2 \tag{4.23}$$

$$RMSE = \sqrt{MSE} = \sqrt{\frac{1}{N}\sum_{i=1}^{N} \left(\hat{Y}_i - Y_i\right)^2} \tag{4.24}$$

4.6 Conclusion

We showed that it is possible to create a metamodel that predicts the optimal solutions of a schedule recovery model. In this chapter, we have provided a proof of this concept. Although we could not accurately predict the exact solution in every instance, we showed that it is possible to use such an approach to aid the decision-making process of an AOCC. The ML metamodels presented in the case study were applied to airline operations. However, such an approach can potentially be applied to other optimization problems which run repeatedly and thus generate large databases of optimization problems and solutions. We developed a general methodology that allows the creation of databases ready to be used by an ML model as training set. The methodological concept presented in this work, Section 4.3, contains a standardized way to feed multiple optimization problems with varying dimensions to an ML metamodel so the ML metamodel can predict optimal solutions. This methodology solves the issue of varying problem dimensions, changing number of variables and constraints, and the interchangeable nature of the columns and rows of a problem matrix of an optimization problem.

Classification ANNs are advantageous due their model prediction certainty parameter. This implies that, besides the class, it returns the level of certainty that the model has of its prediction. From the case study results, we find that if we choose a high enough cutoff value for the model certainty, we can have a nearly perfect prediction accuracy level. This would be helpful for confidently using such a metamodel in real life.

When comparing the classification models, ANN1 and ANN2, we couldn’t conclude that the model with the higher complexity is necessarily better at prediction, because there was only a small increase in accuracy with the increased complexity, and even after accounting for different certainty levels, the accuracy is lower for ANN2 than for ANN1, for the highest certainty. So, in this particular case, we can say that the simpler model might sometimes be preferred over the more complex one, depending on the utilization, topology, and data used to train them.

Our regression ANN’s had difficulties in predicting the exact value for costs and delays. After digging deeper into the results, it was noted that too much similarity across the training problems led to an effective reduction in the size of the database. The data was just not sufficiently diverse for the models to learn and accurately predict the values of costs or delays. The unbalanced nature of the data, having significant more instances with one type of class than another, created additional difficulties for the classification ANNs. However, our ANNs presented promising results.

The artificial neural networks presented in the case study are only intended as proofs of the concept that they could be used in real life. Since we are only predicting one value at a time, the scope is limited. However, this could be used as information to accelerate the performance of the recovery optimization model, for instance, by using the results from the classification ANNs (ANN1 and ANN2) as extra constraints. The results from the regression ANNs (ANN3 and ANN4) could be used to provide a solution of this variable directly to the recovery model. Both potentially could reduce the computational time of the recovery model.

Our methodology is possible to be applied to any optimization problem as long as there exists a large set of historical problems and solutions to feed it and train the ML model. Thus, the methodological approach as described in Section 4.3 can leverage historical data and extract the patterns that exist in these datasets to predict optimization problem solutions.

4.7 Future Work

The present case study is not an exhaustive listing of all possible ANN topologies. Additional types of ANNs could be tested and compared with these results. Consequently, the next steps would be to further optimize the topology of the neural network to see if this could lead to improved predictive accuracy. As mentioned before, artificial neural networks are data hungry. More data and more diverse data, e.g. simulation of additional days and multiple types of delays, might be beneficial. It would also be interesting to perform an analysis of the transferability of the trained ML metamodels to the data from other airlines. It is important to also note that other types of machine learning models, besides artificial neural networks, could also be used as ML metamodel, and compared to ANNs to see if there is an advantage in using any one of them.

One important issue in an ML metamodel for optimization problems, especially combinatorial problems, is the feasibility of the solutions, i.e., the question of how to guarantee that the ML metamodel would generate feasible solutions. One way to proceed is checking the final solutions for feasibility and discard the ones that are not feasible. This is simple and fast method to implement but with a high risk of not producing usable solutions. Another enhancement to be made is to train the ML metamodel with a penalizing function for unmet constraints. Both options need to eb compared and the results assessed.

5 CONCLUSION

In this chapter, we gather the main conclusions of the thesis and propose further improvements and possible continuations on the research questions we have addressed in this document.

5.1 Main Conclusions

The work presented in this thesis was able in general to achieve the objectives defined when we initiated our research.

With the robust crew pairing model presented in Chapter 2, we showed that there is an optimal level of robustness to be added to schedules. Additionally, it was shown that crew schedules due to their constraints – legal framework, complex working rules, etc. – are the ideal candidates to add such robustness. Although the answer to uncertainty always involves balancing proactive measures and the ability to reactively respond to the disruptions, we showed that there is a level of robustness (buffers) that should be added to a schedule making it less brittle and able to absorb some of the undesirable consequences of the disruptions caused by uncertainty. Since we were able to keep the model in linear form, it can be used by any airline directly using off the shelf commercial optimization software.

Similarly, it was possible to show the benefits of the simulation platform presented in Chapter 3. It is an easy to use platform that allows any interested party, being it academic researchers or industry practitioners, to use the simulation framework with minimal effort. Likewise, the simulation model showed its capabilities to benchmark schedule recovery solutions. Consequently, it also benchmarks the models that generate these solutions. This is important because, otherwise it is hard to compare recovery models without implementing their outcomes in practice.

We feature an analysis of the behavior of an airline in the model of Chapter 3. We confirmed the idea that an agent-based formulation is a suitable way to model an enterprise like an airline. We also showed how variations in schedule recovery models can greatly influence the outcomes, the decisions that are made and the consequent costs of the operation of an airline.

In the final part of the thesis, Chapter 4, we coupled an airline schedule recovery model with a machine learning metamodel, specifically an artificial neural network. We demonstrated the potential of such metamodel to be used based on the historical database of schedule recovery problems and solutions, an underused asset of the airlines. We also conceived an approach that is able to capture the patterns in these data, making it a stepping stone for further work to be done in the area of airline schedule recovery.

5.2 Further improvements

As valuable as the models, methodologies and conclusions presented here can be, there are still further improvements that can be made.

Relating to the work described in Chapter 2, one of the most important improvements is to add robustness to the schedules of other resources of an airline, in particular, the aircraft. This would allow to compare the effectiveness of the added robustness for each schedule and to compare them which each other. Of course, an integrated way of adding robustness to the utilization of all possible resources would be preferable and would be an obvious further improvement that could generate interesting conclusions, especially when comparing with partial robustness of one resource like the crews, specifically crew pairings, as presented in this thesis.

As referred before, the answer to uncertainty is always a combination of proactive and reactive measures. The most important future work is to understand how robust scheduling solutions and schedule recovery solutions interact with each other. This means that it is not clear that the robust schedules (inducing buffers, redundancy or flexibility in the schedules) will always have the same outcomes with the different schedule recovery models used to correct disruptions. These interactions between the two should be tested and understood further. This is one of the applications where all three models presented in this thesis would come in handy: a robust proactive approach exemplified with our robust crew paring model; a platform that is able to simulate the daily operations of an airline in great detail; and a machine learning metamodel that is potentially able to mimic any reactive model.

Finally, we showed that it is possible to learn patterns from the historic database of optimization problems and solutions. However, the model we used as proof of concept needs to be further developed so it can be used confidently as an AOCC metamodel.

6 REFERENCES

Abar, S., Theodoropoulos, G.K., Lemarinier, P., O'Hare, G.M.P., 2017. Agent Based Modelling and Simulation tools: A review of the state-of-art software. Comput. Sci. Rev. 24, 13–33.

Abdelghany, K.F., Abdelghany, A.F., Ekollu, G., 2008. An integrated decision support tool for airlines schedule recovery during irregular operations. Eurpean J. Oper. Res. 185, 825–848.

AhmadBeygi, S., Cohn, A., Lapp, M., 2010. Decreasing airline delay propagation by re-allocating scheduled slack. IIE Trans. 42, 478–489.

AhmadBeygi, S., Cohn, A., Weir, M., 2009. An integer programming approach to generating airline crew pairings. Comput. Oper. Res. 36, 1284–1298.

Airlines of America, 2015. Per-minute cost of delays US airlines. http://airlines.org/data (accessed 1.12.15).

Alaska Airline, 2017. Alaska Airline website. www.alaskaair.com

Anylogic, 2018. AnyLogic website. www.anylogic.com

Argüello, M.F., Bard, J.F., Yu, G., 1997. A GRASP for aircraft routing in response to groundings and delays. J. Comb. Optim. 1, 211–228.

Baden, W.A., Bodoh, D.J., Williams, A.G., Kuzminski, P.C., 2011. SystemwideModeler: A fast-time simulation of the NAS. ICNS 2011 - Integr. Commun. Navig. Surveill. Conf. Renov. Glob. Air Transp. Syst. Proc. 1–6.

Ball, M., Barnhart, C., Dresner, M., Neels, K., Odoni, A., Peterson, E., Sherry, L., Trani, A., Zou, B., 2010. Total Delay Impact Study: A Comprehensive Assesment of the Costs and Impact of Flight Delay in the United States.

Ball, M., Barnhart, C., Nemhauser, G., Odoni, A., 2006. Air Transportation : Irregular Operations and Control, in: Handbooks of Operations Research and Management. pp. 1–71.

Barnhart, C., Fearing, D., Vaze, V., 2014. Modeling Passenger Travel and Delays in the National Air Transportation System. Oper. Res. 62, 580–601.

Barnhart, C., Vaze, V., 2015. The Global Airline Industry, in: Belobaba, P., Odoni, A., Barnhart, C. (Eds.), The Global Airline Industry, 2nd Edition. Wiley, pp. 263–286.

Barton, R.R., 1994. the 1994 Winter Simulation Conference.

Bertsimas, D., Sim, M., 2004. The Price of Robustness. Oper. Res. 52, 35–53.

Bertsimas, D., Thiele, A., 2006. Robust and Data-Driven Optimization: Modern Decision Making Under Uncertainty, in: Models, Methods, and Applications for Innovative Decision Making. INFORMS, pp. 95–122.

Bisaillon, S., Pasin, F., Laporte, G., 2010. A Large Neighbourhood Search Heuristic for the Aircraft and Passenger Recovery Problem. 4OR Q. J. Belgian, French Ital. Oper. Res. Soc. 9, 139–158.

Bonabeau, E., 2002. Agent-based modeling: methods and techniques for simulating human systems. Proc. Natl. Acad. Sci. 99, 7280–7287.

Box, G., Draper, N., 1987. Empirical modeling and response surfaces 424.

Bratu, S., Barnhart, C., 2006. Flight operations recovery: New approaches considering passenger recovery. J. Sched. 9, 279–298.

BTS, 2018. Bureau of Tranportation Statistic On-Time Performance - Flight Delays at a

Glance. https://www.transtats.bts.gov/homedrillchart.asp (accessed 11.5.18).

BTS, 2016. Airline On-Time Performance. http://www.transtats.bts.gov (accessed 1.1.16).

Castro, A.J.M., Oliveira, E., 2011. A new concept for disruption management in airline operations control. Proc. Inst. Mech. Eng. Part G J. Aerosp. Eng. 225, 269–290.

Castro, A.J.M., Rocha, A.P., Oliveira, E., 2012. Towards an autonomous and intelligent Airline Operations Control, in: 2012 15th International IEEE Conference on Intelligent Transportation Systems. Ieee, pp. 1429–1434.

Çetinkaya, D., Camci, I., 2014. Towards a metamodel for airport modeling and simulation. 26th Eur. Model. Simul. Symp. EMSS 2014.

