UNIVERSITAT POLITÈCNICA DE CATALUNYA
BARCELONATECH
Escola d'Enginyeria de Telecomunicació
i Aeroespacial de Castelldefels

Final Degree Thesis

TFG TITLE: Dynamic Air Holding Program for Delay Assignment applied to Barcelona Airport Case

DEGREE: Degree in Air Navigation Engineering

AUTHOR: Zheng Danlin

ADVISOR: Adeline de Villardi de Montlaur

DATE: September, 2015

Title : Dynamic Air Holding Program for Delay Assignment applied to Barcelona Airport Case

Author: Zheng Danlin

Advisor: Adeline de Villardi de Montlaur

Date: September, 2015

Overview

Nowadays, the aircraft is the most used transport to travel between two very distanced places since it is the fastest way. For these last years, the demand of flights has been increasing immensely, so the airports have been also suffering the same increase in workload. Congestion is the word for the imbalance between demand and capacity. After applying a Ground Holding Program (GHP) optimization process before departure to solve this problem in a previous work [1], in this project the Air Holding Program (AHP) is presented in order to minimize the economic costs once in the landing phase.

The main goal of this final degree project is to implement a basic model of AHP to minimize the delay costs near the single destination airport, Barcelona-El Prat in this case. The model of AHP is based on a dynamic stochastic GHP. The problem is solved using Gurobi libraries.

Firstly, a pre-tactical phase in [1] of GHP is applied to make flights wait on ground according to capacity constraints in the Barcelona Airport. Then, once the flights are already flying towards the destination and entering the analysed area (500 km from destination airport), these flights will perform a Free Route Airspace (FRA) route in order to make them fly a straight route trajectory to possibly shorten the flight duration, which will cause that the arrival queue be again disordered making the capacity constraint again violated. In order to minimize the air delay costs, a second optimization to assign air delay is applied called AHP. Once the flights are leaving this airspace, they will be communicated their final assigned holding. Finally when they are entering the landing phase, they should first pass through the holding procedure to hold the time needed according to the delay assigned.

Some of the conclusions after this research is that if capacity is reduced in destination airport, firstly a GHP can be applied before departure [1]. Then, an AHP can be applied for a second optimization purpose. If traffic is directly taken to be applied an AHP, the capacity flexibility is very reduced, while using pre-tactical regulation the capacity could be as small as desired. This outlines the advantage of using GHP and AHP together. Furthermore, although the main objective is to minimize the delay costs, if the flights are arriving earlier than scheduled in the most optimal case, the delays could be negative values, which means that costs are saved.

Título: Programa de Espera en Aire Dinámico para Asignación de Retrasos aplicado al Aeropuerto de Barcelona

Autor: Zheng Danlin

Director: Adeline de Villardi de Montlaur

Fecha: Septiembre, 2015

Resumen

Hoy en día, el avión es el transporte más usado para viajar entre dos sitios muy lejanos ya que es la manera más rápida. En estos último años, la demanda de vuelos ha ido incrementando inmensamente, por lo que los aeropuertos también han sufrido este aumento de carga de trabajo. El desequilibrio entre demanda y capacidad se denomina congestión. Después de aplicar un Ground Holding Program (GHP) para resolver este problema en un trabajo anterior [1], en este proyecto se presenta el Air Holding Program (AHP) para minimizar los costes económicos una vez entrando en la fase de aterrizaje.

El principal objetivo de este proyecto de final de grado es implementar un modelo básico de AHP para minimizar los costes económicos cerca de un único aeropuerto de destino, Barcelona-El Prat. El modelo de AHP está basado en un GHP dinámico y estocástico. El problema se ha resuelto usando la librería de Gurobi.

Para empezar, se aplica un GHP en fase pretáctica para que los aviones esperen en tierra de acuerdo con la capacidad en el aeropuerto de Barcelona. Después, una vez que los aviones estén volando y entrando en el área de estudio (500 km desde aeropuerto de destino), volarán en una ruta de Espacio Aéreo Libre (FRA) haciendo posible que puedan llegar antes que su tiempo planificado. Esto puede causar que los aviones que estaban ordenados de manera que llegaran de acuerdo con la capacidad inicial, lleguen en una secuencia desordenada, haciendo incumplir la restricción de capacidad. Para minimizar los costes de espera en aire, una segunda optimización de asignación de retrasos en aire debe ser aplicada. Una vez que el avión salga del área de estudio, se le comunicará su ”holding”(tiempo de espera en aire) final asignado, para que cuando lleguen al procedimiento de ”holding”, esperen el tiempo suficiente de acuerdo con el retraso asignado.

Algunas de las conclusiones más importantes después de este estudio es que si la capacidad es reducida, se puede implementar primeramente un GHP antes de su salida [1]. Y una vez que estén llegando, una segunda optimización de espera en aire. Si la optimización de la fase táctica fuese aplicada directamente sin haber pasado por la fase pretáctica, puede conllevar la reducción de flexibilidad en cuanto a la consideración de la capacidad. Si se han aplicado retrasos antes de despegar, la capacidad puede ser tan pequeña como sea la de fase pretáctica. Esto remarca la ventaja de trabajar conjuntamente la fase pretáctica y la fase táctica. Otra ventaja es que aunque la idea principal sea reducir costes por retrasos en aire, si estamos en el óptimo caso de que los aviones llegan antes de su tiempo planificado, se puede llegar a ahorrar costes económicos respecto el plan de vuelo planificado.

ACRONYMS

AA: Absorption Areas

AAR: Airport Acceptance Rate

AHP: Air Holding Program

AIRAC: Aeronautical Information Regulation And Control

AMAN: Arrival Manager

AMSTAR: Airborne Merging and Spacing for Terminal Arrivals

ATC: Air Traffic Centre

ATCo: Air Traffic Controller

ATFM: Air Traffic Flow Management

ATM: Air Traffic Management

CDM: Collaborative Decision Making

CTA: Controlled Time of Arrival

ETA: Estimated Time of Arrival

FRA: Free Route Airspace

FSFS: First Scheduled First Served

GDP: Ground Delay Program

GHP: Ground Holding Program

GHPP: Ground Holding Policy Problem

ID: Identification

LP: Linear Programming

NATS: National Air Traffic Services

OPS: Air Operations

RBS: Ratio by Scheduled

STAR: Standard Terminal Arrival Route

TFM: Traffic Flow Management

To those who are still amazed
by the flying airplanes in the sky.

CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGEMENT

After four years of effort and work, this thesis is the best product of my interest, imagination and will. The world of Air Traffic Management (ATM) has always been a very attractive, unknown and endless field to be explored. Maybe this thesis is not even a million part of it, but I am totally grateful to own it and to be just a million part of the world I have been always amazed by.

When it comes the time to acknowledgements I am not sure from whom to start with. All I can say is that two persons made a crucial and essential roll in this thesis production.

The first person is my advisor of the thesis, Adeline Villardi, who also had been my professor in many subjects related with ATM. She was the first person to introduce me into the world of ATM and thus, to this thesis about Air Holding Program (AHP). She proposed me this idea of research and opened so many unknown doors that I had never dreamt of. As the project proceeded, I have learnt so many things that I have never thought they could even exist. This should have been the best way to explore this world so fantastic to me. Therefore, I thank her for giving me the chance to prove myself that I can do things I never thought of and for choosing me to be her assistant to make real this research dream.

The other person who deserves the same gratitude is Guillem Tarín, who had been my partner during all these months together in the thesis, since we have been side by side to make possible her initial thoughts about this research. He has been the best mate I have ever got, he was the person who helped me when I could not continue with this project, when I thought it was impossible. Both together, we did it and I thank him for all his patience and work.

Despite the help they gave me in this thesis, I would also like to thank my parents, for their patience and understanding of what I was facing to during these months. For their company and spiritual support day after day. And finally, I would like to thank my younger sister, who did not do anything but staying with me all the time, making me feel warmth and at home.

This thesis is dedicated to all my family and friends, to whom loves me and I love.

INTRODUCTION

Nowadays, the fastest, safest and therefore the best way to travel from a place to another of the world is using an aircraft. In these last decades, where the technology has been improving day by day, it is not surprising that the most used transport is in the air. For these last years, the demand of flights has been increasing immensely, so the airports have been also suffering the same increase in workload.

Since the capacity, a number of aircraft during a period of time within an area, either in the air or at the airports is limited, what could be improved to benefit from the maximum capacity is how the management of these flights are done. The airspace is divided into several sectors, each one being supervised by the Air Traffic Controllers (ATCo) and they have their maximum handleable workload. The capacity of each sector comes from this workload. The same could be said for the airports, since the tower controllers have the same limitations. Therefore, what should be studied with more importance among all other possible improvements to the Air Traffic Management (ATM) is how to arrange all the demanding traffic according to the capacity of the sectors, airports, etc.

Congestion is the word for the imbalance between demand and capacity. After applying a Ground Holding Program (GHP) optimization process before departure to solve this problem in a previous work [1], in this project the Air Holding Program (AHP) is presented in order to minimize the economic costs once in the landing phase.

In this Final Degree Project, the studies were done for a single airport model. In this case the Airport of Barcelona (LEBL) is chosen due to its proximity to the research laboratories and a major knowledge of its infrastructure.

The Ground Delay Program (GDP) is used as a solution method for congestions when destination airport has a decrement of capacity. It is a traffic management procedure where aircraft are delayed at their departure airports in order to manage demand and capacity at their arrival airport. The Ground Holding Program (GHP) is finding the most optimal way to apply the Ground Delay Program. If there is a congestion so the aircraft has to wait in the air when arriving at destination airport due to capacity constraints, why not make it wait on the ground, since it reduces costs of fuel and safety with respect to air holding.

The optimal method taking into account the meaning of delays to airlines, airspace and passengers is studied in a previous work as the pre-tactical phase (before departure) of this new GHP concept [1]: the pre-tactical phase study was a static (once and for all flights of a day) model focusing its attention on the cost function of the GHP when assigning new slots of departure to flights that should be delayed due to capacity limitation.

Despite this solution applied before departure, the performance of a flight during the whole trip is unpredictable. Therefore, when arriving at destination airport and the airspace and airport are still congested, then the Air Holding Program might be the most direct and simplest method to at least minimize these air delays.

Unlike the pre-tactical phase model of GHP, in the present document where the tactical phase model of Air Holding Program is presented, this model is using an AHP in a dynamic concept. Once the flights have been submitted to the pre-tactical phase optimization [1] and are arriving at a certain airspace between two radii near LEBL, the aircraft are supposed to manage a free-route flight with a calculated average speed. Since the regulated slots of arrival have been disordered because of flying in straight line with free-route (they will not arrive as regulated according to capacity), this new tactical model of AHP is optimizing again the delays for those flights for the same capacity.

The idea of this model is a simulator that optimizes the delay according to cost function of each flight inside this airspace. Every time that a new flight enters, the simulator is supposed to assign again the delays considering this new aircraft.

The aircraft is supposed to fly with the average speed until holding point according to the Standard Terminal Arrival Route (STAR) procedure of LEBL, where it should make the assigned air holding.

As a summary, once the studied flights of a given day have been delayed according to the pre-tactical model of GHP [1], the main goal of this study is to delay for a second time their arrival times making air holdings if necessary, which is due to free-route concept applied.

Note that since this research is merely a theoretical and strategical first approach to an AHP model, it does not suppose any environmental impact to the nature or the society.

In the next chapters, the statement of the art review, the whole process of data acquisition, problem and hypothesis statements, mathematical formulation, final results, conclusions and future work will be accurately explained.

CHAPTER 1. STATEMENT OF THE ART REVIEW

In recent years, the research on the Air Traffic Management (ATM) has been increasing in a fearsome speed. The most studied problem is the congestion caused by the imbalance between flight demand and airspace and airport capacities.

The main goal of this research is focused on minimizing the costs of air delay by applying Air Holding Program (AHP) in a tactical phase.

The idea of tactical phase is different in our study: the pre-tactical phase is considered as the time window of up to 7 days before the day of departure and the tactical phase the day of operation, as explained in [2]. But in our case, the tactical phase starts when arriving at 500 km from destination airport, in an en route time window.

The tactical time window for delay optimization (make the aircraft wait in the air) is not as studied as the pre-tactical time window (make them wait on ground) since it means more economic costs. But as a general review, it is recommendable to search for the previous works and studies done for the both concepts, as well as for the general congestion solutions presented.

One of the studies to solve congestion problems is presented in [3], they consider reducing delays due to congestion applying a method of trade-off between the capacity rate of departure and arrival. That is, under Collaborative Optimization, arrivals and departures are considered jointly and are treated as interdependent operations. When arrival capacity can be traded for departure capacity, Collaborative Optimization provides the best allocation of capacity to arrivals and departures to maximize the airport throughput and minimize delays. The resulting solutions provide the best capacity allocation strategies that take into account user priorities (e.g., preserving integrity of banks at hub airports). This presented optimization model can be used as a decision support tool for solving Traffic Flow Management (TFM) problems at airports, which is the same as our study, providing support tool for deciding delays assignment but in a tactical window.

The other study found dealing with this problem is explained in [4], where an algorithm, called compression, restores system efficiency by moving flights earlier in time (and never later) to fill arrival slots vacated by cancelled or delayed flights. This is the exposed idea of Slot Exchange Mechanism.

Regarding the slot allocation process in a pre-tactical (before departure) time window, many studies are found about this method: in [2], they use a new concept of Absorption Areas (AA) to absorb flights not using their slot due to disturbances such as weather or capacity constraints, by leaving unfilled slots during slot allocation process. Their results showed that it is a interesting way to improve air traffic throughput management.

Some other studies focused their interest in predicting delays, which is slightly different from what is studied in this document. Here the delays are assigned regarding capacity constraints, and in this study [5] they present a new model for predicting departure, en-route, and arrival delays before departure. What is interesting for our research is that they also used the real European flight information ALL_FT+ data set for modelling process. Results show that arrival time can be predicted with 70 % confidence and ± 30 minute tolerance 3 hours before departure.

But if the tactical (en route flights) time window is considered, studies are found such as in [6]: four algorithms have been designed to reduce the amount of delay in the terminal phase of light. The main idea here is to transfer and eliminate delay. They propose a strategy for transferring data between the systems and command centres, which are taken as the strategy proposed in this study regarding data transmission, and develop a process for assigning Controlled Times of Arrivals (CTA) to flights en route. Each algorithm is designed to prioritize criteria of fuel savings, throughput and equity but assigns different weights to these criteria. Their analysis suggests that RBS-based, that is landing slots are assigned in a first scheduled, first served (FSFS) manner known as Ration by Schedule (RBS), algorithms provide strong throughput performance. This improvement in throughput is likely achieved, however, at the expense of fuel usage. This study is very similar to ours. They consider flights when arriving at 1000 nmi (500 km for us) from destination airport and start the slot assignment process when reaching 500 nmi (50 km for us). They update this information every 15-30 minutes, we do it every time that a flight enters the 500 km radius, which is much more precise and realistic. The whole simulation is an iterative process to assign dynamically new slots (delays from scheduled planning times).

Another paper [7] talked about the new airborne precision spacing model at terminal arrival routes. In other words, they studied the methods to space in time or distance the aircraft from the threshold, which is almost the same as we do but from 50 km of destination airport. This study investigated Airborne Merging and Spacing for Terminal Arrivals (AM-STAR) performance in long arrival sequences composed of diverse aircraft types ranging from light jets to heavy transports. Three arrival routes with two merge points were modelled, and two different merge frequencies were evaluated. Results of the study indicated that inter-arrival spacing was achieved within 10 seconds even with a diverse fleet of aircraft types having dissimilar final approach speeds and unequal spacing assignments.

Reviewing the dynamic solution to GHP, in the study [8], they commented that research on probabilistic models of the Ground Holding Policy Problem (GHPP) has been limited to static solutions, producing ”once and for all” ground-holds at the beginning of daily operations. In their thesis, they present first a dynamic solution to the probabilistic GHPP based on a dynamic programming algorithm that exercises control on individual planes by deciding whether or not flights should be allowed to depart at each time period. The algorithms allow for general ground-hold cost functions for several aircraft classes and the cost of air delays is assumed to be identical for all planes, which is not the same as our study, where each flight will have a different cost for air delay due to fuel consumption. As a result, the dynamic stochastic programming algorithm performed significantly better than the static motivating the development and implementation of a very fast dynamic heuristic that works with a deterministic forecast. With this, the consistency of our dynamic stochastic (due to randomness of parameters such as Delay Noise from pre-tactical phase, as it will be explained in the main core of the document) model is ensured.

Unlike the previous research, in [9] and [10], studies about cost functions are explained. The foundational principle of all basic and fair slot allocation processes is known as Ration by Schedule (RBS). The airlines consider this method to be fair. But in our cost functions more factors are considered to assign delays since RBS method is fair enough but not the most optimal method. Therefore, the airlines would also take advantage from our cost function, since they will save economic costs. For this study, they think that the equitable treatment of passengers is a topic much less considered, even though passengers are also important stakeholders in the National Air Traffic Services (NATS). In the same way, in a chapter of [11], Hanowsky considers flight cost functions that are proportional to the number of passengers per flight.

Regarding weather limitations in [12], an optimal algorithm is presented to update ground holding decisions as time progresses and more accurate weather (hence capacity) forecasts become available. In this study it is shown that the weather factor is a crucial limitation that directly affects the capacity for all stages of the flight. For instance, this research focused on a weather variable factor environment. The same could be found in [13], where Nilim et al. addressed weather related uncertainty in airspace condition in routing individual flights. They formulated a robust Markov Decision Process problem for dynamically re-routing an aircraft across convective weather impacted region. Decisions in their problem are aircraft speed and heading. Nevertheless, in our study the convective weather is not taken into account in order to not complicate the scenario in a very basic idea of dynamic AHP.

Finally, in the academic literature, ground holding in Air Traffic Flow Management (ATFM) for a single airport was first introduced by Odoni in 1987 in [14], but as all previous research on this GHP problem, it has been restricted to the single-airport case, which neglects ”down-the-road” effects due to transmission of delays between successive flights performed by the same aircraft. This study has been done in [15], which was in a more realistic environment and the multi-airport case is of a major importance. Since each of a large number of aircraft performs more than one flight on any given day, ”network”(or ”down-the-road”) effects may be important: when a specific aircraft is delayed, in many cases the next flight performed by the same aircraft will also be delayed. Moreover, at a hub airport, a late-arriving aircraft may delay the departure of several flights, given current airline scheduling practices which emphasize passenger transfers. But in this study, huge scenarios has its inconvenience, which is of being a deterministic study (they presented the capacity as deterministic to simplify the problem statement), when a simpler scenario would allow stochastic modelling such as in our case (noise factor will be used to simulate real flight performance).

After this brief summary of solutions studied in recent years for congestion problems and delays optimization, it is proven that almost all the researches and works were done for the time window before departure. The tactical time window (en route) solutions were rarely found, which makes more important and interesting this thesis on this field.

Once the overall situation of the research on these ideas are reviewed, in the following chapters the main content of our study should be explained in detail.

CHAPTER 2. EUROCONTROL DATA PROCESSING WITH AWK

Before starting with the optimization process, it is necessary to obtain the information data of the flights that will be used and analysed using Matlab program. Since the goal is to assess the air traffic in Europe, the Eurocontrol flight data information is chosen to be considered as input data in this thesis.

Eurocontrol flight data information is a source available to public. The only requirement is to be a related person such as students or researches, for industrial or academic purposes. The complete process to be followed to obtain the user account, to subscribe to DDR2/3 repository data and then be able to download the flight data is explained in the appendices (see A.1.). The file format of the flight data studied is ALL_FT+, since it has the most complete and detailed structure of the data.[1]

Once the ALL_FT+ flight data files are downloaded and ready to be used. The next step is to filter the huge information it contains and only save the ones needed as Matlab input data. Since these files are big size files (more than 1GB), it is recommendable to use AWK software to read line by line the huge file and filter the data needed. The AWK utility interprets a special-purpose programming language that makes it easy to handle simple data-reformatting jobs. Therefore, it is the best option when dealing with big files with many unnecessary additional data.

AWK script

Because of the huge data to be filtered, it is recommendable to write a script[2] code with all the requirements to be met to obtain the input data for Matlab.

As a first approach to the optimization model pretended to be studied, the input parameters in the script should be for a single-airport case. Thus, the hypothesis considered are:

- Destination Airport: ”LEBL” (single-airport case for Barcelona Airport)
- Origin Airport cannot be ”LEBL”. Avoids the flights that depart and also arrive at the same airport (LEBL) because these ones shall not be considered as commercial flights.
- Origin Airport cannot be less than 50 km in distance to LEBL.
- Date to be analysed should be defined every time the script is executed: ”20150524”. This makes that only the flights with Estimated Time of Arrival (ETA) on this day are considered. Departure times of these flights might be from a day before (long-haul flights).

[1] The last update of this format is for DDR3 repository, so the days to be analysed will be always from April 2015 onwards. This is the most detailed and corrected format of ALL_FT+ files.

[2] Note that the script is compatible with the previous thesis of the pre-tactical phase, which needs the departure flight information from Barcelona. Thus, in the script it will appear code only useful for pre-tactical phase thesis.

- Entry Point Radius from LEBL: 500 km. From this point of the trajectory, the aircraft is considered for optimization process making a free-route, this is, to fly in straight line towards LEBL.
- Exit Point Radius from LEBL: 50 km. When arriving at this point flying with free-route, the delay (negative or positive) of this flight is assigned.
- These parameters shall be changed for any other single airport case, modifying for different airports, dates and tactical area of optimization.

The idea is to save the flight data of the aircraft entering this studied area near Barcelona airport to be able to make a sequence of arrival flights according to capacity constraints of the destination airport. To do so, the script is coded to read line by line the ALL_FT+ data file, where each line is the whole information of the trajectory of one aircraft from origin to destination airport, which should be Barcelona airport. Each line contains many columns with different data. The idea is to filter the ones needed and calculate others for the next steps in Matlab programming.

For each line the script will search for the first point of the flight trajectory entering the Entry Point radius and save its point information. When the Entry point is located, then the script goes for the point of the trajectory that arrives to the Exit Point radius, and again saves the point information.

The coordinates of these points are given in degrees minutes and seconds, for instance ”385222N0012223E”. Firstly these units should be converted to decimal degrees and afterwards to radians. Once this is done, to convert these units to kilometres from destination airport in straight line (i.e.: ”276.456 km”) the ”haversine” formula [16] to calculate the great-circle distance between two points – that is, the shortest distance over the earth’s surface – giving an ”as-the-crow-flies” distance between the points is used.

The mathematical formulation is shown below:

$$a = \sin^2\left(\frac{\Delta\varphi}{2}\right) + \cos\varphi_1 \cdot \cos\varphi_2 \cdot \sin^2\left(\frac{\Delta\lambda}{2}\right) \tag{2.1}$$

$$c = 2 \cdot \arctan\left(\frac{\sqrt{1-a}}{\sqrt{a}}\right) \tag{2.2}$$

$$d = R \cdot c \tag{2.3}$$

Whereφ is latitude, λ is longitude, R is earth’s radius (mean radius = 6371 km), $\Delta\varphi$ is the difference between the latitudes of the point and the LEBL airport. The same with the $\Delta\lambda$ as difference of longitudes[3].

The test of this method’s feasibility is shown in figure 2.1, where it can be proved that from ”385222N0012223E” (Ibiza Airport) to Barcelona Airport is 149 NM, which is about 276 kilometres in straight line.

When all these information are saved, it continues with calculating the average speed of the flight inside this area by getting the distance in scheduled trajectory between the Entry

[3] Note that angles need to be in radians to pass to trig functions.

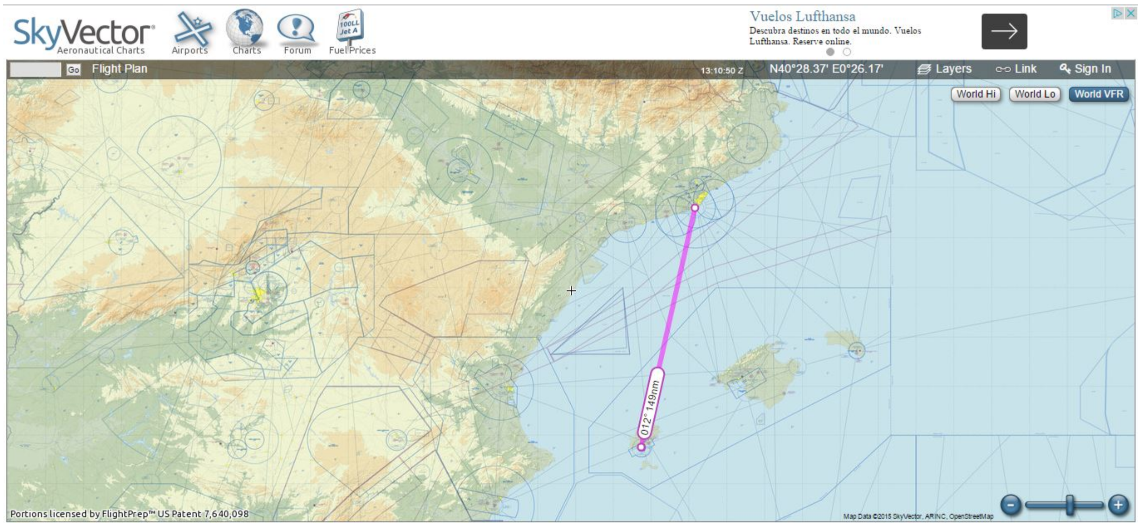


Figure 2.1: Skyvector testing Harvesine formulae veracity [17]

and Exit Point in kilometres and also the time when arriving at each point to obtain the duration to travel this distance in trajectory:

$$AverageSpeed(km/h) = \frac{Distance(km)}{Duration(hours)} \tag{2.4}$$

This speed is used as the average speed the aircraft will fly at in straight line from Entry to Exit Point.

Using the Harvesine Formulas the distance from Entry Point to LEBL and the distance from Exit Point to LEBL in straight line in kilometres are calculated. And then calculate the distance between these two points in straight line as simple as:

$$Distance(km) = DistanceEntryPointtoLEBL - DistanceExitPointtoLEBL \tag{2.5}$$

When this is done, the next step is to calculate the time needed to fly from Entry to Exit Point in straight line:

$$Duration(hours) = \frac{Distance(km)}{AverageSpeed(km/h)} \tag{2.6}$$

Finally, it only remains to compute the new arrival time to Exit Point flying in straight line from Entry Point. This is as easy as getting the Entry Point Time (hhmmss), convert it into hours, sum the duration of equation 2.6 to this time and later convert it again into hhmmss format.

When this new arrival time to Exit Point is calculated, this should be the new time to be considered as Exit Point Time for the Matlab optimization process to assign delays according to capacity and cost function.[4]

Since in the Eurocontrol database they made a floor round to all trajectory distances, the flights that also fly in straight line inside this area by scheduled trajectory will have a slightly

[4] Note that scheduled and real trajectory data will be saved to generalize the problem, if real flight data is empty, all the data saved for real data will be -1.

shorter distance than the calculated one (see Equation 2.5) when they are supposed to be the same distance. This will cause that the duration calculated in 2.6 will be also slightly longer, which will mean that in these cases their New Exit Point Time will be some few seconds later than their Scheduled Exit Point Time. But this problem will be solved when processing with Matlab Program.

Like the Eurocontrol data acquisition process, the complete explanation of AWK utility, script code and output files format are available in A.2..

Furthermore, in order to verify the consistency of the data of the output files, tests with NEST are done to check if the data filtered correspond to the official ones in NEST (see Appendix A.3.).

When all the information needed are saved and accessible for Matlab as input data, in the following chapter the main optimization process will be explained in detail.

CHAPTER 3. DYNAMIC AIR HOLDING PROGRAM USING MATLAB

The concept of Dynamic Air Holding Program in a Tactical Time Window is dynamically assign delay to make flights hold in the air when searching to minimize the costs of air delay near the airport due to capacity constraints.

As explained in the introduction, the proposed solution for cost minimization is air holding at tactical phase (500 km from destination airport) taking into account a cost function. In the following sections, the theoretical concept of this optimization method, the hypothesis taken into account, the mathematical formulation, the parameters to be analysed and the whole optimization process are explained in detail.

3.1. Problem statement

Firstly it is necessary to understand the main concept of using AHP for delay assignment optimization, together with the idea of introducing free-route to make this model possible.

3.1.1. Theoretical concept of Air Holding Program and Free Route Airspace

First of all, before starting with the concepts of this research, let's remind what was done in a previous step [1] since the major advantage of this research idea is the collaboration between pre-tactical and tactical phase. Therefore, understanding what had been studied on the previous thesis is crucial.

A main focus of research in ATFM is how to respond when inclement weather or other circumstances reduce the number of aircraft that can safely land at an airport. The problem of how much ground delay should be assigned to each flight during a regulation is typically referred to as the Ground Holding Problem (GHP). This is, when optimization is considered.

In the GHP model, each flight has a cost (normally expressed in units of €) depending on different parameters such as fuel, connections, passengers, etc.

In the pre-tactical phase, the GHP implemented for flights arriving at LEBL had a cost function taking into account the number of passengers according to aircraft type and also the turn around time of the aircraft once landed at LEBL.

Now, in this thesis, for the tactical phase, since the aircraft is already flying towards the destination airport, the cost function is trying to minimize the delay, for the safety because the more time it spends in the air the more dangerous it is, and the fuel consumption at holding altitude (normally 1500 ft).

The software used to optimize this cost function, in this case to minimize it, is Gurobi Optimizer. It allows users to state their toughest business problems as mathematical models, and then automatically considers billions or even trillions of possible solutions to find the best one. The solver can be used as both a decision-making assistant, to help guide the choices of a skilled expert, or as a fully automated tool to make decisions with no human intervention [18]. In this thesis, this optimizer is chosen because of its speed when computing solutions compared with traditional Matlab Optimizing Functions.

Another concept is very important to make feasible this air delay idea, the Free Route Airspace (FRA). FRA is a concept of providing air traffic services in which an operator can choose its route subject to only a few limitations (e.g. fixed entry and exit points and the need to avoid danger areas) as opposed to the situation where standard airways should be used. In most cases the straight line between an entry point and an exit point will be chosen. If for some reason this is not appropriate (e.g. a danger area needs to be avoided) additional turning points can be specified. These can be navigational aids, published navigational points or points with specified coordinates. The following figure 3.1 gives an overview of the main FRA rules:

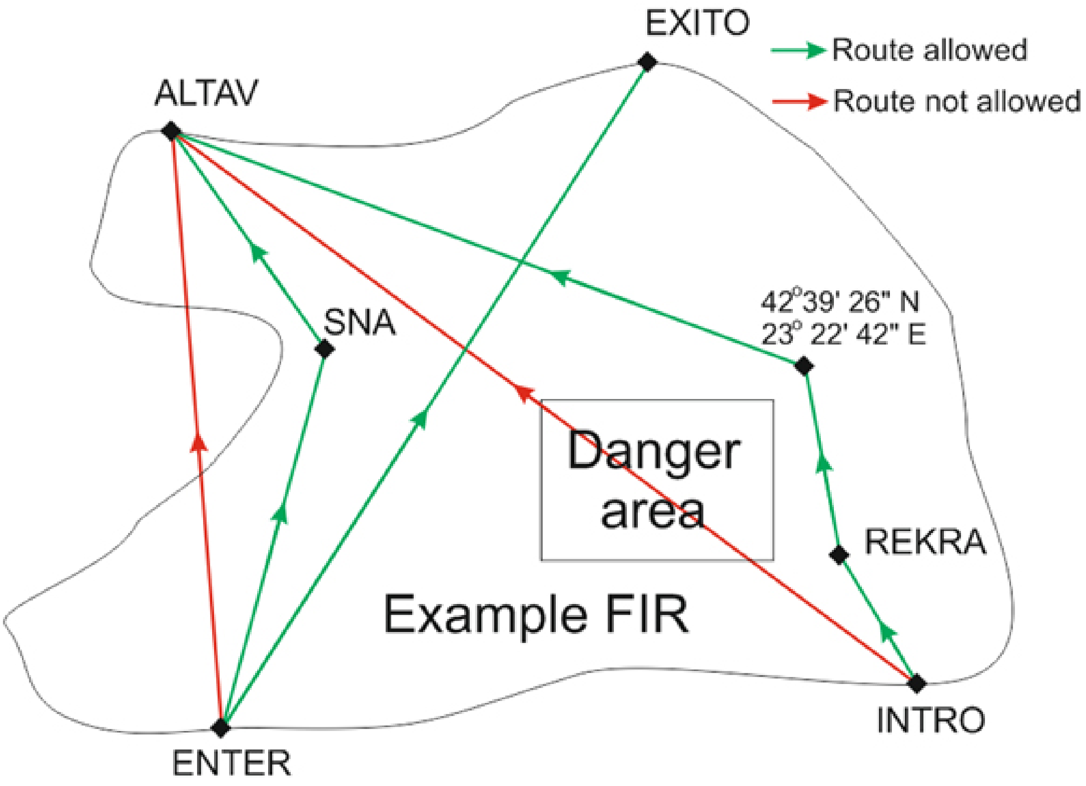


Figure 3.1: Example of allowed and not allowed FRA routes to be considered during the pre-flight planning [19].

In the example Flight Information Region (FIR) depicted, INTRO and ENTER are entry points, ALTAV and EXITO are exit points, SNA is a VHF Omnidirectional Radio Range (VOR) and REKRA is an Area Navigation (RNAV) point. When FRA is implemented, the green routes would be accepted and the red routes would be rejected by the Air Traffic Centre (ATC) flight plan processing system. The reasons for rejection include the crossing of a danger area (INTRO-ALTAV) and the requested route not remaining within the FRA (ENTER-ALTAV). The approved routes can be either direct from an entry to an exit point (e.g. ENTER-EXITO) or with intermediate points (navigational aids (SNA), published points (REKRA) or randomly selected points) [19].

This idea is considered because it is the easiest way to fly between our Entry and Exit

Points. Moreover, it is the fastest way too, when flying straight route path. This makes possible that the flights with not straight scheduled trajectory till destination could also fly in a shortest route path, which is a straight line.

Having flights that could arrive earlier than scheduled time (considering regulation and noise) makes necessary the optimization assignment for a second time, because the flights that were supposed to arrive according to regulation applied in the pre-tactical phase are now arriving in a disordered queue, which means incompatibility with the capacity of the airport. Moreover, it also makes possible the assignment of negative delays, that is to say, to land earlier than original ETA and which means to save time and fuel. Furthermore, since this simulator is intended to be applied by the destination airport involved stakeholders such as Arrival Manager (AMAN) when the flights are already arriving, if the capacity of the airport at that moment is increased, the model is able to adapt its AAR to the actual capacity and again save economic costs making them arrive earlier in the best cases. This will be explained in detail in the next sections.[1]

In the figure 3.2, this FRA idea is shown. The aircraft with ID BAW321 has a scheduled trajectory with estimated exit point time at 9:15 (already taking into account the regulation delay and noise) but if the FRA is applied, it is flying in a straight line till exit point, then the new exit time will be 9:10.