Chen, X., Sim, M., Sun, P., 2007. A Robust Optimization Perspective on Stochastic Programming. Oper. Res. 55, 1058–1071.

Clarke, J., 2004. MIT Extensible Air Network Simulation (MEANS) Presentation - AGIFORS Airline Operations Study Group.

Clausen, J., Larsen, A., Larsen, J., Rezanova, N.J., 2010. Disruption management in the airline industry—Concepts, models and methods. Comput. Oper. Res. 37, 809–821.

Desaulniers, G., Desrosiers, J., Solomon, M.M., 2005. Column Generation. Springer US, Boston, MA.

Devore, M.D., Feigh, K., Pritchett, A., 2006. Design of Support Systems for Dynamic Decision Making in Airline Operations, in: Systems and Information Engineering Design Symposium. pp. 136–141.

Dunbar, M., Froyland, G., Wu, C.-L., 2012. Robust Airline Schedule Planning: Minimizing Propagated Delay in an Integrated Routing and Crewing Framework. Transp. Sci. 46, 204–216.

Ehrgott, M., Ryan, D.M., 2002. Constructing robust crew schedules with bicriteria optimization. J. Multi-Criteria Decis. Anal. 11, 139–150.

EUROCONTROL, 2018. Coda Digest 2017: All-Causes Delay and Cancellations to Air Transport in Europe - 2017.

FAA, 2017. FAA Registry N-Number Inquiry. URL http://registry.faa.gov/aircraftinquiry

FAA, 2014. System-Wide Analysis Capability (SWAC).

FAA, 2006. FAA System-Wide Modeling.

Funahashi, K.I., 1989. On the approximate realization of continuous mappings by neural networks. Neural Networks 2, 183–192.

Gao, C., Johnson, E., Smith, B., 2009. Integrated Airline Fleet and Crew Robust Planning. Transp. Sci. 43, 2–16.

Geem, Z.W., Kim, J.H., 2001. A New Heuristic Optimization Algorithm: Harmony Search. Simulation 76, 60–68.

Google, 2018. TensorFlow. https://www.tensorflow.org (accessed 8.29.18).

Hasani, A., Zegordi, S.H., Nikbakhsh, E., 2012. Robust closed-loop supply chain network design for perishable goods in agile manufacturing under uncertainty. Int. J. Prod. Res. 50, 4649–4669.

Hornik, K., Stinchcombe, M., White, H., 1989. Multilayer feedforward networks are universal approximators. Neural Networks 2, 359–366.

IATA, 2018. Fact Sheet Industry Statistics 2017–2018.

Jarrah, A.I.Z., Yu, G., Krishnamurthy, N., Rakshit, A., 1993. A Decision Support Framework for Airline Flight Cancellations and Delays. Transp. Sci. 27, 266–280.

Jennings, N.R., 2000. On agent-based software engineering. Artif. Intell. 117, 277–296.

Kasirzadeh, A., Saddoune, M., Soumis, F., 2014. Airline crew scheduling: Models , algorithms , and data sets, Les Cahiers du GERAD. Montreal.

Kirkpatrick, S., 1984. Optimization by Simulated Annealing : Quantitative Studies. J. Statisical Phys. 34, 975–986.

Kohl, N., Larsen, A., Larsen, J., Ross, A., Tiourine, S., 2007. Airline disruption management—Perspectives, experiences and outlook. J. Air Transp. Manag. 13, 149–162.

Lee, L.H., Huang, H.-C., Lee, C., Chew, E.-P., Jaruphongsa, W., Yong, Y.Y., Liang, Z., Leong, C.-H., Tan, Y.P., Namburi, K., Johnson, E., Jerry, B., 2003. Discrete Event Simulation Model for Airline Operations: SIMAIR, in: Proceedings of the 2003. pp. 1656–1662.

Lu, D., Gzara, F., 2015. The robust crew pairing problem: model and solution methodology. J. Glob. Optim. 62, 29–54.

Massachusetts Institute of Technology, 2014. Airline Data Project- Global Airline Industry Program. airlinedataproject.mit.edu (accessed 2.12.15).

Mathaisel, D.F.X., 1996. Decision support for airline system operations control and irregular operations. Comput. Oper. Res. 23, 1083–1098.

Melo, A.P., Cóstola, D., Lamberts, R., Hensen, J.L.M., 2014. Development of surrogate models using artificial neural network for building shell energy labelling. Energy Policy 69, 457–466.

NASA, 2002. Future Air Traffic Management Concepts Evaluation Tool (FACET).

National University of Sinapore, Georgia Institute of Technology, 2013. SIMAIR. http://www.ise.nus.edu.sg/project/simair/ (accessed 8.19.13).

Nordin, J.P., 1980. A fexible Simulation Model of Airport airside Operations.

Massachusetts Institute of Technology.

Pan, I., Babaei, M., Korre, A., Durucan, S., 2014. Artificial Neural Network based surrogate modelling for multi- objective optimisation of geological CO 2 storage operations. Energy Procedia 63, 3483–3491.

Petersen, J.D., Sölveling, G., Johnson, E.L., Clarke, J.-P., Shebalov, S., 2010. An Optimization Approach to Airline Integrated Recovery. Transp. Sci. 46, 482–500.

Power, D.J., Sharda, R., 2007. Model-driven decision support systems: Concepts and research directions. Decis. Support Syst. 43, 1044–1061. https://doi.org/10.1016/j.dss.2005.05.030

Rashedi, N., Vaze, V., 2019. A Supervised Machine Learning Approach for Solving the Aircraft Recovery Problem. Transp. Sci. (under revison).

Rauch, L., Górecki, G., Pietrzyk, M., Macioł, P., 2016. Metamodelling with artificial neural networks by using high performance computing infrastructures. AIP Conf. Proc. 1769.

Rosenberger, J.M., Schaefer, A.J., Goldsman, D., Johnson, E.L., Kleywegt, A.J., Nemhauser, G.L., 2002. A Stochastic Model of Airline Operations. Transp. Sci. 36, 357–377.

Schaefer, A.J., Johnson, E.L., Kleywegt, A.J., Nemhauser, G.L., 2005. Airline Crew Scheduling Under Uncertainty. Transp. Sci. 39, 340–348.

Shebalov, S., Klabjan, D., 2006. Robust Airline Crew Pairing: Move-up Crews. Transp. Sci. 40, 300–312.

Sim, M., 2004. Robust Optimization. Springer US, Boston, MA.

Sprinkle, J., Rumpe, B., Vangheluwe, H., Karsai, G., 2014. Metamodelling: State of the Art and Research Challenges. CoRR abs/1409.2.

Tam, B., Ehrgott, M., Ryan, D., Zakeri, G., 2011. A comparison of stochastic programming and bi-objective optimisation approaches to robust airline crew scheduling. OR Spectr. 33, 49–75.

Teodorović, D., Stojković, G., 1990. Model for operational daily airline scheduling. Transp. Plan. Technol. 14, 273–285.

Tu, Y., Ball, M.O., Jank, W.S., 2008. Estimating Flight Departure Delay Distributions—A Statistical Approach With Long-Term Trend and Short-Term Pattern. J. Am. Stat. Assoc. 103, 112–125.

Vance, P.H., Barnhart, C., Johnson, E.L., Nemhauser, G.L., 1997. Airline Crew Scheduling : A New Formulation and Decomposition Algorithm.

Virgin America, 2016. Virgin America website. www.virginamerica.com

Visentini, M.S., Borenstein, D., Li, J.Q., Mirchandani, P.B., 2014. Review of Real-Time vehicle schedule recovery methods in transportation services. J. Sched. 17, 541–567.

Wang, G.G., Shan, S., 2007. Review of Metamodeling Techniques in Support of Engineering Design Optimization. J. Mech. Des. 129, 370.

Weide, O., Ryan, D., Ehrgott, M., 2010. An iterative approach to robust and integrated aircraft routing and crew scheduling. Comput. Oper. Res. 37, 833–844.

Werbos, L., Kozma, R., Silva-Lugo, R., Pazienza, G.E., Werbos, P.J., 2012. Metamodeling and the Critic-based approach to multi-level optimization. Neural Networks 32, 179–185.

Wooldridge, M., 2008. An Introduction to MultiAgent Systems. Cambridge University Press, Cambridge.

World Bank, 2018. Air transport, passengers carried. https://data.worldbank.org/indicator (accessed 10.31.18).

World Bank Group, 2017. Air Transport: Annual Report 2017.

Yan, C., 2017. Airline Scheduling and Air Traffic Control : Incorporating Uncertainty and Passenger and Airline Preferences. Massachusets Institute of Technology.

Yan, C., Kung, J., 2016. Robust Aircraft Routing. Transp. Sci. 1–17.

Ye, Y., Li, J., Li, Z., Tang, Q., Xiao, X., Floudas, C.A., 2014. Robust optimization and stochastic programming approaches for medium-term production scheduling of a large-scale steelmaking continuous casting process under demand uncertainty. Comput. Chem. Eng. 66, 165–185.

Yen, J.W., Birge, J.R., 2006. A Stochastic Programming Approach to the Airline Crew Scheduling Problem. Transp. Sci. 40, 3–14.

7 APPENDIX I

In this appendix, we present the remaining parameters, variables and constraints in our PP formulation that were not presented in Section 2 of the Thesis. Some of these are borrowed directly from the base formulation. Others are added or modified by us but are not directly related to the robustness considerations. The parameters, sets and decision variable listed here are similar to those in the paper by AhmadBeygi et al. (2009). We retained most of them exactly as in the original model. For a small number of them, we simply changed their notations to avoid naming conflicts with the variables and parameters that were added by us (see Section 4 for such added notations).

I.a Sets

Ω: Set of crew bases;

$\mathcal{D} = \{1, \dots, D\}$: Set of all possible duty indices;

$\mathcal{D}' = \{1, \dots, D-1\}$: Set of all duty indices except the last duty;

$\mathcal{D}'' = \{2, \dots, D\}$: Set of all duty indices except the first duty;

$\mathcal{D}''' = \{2, \dots, D-1\}$: Set of all duty indices except the first and the last duties;

$\mathcal{B}$: Set of all flights originating at a crew base;

$\mathcal{E}$: Set of all flights terminating at a crew base;

$\mathcal{O}$: Set of all flight pairs that can end a duty and start the next one;

$\mathcal{L}$: Set of all flight pairs that can form a permissible sequence within a duty.

I.b Parameters

$\varphi \in \mathbb{R}^+$: Maximum duty flying time;

$D \in \mathbb{Z}^+$: Maximum allowable number of duties in the pairing.

I.c Decision Variables

$v_f \in \{0,1\}$: Binary decision variable indicating if flight f is the first flight in the pairing;

$u_d \in \{0,1\}$: Binary decision variable indicating if the pairing contains at least d duties.

I.d Cost Constraints

$$c_d \geq \theta \cdot u_d \qquad \forall d \in \mathcal{D} \quad \text{(I.1)}$$

$$c_d \geq \sum_{f \in \mathcal{F}} b_f \cdot z_{fd} \qquad \forall d \in \mathcal{D} \quad \text{(I.2)}$$

$$c_d \geq \beta \cdot e'_d \qquad \forall d \in \mathcal{D} \quad \text{(I.3)}$$

$$c'' \geq \sum_{d \in \mathcal{D}} c_d \quad \text{(I.4)}$$

$$c'' \geq \alpha \cdot T' \quad \text{(I.5)}$$

Constraints (I.1) to (I.3) set the cost of each duty to be the maximum of 1) the minimum guaranteed payment per duty; 2) the total block time of the duty; and 3) a fraction, β, of the elapsed time of the duty. Similarly, constraints (I.4) and (I.5) set the cost of the pairing, c'', to be the maximum of 1) the sum of all duty costs; and 2) a fraction, α, of the TAFB.