The flights arriving at a radii of 500 km from LEBL are considered in the list of optimization assignment, in the case of the figure 3.2, there are 3 flights in the list. The optimization is executed every time a flight enters this radii. In this way, all the flights inside this area will be considered together to be assigned delays. This means that each time that optimization is done, the delays for each flight are different from their delay assigned in the previous optimization, since the new flight that entered the area could have changed the cost function. In this example case, the newest flight is KLM591 and enters at 09:00 (Entry Point Time = Actual simulation time) and its entrance to the airspace could have changed the delays assigned to other 2 flights in the previous optimization. In this case, the delay of the flight SWR352 was 5 minutes before KLM591 entered and now its delay is 10 minutes when considering also KLM591 together with others.

Once the flight arrives to Exit Point, the pilot should be communicated the holding corresponding to the delay assigned in the last optimization (10 minutes for SWR352 if after optimization of ”i” iteration it leaves the airspace) and it should fly at the same average speed calculated till holding procedure, where it will hold in air the time needed according to the delay assigned. This flight will be deleted from the list of optimization flights since it is already outside the area, leaving 2 flights in the area in this case. This is the idea of dynamic in this thesis.

According to this idea, each flight will have a delay assigned so that every flight should hold the time needed to minimize the total cost of air delays due to capacity constraints at LEBL airport, in this case.

3.1.2. Hypothesis with the corresponding mathematical formulations

The main idea of this thesis is to generate a fast-time-simulator for tactical time window of delay assignment optimization.

[1] This is only applicable to situations where AAR is increased with respect the pre-tactical AAR considered.

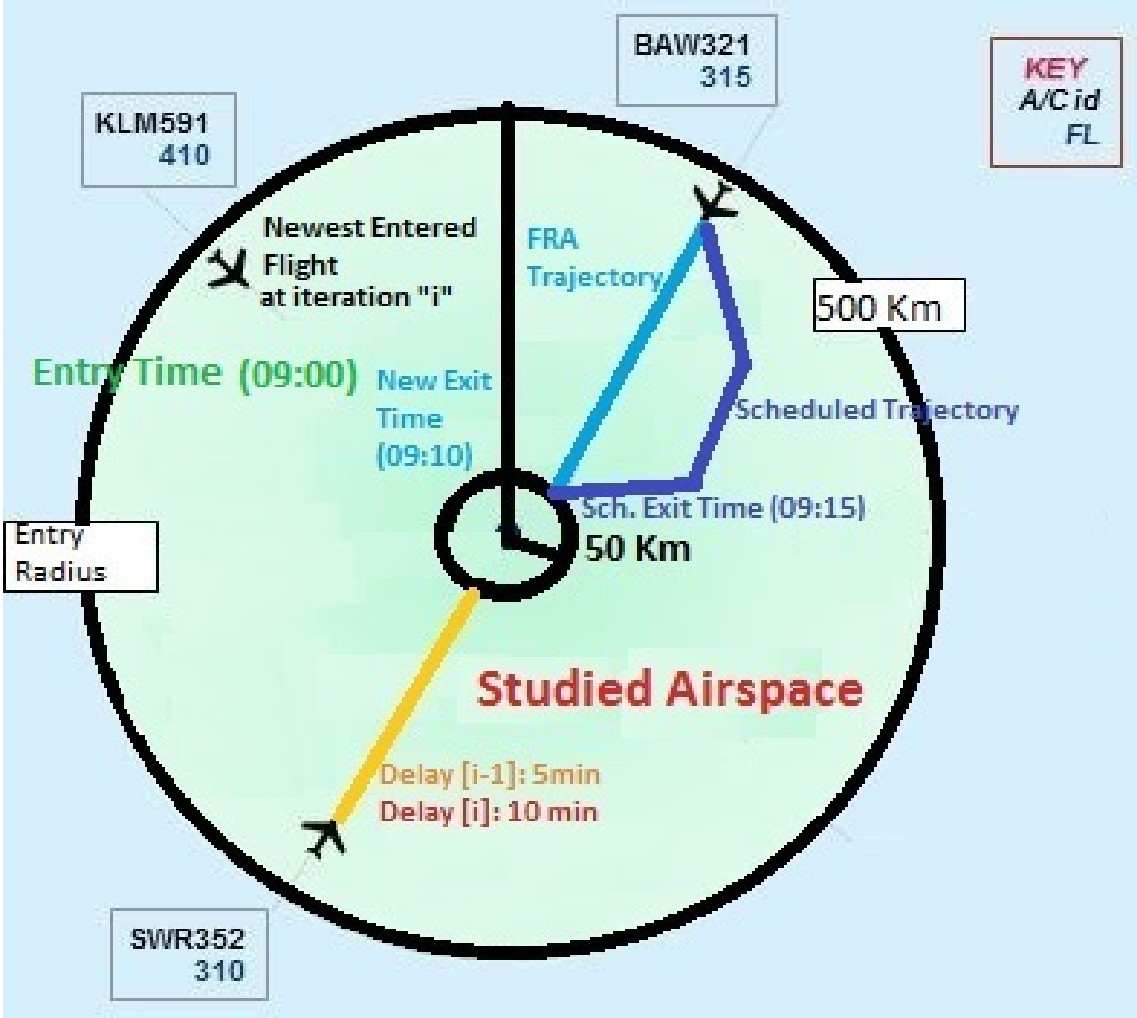


Figure 3.2: Example case of AHP and FRA concept

In the previous thesis [1], in a pre-tactical time window, the idea was to generate a regulation to the flights arriving at LEBL in a determined day before their departure time according to the Airport Acceptance Rate (AAR) of LEBL, which is its capacity, from the scheduled flight plan.

In the language of Eurocontrol database, the M1 is the flight plan of the scheduled flights, the M2 is for the regulated flights by the Air Traffic Flow Management (ATFM) centre and M3 is for the real flight performed, see figure 3.3. Therefore, in the pre-tactical thesis, the idea was to create a M2 trying to minimize its own cost function from a M1 downloaded from Eurocontrol DDR2/3 repository data source. The result was a list of delays assigned to each flight of M1, that is the same as a list of new departure slots for each flight, making its own simulated M2 model.

In this thesis of tactical time window, the idea is to create a simulated M3 model for the flights of M1 adding the regulation (simulated M2) of the pre-tactical phase. That is to say, first read M1 flight information from Eurocontrol, then add the delays of pre-tactical phase creating a M2 and add a noise factor to M2 to create a virtual M3, see figure 3.4. This virtual M3 is the one considered before entering the 500 km radius.[2] From this virtual

[2] From here onwards, all the times are referred as the M1 times plus the regulation and noise.

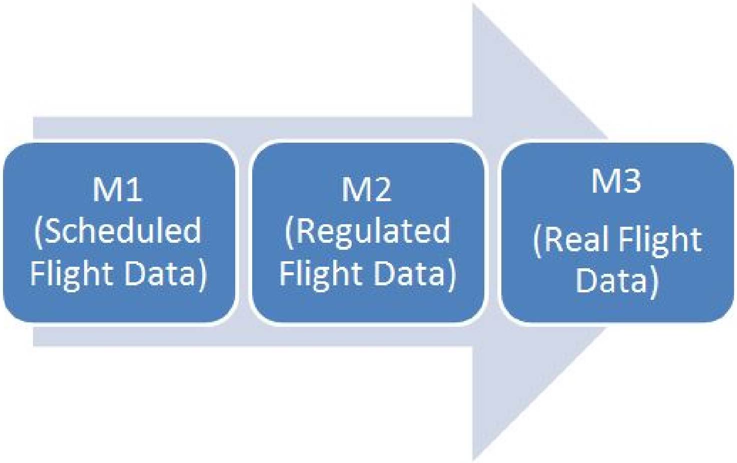


Figure 3.3: M1, M2 and M3 Flight Data Model of Eurocontrol

M3, apply the AHP optimization model to assign delays in air for these flights, obtaining eventually a simulated M3, see figure 3.5. **This AHP model is applying delays with respect the virtual M3 flight plan. This means that all the delays calculated and shown are tactical delays.**

For an easier understanding of these concepts, an example of the Estimated Time of Arrival (ETA) evolution for 3 flights is shown in figure 3.6. The delay shown in the last step (Simulated ETA) is the total delay assigned to the aircraft with respect the original ETAs. But the delay calculated in this study is only the delay with respect the Virtual ETA, in this case 1 minute for ID1, 3 minutes for ID2 and 1 minute for ID3.

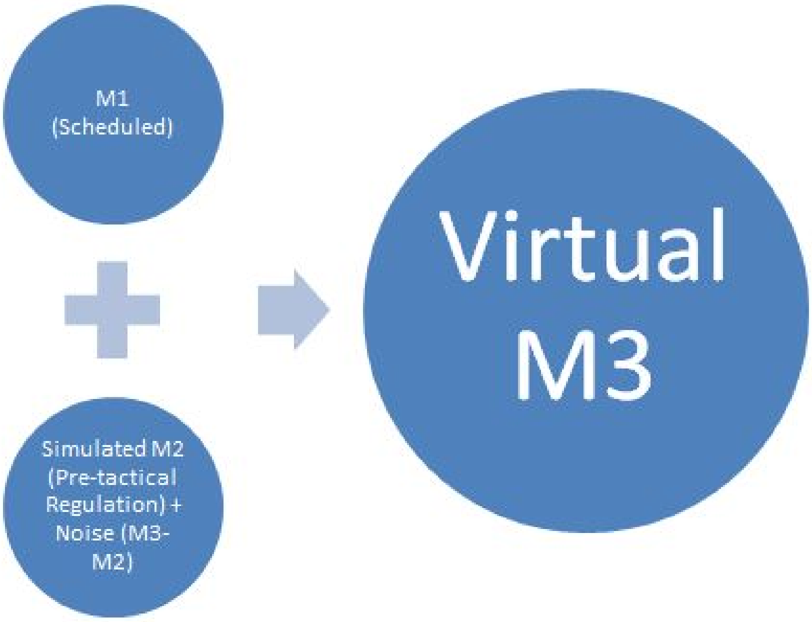


Figure 3.4: Process of creating Virtual M3

The model of AHP is based on a stochastic GHP due to the fact that the Noise Factor

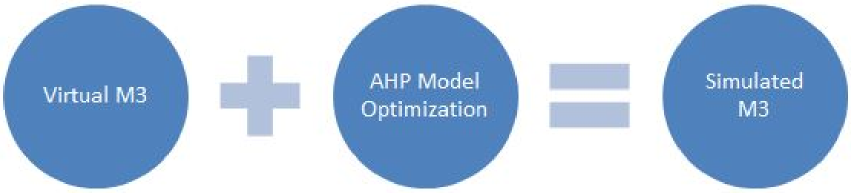


Figure 3.5: Process of creating Simulated M3

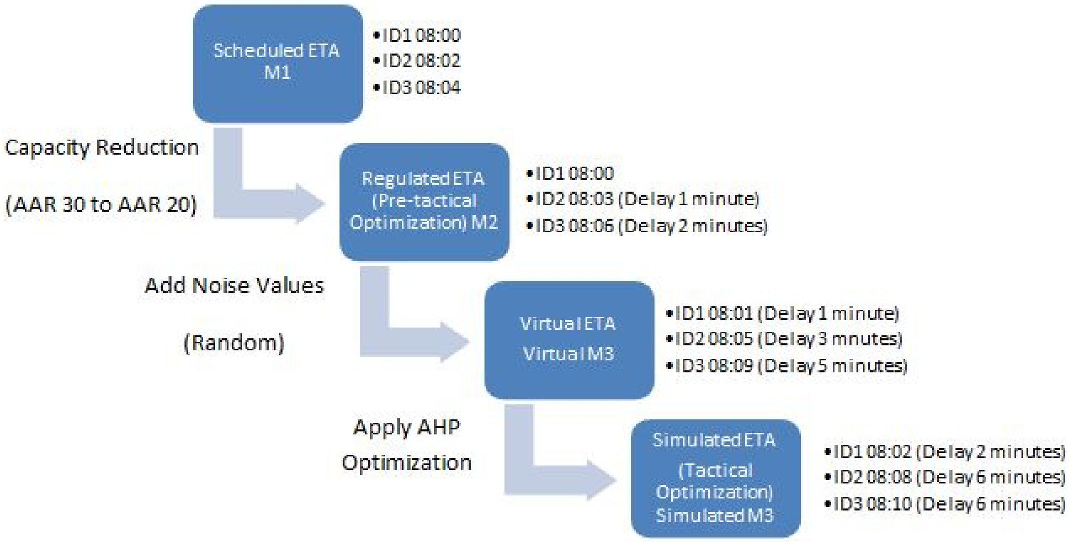

Scheduled ETA M1
•ID1 08:00
•ID2 08:02
•ID3 08:04
Capacity Reduction
(AAR 30 to AAR 20)
Regulated ETA (Pre-tactical Optimization) M2
•ID1 08:00
•ID2 08:03 (Delay 1 minute)
•ID3 08:06 (Delay 2 minutes)
Add Noise Values
(Random)
Virtual ETA
Virtual M3
•ID1 08:01 (Delay 1 minute)
•ID2 08:05 (Delay 3 mnutes)
•ID3 08:09 (Delay 5 minutes)
Apply AHP
Optimization
Simulated ETA
(Tactical Optimization)
Simulated M3
•ID1 08:02 (Delay 2 minutes)
•ID2 08:08 (Delay 6 minutes)
•ID3 08:10 (Delay 6 minutes)

Figure 3.6: Example of whole optimization process

added to the flights is randomly created in each simulation. The noise distribution (M3-M2) is studied in [1] to make more realistic the scenario since the regulated time is always different from the real performed flight profile. Thus, this is an intent of imitating the randomness of the real performance adding to the regulated (M2) time an extra time. Since the distribution study has found that there is no correlation between the different days of a week, it was impossible to find any known distribution patron followed by the different weekdays. For this reason, in a deeper study, only one day of a week during a month was studied (in our case we selected Saturdays in a first intent) and found that they follow a gamma distribution. But there were two different distribution function calculated since the set of Saturdays taken to be analysed their flights' noise distribution has either negative and positive values. Thus, two different gamma distribution function were defined:

- A total of 2115 flights were considered for this study: 879 flights with negative noise values and 1236 flights with positive noise values.
- The negative values are inverted to positive values and follow a gamma distribution

function with parameters of shape (1.220) and scale (5.533).

- The positive values follow a gamma distribution function with parameters of shape (1.195) and scale (8.408).
- An randomly created (following a Uniform Distribution) value from 0 to 1 is considered to classify if a flight is assigned with the negative (when value is equal or less than 879/2115) or positive (when this value is bigger than 879/2115) distribution function.
- The noise is created for each flight using a random function following a gamma distribution with the corresponding parameters.

Therefore, taking into account that only Saturdays were analysed the days to be optimized by the AHP model should be also Saturdays to keep this distribution consistent.

The problem is solved using Linear Programming (LP) taking advantage of Gurobi libraries. The AHP is named after GHP in a tactical window, which means Air Holding. But the mathematical formulations are supposed to be the same as GHP model. It is designed to assign flight slots to a set of aircraft during its landing phase to an airport.

At the beginning of the simulation, the whole day flight data is available and the user can filter the flights by their Entry Point Times. Once filtered, the N number of flights considered are defined as In={1,2,3,...,N}, indexed by ”i”.

Since this thesis is dealing with dynamic optimization, some auxiliary vectors should be defined and the size of these vectors will always change for every iteration ”i”. A new iteration is started when a new aircraft of vector ”In” enters the studied airspace. This means that for every iteration ”i”, the size of the list of flights in that iteration is always variable. This auxiliary list of the considered flights of one iteration depending on ”i” is defined as L={1,2,3,...,n}, indexed by ”l”. The ”n” number of flights in L depends on the number of flights inside the airspace at the moment of the simulation (see Entry Time (09:00) in figure 3.2). All the flights that their new Exit Point Times are later than the actual time, which is the Entry Point Time of the newest entered flight, will be in this list L.

A list of slots should be created St={1,2,3,...,s+T}, indexed by ”j”. The size of ”s+T” is from the earliest New Exit Point Time till the latest Scheduled Exit Point Time plus ”T” minutes (30 minutes of maximum holding time set as default) of the aircraft on list L. The slots must be of 1 minute interval time as default. For instance, if the earliest New Exit Point Time of all aircraft in the list L is 08:00 (being this slot the first element of the list St) and the latest Scheduled Exit Point Time is 08:30 (being this the slot corresponding to size ”s”), then the list St={08:00, 08:01, 08:02, ..., 09:00}, having size s+T=60.

Furthermore, all the slots that were previously assigned to the already left flights should not be considered, as well as their neighbours due to capacity separation constraint. That is to say, if only one slot 08:03 was assigned and the capacity is 1 aircraft every 2 minutes (AAR 30) then the set of slots from 08:02 to 08:04 is not assignable neither. In the same example as before, the list is St={08:00, 08:01, 08:05, 08:06, ..., 09:00}, having size s+T=57. This will be the final size of the list St in the iteration ”i”.