I.e Duty Constraints

$$\sum_{f\in\mathcal{F}} b_f \cdot z_{fd} \leq \varphi \qquad \forall d \in \mathcal{D} \tag{I.6}$$

$$e'_d \leq \lambda \qquad \forall d \in \mathcal{D} \tag{I.7}$$

Constraints (I.6) ensure that the total flying time in the d^{th} duty does not exceed the maximum permitted flying time. Constraints (I.7) prevent the d^{th} duty from violating the maximum elapsed time restriction. These two constraints are directly borrowed from the base formulation.

I.f Pairing Constraints

$$\sum_{d\in\mathcal{D}} z_{fd} \leq 1 \qquad \forall f \in \mathcal{F} \tag{I.8}$$

$$\sum_{d\in\mathcal{D}} \sum_{f\in\mathcal{F}} z_{f,d} = v \tag{I.9}$$

Constraints (I.8), which are directly borrowed from the base formulation, ensure that each flight f can appear at most once across all duties. Constraint (I.9), which ensures that the pairing has exactly v flights in it, is a new constraint that we have added to the base formulation.

I.g Balance constraints

$$\sum_{f\in\mathcal{F}} v_f = 1 \qquad \forall f \in \mathcal{F} \tag{I.10}$$

$$\sum_{f_1\in\mathcal{F}:(f_1,f)\in\mathcal{L}} x_{f_1f1} = \sum_{f_1\in\mathcal{F}:(f,f_1)\in\mathcal{L}} x_{ff_11} + \sum_{f_1\in\mathcal{F}:(f,f_1)\in\mathcal{O}} y_{ff_11} \qquad \forall f \in \mathcal{F}\backslash(\mathcal{B}\cup\mathcal{E}) \tag{I.11}$$

$$v_f + \sum_{f_1 \in \mathcal{F}:(f_1,f) \in \mathcal{L}} x_{f_1 f 1} = \sum_{f_1 \in \mathcal{F}:(f,f_1) \in \mathcal{L}} x_{f f_1 1} + \sum_{f_1 \in \mathcal{F}:(f,f_1) \in \mathcal{O}} y_{f f_1 1} \qquad \forall f \in \mathcal{B} \backslash \mathcal{E} \tag{I.12}$$

$$\sum_{f_1 \in \mathcal{F}:(f_1,f) \in \mathcal{L}} x_{f_1 f 1} = \sum_{f_1 \in \mathcal{F}:(f,f_1) \in \mathcal{L}} x_{f f_1 1} + \sum_{f_1 \in \mathcal{F}:(f,f_1) \in \mathcal{O}} y_{f f_1 1} + w_{f 1} \qquad \forall f \in \mathcal{E} \backslash \mathcal{B} \tag{I.13}$$

$$v_f + \sum_{f_1 \in \mathcal{F}:(f_1,f) \in \mathcal{L}} x_{f_1 f 1} = \sum_{f_1 \in \mathcal{F}:(f,f_1) \in \mathcal{L}} x_{f f_1 1} + \sum_{f_1 \in \mathcal{F}:(f,f_1) \in \mathcal{O}} y_{f f_1 1} + w_{f 1} \ \forall f \in \mathcal{E} \cap \mathcal{B} \tag{I.14}$$

$$v_f + \sum_{d \in \mathcal{D}} w_{fd} \leq 1, \forall f \in \mathcal{F} \tag{I.15}$$

$$\sum_{f_1 \in \mathcal{F}:(f_1,f) \in \mathcal{L}} x_{f_1 f d} + \sum_{f_1 \in \mathcal{F}:(f_1,f) \in \mathcal{O}} y_{f_1 f, d-1} = = \sum_{f_1 \in \mathcal{F}:(f,f_1) \in \mathcal{L}} x_{f f_1 d} + \sum_{f_1 \in \mathcal{F}:(f,f_1) \in \mathcal{O}} y_{f f_1 d} \quad \forall f \in \mathcal{F} \backslash \mathcal{E}, \forall d \in \mathcal{D}''' \tag{I.16}$$

$$\sum_{f_1 \in \mathcal{F}:(f_1,f) \in \mathcal{L}} x_{f_1 f 1} + \sum_{f_1 \in \mathcal{F}:(f_1,f) \in \mathcal{O}} y_{f_1 f, d-1} = = \sum_{f_1 \in \mathcal{F}:(f,f_1) \in \mathcal{L}} x_{f f_1 d} + \sum_{f_1 \in \mathcal{F}:(f,f_1) \in \mathcal{O}} y_{f f_1 d} + w_{fd} \quad \forall f \in \mathcal{E}, \forall d \in \mathcal{D}''' \tag{I.17}$$

$$\sum_{f_1 \in :\mathcal{F},(f_1,f) \in \mathcal{L}} x_{f_1 f D} + \sum_{f_1 \in \mathcal{F}:(f_1,f) \in \mathcal{O}} y_{f_1 f, D-1} = \sum_{f \in \mathcal{F}:(f,f_1) \in \mathcal{L}} x_{f f_1 D} + w_{fD} \quad \forall f \in \mathcal{E} \tag{I.18}$$

$$\sum_{f_1 \in \mathcal{F}:(f_1,f) \in \mathcal{L}} x_{f_1 f D} + \sum_{f_1 \in \mathcal{F}:(f_1,f) \in \mathcal{O}} y_{f_1 f, D-1} = \sum_{f \in \mathcal{F}:(f,f_1) \in \mathcal{L}} x_{f f_1 D} \qquad \forall f \in \mathcal{F} \backslash \mathcal{E} \tag{I.19}$$

$$\sum_{f \in \mathcal{F}:o_f = \omega} v_f = \sum_{f \in \mathcal{F}:d_f = \omega} \sum_{d \in \mathcal{D}} w_{fd} \qquad \forall \omega \in \Omega \tag{I.20}$$

Constraints (I.10) to (I.20) ensure that the flow of crew across the flight network is balanced and also ensure that the crew pairing represents a feasible sequential path in the flight network. Apart from a few adjustments, these constraints (other than constraints (I.14), (I.15) and (I.20)) are mostly borrowed from the base formulation. These adjustments were required to ensure that our model can correctly handle flight networks with multiple crew bases. Constraints (I.10) ensure that there is only one first flight in each pairing. Constraints (I.11) ensure flow balance for every flight that is in the first duty but neither starts nor ends at a crew base. Constraints (I.12), (I.13), and (I.14) ensure flow balance for the flights in the first duty that, respectively, start but do not end at a crew base, end but do not start at a crew base, and start as well as end at a crew base. Constraints (I.14) did not exist in the base formulation because they correspond to a special case that the base formulation did not attend to. Constraints (I.15) which are also new constraints, ensure that a flight may start or end a pairing, but it cannot do both. This ensures that we do not have pairings consisting of a single flight. Constraints (I.16) ensure flow balance for the flights in duties other than the first and the D^{th} one that do not end at a crew base, while constraints (I.17) do the same for those which do end at a crew base. Constraints (I.18) ensure flow balance for flights in the D^{th} duty which terminate at a crew base while constraints (I.19) achieve the same for those which do not. Constraints (I.20), which we added as new constraints, ensure that each pairing starts and ends at the same crew base. This was a small modification we made so that the model would handle multiple crew bases.

I.h Auxiliary Constraints

$$u_1 = 1 \tag{I.21}$$

$$u_d = \sum_{(f_1,f_2)\in \mathcal{O}} y_{f_1 f_2, d-1} \qquad \forall d \in \mathcal{D}'' \tag{I.22}$$

$$z_{f1} = \sum_{f_1 \in \mathcal{F}:(f_1,f)\in \mathcal{L}} x_{f_1 f 1} \qquad \forall f \in \mathcal{F}\backslash\mathcal{B} \tag{I.23}$$

$$z_{f1} = v_f + \sum_{f_1 \in \mathcal{F}:(f_1,f)\in \mathcal{L}} x_{f_1 f 1} \qquad \forall f \in \mathcal{B} \tag{I.24}$$

$$z_{fd} = \sum_{f_1 \in \mathcal{F}:(f,f_1) \in \mathcal{O}} y_{ff_1d} + \sum_{f_1 \in \mathcal{F}:(f,f_1) \in \mathcal{L}} x_{ff_1d} \quad \forall f \in \mathcal{F} \backslash \mathcal{E}, \forall d \in \mathcal{D}''' \tag{I.25}$$

$$z_{fd} = w_{fd} + \sum_{f_1 \in \mathcal{F}:(f,f_1) \in \mathcal{O}} y_{ff_1d} + \sum_{f_1 \in \mathcal{F}:(f,f_1) \in \mathcal{L}} x_{ff_1d} \quad \forall f \in \mathcal{E}, \forall d \in \mathcal{D}''' \tag{I.26}$$

$$z_{fD} = \sum_{f_1 \in \mathcal{F}:(f,f_1) \in \mathcal{L}} x_{ff_1D} \quad \forall f \in \mathcal{F} \backslash \mathcal{E} \tag{I.27}$$

$$z_{fD} = w_{fD} + \sum_{f_1 \in \mathcal{F}:(f,f_1) \in \mathcal{L}} x_{ff_1D} \quad \forall f \in \mathcal{E} \tag{I.28}$$

Constraints (I.21) to (I.28) are unchanged from the base formulation. They implement the relationships between all the flight-specific or duty-specific decision variables of the base formulation, namely, $x_{f_1f_2d}$, $y_{f_1f_2d}$, z_{fd}, v_f, w_{fd} and u_d. Constraints (I.21) ensure that there is at least one duty in the chosen pairing, while constraints (I.22) ensure that, for all subsequent duties, there exists exactly one y variable that connects it to the previous duty. Constraints (I.23) and (I.24) ensure that if a flight, which is covered by the first duty in a pairing, starts at a crew base then it can be the first flight of the pairing, and otherwise it must be preceded by another flight in the same duty. Analogously, constraints (I.25) and (I.26) ensure that if a flight ends at a crew base then it can be the last flight of the pairing, and otherwise it must be followed by another flight in the same or the next duty. Finally, constraints (I.27) and (I.28) ensure that, for any flight covered by the last duty of the pairing, if it ends at a crew base then it can be the last flight in the pairing, and otherwise it must be followed by another flight in the same duty.

8 APPENDIX II

Appendix II relates to the Section 3 of the Thesis

II.a Recovery models

II.a.i DSTAR

The DSTAR model is based on the work done by Abdelghany et. al(Abdelghany et al., 2008):

Abdelghany, K. F., Abdelghany, A. F., & Ekollu, G. (2008). An integrated decision support tool for airlines schedule recovery during irregular operations. European Journal of Operations Research, 185, 825–848.

The recovery process is divided into stages where at end of each stage, in case any flight is disrupted in this stage, resources (i.e. aircrafts, cockpit crew pairs, cabin crew pairs) are rescheduled, delayed. If necessary, flights canceled to be able to cope with the delays. Every stage is limited by independent flights in terms of time and space. The optimization algorithm is developed as a mixed integer formulation. This model gave us a good framework how to insert the optimization algorithm into the simulation framework, using the staging scheme (view Figure II.1).

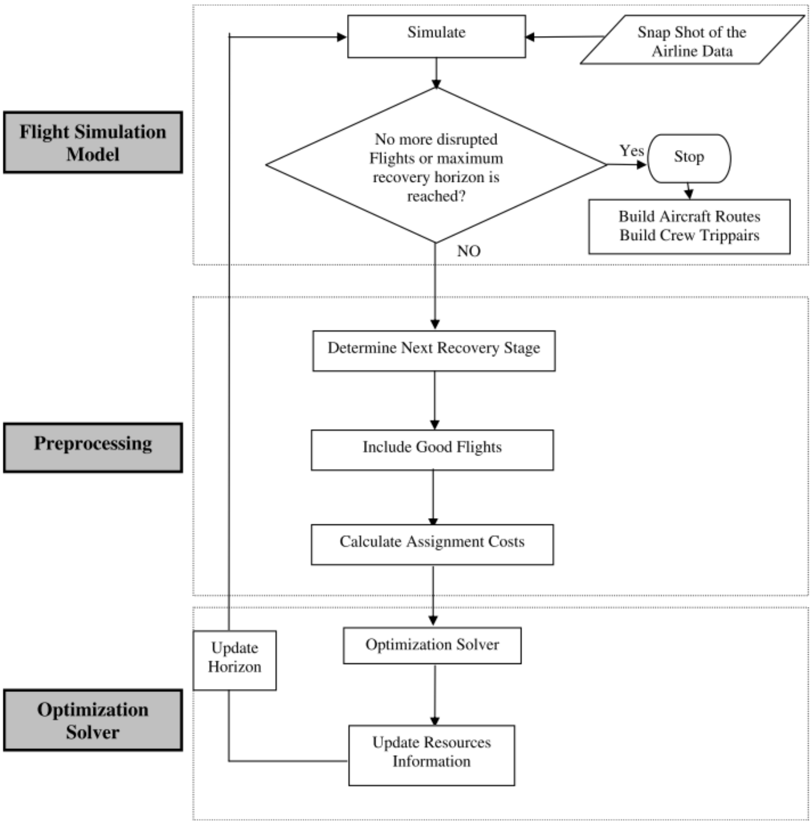

Flight Simulation Model
Simulate
Snap Shot of the Airline Data
No more disrupted Flights or maximum recovery horizon is reached?
Yes
Stop
Build Aircraft Routes Build Crew Trippairs
NO
Preprocessing
Determine Next Recovery Stage
Include Good Flights
Calculate Assignment Costs
Optimization Solver
Update Horizon
Optimization Solver
Update Resources Information

Figure II.1. Framework how the DSTAR model works (source (Abdelghany et al., 2008))

In the AOSM the model does not run all stages until the end. We decided to only run a stage at the time since the model was designed to know beforehand the future delays. In a stochastic model, as AOSM, this is not the case. It leads to a more myopic version compared the original DSTAR model, but it better represents the reality faced by an AOCC of an airline. They do not know the delays in advance.

II.a.i.i Sets and indices

F_s: set of flights f in stage s (in our case the is only one stage each time the DSTAR model is called);

k: defines the type of resource (aircraft, crew pairing);

r_k: represents the resource r of type k;

II.a.i.ii Variables

$x_{r,f}$: binary variable that indicates that resource r is covering flight f;

$b_{r,f}$: binary variable that indicates if resource can cover any part of trip pair r;

$p_{r,f}$: the part of the crew paring that can be covered by resource r;

L_f: binary variable that indicates if flight f is canceled or not;

m_f: variable that indicates the actual departure time of flight f;

n_f: variable that indicates the actual arrival time of flight f;

II.a.i.iv Parameters

$a_{r,f}$: ready time of resource r to operate flight f;

$c_{r,f}$: swapping cost of resource r to flight f;

cd_f: 1 minute of delay cost for flight f;

cc_f: cost for canceling flight f;

v_r: available time in duty that resource r still has left;

t_f: planned departure time of flight f;

T_f: block time of flight f;

II.a.i.v Objective Function

Without any major modification we were able to incorporate the model. In (II.1) the objective function of the model is given:

$$\min \sum_{f \in F_S} \sum_{r} c_{r,f} \cdot x_{r,f} + \sum_{f \in F_S} cd_f \cdot [m_f - t_f] + \sum_{f \in F_S} cc_s \cdot L_f \qquad \text{(II.1)}$$

The objective function minimizes is the sum of three components. First parcel minimizes the swapping of resources the second is the cost relative to delays and the final one captures the cancelation cost.

II.a.i.vi Constraints

The objective function is subjected to the following constrains:

$$x_{r,f} \leq b_{r,f} \qquad \forall f \in F_s, r \quad \text{(II.2)}$$

The first constraint, equation (II.2), indicates that the resource $x_{r,f}$ is only possible to be used if it is available.

$$\sum_{f \in F_S} x_{r,f} \leq 1 \qquad \forall f \in r \quad \text{(II.3)}$$

In (II.3) the constrain obliges the model to only assign resource $x_{r,f}$ to a single flight.

$$[1 - L_f] \cdot g_{k,f} = \sum_{r_k} x_{r_k,f} \qquad \forall f \in F_s, k \quad \text{(II.4)}$$

The constraint (II.4) assures that if a flight is cancelled there cannot be assigned any resources to that flight.

$$m_f \geq a_{r,f} \cdot x_{r,f} \qquad \forall f \in F_s, r \quad \text{(II.5)}$$

In constraint given by (II.5) the model is forced to for any given flight the departure time is later that the allocated resources latest ready time.

$$n_f \leq v_r \cdot x_{r,f} + (1 - x_{r,f}) \cdot UB(n_f) \qquad \forall f \in F_s, r \quad \text{(II.6)}$$

Constraint in (II.6) assures that in case of delays the maximum duty time is not violated.

$$n_f = m_f + T_f \qquad \forall f \in F_s \quad \text{(II.7)}$$

Constraint described in (II.7) computes the arrival time. It is equal the real departure time plus the block time.

$$m_f \geq t_f \qquad \forall f \in F_s \quad \text{(II.8)}$$

In constraint (II.8) all flight must depart at or after their scheduled departure. This means that there are no early departures, i.e. negative delays.

$$x_{r,f}, L_f \in \{0,1\} \qquad \forall f \in F_s, r \quad \text{(II.9)}$$

$$m_f, n_f \in \mathbb{Z} \qquad \forall f \in F_s \quad \text{(II.10)}$$

In(II.9) and (II.10)we assure that $x_{r,f}$ and L_f are binary and in that m_f and n_f are integers.

II.a.ii DPM model

The DPM model currently integrated into de AOSM is based on the following work from Bratu and Barnhart (2006):

> *Bratu, S., & Barnhart, C. (2006). Flight operations recovery: New approaches considering passenger recovery. Journal of Scheduling, 9(3),279-298.*

These authors developed a recovery model that it is passenger centric. The paper contains 2 models: the passenger delay model (PDM) where the cost for passenger are modelled in high detail and the disrupted passenger metric (DPM) where these costs are captured in a simplified manner. It allows to delay or cancel departures and assign reserve crews to individual flight legs. The computational tests showed that the more detailed version (PDM) has tractability issues not able to be solved in a real-time environment where as the DPM is able to produce solutions in an efficient manner. As we can see in Figure II.2 the model is also developed to be solved inside a simulation and re-run in steps (t1, t2, t3,.., te). One other important detail is that this model is not linear.

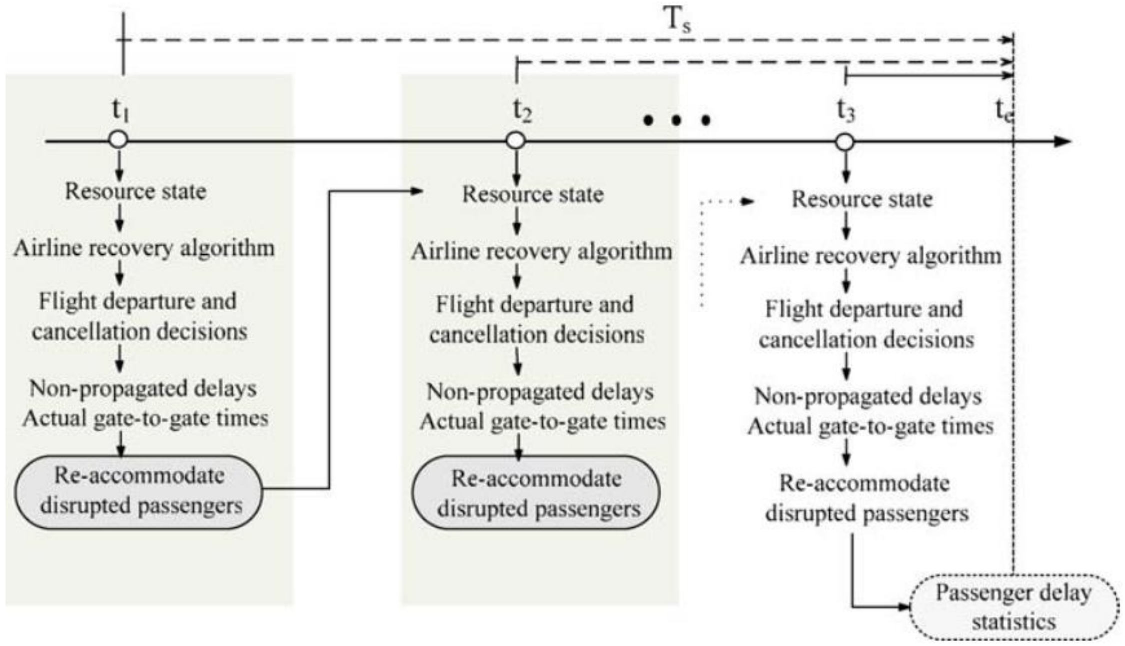

Ts
t1
t2
t3
te
Resource state
Airline recovery algorithm
Flight departure and cancellation decisions
Non-propagated delays
Actual gate-to-gate times
Re-accommodate disrupted passengers
Resource state
Airline recovery algorithm
Flight departure and cancellation decisions
Non-propagated delays
Actual gate-to-gate times
Re-accommodate disrupted passengers
Resource state
Airline recovery algorithm
Flight departure and cancellation decisions
Non-propagated delays
Actual gate-to-gate times
Re-accommodate disrupted passengers
Passenger delay statistics

Figure II.2. Framework of the DPM model (source (Bratu and Barnhart, 2006))

II.a.ii.i Sets and indices

K: set of fleet types;

A: set of airports;

$F_{n+}^{k,a}$: set of flight arcs of fleet type k departing form node n of airport a;

$F_{n-}^{k,a}$: set of flight arcs of fleet type k arriving form node n of airport a;

$N(k,a)$: Set of nodes with in or outflow at airport a for fleet type k;

R: set of crews;

$f_c(m)$: m^{th} flight leg scheduled to be operated by crew c:

$M(c,m,t)$: set of copies of light leg $f_c(m+1)$ to which crew c cannot connect with.