The capacity is a fixed scalar parameter during the whole simulation, for every optimization iteration ”i”. It is defined depending on AAR and S:

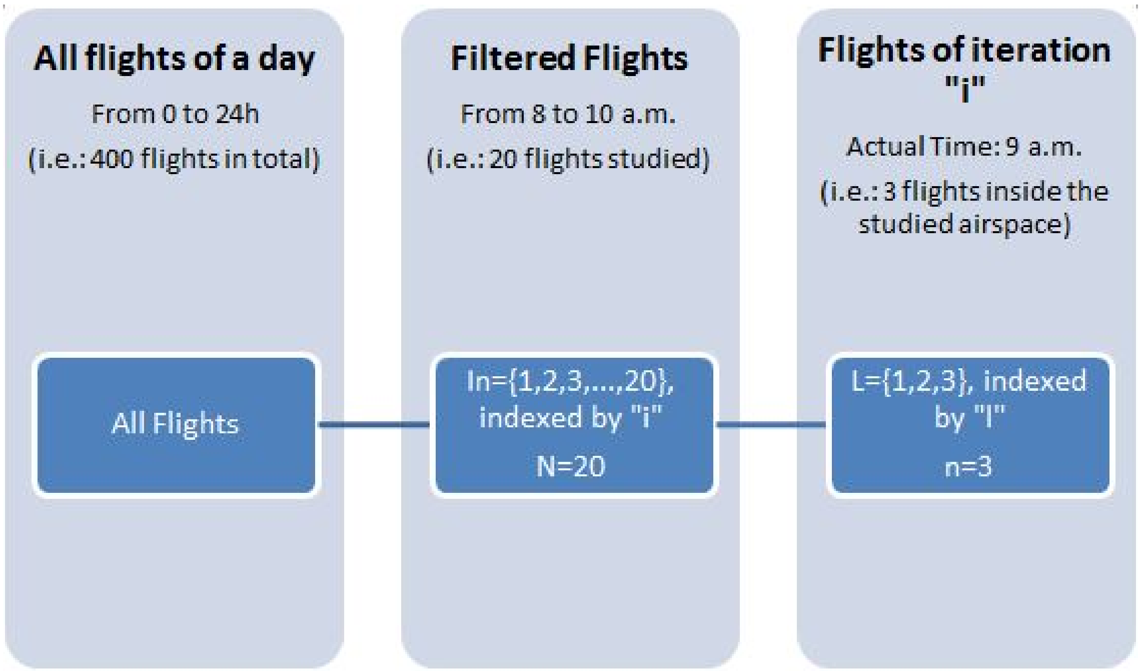

All flights of a day
From 0 to 24h
(i.e.: 400 flights in total)
All Flights
Filtered Flights
From 8 to 10 a.m.
(i.e.: 20 flights studied)
In={1,2,3,...,20}, indexed by "i"
N=20
Flights of iteration "i"
Actual Time: 9 a.m.
(i.e.: 3 flights inside the studied airspace)
L={1,2,3}, indexed by "l"
n=3

Figure 3.7: Process of creating auxiliary List ”L”

$$C = floor(\frac{AAR}{3600}) \cdot S$$

Since the slots are of 1 minute, the capacity C has always to be 1 too. Thus, the capacity is always defined as 1 aircraft every S seconds. The S has to be always seconds that are exact minutes (e.g. 60, 120, 180, etc.). For instance, when capacity is 1 aircraft every 2 minutes, AAR and S has to be combined to make C=1. In this case, AAR should be 30 and S 120. Therefore, the real capacity here is defined depending on S, which is the minutes that the aircraft has to be separated in the landing process.[3]

Variables definition:

$$x_{lj} \text{= 1 or 0, } \forall \text{ l} \in \{\text{1,...,n}\} \text{ and } \forall \text{ j} \in \{\text{1,...,s+T}\}$$

- x_{lj}: Binary variable. It is ”1” if flight ”l” is assigned to slot St_j; ”0” otherwise.
- New_l: New Exit Point Time slot in which flight ”l” was calculated to arrive.
- $(Sch+T)_l$: Scheduled slot (”Sch”) in which flight ”l” was planned to arrive plus 30 minutes (”T”).
- S: Capacity of the destination airport, expressed as 1 aircraft every S seconds (exact minutes).

[3] Note that for LEBL airport the maximum capacity is AAR 42 but since in this study only exact minutes are considered for the capacity parameter then AAR 30 is selected, which is 1 aircraft every 2 minutes. But for bigger airports where its capacity is bigger (i.e.: AAR 120, 2 aircraft every minute) this model is not applicable because C has to be always 1.

Problem Equations:

$$\sum_{j=New_l}^{(Sch+T)_l} x_{lj} = 1, \forall l \in \{1, ..., n\} \tag{3.1}$$

Equation 3.1 is for the unicity constraint, which means that every aircraft only can land once over all its possible landing slots.

$$\sum_{l=1}^{n} x_{lj} \leq \frac{1}{S/60}, \forall j \in \{1, ..., s+T\} \tag{3.2}$$

Equation 3.2 is for the capacity constraint, which means that every aircraft only can arrive according to capacity of the destination airport. This sum should be less or equal than 1 divided by the capacity. This is limiting the number of aircraft that can enter in a slot depending on the capacity, in order to have them properly distributed among all the slots in list St. For instance, if S=120 (2 minutes) then in every slot only 0.5 aircraft can land, which is limiting 1 aircraft every 2 minutes conceptually.

These slots are virtual slots since the important parameter here is the delay. The aircraft are supposed to fly till holding procedure and then make the holding corresponding to the delay assigned. Thus, the slot concept in this thesis is merely strategical, but useful to verify if the capacity constraint is met when analysing the results.

3.1.3. Tactical AHP Cost Function

The cost function in tactical phase is looking for minimizing the cost of fuel consumption at holding altitude for the aircraft. Thus, first of all the fuel consumption should be defined, in an unit of litres per hour (L/h):

$$\sum_{l=1}^{n} [K1_l \cdot K2_l \cdot K3_l]$$

Where:

- K1 is the fuel in litres (kerosene) consumed per kilometre travelled for one seat. It varies depending if the trip is a short (0.02408 L/km), medium (0.02772 L/km) or long haul (0.02948 L/km). These are average values, considered from a table of aircraft types for each trip type [20].

- K2 is the speed in km/h at holding altitude depending on different categories of the aircraft [21]. For category ”L” it is 388.92 km/h, for ”M” it is 438.92 km/h and for ”H” it is 490.78 km/h.

- K3 is the average seats the aircraft should have depending on the trip type [20]. For short it is 168 seats, for medium it is 260 seats and for long haul it is 346 seats.

This will give a fuel consumption function of units [L/h].

Then, the cost of this fuel consumption function should be defined as:

$$\sum_{l=1}^{n}[K1_l \cdot K2_l \cdot K3_l \cdot K4]$$

Where K4 is the price in euros (€) per litre of the kerosene for propellant purposes [22]. In this case, 0.33 € per litre of fuel consumed. This gives the cost function units of [€/h].

And finally, the eventual cost function should be:

$$min \sum_{l=1}^{n} \sum_{j=New_l}^{(Sch+T)_l} [K1_l \cdot K2_l \cdot K3_l \cdot K4] \cdot D_{lj} \cdot x_{lj} \tag{3.3}$$

Where D is:

$$D_{lj}[h] = \left[St_j - (Sch)_l\right]^{(1+\varepsilon)} \tag{3.4}$$

D is a vector of delays for every slot of every aircraft to see if the slot ”j” is assigned to aircraft ”l”, what would be the delay with respect its scheduled exit point time. Here ε is defining the linearity of the delay function, set as ”0” as default.

The cost function is expressed in units of euros (€) to make more understandable to users.

3.1.4. Maximum Holding Time

Since the Air Holding is considered in this thesis, the fact of making flights wait in air is unavoidable. Thus, if the aircraft have to wait in air, then how much time an aircraft can hold is a very critical point.

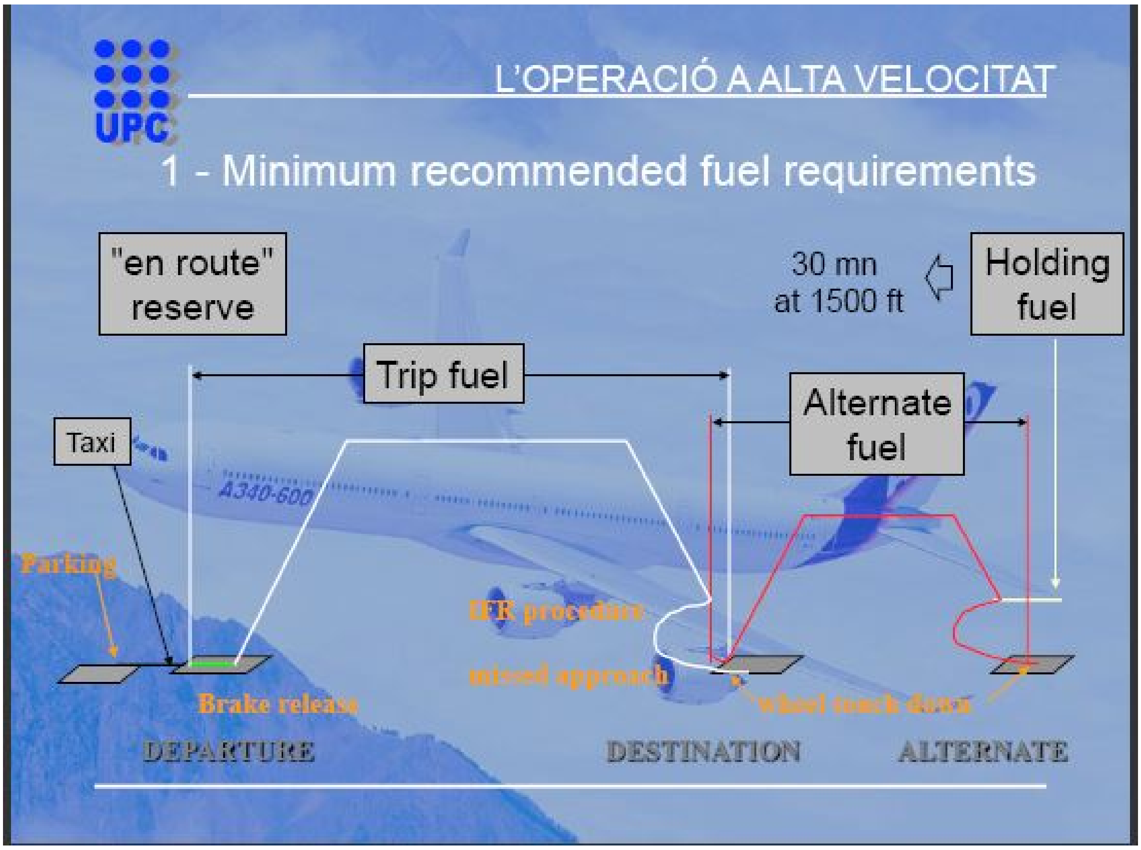
The time an aircraft can fly is directly proportional to its fuel reserve, thus here the maximum delay time of 30 minutes at holding point is considered.

From the subject of Air Operations (OPS) [23], every flight must have enough fuel to travel to destination airport, then to the alternate airport and still have fuel to hold at 1500 ft as shown in figure 3.8. If the flight is not deferred to alternate airport and enters to holding procedure to land, the 30 minutes of holding fuel reserved to be used in alternate airport could be used for this purpose of AHP. Thus, for the sake of safety, only 30 minutes of delay is allowed.

3.1.5. Capacity Limitations

Since the capacity is the main problem when dealing with airspace congestion, in this section the capacity flexibility is explained.

Although the main purpose of this thesis is minimizing the air delay according to cost function applying the free route concept, another very important objective is to adapt the capacity considered in the simulation to the real capacity at that moment at the destination airport.



UPC

L'OPERACIÓ A ALTA VELOCITAT

1 - Minimum recommended fuel requirements

Figure 3.8: Capture from Operations (OPS) subject slides of fuel reserves [23]

Generally, the capacity considered in tactical phase should be equal than the same considered in pre-tactical phase since the flights at departure were delayed according to this capacity. But now that they are already entering the studied area, if the capacity increases in this moment, it is still time to change the capacity of the simulator to make aircraft arrive even earlier than expected with pre-tactical capacity. Therefore, this will save more economic costs.

But this is only useful when the capacity is increased with respect the capacity considered in pre-tactical phase, because it will not find any solution to this problem with these conditions if capacity is reduced and maximum holding time is still 30 minutes. If the reduced capacity is desired to be considered in tactical model, then the maximum holding time should be increased. This is a trade off between the capacity and the maximum holding time, since they are directly related.

Nevertheless, if the maximum holding time is bigger than 30 minutes, for instance 45 to 60 minutes, this is breaking the safety requirement and consuming more fuel. Moreover, this is not even the optimal method to solve this problem when capacity is reduced respect pre-tactical phase, because a better solution is to deviate the flight to the nearest available airport instead of making it hold in air more than 30 minutes.

The conclusion of this analysis is that for capacities smaller than the one considered in pre-tactical phase, the flights shall better be deferred to alternate airport more than keep waiting in air at holding point more than 30 minutes.

3.2. Optimization process explanation with some other hypothesis

Coming up next, the whole process of Matlab optimization assignment is explained step by step, showing some not mentioned hypothesis:

1. First of all, the data of M1 filtered using AWK are read saving them into vectors as variables. Also, the delay table with the list of assigned delays from pre-tactical phase is read.
2. **The capacity is supposed to be equal or bigger than in Pre-tactical Phase**: 1 aircraft every S minutes (how many aircraft the airport can accept each minute).
3. Remove the cancelled flights because they do not exist in tactical phase: delete the flight with -1.
4. Convert all the times needed for optimization from hhmmss format into seconds.
5. Solve the problem of Scheduled Exit Point Time (in trajectory) earlier than new Scheduled Exit Point Time (straight line): if it is earlier, make them equal. **The new Exit Point Time can never be later than Scheduled.** This might be already solved in step 4, when converting hhmmss to seconds and round function is applied to make all times in seconds be exact minutes.
6. Create Noise values for all the flights. **Use the Gamma Distribution found in the data of $M3-M2$ to generate these values.** The distribution is only analysed for days on Saturday in this research to simplify the problem, so the days available to be studied should be also on Saturdays. These values will make every simulation have different results (stochastic model) since the noises are generated randomly.[4]
7. Add delay and noise to all the flights. See [1] for more information.
8. **If the Entry, new Exit and Exit Point Times are later than ETA, then it means that this flight is departing a day before the day studied, thus, subtract 24 hours to those flights.** Here negative values may appear.
9. Make a RBS delay assignment to all the flights to know the total delay with no cost function to be able to verify if the AHP is properly programmed (when unitary cost function is applied). **Here the same capacity is considered.**
10. Order all the flights from earliest to latest according to Entry Point Times. **The flight should be considered once entered the 500 km radius. So they are all ordered respect their Entry Point Time to know which flights enter first.**
11. Optimization loop starts. Add the first flight entering the Entry Point Radius to the optimization list (L).
12. **The actual time during this simulation should be the entry time of this flight.**
13. **Linear cost function is considered**: $\varepsilon = 0$.

[4] This study of noise distribution is done in the pre-tactical phase [1].

14. Create the matrices for unicity and capacity constrains as defined in the equations 3.1 and 3.2.

15. Once determined all these vectors and matrices, the next step is to pass them to Gurobi Optimizer to obtain the resulting vector with the slots allocated for the flights in the list. Also the final cost is obtained in this step.

16. According to the vector of results from Gurobi, it is possible to create the vector of holding for each flight and also the cost.

17. From the holding vector, the delay vector is created. **The holding vector is the holding time the aircraft should wait in air when arriving at holding procedure point. The delay vector is the delay time the flight will have once landed after doing the holding in air. It can be a negative value, when landing before scheduled time.**

18. All these vectors are saved into a structure called Results, which will be increasing in size because in each iteration it will save the results of that simulation. The size of this structure should be the same as the number of flights of the day studied, because **every time that a flight enters the area and is added to the list, it simulates one result for those flights in the list.**

19. When the following flight is inside the airspace considered, the whole simulation starts again. **Here the new flight is added to the list but also the flights that are arriving at 50 km and leaving the airspace, that is, when their new Exit Point Time is later than the actual time, are deleted from the list saving their last holdings, delays and other parameters assigned as final holdings and delays for those flights.** These holdings and delays will be the ones to be communicated to the pilots.

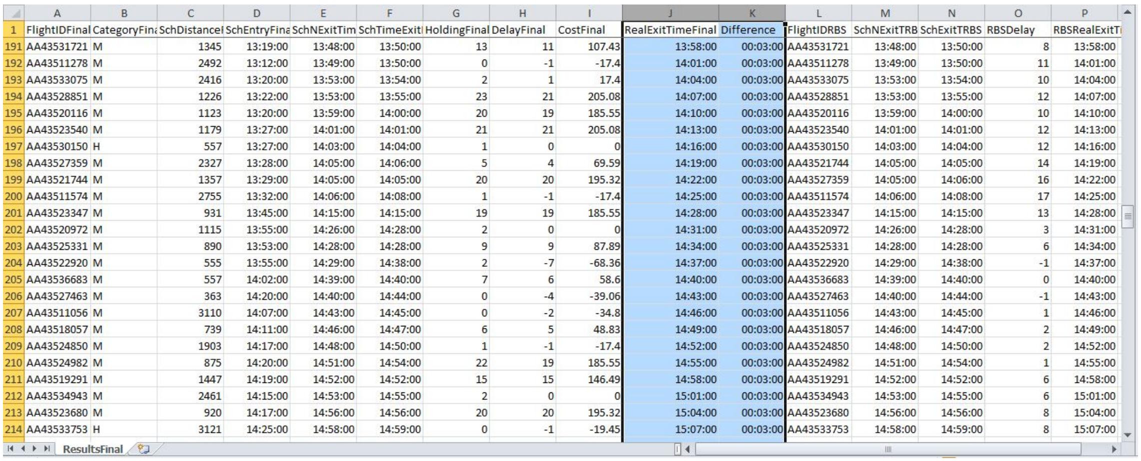
20. **The simulation will iterate as many times as the number of flights of a day studied has, if filter is not applied.**

Once finished the execution of Matlab Program containing all the steps mentioned above and obtained the desired results, the results are saved in two different tables for different analysis purposes.