$I\big(f_c(m)\big)$: Set of flight copies of flight leg $f_c(m)$;

$IT(p)$: set of flight legs in itinerary p;

P: set of passenger itineraries;

II.a.ii.ii Variables

x_f^t: binary variable that indicates if flight f will depart at the tth offset time;

z_f: binary variable that indicates if flight f is canceled or not;

$y_t^{k,a}$: integer variable that indicates the number of aircrafts on the ground of type k, at airport a, at time t;

q_m^c: integer variable that indicates a reserve crew is necessary to operate the mth flight leg of crew c original scheduled flights due to work rule violation;

r_m^c: integer variable that indicates a reserve crew is necessary to operate the mth flight leg of crew c original scheduled flights due to cancelation;

II.a.ii.iii Parameters

e_c^m: cost of calling a reserve crew to operate mth flight of crew c schedule;

s_p: Cost per disrupted passenger;

o_{tf}: Cost of delaying flight f, t time-steps;

n_p: Number of passengers in itinerary p;

n_f: Number of itineraries that end with flight f;

β_f^p: Indicates if itinerary p terminates with flight f;

d_f^t: delay cost per non- disrupted passenger of flight f, time offset t.

II.a.ii.iv Objective function

The objective function given in (II.11) minimizes the sum of 3 parcels. First the cost of disrupted passengers, second is the cost of delay and the 3rd related to the cost of extra crew pairing necessary to cover the flights:

$$\begin{aligned}\min \sum_{p\in P} s_p \cdot n_p \cdot \lambda_p \\ + \sum_{f\in F}\sum_{t\in I(f)}\left(o_f^t + \left(n_f - \sum_{p\in P} n_p \cdot \beta_f^p \cdot \lambda_p\right)\right)\cdot x_f^t \\ + \sum_{c\in R}\sum_{m=1,\dots,n_c} e_m^c \cdot (r_m^c + q_m^c)\end{aligned} \tag{II.11}$$

II.a.ii.v Constraints

$$\sum_{(f,t)\in F_{n-}^{k,a}} x_f^t + y_{n-}^{k,a} - \sum_{(f,t)\in F_{n+}^{k,a}} x_f^t - y_{n+}^{k,a} = 0 \quad \forall k \in K, \forall a \in A, \forall n \in N(k,a) \tag{II.12}$$

$$\sum_{(f,t)\in F_{b+}^{k,a}} x_f^t + y_{n+}^{k,a} = N_b^{k,a} \qquad \forall k \in K, \forall a \in A \tag{II.13}$$

$$\sum_{(f,t)\in F_{e-}^{k,a}} x_f^t + y_{e-}^{k,a} = N_e^{k,a} \qquad \forall k \in K, \forall a \in A \tag{II.14}$$

Constraint (II.12) to (II.14) are aircraft flow balance constrains. The authors mention in the original paper that if necessary, it is possible to add spare aircrafts to the model by adding them to the right-hand side of these constrains.

$$\sum_{t \in I(f)} x_f^t + z_f = 1 \qquad \forall f \in F \qquad \text{(II.15)}$$

Constraints in (II.15) guarantees that each aircraft is assigned to a flight leg or cancelled.

$$x_{IT(p,1)}^t \sum_{u \in MC(IT(p,1),t)} x_{IT(p,2)}^u - \lambda_p \leq 1 \qquad \forall p \in C, \forall t \in I(IT(p,q)) \qquad \text{(II.16)}$$

$$\lambda_p \geq z_f \qquad \forall p \in P, \forall t \in I(IT(p)) \qquad \text{(II.17)}$$

When there are cancelled flights, or insufficient connection time the passengers on that route are classified as disrupted this is handled by constraints (II.16) and (II.17), respectively.

$$x_{f,c(m)}^t + \sum_{i \in M(c,m,t)} x_{f,c(m)}^u - q_{m+1}^c \leq 1, \ \forall c \in R, \forall m \in 1, \dots, n_c - 1, \forall t \in I\big(f_c(m)\big) \qquad \text{(II.18)}$$

Constraint given by (II.18) assures that a reserve crew is placed at a flight that does not sufficient time to connect to the next flight and so one or more work rules would be violated.

$$\sum_{m=1,\dots,i} \sum_{t \in I(f_c(m))} x_{f,c(m)}^t \cdot \gamma_{a1_c(m)}^{a1_c(i)} - x_{f,c(m)}^t \cdot \gamma_{a2_c(m)}^{a1_c(i)} - \sum_{m=1,\dots,i} r_m^c \cdot \gamma_{a1_c(m)}^{a1_c(i)} < 0 \qquad \text{(II.19)}$$

$$\forall c \in R, \forall i = 2, \dots, n_c$$

In (II.19) we present the constraints that handles the reserve crews needed due to cancellations.

$$\sum_{c \in R} \sum_{m=1,\dots,n_c} (r_m^c + q_m^c) \cdot \gamma_{a1_c(m)}^a \cdot \delta_c^k \leq E(a,k) \ \ \forall a \in A', \forall k \in K' \qquad \text{(II.20)}$$

The constraint in (II.20) limits the new assigned reserve crew due to work rule violations or cancelations to the maximum available crews.

$$0 \leq \lambda_p \leq 1 \qquad \forall p \ \in P, \tag{II.21}$$

$$0 \leq r_c^m, q_c^m \leq 1 \qquad \forall m \ \in 1, \dots, n_c, \forall c \ \in R \tag{II.22}$$

$$x_f^t \in \{0,1\} \qquad \forall f \ \in F, \forall t \ \in I(f) \tag{II.23}$$

$$z_f \geq 0 \qquad \forall f \ \in F \tag{II.24}$$

$$y_n^{k,a} \geq 0 \qquad \forall k \ \in K, \forall a \ \in A, \forall a \ \in A, \forall n \ \in N(k,a) \tag{II.25}$$

Constraints (II.21) to (II.25) grant that x_f^t is binary, $y_n^{k,a}$ and $z_f \geq 0$ are non-negative, and that r_c^m, q_c^m, λ_p are between 0 and 1.

II.b Delay modeling details

II.b.i p_y– Yearly pattern

Table II.1. Result of the Box-Pierce test for the yearly pattern of Alaska Airlines

Lag	1	2	3	4	5	6	7	8	9	10
ABQ	0.8088	0.7946	0.9064	0.6621	0.7696	0.5923	0.7036	0.6444	0.7324	0.7797
ADK	0.3081	0.4674	0.5459	0.2679	0.2306	0.2434	0.2385	0.3166	0,1577	0,1655
ADQ	0,6330	0,6012	0,4510	0,4918	0,3112	0,4284	0,3389	0,3859	0,4391	0,5340
ANC	0,1738	0,0026	0,0018	0,0022	0,0025	0,0034	0,0046	0,0022	0,0035	0,0054
ATL	0,3648	0,3374	0,2098	0,3386	0,4736	0,4888	0,5826	0,3403	0,4348	0,4904
AUS	0,0269	0,0300	0,0712	0,1278	0,1917	0,2843	0,3383	0,3678	0,4026	0,4471
BET	0,3381	0,3469	0,0632	0,1159	0,1893	0,2222	0,3120	0,2042	0,2767	0,3584
BNA	0,1474	0,3062	0,4887	0,4915	0,6360	0,7524	0,8314	0,8477	0,9012	0,9255
BOS	0.3677	0.5930	0.5222	0.0306	0.0317	0.0567	0.0815	0.0605	0.0589	0.0505
BRW	0.5067	0.5018	0.3466	0.0563	0.0273	0.0428	0.0713	0.1033	0.1091	0.1403
BUR	0.7461	0.9144	0.7589	0.6891	0.6853	0.6101	0.7183	0.8065	0.3066	0.3222
BWI	0.9644	0.8473	0.8550	0.9122	0.8418	0.8396	0.8448	0.9041	0.4660	0.2026
CDV	0.0276	0.0842	0.1679	0.2810	0.3118	0.3942	0.4914	0.5106	0.6115	0.3628
CHS	0.1660	0.3768	0.4505	0.5712	0.6829	0.7709	0.8563	0.8432	0.8626	0.8872
DCA	0.7141	0.2955	0.2605	0.3811	0.4695	0.5159	0.5944	0.2099	0.0867	0.0630
DEN	0.2579	0.0135	0.0289	0.0545	0.0912	0.1470	0.0083	0.0023	0.0041	0.0061

DFW	0.8484	0.0083	0.0096	0.0033	0.0075	0.0109	0.0015	0.0030	0.0051	0.0056
DTW	0.0729	0.0901	0.1337	0.1927	0.2844	0.3910	0.4851	0.3608	0.4111	0.4841
EWR	0.5945	0.6678	0.8133	0.0028	0.0056	0.0111	0.0199	0.0174	0.0119	0.0194
FAI	0.0276	0.0848	0.0803	0.0112	0.0182	0.0340	0.0238	0.0398	0.0585	0.0835
FLL	0.0019	0.0068	0.0153	0.0336	0.0621	0.0313	0.0530	0.0840	0.0337	0.0260
GEG	0.2982	0.3038	0.4641	0.5458	0.6347	0.5818	0.5498	0.6245	0.6864	0.7688
HNL	0.0247	0.0781	0.1004	0.1709	0.2105	0.0477	0.0338	0.0449	0.0638	0.0817
IAD	0.8202	0.8553	0.6757	0.1762	0.1093	0.1720	0.2156	0.2970	0.2176	0.2795
IAH	0.0512	0.1234	0.2134	0.2317	0.3326	0.3120	0.2463	0.3259	0.4173	0.3458
JFK	0.7822	0.5611	0.7415	0.8406	0.8870	0.9288	0.9570	0.9787	0.9883	0.0151
JNU	0.6610	0.8996	0.9721	0.5254	0.6526	0.3323	0.4367	0.5448	0.6072	0.6296
KOA	0.0018	0.0033	0.0027	0.0027	0.0057	0.0115	0.0090	0.0064	0.0101	0.0113
KTN	0.0015	0.0009	0.0008	0.0021	0.0040	0.0020	0.0040	0.0036	0.0011	0.0005
LAS	0.9336	0.0008	0.0003	0.0009	0.0021	0.0046	0.0003	0.0004	0.0006	0.0011
LAX	0.4066	0.0442	0.0967	0.0024	0.0009	0.0017	0.0014	0.0028	0.0040	0.0057
LIH	0.1715	0.0143	0.0003	0.0010	0.0023	0.0037	0.0055	0.0056	0.0090	0.0101
MCI	0.9193	0.9125	0.3061	0.3756	0.3824	0.5042	0.6203	0.6914	0.4862	0.4505
MCO	0.1871	0.1674	0.3063	0.4336	0.5004	0.3004	0.4043	0.4999	0.5807	0.6724
MSP	0.0052	0.0200	0.0479	0.0843	0.0999	0.0103	0.0103	0.0173	0.0291	0.0076
MSY	0.0013	0.0007	0.0022	0.0010	0.0007	0.0016	0.0029	0.0036	0.0059	0.0083
OAK	0.0281	0.0896	0.0004	0.0008	0.0016	0.0022	0.0045	0.0078	0.0124	0.0193
OGG	0.0532	0.0756	0.1601	0.0309	0.0015	0.0021	0.0028	0.0018	0.0012	0.0023
OMA	0.0004	0.0016	0.0026	0.0061	0.0110	0.0089	0.0058	0.0053	0.0060	0.0103
OME	0.5677	0.7811	0.8867	0.8693	0.8175	0.8110	0.8199	0.6112	0.6460	0.3495
ONT	0.2199	0.4685	0.6763	0.8219	0.8140	0.8594	0.9152	0.9165	0.9498	0.9699
ORD	0.2850	0.0105	0.0274	0.0519	0.0837	0.1115	0.1699	0.2395	0.2859	0.3309
OTZ	0.8803	0.9026	0.9537	0.7888	0.6064	0.7277	0.5639	0.5676	0.3836	0.2684
PDX	0.0253	0.0395	0.0742	0.0969	0.1395	0.2100	0.1319	0.1914	0.2561	0.3301
PHL	0.0327	0.0938	0.1486	0.1088	0.1567	0.1408	0.0928	0.1212	0.1748	0.1956
PHX	0.0388	0.0475	0.0220	0.0138	0.0217	0.0282	0.0487	0.0212	0.0201	0.0326
PSG	0.6776	0.6582	0.8082	0.9140	0.9488	0.9739	0.8641	0.9195	0.8820	0.8841
PSP	0.5502	0.8339	0.9324	0.9224	0.3347	0.3929	0.1260	0.1679	0.2129	0.2530
RDU	0.9836	0.3995	0.5914	0.2607	0.3123	0.3985	0.1868	0.2515	0.3348	0.4225
SAN	0.0058	0.0168	0.0115	0.0200	0.0043	0.0073	0.0111	0.0189	0.0157	0.0248
SAT	0.1395	0.2201	0.3352	0.4932	0.4975	0.0986	0.1522	0.2169	0.2785	0.3593
SCC	0.0788	0.0251	0.0384	0.0704	0.1006	0.0821	0.1297	0.1658	0.2292	0.2363
SEA	0.3262	0.1736	0.2478	0.0881	0.0667	0.0961	0.0499	0.0748	0.1123	0.0857
SFO	0.1780	0.3624	0.2613	0.3239	0.4242	0.4453	0.5503	0.6569	0.7161	0.7903
SIT	0.6821	0.0017	0.0032	0.0078	0.0076	0.0120	0.0124	0.0140	0.0237	0.0367
SJC	0.0295	0.0761	0.0396	0.0141	0.0286	0.0415	0.0698	0.0785	0.1179	0.0281

SLC	0.3818	0.6820	0.6470	0.3700	0.4976	0.5577	0.4176	0.2847	0.3715	0.4606
SMF	0.0353	0.0660	0.0735	0.1095	0.0738	0.0679	0.0733	0.1070	0.1428	0.1984
SNA	0.0196	0.0647	0.0114	0.0165	0.0330	0.0557	0.0706	0.1096	0.1132	0.1572
STL	0.5132	0.5329	0.7162	0.8515	0.9134	0.9522	0.8558	0.9079	0.7078	0.7464
TPA	0.0261	0.0320	0.0037	0.0045	0.0019	0.0031	0.0062	0.0113	0.0195	0.0278
TUS	0.3178	0.3974	0.6051	0.7629	0.8662	0.9254	0.9594	0.9284	0.9490	0.9129
WRG	0.1357	0.3284	0.5148	0.2630	0.1734	0.1771	0.2091	0.1762	0.0657	0.0967
YAK	0.0604	0.1517	0.0843	0.0294	0.0417	0.0478	0.0354	0.0567	0.0846	0.1059
AKN	0.8588	0.9803	0.7630	0.8377	0.8590	0.9017	0.9267	0.9375	0.9624	0.9000
DLG	0.6350	0.8021	0.9219	0.9252	0.9705	0.9198	0.9357	0.9666	0.9562	0.9034
GST	0.3760	0.2714	0.4336	0.5765	0.7027	0.7862	0.8658	0.8330	0.8045	0.8645

Table II.2. Result of the Lijung-Box test for the yearly pattern of Alaska Airlines

Lag	1	2	3	4	5	6	7	8	9	10
ABQ	0.8081	0.7927	0.9050	0.6554	0.7636	0.5809	0.6931	0.6303	0.7196	0.7673
ADK	0.3053	0.4628	0.5396	0.2580	0.2192	0.2299	0.2231	0.2982	0.1407	0.1465
ADQ	0.6316	0.5980	0.4453	0.4848	0.3019	0.4177	0.3260	0.3714	0.4230	0.5173
ANC	0.1721	0.0024	0.0016	0.0020	0.0022	0.0030	0.0041	0.0019	0.0030	0.0047
ATL	0.3628	0.3338	0.2051	0.3323	0.4663	0.4798	0.5731	0.3258	0.4189	0.4732
AUS	0.0263	0.0291	0.0692	0.1245	0.1871	0.2784	0.3308	0.3585	0.3915	0.4343
BET	0.3361	0.3433	0.0606	0.1117	0.1832	0.2146	0.3026	0.1936	0.2638	0.3437
BNA	0.1457	0.3031	0.4848	0.4856	0.6303	0.7473	0.8271	0.8424	0.8971	0.9217
BOS	0.3658	0.5903	0.5172	0.0286	0.0293	0.0528	0.0761	0.0553	0.0533	0.0449
BRW	0.5049	0.4983	0.3410	0.0532	0.0252	0.0395	0.0663	0.0964	0.1011	0.1303
BUR	0.7450	0.9137	0.7552	0.6831	0.6777	0.5993	0.7085	0.7983	0.2872	0.3010
BWI	0.9643	0.8458	0.8527	0.9102	0.8370	0.8336	0.8379	0.8990	0.4444	0.1833
CDV	0.0269	0.0825	0.1649	0.2768	0.3059	0.3870	0.4835	0.5007	0.6017	0.3458
CHS	0.1630	0.3715	0.4428	0.5622	0.6737	0.7621	0.8494	0.8328	0.8513	0.8760
DCA	0.7130	0.2916	0.2553	0.3745	0.4618	0.5068	0.5846	0.1968	0.0776	0.0550
DEN	0.2560	0.0129	0.0276	0.0522	0.0877	0.1419	0.0072	0.0019	0.0034	0.0051
DFW	0.8478	0.0079	0.0091	0.0030	0.0068	0.0100	0.0013	0.0026	0.0044	0.0047
DTW	0.0717	0.0882	0.1306	0.1882	0.2784	0.3839	0.4771	0.3490	0.3976	0.4696
EWR	0.5930	0.6651	0.8109	0.0025	0.0050	0.0100	0.0180	0.0154	0.0103	0.0169
FAI	0.0270	0.0831	0.0780	0.0104	0.0169	0.0317	0.0218	0.0366	0.0539	0.0773
FLL	0.0018	0.0066	0.0147	0.0323	0.0600	0.0294	0.0501	0.0796	0.0306	0.0231
GEG	0.2962	0.3003	0.4596	0.5402	0.6285	0.5727	0.5380	0.6125	0.6742	0.7580
HNL	0.0241	0.0765	0.0979	0.1669	0.2052	0.0443	0.0308	0.0409	0.0582	0.0746

IAD	0.8195	0.8539	0.6711	0.1696	0.1033	0.1636	0.2052	0.2844	0.2041	0.2636
IAH	0.0503	0.1212	0.2099	0.2268	0.3263	0.3038	0.2365	0.3143	0.4044	0.3300
JFK	0.7813	0.5575	0.7381	0.8376	0.8841	0.9264	0.9552	0.9777	0.9876	0.0118
JNU	0.6597	0.8988	0.9717	0.5171	0.6444	0.3199	0.4228	0.5303	0.5922	0.6131
KOA	0.0017	0.0032	0.0025	0.0025	0.0053	0.0108	0.0082	0.0057	0.0091	0.0100
KTN	0.0014	0.0008	0.0008	0.0019	0.0038	0.0018	0.0036	0.0032	0.0009	0.0004
LAS	0.9333	0.0007	0.0003	0.0008	0.0019	0.0042	0.0002	0.0004	0.0005	0.0010
LAX	0.4046	0.0428	0.0938	0.0021	0.0008	0.0015	0.0012	0.0024	0.0034	0.0049
LIH	0.1697	0.0137	0.0003	0.0009	0.0020	0.0034	0.0049	0.0050	0.0081	0.0089
MCI	0.9190	0.9116	0.3000	0.3680	0.3732	0.4942	0.6105	0.6815	0.4691	0.4307
MCO	0.1853	0.1645	0.3019	0.4282	0.4938	0.2906	0.3930	0.4878	0.5681	0.6605
MSP	0.0050	0.0193	0.0466	0.0820	0.0966	0.0093	0.0092	0.0156	0.0263	0.0064
MSY	0.0012	0.0007	0.0021	0.0009	0.0007	0.0015	0.0026	0.0032	0.0053	0.0074
OAK	0.0275	0.0878	0.0004	0.0007	0.0015	0.0020	0.0040	0.0070	0.0112	0.0175
OGG	0.0522	0.0739	0.1570	0.0292	0.0013	0.0018	0.0024	0.0015	0.0010	0.0019
OMA	0.0003	0.0014	0.0023	0.0056	0.0100	0.0078	0.0048	0.0043	0.0048	0.0083
OME	0.5661	0.7794	0.8851	0.8665	0.8123	0.8044	0.8123	0.5951	0.6291	0.3266
ONT	0.2180	0.4656	0.6734	0.8196	0.8101	0.8555	0.9123	0.9127	0.9471	0.9680
ORD	0.2830	0.0100	0.0262	0.0498	0.0804	0.1070	0.1637	0.2316	0.2764	0.3196
OTZ	0.8798	0.9016	0.9529	0.7839	0.5968	0.7191	0.5498	0.5518	0.3638	0.2482
PDX	0.0247	0.0384	0.0721	0.0939	0.1351	0.2041	0.1254	0.1829	0.2457	0.3180
PHL	0.0320	0.0919	0.1456	0.1052	0.1517	0.1348	0.0870	0.1137	0.1650	0.1840
PHX	0.0380	0.0462	0.0209	0.0129	0.0203	0.0262	0.0456	0.0191	0.0178	0.0291
PSG	0.6764	0.6554	0.8056	0.9124	0.9475	0.9730	0.8576	0.9149	0.8742	0.8753
PSP	0.5486	0.8326	0.9316	0.9207	0.3238	0.3805	0.1166	0.1560	0.1984	0.2362
RDU	0.9835	0.3954	0.5869	0.2534	0.3035	0.3881	0.1762	0.2384	0.3195	0.4056
SAN	0.0056	0.0162	0.0109	0.0190	0.0040	0.0066	0.0101	0.0173	0.0140	0.0223
SAT	0.1379	0.2172	0.3309	0.4882	0.4906	0.0922	0.1433	0.2056	0.2649	0.3438
SCC	0.0775	0.0242	0.0369	0.0678	0.0969	0.0780	0.1238	0.1583	0.2198	0.2250
SEA	0.3242	0.1705	0.2433	0.0842	0.0630	0.0908	0.0458	0.0690	0.1042	0.0778
SFO	0.1763	0.3593	0.2565	0.3177	0.4167	0.4361	0.5405	0.6477	0.7066	0.7820
SIT	0.6808	0.0016	0.0030	0.0072	0.0071	0.0110	0.0113	0.0127	0.0216	0.0335
SJC	0.0289	0.0745	0.0381	0.0132	0.0269	0.0390	0.0659	0.0737	0.1112	0.0247
SLC	0.3799	0.6798	0.6428	0.3622	0.4889	0.5480	0.4043	0.2699	0.3546	0.4423
SMF	0.0346	0.0645	0.0713	0.1062	0.0703	0.0640	0.0686	0.1005	0.1344	0.1879
SNA	0.0192	0.0632	0.0108	0.0155	0.0312	0.0528	0.0668	0.1042	0.1067	0.1488
STL	0.5114	0.5295	0.7129	0.8490	0.9114	0.9508	0.8495	0.9030	0.6920	0.7305
TPA	0.0255	0.0310	0.0034	0.0041	0.0017	0.0028	0.0055	0.0103	0.0178	0.0253
TUS	0.3159	0.3939	0.6013	0.7596	0.8638	0.9236	0.9582	0.9250	0.9460	0.9066
WRG	0.1341	0.3254	0.5111	0.2566	0.1665	0.1690	0.1993	0.1656	0.0588	0.0872