The table called ”Results” contains important parameters such as Flight Identification (ID), Holding time (min), Total Holding (min), Delay Time(min), Total Delay (min), Cost (€), Total Cost (€), Real Exit Time (hhmmss) and others of each flight in each simulation. This table provides the evolution of the flights delays when proceeding with the whole simulation.

The table called ”ResultsFinal” contains almost the same parameters but only the last ones assigned when leaving the studied area. This is the most important table since it provides the final results after simulation. Also together with the results using AHP, the results of the RBS method simulation is also available here, in order to be able to compare them.

The Real Exit Time in hhmmss (slots assigned) is not a analysable parameter but useful to check if the capacity constraint is properly met (see figure 3.9). It simulates as the aircraft has to wait at 50 km instead of holding point and indicates the time it should continue flying towards destination airport once hold that time there. But since the real idea is to continue flying when arriving at 50 km till holding procedure and then make the holding there, this is just a parameter to check capacity veracity. Actually, the Exit Point should have been at holding point but in order to make simpler the problem, 50 km of radius is used.



	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P
1	FlightIDFinal	CategoryFina	SchDistanceI	SchEntryFina	SchNExitTim	SchTimeExit	HoldingFinal	DelayFinal	CostFinal	RealExitTimeFinal	Difference	FlightIDRBS	SchNExitTRB	SchExitTRBS	RBSDelay	RBSRealExitTi
191	AA43531721	M	1345	13:19:00	13:48:00	13:50:00	13	11	107.43	13:58:00	00:03:00	AA43531721	13:48:00	13:50:00	8	13:58:00
192	AA43511278	M	2492	13:12:00	13:49:00	13:50:00	0	-1	-17.4	14:01:00	00:03:00	AA43511278	13:49:00	13:50:00	11	14:01:00
193	AA43533075	M	2416	13:20:00	13:53:00	13:54:00	2	1	17.4	14:04:00	00:03:00	AA43533075	13:53:00	13:54:00	10	14:04:00
194	AA43528851	M	1226	13:22:00	13:53:00	13:55:00	23	21	205.08	14:07:00	00:03:00	AA43528851	13:53:00	13:55:00	12	14:07:00
195	AA43520116	M	1123	13:20:00	13:59:00	14:00:00	20	19	185.55	14:10:00	00:03:00	AA43520116	13:59:00	14:00:00	10	14:10:00
196	AA43523540	M	1179	13:27:00	14:01:00	14:01:00	21	21	205.08	14:13:00	00:03:00	AA43523540	14:01:00	14:01:00	12	14:13:00
197	AA43530150	H	557	13:27:00	14:03:00	14:04:00	1	0	0	14:16:00	00:03:00	AA43530150	14:03:00	14:04:00	12	14:16:00
198	AA43527359	M	2327	13:28:00	14:05:00	14:06:00	5	4	69.59	14:19:00	00:03:00	AA43521744	14:05:00	14:05:00	14	14:19:00
199	AA43521744	M	1357	13:29:00	14:05:00	14:05:00	20	20	195.32	14:22:00	00:03:00	AA43527359	14:05:00	14:06:00	16	14:22:00
200	AA43511574	M	2755	13:32:00	14:06:00	14:08:00	1	-1	-17.4	14:25:00	00:03:00	AA43511574	14:06:00	14:08:00	17	14:25:00
201	AA43523347	M	931	13:45:00	14:15:00	14:15:00	19	19	185.55	14:28:00	00:03:00	AA43523347	14:15:00	14:15:00	13	14:28:00
202	AA43520972	M	1115	13:55:00	14:26:00	14:28:00	2	0	0	14:31:00	00:03:00	AA43520972	14:26:00	14:28:00	3	14:31:00
203	AA43525331	M	890	13:53:00	14:28:00	14:28:00	9	9	87.89	14:34:00	00:03:00	AA43525331	14:28:00	14:28:00	6	14:34:00
204	AA43522920	M	555	13:55:00	14:29:00	14:38:00	2	-7	-68.36	14:37:00	00:03:00	AA43522920	14:29:00	14:38:00	-1	14:37:00
205	AA43536683	M	557	14:02:00	14:39:00	14:40:00	7	6	58.6	14:40:00	00:03:00	AA43536683	14:39:00	14:40:00	0	14:40:00
206	AA43527463	M	363	14:20:00	14:40:00	14:44:00	0	-4	-39.06	14:43:00	00:03:00	AA43527463	14:40:00	14:44:00	-1	14:43:00
207	AA43511056	M	3110	14:07:00	14:43:00	14:45:00	0	-2	-34.8	14:46:00	00:03:00	AA43511056	14:43:00	14:45:00	1	14:46:00
208	AA43518057	M	739	14:11:00	14:46:00	14:47:00	6	5	48.83	14:49:00	00:03:00	AA43518057	14:46:00	14:47:00	2	14:49:00
209	AA43524850	M	1903	14:17:00	14:48:00	14:50:00	1	-1	-17.4	14:52:00	00:03:00	AA43524850	14:48:00	14:50:00	2	14:52:00
210	AA43524982	M	875	14:20:00	14:51:00	14:54:00	22	19	185.55	14:55:00	00:03:00	AA43524982	14:51:00	14:54:00	1	14:55:00
211	AA43519291	M	1447	14:19:00	14:52:00	14:52:00	15	15	146.49	14:58:00	00:03:00	AA43519291	14:52:00	14:52:00	6	14:58:00
212	AA43534943	M	2461	14:15:00	14:53:00	14:55:00	2	0	0	15:01:00	00:03:00	AA43534943	14:53:00	14:55:00	6	15:01:00
213	AA43523680	M	920	14:17:00	14:56:00	14:56:00	20	20	195.32	15:04:00	00:03:00	AA43523680	14:56:00	14:56:00	8	15:04:00
214	AA43533753	H	3121	14:25:00	14:58:00	14:59:00	0	-1	-19.45	15:07:00	00:03:00	AA43533753	14:58:00	14:59:00	8	15:07:00

Figure 3.9: Example of Capacity of 1 aircraft every 3 minutes (S=180 seconds) Verification using Real Exit Time

As seen in figure 3.9, the first marked blue column is the Real Exit Time Slots in ascending order and the second one is the difference in time between two consecutive slots (rows). It is proved that the minimum capacity requirement is met since the slots minimum separation time is 3 minutes, for S=180 seconds which means 1 aircraft every 3 minutes.

The complete and commented main code of Matlab and the output files format of the tables of results are available in Appendix B.1. and B.2..

In the following chapter, the different results and analysis will be presented.

CHAPTER 4. RESULTS AND ANALYSIS

The results obtained from the simulation must be evaluated together with others by changing some parameters to verify its veracity and feasibility.

The analysis is focused on the Saturdays from April 2015 onwards, due to Eurocontrol DDR3 ALL_FT+ format last update and noise distribution analysis.

First of all, the consistency of the AHP program has to be proven. Then the analysis of some remarkable and interesting parameters will be presented. For these two sections, the pre-tactical delays will be taken into account and always considering the standard parameters values[1]. Finally, the advantage of combining the Pre-tactical Optimization with the Tactical Phase shall be explained in detail.

For the general results, in order to validate the veracity and consistency of the output data and also to analyse some interesting aspects of the parameters, only one day has been chosen to be analysed in detail: 11-07-2015.

4.1. AHP Program Testing

4.1.1. Flight Delay Evolution to test Dynamic Model

The simulation standard parameters for this subsection are:

- Interval of study: 00:00:00 – 23:59:59 of 11/07/2015
- Airport: Barcelona (LEBL)
- Maximum Holding Time: 30 minutes
- Tactical AAR: 20 (constant during the whole day)
- Entry Point Radius: 500 km
- $\varepsilon = 0$
- AHP Cost Function: $min \sum_{l=1}^{n} \sum_{j=New_l}^{(Sch+T)_l} [K1_l \cdot K2_l \cdot K3_l \cdot K4] \cdot D_{lj} \cdot x_{lj}$
- Pre-tactical Delay (AAR 20) and Noise applied.

From the most basic results of the table ”Results” (see Appendix B.2. for format definition and examples), it is possible to analyse the evolution of the flights along all the iterations, since it enters the airspace till it leaves it. After having optimized for all the flights of a day, 5 flights are selected to be shown their delay evolution to prove the dynamic method. A peak hour has been chosen (about 2 p.m.) in order to see more clearly the delay evolution of every flight since there will be more congestion.

In figure 4.1, there is a plot for the flights delays through each iteration. The general tendency of all flights are from smaller values to bigger values, since as more iterations,

[1] Note that all delays and costs showed are tactical delays and costs until section 4.3. (delays and costs with respect M1+Regulation+Noise).

more flights are involved in the airspace and more delay will be assigned. The number of iterations is not regular for all flights, since it depends on the time it spends inside the airspace, which depends on its speed and trajectory.

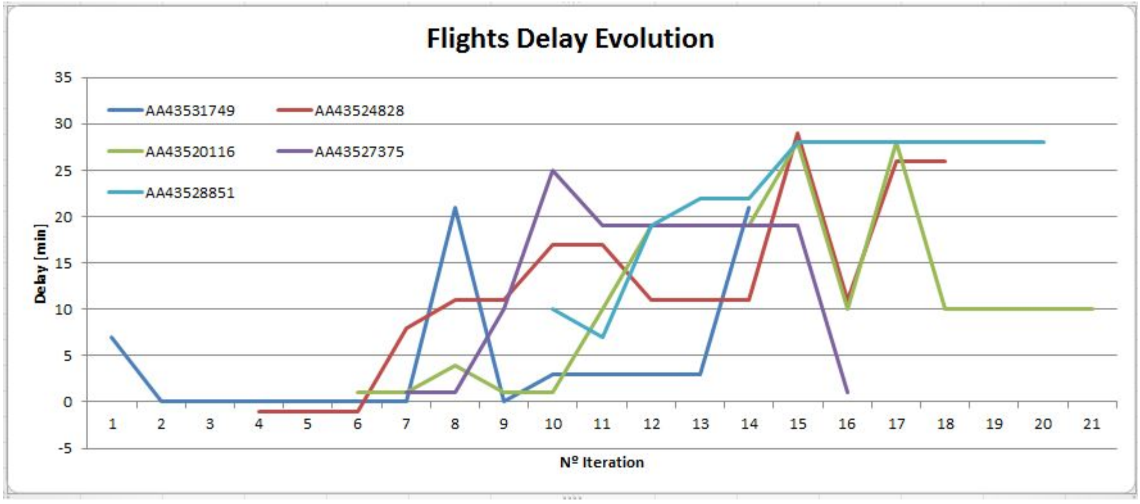
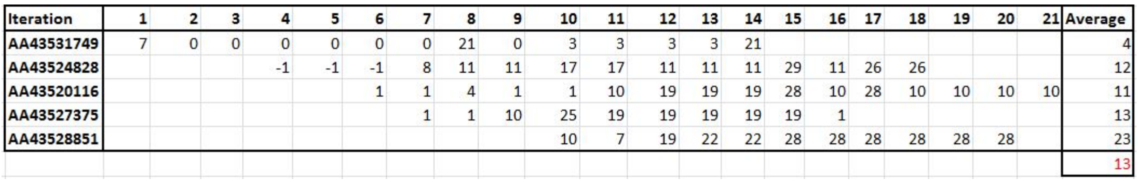


Figure 4.1: Plot of Delay Evolution

The consistency of the dynamic model is proved since all the flights delays are evolving when more flights are entering the studied airspace. In this example, the aircraft with ID AA43531749 (dark blue line) is keeping its delay ”0” till other flights entered the airspace and made the demand be imbalanced. The general tendency of the delays is to grow when more flights are involved, as expected.

Furthermore, to more deeply see this evolution process, a table is presented in 4.2 where it can be seen that reading from iteration 1 to 21, the delays are evolving from negative values at first to positive values and then keep increasing as more flights are added. Moreover, the average delay of the first flight is very small but when adding new flights the averages are also increasing in the new added flights. The total average of these 5 flight is 13 minutes. With this, the conclusion of the more flights the more delay will be assigned to each one in average can be assumed.



Iteration	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	Average
AA43531749	7	0	0	0	0	0	0	21	0	3	3	3	3	21								4
AA43524828				-1	-1	-1	8	11	11	17	17	11	11	11	29	11	26	26				12
AA43520116						1	1	4	1	1	10	19	19	19	28	10	28	10	10	10	10	11
AA43527375							1	1	10	25	19	19	19	19	19	1						13
AA43528851										10	7	19	22	22	28	28	28	28	28	28		23
																						13

Figure 4.2: Table of Delay Evolution Average

4.1.2. AHP and RBS method Comparison

4.1.2.1. AHP Unitary Cost Function VS. RBS

The simulation standard parameters for this subsection are:

- Interval of study: 00:00:00 – 23:59:59 of 11/07/2015

- Airport: Barcelona (LEBL)
- Maximum Holding Time: 30 minutes
- Tactical AAR: 20 (constant during the whole day)
- Entry Point Radius: 500 km
- ε = 0
- AHP unitary Cost Function: $min \sum_{l=1}^{n} \sum_{j=New_l}^{(Sch+T)_l} D_{lj} \cdot x_{lj}$
- Pre-tactical Delay (AAR 20) and Noise applied.

From the "Results Final" table using standard values, firstly the veracity of the computation using Matlab Code should be tested. For this cause, in figure 4.3 the difference of delays between AHP (with unitary Cost Function) and RBS for each aircraft is shown. It can be proved that although the delays are not the same (in AHP the flights were listed by their Entry Point Time and in RBs they were listed by their New Exit Point Time, thus the assignment was quite different), the average difference is still 0 because the positive values balance the negative ones. Moreover the sum of the delays in both methods, the total (2351 minutes) delay is the same for both cases. This can prove that the AHP with unitary cost function and RBS have the same results, thus the assignment must be also correct.

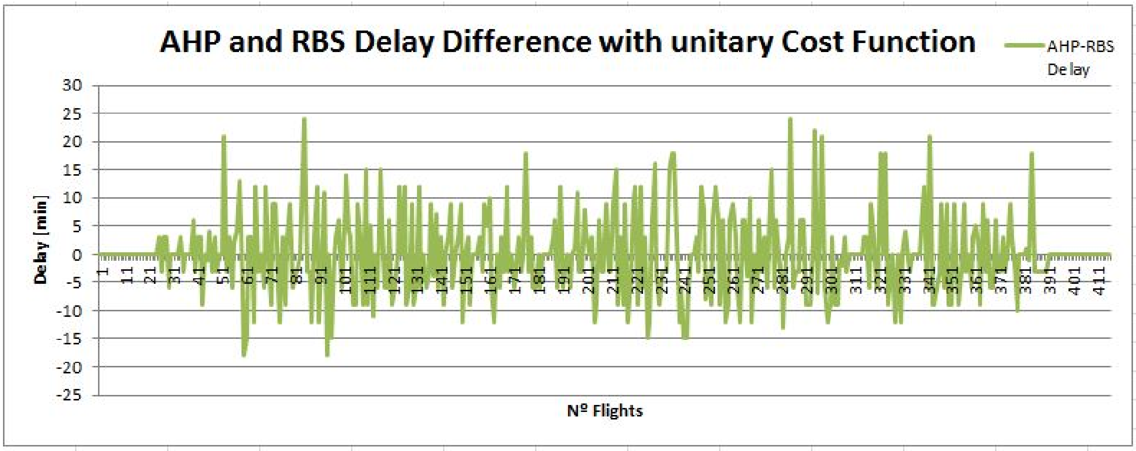


Figure 4.3: Plot of AHP (with unitary Cost Function) and RBS Delay Difference (AAR 20)

4.1.2.2. AHP Non-Unitary Cost Function VS. RBS

The simulation standard parameters for this subsection are:

- Interval of study: 00:00:00 – 23:59:59 of 11/07/2015
- Airport: Barcelona (LEBL)
- Maximum Holding Time: 30 minutes
- Tactical AAR: 20 (constant during the whole day)

- Entry Point Radius: 500 km
- ε = 0
- AHP non-unitary Cost Function: $min \sum_{l=1}^{n} \sum_{j=New_l}^{(Sch+T)_l} [K1_l \cdot K2_l \cdot K3_l \cdot K4] \cdot D_{lj} \cdot x_{lj}$
- Pre-tactical Delay (AAR 20) and Noise applied.

The next step is to compare the difference between using non-unitary Cost Function in AHP and RBS, this is when fuel consumption is being minimized.

In figure 4.4, for a sample simulation of same capacity than in section 4.1.2.1. the difference in Delay and Cost for AHP with non-unitary cost function and RBS are shown. The difference in delay is almost the same in average (average 0.0048 minutes) since the positive values compensate the negative ones. But for the cost difference (average -18.55 €), it can be seen that there are more negative values, which means that the AHP method is saving more costs than RBS method.

The total delay for AHP and RBS are likely the same (about 6360 minutes). Taking into account the sub-figure 4.4(b), the total cost for AHP (total 30649 €) is smaller than for RBS (total 38368 €). This is, actually, the real achievement desired when applying cost function because with the same delay, the AHP is reducing more economic costs (7719 €) than RBS. The order in which the slots are assigned to each aircraft taking into account the cost function is making possible this optimization.

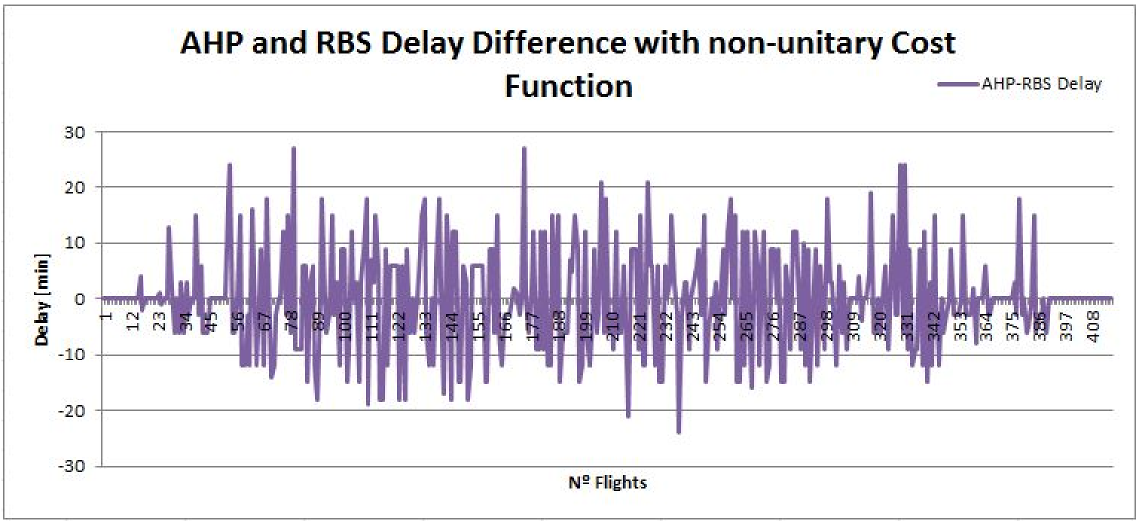
4.2. Parameters Dependant Results

In this section, the results depending on the variables will be presented. The parameters from pre-tactical phase will remain fixed.

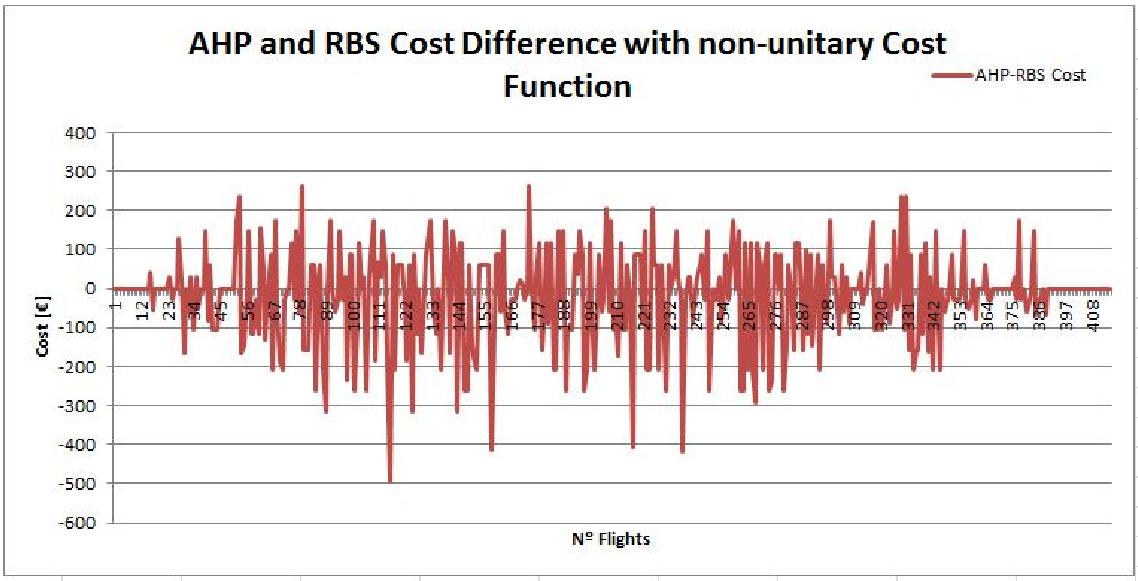
4.2.1. Decreasing Holding Time

The simulation standard parameters for this subsection are:

- Interval of study: 00:00:00 – 23:59:59 of 11/07/2015
- Airport: Barcelona (LEBL)
- Maximum Holding Time: 30, 25, 20 and 15 minutes
- Tactical AAR: 20 (constant during the whole day)
- Entry Point Radius: 500 km
- ε = 0
- AHP non-unitary Cost Function: $min \sum_{l=1}^{n} \sum_{j=New_l}^{(Sch+T)_l} [K1_l \cdot K2_l \cdot K3_l \cdot K4] \cdot D_{lj} \cdot x_{lj}$
- Pre-tactical Delay (AAR 20) and Noise applied.



(a) AHP and RBS Delay Difference

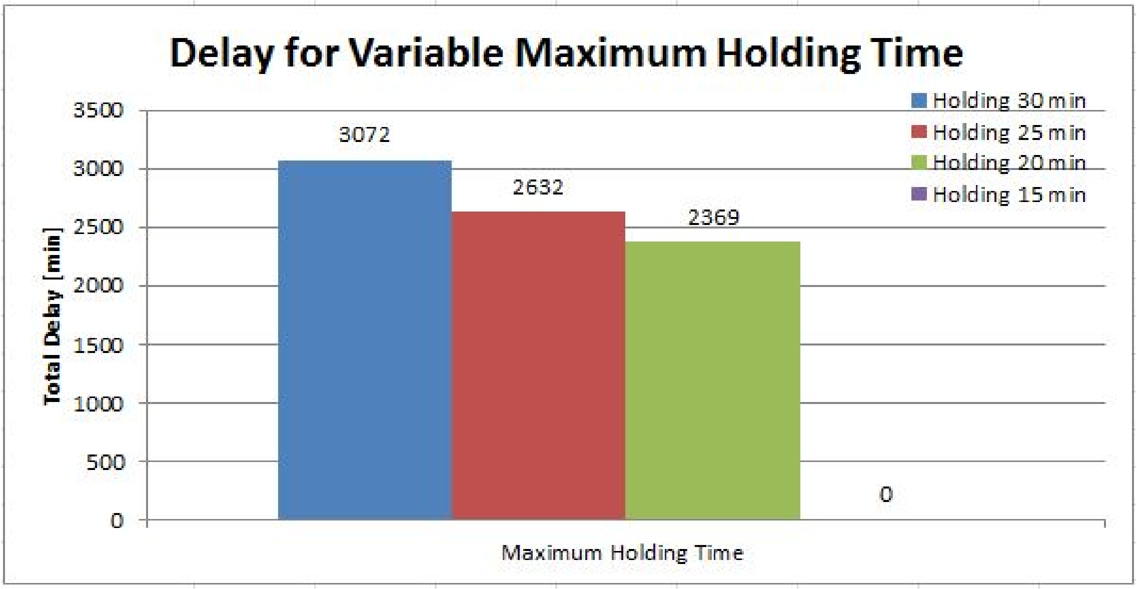


(b) AHP and RBS Cost Difference

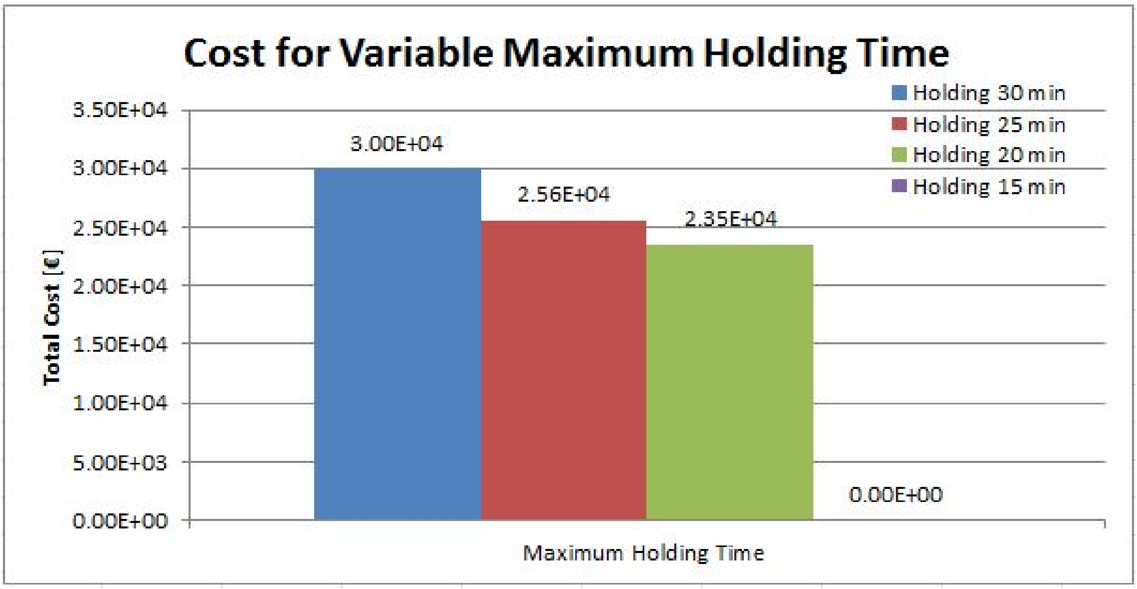
Figure 4.4: Plots of AHP and RBS Delay and Cost Difference (AAR 20)

The first parameter to be analysed is the holding time. If the maximum holding time for every aircraft is decreased, the delay and cost might also decrease but when keeping reducing this time, it comes to a worst case where the simulation might not converge because there is not any solution for that maximum holding time due to large amount of flights demand. In this case, if the maximum holding time had to be kept, the solution would be to increase capacity.

As proved in figure 4.5, as holding time decreases, the delay and cost decreases too. For the 15 minutes case of 0 delay and cost, the optimization has no solution due to the imbalance between demand and maximum holding time, which can be also expressed as a kind of capacity.



(a) Delay for variable Holding Time



(b) Cost for variable Holding Time

Figure 4.5: Delay and Cost for variable Holding Time

4.2.2. Increasing AAR

The simulation standard parameters for this subsection are:

- Interval of study: 00:00:00 – 23:59:59 of 11/07/2015
- Airport: Barcelona (LEBL)
- Maximum Holding Time: 30 minutes
- Tactical AAR: 12, 20 and 30 (constant during the whole day)
- Entry Point Radius: 500 km

- ε = 0
- AHP non-unitary Cost Function: $min\sum_{l=1}^{n}\sum_{j=New_l}^{(Sch+T)_l}[K1_l \cdot K2_l \cdot K3_l \cdot K4] \cdot D_{lj} \cdot x_{lj}$
- Pre-tactical Delay (AAR 12) and Noise applied.

Another interesting point of view is to increase the AAR of the simulation when the AAR of the Pre-tactical Phase is still fixed (for AAR 12 in this case). This is useful when already flying towards the destination airport and the capacity of the airport increases because the circumstances are improved, the simulator is still time the change its capacity parameter to adapt its capacity to the airport's. In this way, more cost could be saved since the capacity is bigger than expected (when applying pre-tactical optimization before departure [1].)

In the figure 4.6, the delay and cost for AAR 12, 20 and 30 is shown. When increasing the capacity, the delay and cost are decreased.

In order to see this more in detail, the table 4.1 is presented. As shown, the total delays and costs are reduced with respect the original considered AAR 12 (in pre-tactical phase). Therefore, if the capacity is improved in tactical window, and the simulator is adapting its capacity to it, the costs to be saved would be very significant, even reaching at 24 € to be saved for each aircraft.

AAR	12 (1 AC/ 5 minutes)	20 (1 AC/ 3 minutes)	30 (1 AC/2 minutes)
Difference in Delay [min]	0	-684	-959
Delay saved per AC [min/AC]	0	-1.6	-2.3
Difference in Cost [€]	0	-7090	-10040
Cost saved per AC [€/AC]	0	-17	-24.1

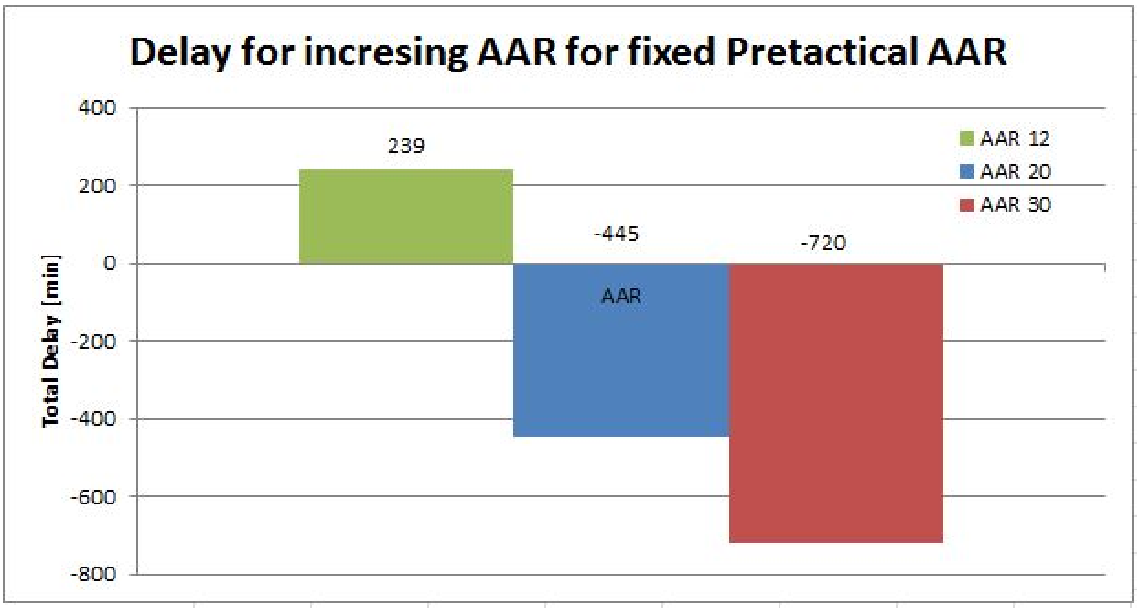
Table 4.1: Delay and Cost Differences with respect AAR 12

Notice that here only increased AAR are applied because of the capacity limitations explained in section 3.1.5..

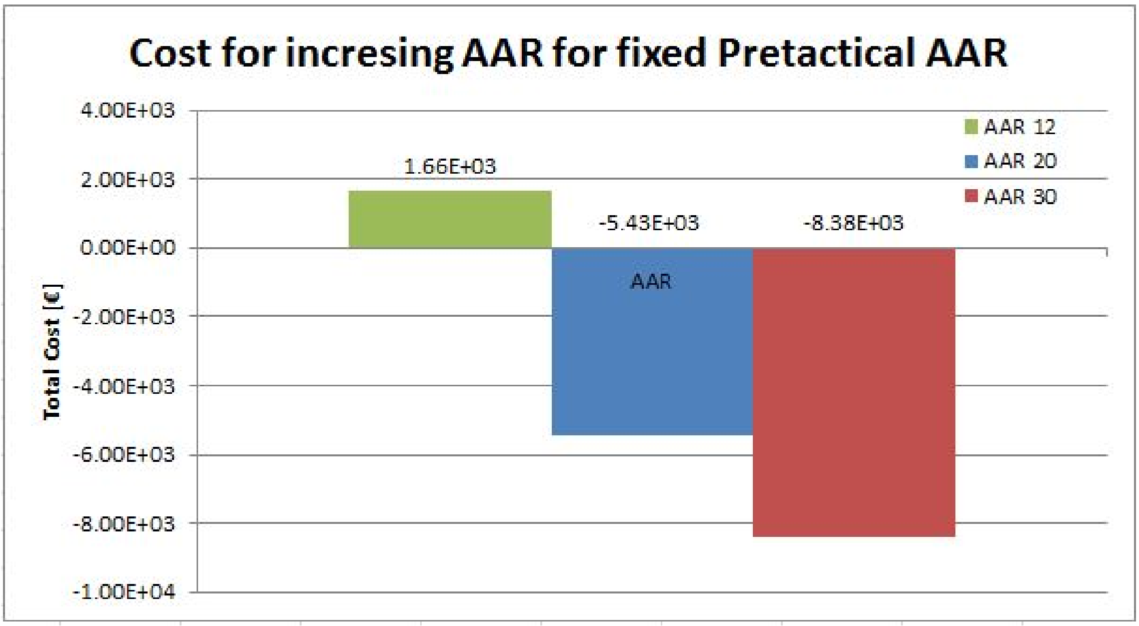
4.2.3. Varying Studied Airspace Radii

The simulation standard parameters for this subsection are:

- Interval of study: 00:00:00 – 23:59:59 of 11/07/2015
- Airport: Barcelona (LEBL)
- Maximum Holding Time: 30 minutes
- Tactical AAR: 20 (constant during the whole day)
- Entry Point Radius: 200, 500 and 800 km
- ε = 0
- AHP non-unitary Cost Function: $min\sum_{l=1}^{n}\sum_{j=New_l}^{(Sch+T)_l}[K1_l \cdot K2_l \cdot K3_l \cdot K4] \cdot D_{lj} \cdot x_{lj}$
- Pre-tactical Delay (AAR 20) and Noise applied.



(a) Delay for increasing AAR



(b) Cost for increasing AAR

Figure 4.6: Delay and Cost for variable AAR when Pre-tactical AAR is fixed

The last parameter to be analysed is the Entry Radius. Since the Exit Radius is 50 km, which is already the most approximated distance for Holding Procedures Point for LEBL case, this value will not be changed.

The idea of modifying the Entry Radius is to know how would it affect the simulation process and the final results. When the radius is increased, it means that the airspace studied is bigger, thus there will be more flights to be considered inside the area for every iteration of the simulation but also more time for every aircraft to evolve. In the same way, when this radius is reduced, the area studied will also decrease, which means that there will be less flights and less time. The situation is that the bigger the airspace, the more time the simulation has to find the optimal assignment for every flight. In this way, it is the same to say that if the airspace is very small, it will not let the program iterate as much as to get the optimal solution.