YAK	0.0594	0.1493	0.0816	0.0277	0.0392	0.0447	0.0325	0.0523	0.0785	0.0981
AKN	0.8556	0.9793	0.7412	0.8169	0.8349	0.8799	0.9061	0.9153	0.9463	0.8472
DLG	0.6289	0.7946	0.9171	0.9176	0.9667	0.9040	0.9199	0.9568	0.9400	0.8626
GST	0.3674	0.2557	0.4127	0.5532	0.6801	0.7645	0.8487	0.8023	0.7595	0.8272

II.b.ii p_d– Daily pattern

Similar as with the before we also ran the same tests, view Table II.3 and Table II.4:

Table II.3. Result of the Box-Pierce test for the yearly pattern of Alaska Airlines

Lag	1	2	3	4	5	6	7	8	9	10
ABQ	0.5807	0.312	0.3439	0.4313	0.4183	0.5412	0.6496	0.7481	0.7944	0.8533
ADK	0.5688	0.8041	0.8734	0.9365	0.9688	0.9883	0.9867	0.9945	0.9978	0.9992
ADQ	0.7048	0.4154	0.587	0.6345	0.7266	0.7436	0.6469	0.7448	0.8034	0.8662
ANC	0.0288	0.0424	0.0963	0.1477	0.1759	0.2428	0.0093	0.0006	0.0012	0.0022
ATL	0.2532	0.4996	0.5075	0.6628	0.7805	0.8174	0.8701	0.8983	0.9381	0.9523
AUS	0.3837	0.4998	0.6755	0.5368	0.5614	0.6175	0.6575	0.7499	0.8244	0.7208
BET	0.0938	0.136	0.0517	0.0651	0.1083	0.0758	0.0567	0.0847	0.0746	0.0643
BNA	0.7775	0.345	0.4208	0.5333	0.6752	0.7883	NA	NA	NA	NA
BOS	0.5743	0.4282	0.4683	0.6138	0.6366	0.6152	0.3539	0.4512	0.3959	0.3755
BRW	0.4382	0.5925	0.6684	0.7615	0.7801	0.8638	0.9231	0.9457	0.9684	0.9787
BUR	0.607	0.6703	0.7078	0.5894	0.5062	0.3109	0.3982	0.4593	0.5375	0.5119
BWI	0.6083	0.5551	0.7524	0.8229	0.6552	0.7643	0.7161	0.7454	0.8191	0.8181
CDV	0.2144	0.1942	0.3507	0.5124	0.5657	0.5842	0.6918	0.7735	0.8445	0.8453
CHS	0.9949	0.9122	0.915	NA	NA	NA	NA	NA	NA	NA
DCA	0.1229	0.0533	0.1174	0.2068	0.2253	0.3152	0.3972	0.5025	0.2855	0.3314
DEN	0.1567	0.3465	0.1447	0.2315	0.2659	0.2521	0.2297	0.0695	0.0295	0.0416
DFW	0.2786	0.5395	0.5747	0.7365	0.7111	0.7586	0.8278	0.8934	0.8618	0.801
DTW	0.3043	0.5875	0.7085	0.8464	0.9257	0.9603	0.9826	0.9926	0.9971	NA
EWR	0.6633	0.2933	0.4635	0.5279	0.4155	0.3713	0.4367	0.5186	0.6196	0.7066
FAI	0.0086	0.0274	0.0572	0.1018	0.1568	0.2106	0.1052	0.0419	0.0493	0.0726
FLL	0.4544	0.5779	0.4541	0.2579	0.3795	0.3761	0.4717	0.573	0.671	0.7508
GEG	0.0229	0.0301	0.0643	0.0856	0.0224	0.024	0.0417	0.0677	0.099	0.1234
HNL	0.2326	0.4895	0.6976	0.6975	0.6895	0.7557	0.8363	0.8578	0.8847	0.8061
IAD	0.5655	0.2657	0.1776	0.2885	0.2826	0.3798	0.4907	0.5955	0.6928	NA
IAH	0.1736	0.2226	0.1849	0.2156	0.3272	0.3973	0.3892	0.4825	0.5321	0.6246
JFK	0.6866	0.384	0.5728	0.733	0.847	0.9183	NA	NA	NA	NA
JNU	0.4535	0.2723	0.3979	0.4552	0.4411	0.5678	0.6733	0.7669	0.82	0.8218

KOA	0.3411	0.571	0.4249	0.5827	0.6755	0.7884	0.6588	0.7357	0.7935	0.8119
KTN	0.0235	0.0518	0.1128	0.1456	0.1586	0.0525	0.0852	0.1266	0.173	0.2139
LAS	0.9846	0.0117	0.0302	0.0317	0.036	0.062	0.0564	0.0894	0.092	0.1328
LAX	0.2208	0.465	0.6749	0.8204	0.9064	0.6954	0.7868	0.8475	0.8197	0.8791
LIH	0.0777	0.209	0.1217	0.0588	0.0889	0.1186	0.1519	0.2103	0.2278	0.3003
MCI	0.4055	0.5039	0.5928	0.7535	0.7396	0.8404	0.9008	0.9432	0.8784	0.9155
MCO	0.2609	0.4446	0.6097	0.7658	0.6006	0.6785	0.7787	0.8562	0.9093	0.9285
MSP	0.1031	0.2158	0.2751	0.4031	0.5457	0.571	0.6732	0.7253	0.7468	0.7727
MSY	0.6515	0.9027	0.8349	0.895	0.8997	0.8341	0.8828	0.9081	0.8889	0.9211
OAK	0.6073	0.0789	0.166	0.0791	0.1133	0.1364	0.139	0.2019	0.2724	0.3527
OGG	0.4367	0.6553	0.5993	0.6087	0.7298	0.7795	0.7801	0.8385	0.4849	0.4862
OMA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA
OME	0.7195	0.1524	0.2196	0.2136	0.3183	0.186	0.2682	0.3202	0.4055	0.4987
ONT	0.4015	0.6352	0.8194	0.9201	0.9274	0.9004	0.9125	0.9511	0.9643	0.9563
ORD	0.0024	0.009	0.0236	0.0333	0.0573	0.0971	0.1085	0.0652	0.0689	0.086
OTZ	0.7778	0.3098	0.3952	0.4963	0.3735	0.2834	0.3799	0.4552	0.29	0.3441
PDX	0.8265	0.8009	0.8893	0.9577	0.9829	0.6864	0.7749	0.8483	0.2782	0.3604
PHL	0.1693	0.3755	0.5252	0.2701	0.247	0.3469	0.4387	0.5446	0.644	0.7267
PHX	0.6449	0.4528	0.6555	0.8054	0.849	0.9193	0.9562	0.9768	0.989	0.9887
PSG	0.3665	0.6467	0.8046	0.5351	0.6363	0.4997	0.5774	0.6744	0.5941	0.6858
PSP	0.981	0.602	0.7894	0.8504	0.8361	0.899	0.9393	0.8218	0.7712	0.8391
RDU	0.3469	0.4009	0.4507	0.6112	0.7478	NA	NA	NA	NA	NA
SAN	0.1161	0.1327	0.1123	0.1752	0.2078	0.2986	0.4015	0.4924	0.5789	0.6386
SAT	0.2269	0.4818	0.5927	0.731	0.8048	0.8857	0.9343	0.966	0.8872	0.9225
SCC	0.9909	0.8876	0.9691	0.8407	0.876	0.927	0.963	0.9807	0.9876	0.9942
SEA	0.5823	0.0932	0.0663	0.021	0.0131	0.0088	0.0037	0.0068	0.0025	0.0017
SFO	0.0712	0.0819	0.0627	0.1084	0.1698	0.2367	0.3285	0.4171	0.4721	0.5675
SIT	0.5206	0.7817	0.4198	0.5872	0.7009	0.8095	0.8167	0.8843	0.8767	0.9152
SJC	0.9987	0.9787	0.9662	0.3321	0.4422	0.5674	0.6724	0.7671	0.8248	0.872
SLC	0.2405	0.2669	0.4501	0.4786	0.5734	0.6714	0.7753	0.8131	0.8771	0.9087
SMF	0.0446	0.1256	0.1876	0.0551	0.0862	0.0984	0.0903	0.0676	0.0485	0.0651
SNA	0.0615	0.1721	0.193	0.3055	0.3553	0.425	0.5237	0.5429	0.6399	0.5069
STL	0.6386	0.6804	0.5539	0.663	0.7433	0.7105	0.8077	0.8754	0.9236	0.9553
TPA	0.5912	0.5668	0.7616	0.7442	0.8552	0.7453	0.8315	0.8952	0.9243	0.9142
TUS	0.0494	0.1444	0.2264	0.1344	0.1013	0.113	0.1728	0.1804	0.1055	0.1171
WRG	0.0173	0.0153	0.0273	0.0562	0.0939	0.1452	0.1627	0.2302	0.2942	0.3678
YAK	0.8381	0.4711	0.6208	0.4218	0.3776	0.4429	0.5601	0.5459	0.4653	0.5606
AKN	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
DLG	0.2714	0.5292	0.5614	0.7244	0.6979	0.7476	0.8194	0.8871	0.8569	0.7954
GST	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000

Table II.4. Result of the Ljung-Box test for the yearly pattern for Alaska Airlines