Here the Evolution Ratio is created to be able to analyse how many times an aircraft is iterated (assigned delay) when it is inside the airspace. This number is computed simply dividing the sum of the total number of iterations for every aircraft for the whole simulation by the total number of iterations (flights) of the simulation. The output is an average value for the evolution of each aircraft. This means how many times in average an aircraft is assigned delay. Therefore, if the Evolution Ratio is high, this will mean that the program will have more time (more iterations) to find the optimal solution. In the figure 4.7, two plots are shown for this purpose.

The sub-figure 4.7(a) shows that when the evolution ratio is high (800 km radius) it means that every aircraft in average is optimized more times (15.18 iterations per AC since it enters till it leaves the studied area) which means that the bigger the airspace the more time spend the flights inside the area, letting possible the computation of an optimal solution. The prove is shown in sub-figure 4.7(b) and 4.7(c), where for the 800 km Entry Radius the delays and costs are the smallest because they have been iterated (optimized) more times and for 200 km the delays and costs are the biggest.

Nevertheless, if a very big airspace is taken, the flights performance is very unpredictable because they are still very far away from the destination airport, thus it is like simulating for a pre-tactical time window already. But if a very small airspace is taken, the flights will be arriving very quickly, not letting the program have enough time (small evolution ratio) to process and find a better solution. Therefore, an intermediate distance (500 km) is considered in order to have a trade off between the optimal solution and the real performance of the flights.

4.3. Pre-tactical and Tactical Optimization Combination

The major advantage of this thesis is that it takes as input data the delay assigned in Pre-tactical Phase according to capacity necessities. Thus, in this section, all the possible modifications to the capacity parameter for both phases will be explained here.

Variable Capacity

The simulation standard parameters for this subsection are:

- Interval of study: 00:00:00 – 23:59:59 of 11/07/2015
- Airport: Barcelona (LEBL)
- Maximum Holding Time: 30 minutes
- Tactical AAR: 30 (constant during the whole day)
- Entry Point Radius: 500 km
- $\varepsilon = 0$

- AHP non-unitary Cost Function: $min\sum_{l=1}^{n}\sum_{j=New_l}^{(Sch+T)_l}[K1_l \cdot K2_l \cdot K3_l \cdot K4] \cdot D_{lj} \cdot x_{lj}$
- Pre-tactical Delay (AAR 30) and Noise applied and not applied.

If the M1 flight data is directly taken into account for tactical optimization (pre-tactical phase neglected), the minimum capacity supported is of AAR 30, which is 1 aircraft every 2 minutes. If AAR 20 is applied, the program will not converge because there is no solution with these conditions (such as 30 minutes of maximum holding time, but not allowed to be increased due to capacity limitations as explained in section 3.1.5.). But if the delay of Pre-tactical Phase (plus noise) is added to this M1 flight data, with the same capacity (AAR 30) the delay on air will be much smaller, even negative, which means that the costs are being saved. Moreover, if capacity is even smaller (less than AAR 30, for instance AAR 20 or 12), the program is not able to find any solution (not convergent) due to demand and capacity imbalance. Nevertheless, adding the pre-tactical delays and noises for those AAR, the tactical optimization would be able to get results successfully, that is assigning delays to flights under desired conditions of maximum holding time, capacity, etc.

This means that without Pre-tactical optimization, the tactical model is only able to assign delays for AAR 30 for the LEBL airport (AAR 60 is not supported by LEBL because maximum capacity supported is AAR 42), but if pre-tactical delays were assigned before departure, the capacity could be as small as desired (AAR 20, 12, etc.).

At the following, in order to show the advantage of the combination of these two phases, two different results will be presented. The same AAR of 30 is taken for both cases, that is 1 aircraft every 2 minutes. One is the tactical result without the pre-tactical optimization process, that is taking directly M1 flight data and apply the tactical optimization. The other one is the result using the pre-tactical optimization, that is taking M1 and adding delay and noise assigned from pre-tactical phase. The difference will be shown in figure 4.8.

As seen in figure 4.8, the delays and costs are interdependent, thus when the difference in delay increases the difference in cost does the same. But the most noteworthy point is that in general the differences in delays and costs are positive values and since these are the delays and costs without pre-tactical optimization minus the ones with them, the plots is confirming that there are less delays and costs when adding pre-tactical delays and noises.

In order to see this result more clearly, the table 4.2 is presented, where the main values[2] to prove this idea are shown.

	Delay Total (min)	Cost Total (€)	Delay per AC (min)	Cost per AC (€)
Without Pretactical	115	167.9	0.27	0.4
With Pretactical	-191	-2912.1	-0.459	-7

Table 4.2: Delay and Cost without and with Pre-tactical Delay (AAR 30)

In this table 4.2 the advantage of using pre-tactical optimization is proved since the total delay and cost are much smaller than taking directly the M1 data, with the same capacity, that is AAR 30.

Now that the statement ”The usage of pre-tactical optimization in tactical phase makes smaller costs than only using M1 data when the same capacity is considered” is proved,

[2] Note that these delays are still delays with respect the virtual M3, which is M1+ Regulated M2+ Noise.

it is time to proceed with decreasing AARs. In figure 4.9, the delays and costs values for decreasing AARs are shown.

The simulation standard parameters for this subsection are:

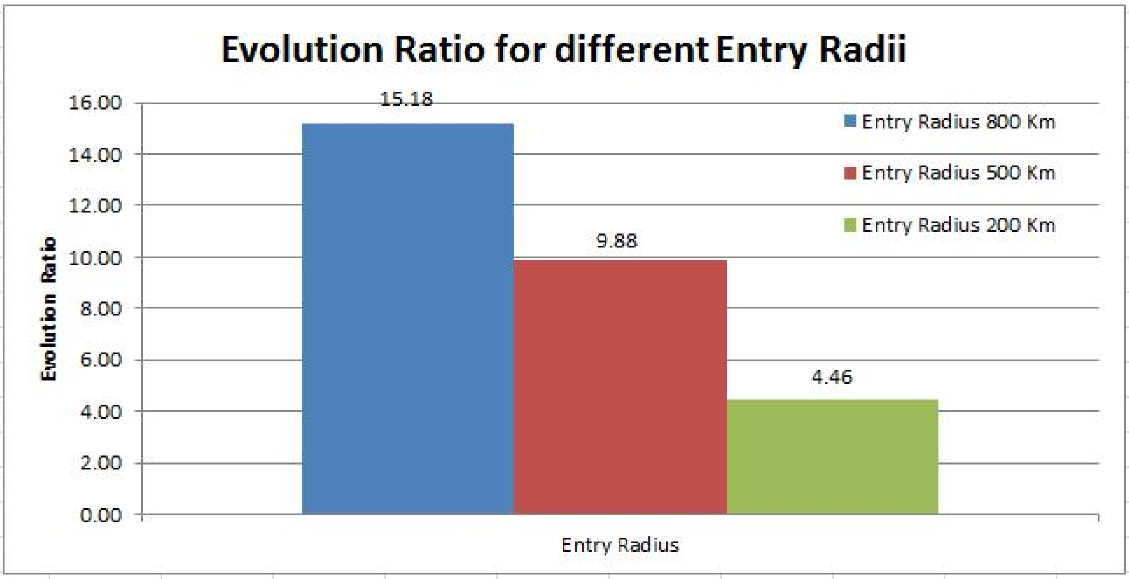
- Interval of study: 00:00:00 – 23:59:59 of 11/07/2015
- Airport: Barcelona (LEBL)
- Maximum Holding Time: 30 minutes
- Tactical AAR: 30, 20 and 12 (constant during the whole day)
- Entry Point Radius: 500 km
- $\varepsilon = 0$
- AHP non-unitary Cost Function: $min \sum_{l=1}^{n} \sum_{j=New_l}^{(Sch+T)_l} [K1_l \cdot K2_l \cdot K3_l \cdot K4] \cdot D_{lj} \cdot x_{lj}$
- Pre-tactical Delay (AAR 30, 20 and 12) and Noise applied.

Notice that in the figure 4.9, the tactical delays and costs are successfully computed for AAR 30, 20 and 12 when directly taking M1 data it was not even possible except for AAR 30. Furthermore, when decreasing capacity, the delay and cost are increased as expected.

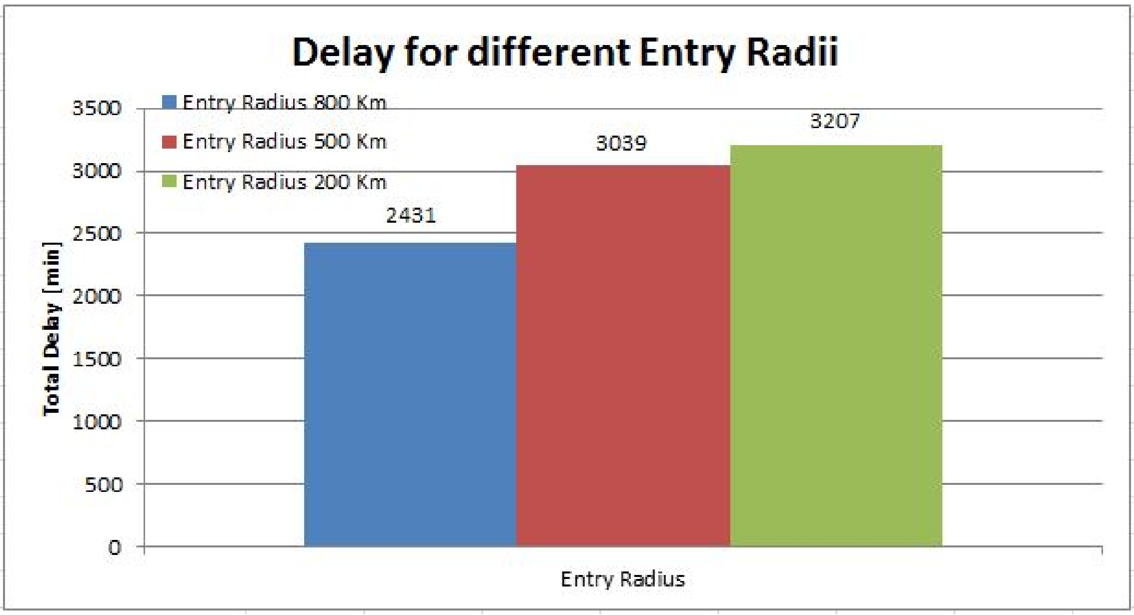
In this case, the combination of pre-tactical and tactical optimization is useful when there is a decrement in capacity in the destination airport. For instance, the capacity was set to be AAR 42 (maximum supported in LEBL) in scheduled flight plan, but due to other circumstances the capacity is decreased to AAR 30. Then if pre-tactical optimization is not applied, there will be congestion at landing phase (115 minutes and 167.9 € in total for tactical optimization). Nevertheless, if pre-tactical optimization is applied, then the congestion problem when arriving at destination airport will be solved (283 minutes and 1860 € in total for pre-tactical optimization). Taking this into account, having delayed all the flights before departure, when arriving at landing phase at destination airport, the delays and costs will be saved (-191 minutes and -2912 € in total for tactical optimization). Therefore, in exchange of assuming delays and costs at pre-tactical phase, delays and costs are saved in tactical phase. The final results of applying these models are a delay saving of 23 minutes and a cost saving of 1220 € with respect the delay and cost when taking directly scheduled flight plans.

In order to see this more in detail, figure 4.10 is presented. The global results of this strategic combination are shown. The sum of GHP [1] and AHP total delay and cost are shown.

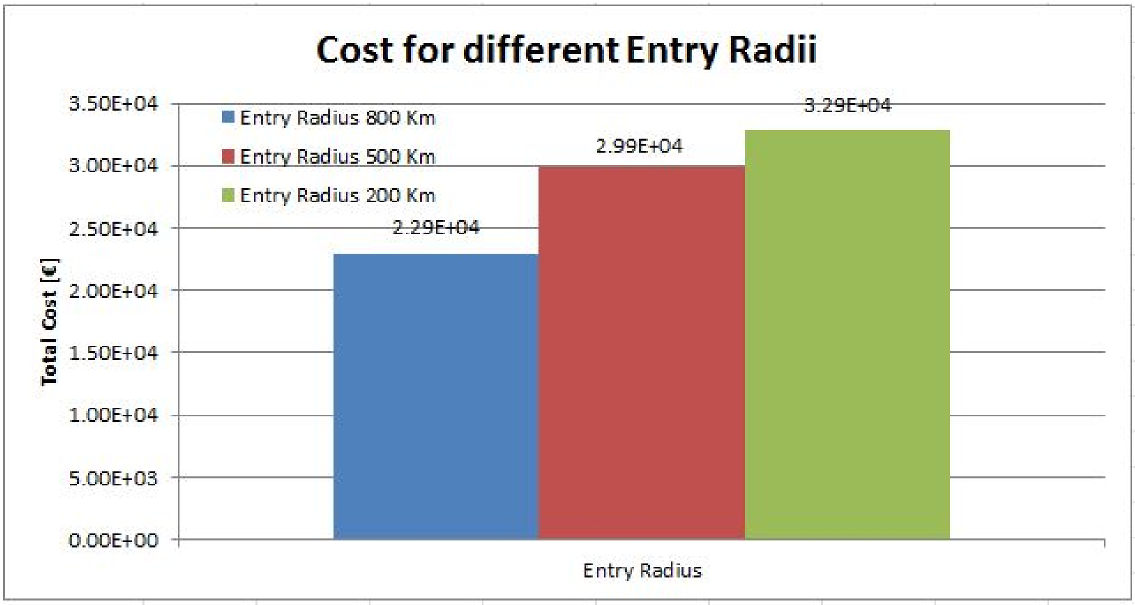
The delays and costs shown in figure 4.10 for each AAR is with respect the ones with capacity considered in M1 flight plan. That is to say, if capacity is reduced to AAR 30, then the global delay after processing through all the optimization simulations will be 92 minutes, which means a global cost of -1052 €.