Lag	1	2	3	4	5	6	7	8	9	10
ABQ	0.5332	0.1997	0.1891	0.2284	0.1585	0.2344	0.3125	0.41	0.3597	0.3984
ADK	0.5273	0.7603	0.8179	0.8918	0.9346	0.9711	0.9408	0.9679	0.983	0.9918
ADQ	0.6719	0.3043	0.4505	0.4505	0.5223	0.4649	0.2093	0.2879	0.3095	0.3905
ANC	0.0277	0.0403	0.0922	0.1413	0.1673	0.2315	0.0072	0.0004	0.0008	0.0015
ATL	0.2255	0.4571	0.4373	0.59	0.7138	0.7358	0.7911	0.8131	0.8724	0.8822
AUS	0.3486	0.439	0.6083	0.4023	0.3936	0.4225	0.43	0.5275	0.6198	0.3252
BET	0.0714	0.0956	0.0228	0.0258	0.0453	0.0194	0.0083	0.0134	0.0066	0.0027
BNA	0.7293	0.149	0.1475	0.1744	0.2687	0.3783	NA	NA	NA	NA
BOS	0.5649	0.4054	0.4367	0.5798	0.5932	0.5571	0.2707	0.3575	0.289	0.2573
BRW	0.4198	0.5633	0.6288	0.719	0.7243	0.8184	0.8906	0.9152	0.9462	0.9592
BUR	0.5935	0.6451	0.6715	0.5236	0.416	0.2062	0.2745	0.3192	0.3849	0.3321
BWI	0.5783	0.4838	0.6855	0.7465	0.4582	0.5755	0.437	0.4264	0.5138	0.4241
CDV	0.1868	0.15	0.2845	0.4344	0.4628	0.4474	0.5559	0.6424	0.73	0.6787
CHS	0.9928	0.7591	0.4656	NA	NA	NA	NA	NA	NA	NA
DCA	0.112	0.0428	0.097	0.1752	0.1835	0.2621	0.3337	0.4336	0.1872	0.2172
DEN	0.1477	0.3301	0.1245	0.202	0.2283	0.2069	0.1779	0.0392	0.0126	0.0182
DFW	0.2665	0.5222	0.5476	0.712	0.6719	0.7154	0.7884	0.8634	0.8101	0.713
DTW	0.2356	0.4917	0.5778	0.7405	0.8525	0.8927	0.9418	0.9674	0.9825	NA
EWR	0.6352	0.213	0.3553	0.3867	0.2184	0.1432	0.1657	0.2052	0.28	0.356
FAI	0.0074	0.0235	0.0494	0.0886	0.1374	0.1842	0.0783	0.0247	0.0282	0.0429
FLL	0.4097	0.5011	0.3158	0.1039	0.1739	0.1321	0.1789	0.2423	0.3224	0.3942
GEG	0.0192	0.0239	0.0518	0.067	0.0129	0.0129	0.0233	0.0396	0.0595	0.0732
HNL	0.2231	0.4751	0.6837	0.6743	0.6564	0.7215	0.8073	0.8254	0.8522	0.7396
IAD	0.507	0.1407	0.05	0.0935	0.0518	0.076	0.1169	0.163	0.2277	NA
IAH	0.148	0.1776	0.1265	0.138	0.2234	0.2683	0.2268	0.2964	0.3151	0.3982
JFK	0.6212	0.1829	0.3099	0.4573	0.6026	0.7255	NA	NA	NA	NA
JNU	0.4446	0.2548	0.374	0.4244	0.4008	0.5254	0.6318	0.73	0.7843	0.7767
KOA	0.33	0.5553	0.3943	0.5494	0.6399	0.7581	0.5957	0.6745	0.7347	0.7469
KTN	0.0211	0.0461	0.1018	0.1295	0.1373	0.0379	0.0631	0.0958	0.133	0.165
LAS	0.9844	0.01	0.0261	0.0266	0.0296	0.0516	0.045	0.0726	0.0724	0.1066
LAX	0.2157	0.457	0.6669	0.8142	0.9021	0.671	0.7653	0.8286	0.7927	0.8577
LIH	0.0669	0.1846	0.0929	0.036	0.0548	0.0724	0.0915	0.1312	0.1332	0.185
MCI	0.3752	0.4506	0.5203	0.688	0.6347	0.7539	0.8296	0.8911	0.6915	0.7475
MCO	0.2438	0.4163	0.5761	0.7364	0.526	0.5992	0.708	0.7987	0.8648	0.8829

MSP	0.0858	0.1804	0.2217	0.3334	0.4688	0.4663	0.5668	0.6048	0.5976	0.5987
MSY	0.6187	0.883	0.7583	0.8194	0.7884	0.5889	0.643	0.6461	0.4679	0.501
OAK	0.5988	0.0669	0.1442	0.0604	0.0866	0.1024	0.0993	0.1493	0.2073	0.2769
OGG	0.429	0.645	0.5799	0.5827	0.7054	0.7534	0.747	0.8084	0.4014	0.3943
OMA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA
OME	0.706	0.1168	0.1665	0.1476	0.2303	0.0964	0.1497	0.1775	0.2382	0.3138
ONT	0.3852	0.6133	0.8019	0.9092	0.9101	0.8646	0.8718	0.9229	0.9382	0.9142
ORD	0.0018	0.0067	0.018	0.0246	0.0428	0.0745	0.0787	0.0373	0.0363	0.0445
OTZ	0.7682	0.2679	0.3381	0.4262	0.2771	0.1755	0.2491	0.3054	0.1298	0.157
PDX	0.825	0.7967	0.8855	0.9556	0.9818	0.6616	0.7528	0.8301	0.2342	0.3096
PHL	0.1299	0.3034	0.4275	0.1202	0.0796	0.1259	0.1694	0.238	0.3175	0.3894
PHX	0.6409	0.4416	0.644	0.796	0.8387	0.9123	0.9515	0.9738	0.9873	0.9862
PSG	0.3482	0.6243	0.7837	0.4639	0.5599	0.387	0.4547	0.5523	0.4263	0.5206
PSP	0.9806	0.5843	0.7744	0.8346	0.8117	0.8796	0.9247	0.7675	0.6942	0.7734
RDU	0.2341	0.1918	0.1408	0.2257	0.3402	NA	NA	NA	NA	NA
SAN	0.1115	0.125	0.1027	0.1609	0.1891	0.2748	0.374	0.4621	0.5472	0.6046
SAT	0.1918	0.4266	0.5154	0.6524	0.718	0.817	0.8812	0.9316	0.6246	0.6815
SCC	0.9906	0.876	0.9641	0.7989	0.8329	0.8939	0.9418	0.9669	0.976	0.9877
SEA	0.5796	0.0891	0.0621	0.0187	0.0112	0.0072	0.0029	0.0054	0.0018	0.0011
SFO	0.0676	0.076	0.056	0.0975	0.1539	0.2158	0.303	0.3879	0.4382	0.533
SIT	0.5081	0.7694	0.3761	0.5401	0.6538	0.7701	0.7659	0.8445	0.8209	0.8685
SJC	0.9987	0.9778	0.9639	0.3021	0.4071	0.5306	0.6367	0.7354	0.7958	0.8466
SLC	0.2291	0.2471	0.424	0.4424	0.532	0.6287	0.738	0.7724	0.8446	0.8785
SMF	0.0397	0.1135	0.1682	0.041	0.0648	0.0717	0.061	0.04	0.0245	0.0331
SNA	0.0572	0.1621	0.178	0.2848	0.3283	0.3922	0.4877	0.4985	0.5962	0.4372
STL	0.6157	0.634	0.4611	0.5597	0.6322	0.5452	0.6607	0.7512	0.8263	0.8841
TPA	0.5512	0.4761	0.6762	0.5935	0.7319	0.439	0.544	0.6466	0.6588	0.4277
TUS	0.0376	0.1144	0.1781	0.0802	0.0471	0.0472	0.0785	0.0704	0.0206	0.0197
WRG	0.0114	0.0083	0.0143	0.031	0.0531	0.085	0.0838	0.1258	0.1633	0.2111
YAK	0.8246	0.3908	0.5249	0.2623	0.1882	0.2167	0.3065	0.2329	0.1072	0.1534
AKN	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
DLG	0.2593	0.5118	0.5339	0.6991	0.6578	0.7033	0.7788	0.856	0.8043	0.7065
GST	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000

Sme issues appeared but further analysis showed that the issue with BNA, CHS, DTW, IAD, OMA, RDU AKN and GST airport where due to insufficient data to properly calculate the test values. We decided that we would still use the spline for all airport except

AKN and GST. Here we used no daily pattern. This means that the daily pattern for the whole day is equal to 0.

II.b.iii ε – Error component (stochastic)

Fixing the patterns, we proceeded to fit the finite mixture model for the error component for the AS airline.

Table II.5. Log-likelihood for the finite mixture models of Alaska Airlines

	k			
	2	**3**	**4**	**5**
ABQ	-3437.505	-3436.503	-3436.5	-3419.25
ADK	-955.6579	-954.2507	-951.8311	-948.2419
ADQ	-5790.552	-5722.039	-5701.59	-5699.17
ANC	-156253.7	-155807	-155764.7	-155798.4
ATL	-7932.816	-7925.439	-7922.06	-7920.21
AUS	-5572.404	-5550.579	-5543.952	-5542.936
BET	-10948.81	-10892.65	-10848.52	-10847.31
BNA	-1945.171	-1933.572	-1940.17	-1930.33
BOS	-21046.75	-21001.34	-21001.11	-20988.2
BRW	-11001.17	-10996.99	-10981.84	-10984.84
BUR	-13336.42	-13273.52	-13258.1	-13256.95
BWI	-5887.021	-5883.094	-5865.37	-5861.2
CDV	-8516.986	-8455.07	-8437.51	-8445.93
CHS	-510.459	-506.0968	-500.05	-499.86
DCA	-19650.81	-19606.12	-19604.16	-19605.3
DEN	-20699.33	-20491.74	-20469.17	-20464.87
DFW	-17282.97	-17230.15	-17227.64	-17226.53
DTW	-3964.272	-3955.431	-3951.069	-3951.069
EWR	-10166.84	-10136.83	-10131.18	-10127.05
FAI	-26429.38	-26330.62	-26324.21	-26321.59
FLL	-4804.094	-4801.839	-4787.598	-4782.498
GEG	-18770.43	-18645.79	-18625.49	-18619.45
HNL	-32128.4	-32125.38	-32090.33	-32084.5
IAD	NA	NA	NA	NA
IAH	-5658.077	-5643.636	-5638.184	-5635.158
JFK	-1988.663	-1978.472	-1978.036	-1976.049
JNU	-51662.43	-51429.61	-51373.85	-51363.84

KOA	-17552.91	-17537.49	-17535.47	-17535.69
KTN	-15942.65	-15906.79	-15904.29	-15927.78
LAS	-17854.39	-17770.16	-17767.08	-17757.3
LAX	-35450.75	-35350.75	-35345.01	-35344.6
LIH	-16094.92	-16071.44	-16065.08	-16064.36
MCI	-5695.804	-5668.558	-5668.557	-5664.618
MCO	-9870.977	-9845.982	-9855.07	-9841.762
MSP	-8858.386	-8809.472	-8807.784	-8799.772
MSY	-3948.679	-3938.192	-3937.737	-3925.364
OAK	-28978.22	-28868.7	-28824.35	-28817.12
OGG	-36577.38	-36536.59	-36534.03	-36533.12
OMA	-201.9061	-196.9246	-196.5673	-193.6135
OME	-8077.813	-8049.229	-8046.699	-8044.305
ONT	-9622.643	-9621.529	-9567.13	-9562.843
ORD	-8811.872	-8769.975	-8764.846	-8763.983
OTZ	-8351.39	-8289.001	-8281.257	-8280.71
PDX	-56939.45	-56876.19	-56666.41	-56666.16
PHL	-5069.193	-5058.731	-5051.576	-5046.101
PHX	-30028.32	-29888.13	-29863.94	-29854.84
PSG	-8535.606	-8498.574	-8497.793	-8490.021
PSP	-20229.46	-20132.72	-20111.01	-20106.41
RDU	-2053.325	-2045.993	-2036.297	-2046.204
SAN	-70702.74	-70460.99	-70389.77	-70386.02
SAT	-4539.918	-4530.645	-4528.098	-4524.323
SCC	-10453.54	-10449.44	-10442.19	-10433.71
SEA	-237383.9	-236946.9	-236829.7	-236821.9
SFO	-64726.76	-64381.32	-64323.46	-64297.62
SIT	-15122.92	-15064.87	-15047.98	-15045.75
SJC	-54596.6	-54249.1	-54175.22	-54170.93
SLC	-25929.86	-25693.47	-25667.52	-25650.98
SMF	-26234.73	-26106.5	-26097.39	-26087.01
SNA	-41688.62	-41512.74	-41493.98	-41492.7
STL	-5762.321	-5739.992	-5732.029	-5730.567
TPA	-4166.229	-4150.096	-4145.569	-4147.959
TUS	-4543.551	-4542.248	-4521.897	-4519.138
WRG	-8251.022	-8211.151	-8202.631	-8206.657
YAK	-8432.935	-8417.134	-8412.59	-8412.893
AKN	NA	NA	NA	NA
DLG	NA	NA	NA	NA
GST	NA	NA	NA	NA

The previous problems propagated inevitably into the error component. Hence, for these airports we decided to turn off the random component. Meaning these airports will not add root delay to the flight. Also, here we went for a three-component model for the same reason as before additionally we wanted to have a similar model, i.e. baseline, for both airlines.