(a) AHP Evolution Ratio

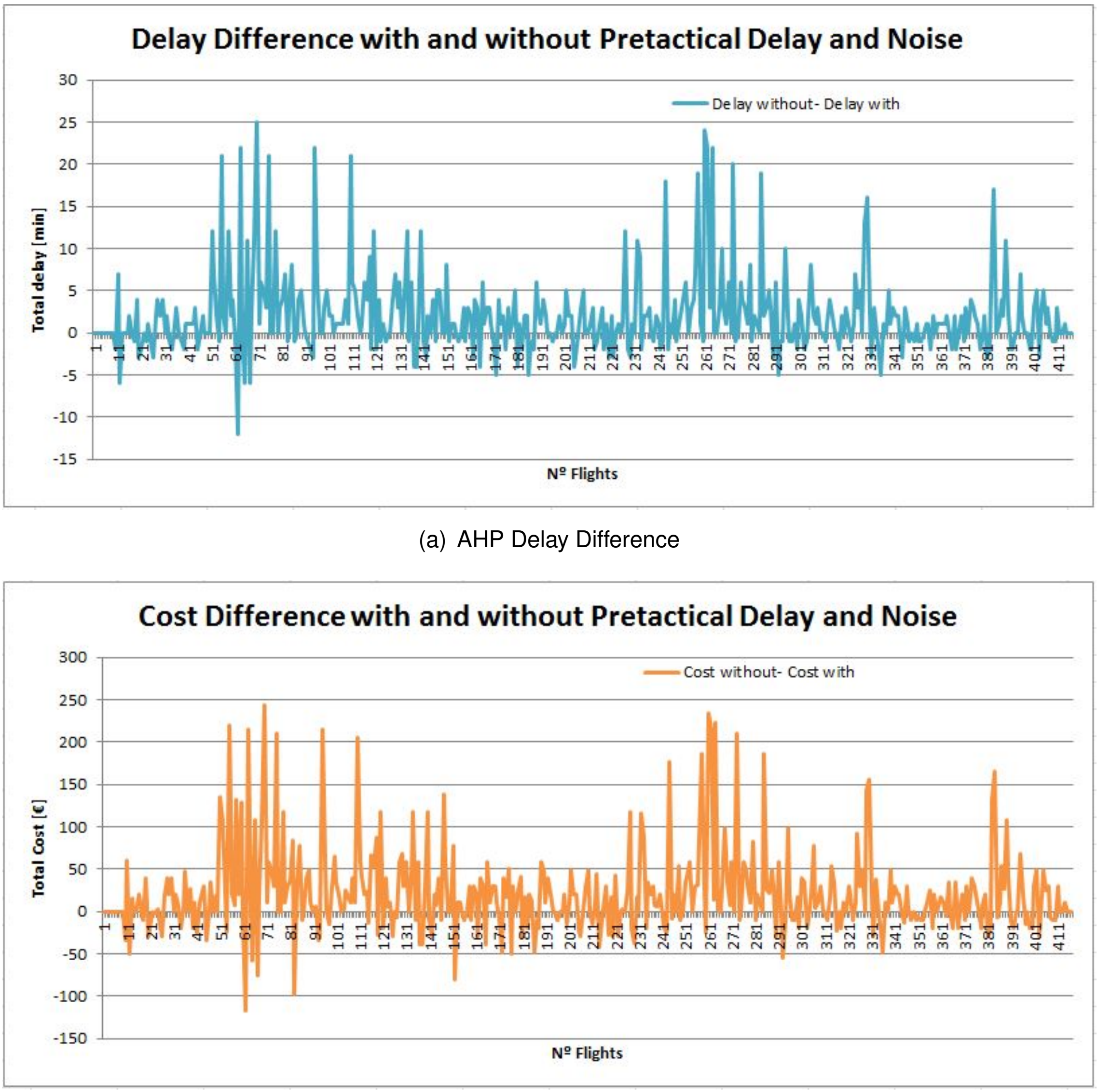


(b) AHP Delay



(c) AHP Cost

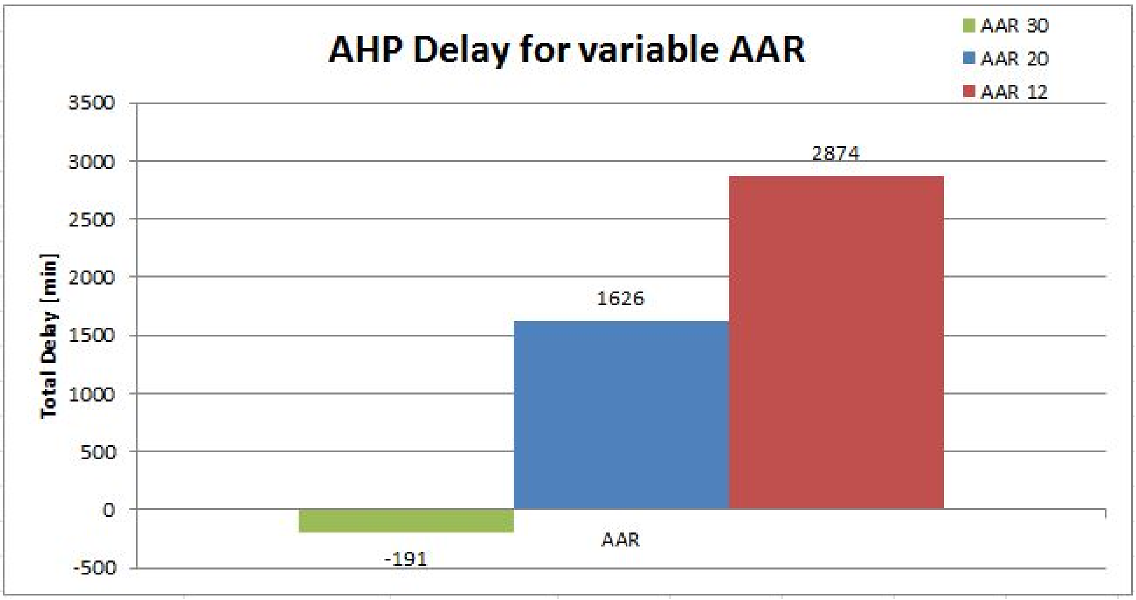
Figure 4.7: Plots of AHP Delay, Cost and Evolution Ratio for different Entry Radii



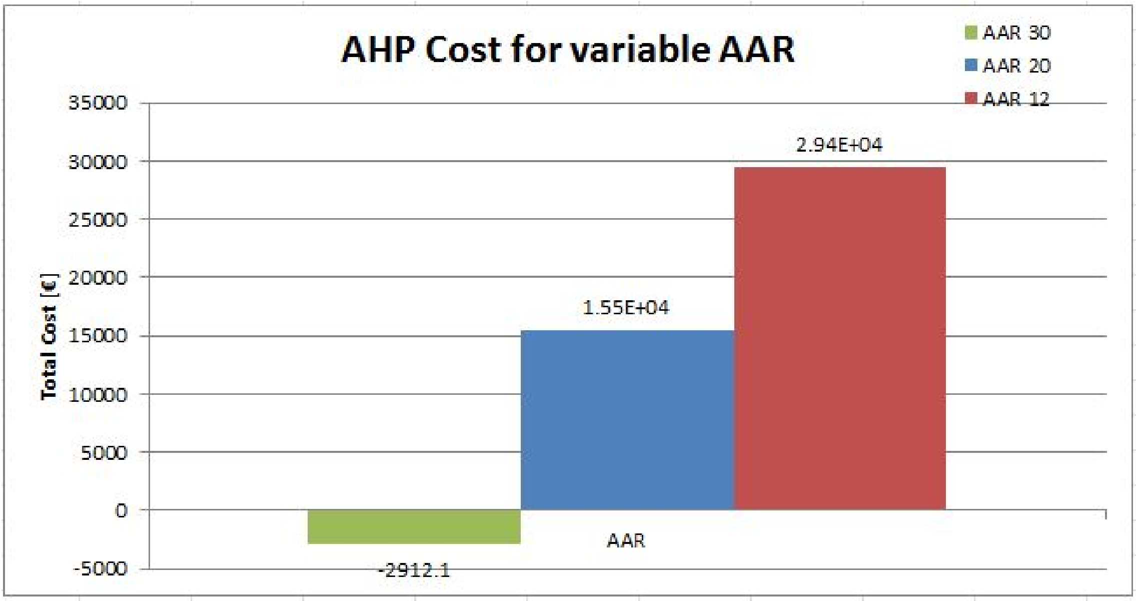
(a) AHP Delay Difference

(b) AHP Cost Difference

Figure 4.8: Plot of AHP Delay and Cost Difference with and without Pre-tactical Optimization

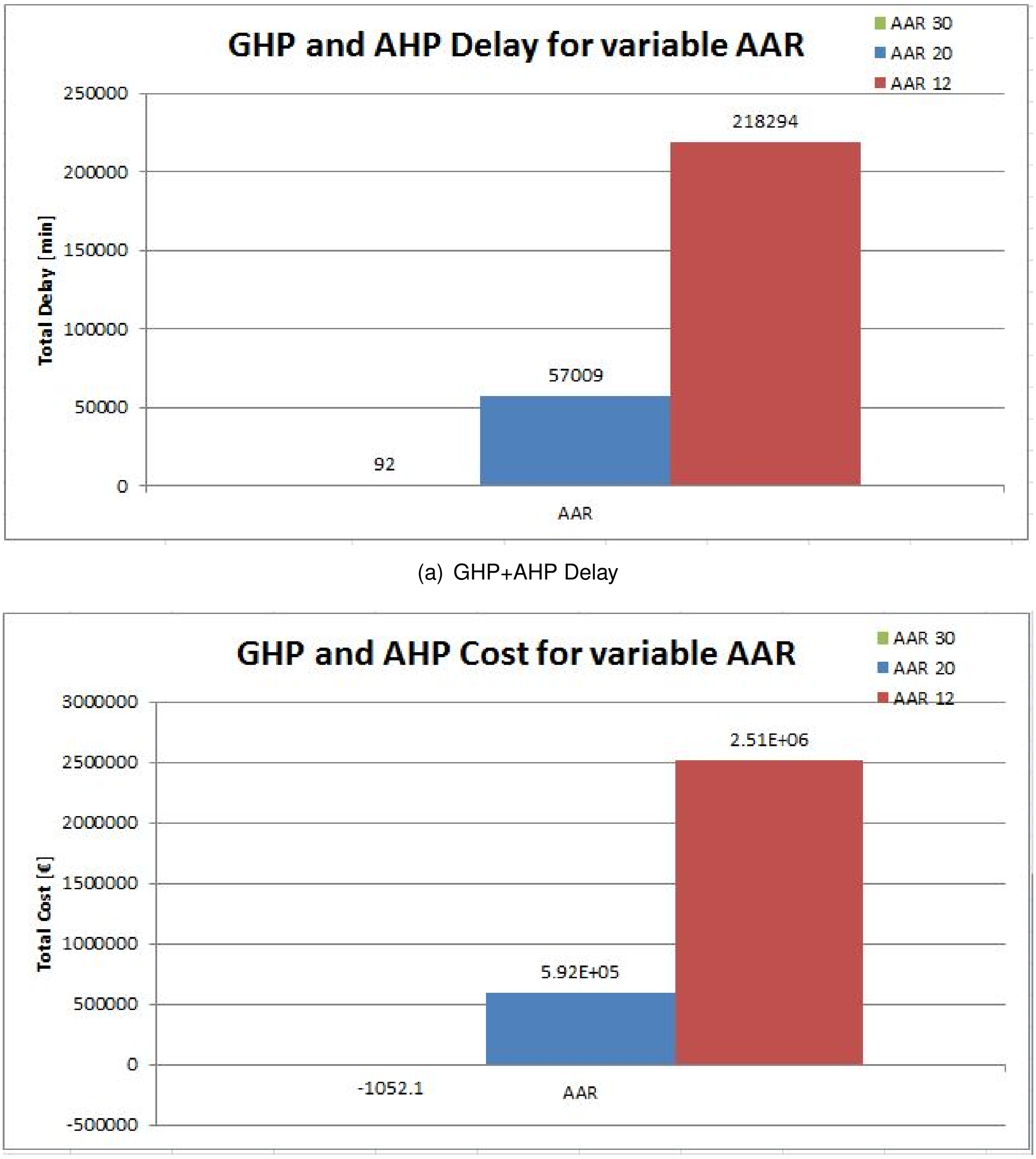


(a) AHP Delay



(b) AHP Cost Difference

Figure 4.9: Plot of AHP Delay and Cost for decreasing AAR from Pre-tactical Optimization



(a) GHP+AHP Delay

(b) GHP+AHP Cost

Figure 4.10: Plot of GHP+AHP Delay and Costs (Total Delay) for variable AAR

CONCLUSIONS

The main goal of this research is to implement a basic model of AHP to reduce economic costs when delaying flights in the air near the single destination airport, Barcelona-El Prat in this case.

Firstly, a pre-tactical phase of GHP is applied to make flights wait on ground according to capacity constraints in the Barcelona Airport. Then, once the flights are already flying towards this destination and entering the analysed area, these flights will perform a FRA route in order to make them fly in the shortest route. This will cause that the already ordered arrival queue according to capacity considered in pre-tactical phase be again disordered because some flights would arrive earlier than scheduled. If these flights could arrive in a disordered sequence, then the capacity limitation pre-considered is now again violated. For this reason, a second optimization to assign air delay is applied called Air Holding Program. This model is delaying again these flights in the air to make them land at destination according to capacity.

This model looks for a dynamic optimization method, which means that not all the flights are optimized ”once and for all” as in pre-tactical phase (traditional GHP optimization model). The aircraft inside the studied airspace have been assigned delays, but if any new flight enters this area, the simulator is automatically optimizing again to take into account this newest entered flight. In this way, the possible variations of delay assignment due to this new flight are taken into account. Furthermore, this model is capable of automatically deleting the flights that are leaving the airspace to not consider them any more because they are already entering the landing phase. These deleted flights will be communicated their final assigned holdings and delays after all the simulation process during their stay inside the area. Once they are entering the landing phase, they should firstly pass through the holding procedure to hold the time needed according to the delay assigned.

The main noteworthy comments and conclusions after processing the whole simulation for several times for different analysis are:

- This model is just a very basic model for a first approach of the AHP research. This means that many parameters and processes are simplified.
- When capacity is reduced in destination airport, firstly a GHP can be applied before departure [1]. Then, an AHP can be applied for a second optimization purpose. If traffic is directly taken to be applied an AHP, the capacity flexibility is very reduced, while using pre-tactical regulation the capacity could be as small as desired. This outlines the advantage of using GHP and AHP together.
- If unitary cost function is applied, then AHP model has the same policy than a RBS model.
- The application of FRA concept in this tactical model is very advantageous since it makes possible the dynamic optimization and the cost saving in most optimal cases.
- The main objective is to minimize the cost function but if the flights are arriving earlier than scheduled in the most optimal case, the delays could be negative values, which means that economic costs are saved with respect scheduled.
- If Maximum Holding Time is reduced, total delay is increased.

- If capacity is increased, total delay is decreased.
- If optimization airspace is decreased, the total delay is increased due to lack of evolution time.
- If capacity is reduced during tactical phase so maximum holding time must more than 30 minutes, then the best solution is to defer the flight to an alternate airport instead of making it hold more than maximum established time.
- Although this model is very basic and limited, the AWK script and Matlab code are very flexible regarding the airport and the day desired to be analysed. Any Saturday from April 2015 and any airport in Europe can be studied.

As explained, this model is very basic and some future improvements should be done if this research is retaken for its further studies. The main improvements to be implemented to this model will be explained in the next section.

FUTURE WORK

The main objective of this thesis is to assess the air traffic by applying an Air Holding Program to the flights when already arriving at destination airport.

This idea of AHP is very basic and generalized since it is the first approach for a tactical phase optimization. Thus, the whole model and all the interesting parameters are very simple and not sophisticated because they are not deeply studied since this research is just a degree thesis project and the given time is not enough to deeply study all the aspects and possible forms this model could take.

Therefore, for this reason, some future work will be detailed here in order to improve this model of AHP to make it richer and more defined:

- Separation between aircraft when sequencing the flights for landing. The category of the aircraft must be taken into account when queueing the aircraft. This will make more realistic the scenario and increase safety issues. Example: if an aircraft of category ”H” is followed by an aircraft of category ”L”, they should have a bigger time distance. This could be considered in the cost function, to avoid these disadvantaged cases.

- Variable slot interval time and capacity parameter. Instead of 1 minute slot, make more flexible the slot interval to be able to adjust it to more capacity cases. Here only 1 aircraft every X minutes could be considered, which lets other capacity cases uncovered, such as 2 aircraft every minute or 3 minutes. This improvement will make the capacity constraint more flexible to be adjusted to the real capacity cases.

- Variable AAR for different period of time studied. Example: AAR 30 from 8 a.m. to 12 a.m., then AAR 20 from 12 a.m. to 5 p.m. depending on the AAR supported during the whole day. This will make the capacity of the model more adjusted to a real case.

- Variable speed when considering the aircraft inside the airspace. In this thesis an average constant speed is considered for each flight to simplify the problem. But it is very recommendable to consider a variable speed scenario. This will make possible a better location of the aircraft and a more real scenario.

- Study deeply the cost function linearity. That is, make ε bigger than 0, which means it will not be linear. For example: $\varepsilon = 1$ (quadratic behaviour). This will improve the cost function performance.

- Variable Holding Time depending on the category of the aircraft. That is, depending on the fuel consumption of each aircraft category, assign a different holding time to every aircraft. In this way, the model will be more flexible to capacity constraints.

- Study other Noise scenarios. Here only Saturdays are studied because other days of the week follow a different distribution, thus only days on Saturday can be optimized. Analysing more scenarios of noise will make more flexible the day available for optimization.

If all these issues were attended, the final model should be more sophisticated and adjusted to a real scenario. The precision will also increase and the solutions will be more optimal since it will take into account more situations and parameters. Therefore, the next step to continue with this research should be the implementation of all these ideas.

BIBLIOGRAPHY

[1] Guillem Tarín ”Pre-tactical time window assignment optimization applied to the Barcelona airport case” September, 2015. iii, v, 3, 4, 13, 16, 18, 24, 33, 37, 43, 60

[2] Eurocontrol, *ATFM Users Manual*, edition n^{o}9.0, april 2003 5

[3] Eugene P. Gilbo, Volpe Center and Kenneth W. Howard, Arcon Corp. *Collaborative Optimization of Airport Arrival and Departure Traffic Flow Management Strategies for CDM* 3rd USA/Europe Air Traffic Management R and D Seminar, Napoli, 13-16 June 2000 5

[4] Michael Ball, University of Maryland, Robert Hoffman, Metron Aviation, Inc. and David J. Lovell, University of Maryland *RESPONSE MECHANISMS FOR DYNAMIC AIR TRAFFIC FLOW MANAGEMENT* 6th USA/Europe Air Traffic Management R and D Seminar 5

[5] Yigit Bekir Kaya and Gokhan Inalhan, Department of Aeronautics and Astronautics *A Dynamic Bayesian Belief Network Approach for Modelling the ATM Network Delays* ICRAT 2014 5

[6] James C. Jones, David J. Lovell and Michael O. Ball, University of Maryland *Algorithms for Dynamic Resequencing of En Route Flights to Relieve Terminal Congestion* ICRAT 2012 6

[7] Karthik Krishnamurthy, Titan Corporation, Bryan Barmore, NASA Langley Research Center and Frank Bussink, National Institute of Aerospace. *AIRBORNE PRECISION SPACING IN MERGING TERMINAL ARRIVAL ROUTES: A FAST-TIME SIMULATION STUDY* 6th USA/Europe Air Traffic Management R and D Seminar 6

[8] Octavio Richetta ”DYNAMIC SOLUTION TO THE GHPP”, *”Ground Holding Strategies for Air Traffic Control Under Uncertainty”* Massachusetts Institute of Technology, May 1991 6

[9] Mallory Jo Soldner ”Passenger-Centric Ground Holding: Including Connections in Ground Delay Program Decisions” MASSACHUSETTS INSTITUTE OF TECHNOLOGY, June 2009 6

[10] Andrea Ranieri, Lorenzo Castelli ,Universita degli Studi di Trieste *A Market Mechanism to Assign Air Traffic Flow Management Slots* Eighth USA/Europe Air Traffic Management Research and Development Seminar (ATM2009) 6

[11] Michael J. Hanowsky *A Model to Design a Stochastic and Dynamic Ground Delay Program Subject to Non-Linear Cost Functions* PhD thesis, Massachusetts Institute of Technology, 2008. 6

[12] Peter B. Vranas, Dimitris J. Bertsimas and Amedeo R. Odoni ”Dynamic Ground-Holding Policies for a Network of Airports” Massachusetts Institute of Technology, June 1992 7

[13] Nilim, A., L. Ghaoui, and V. Duong *Robust Dynamic Routing of Aircraft under Uncertainty* IEE 21st Digital Avionics Systems Conference, 2002 7

[14] A.R. Odoni. "Flow Control of Congested Networks" *"The flow management problem in air traffic control"* pages 269-288, 1987. 7

[15] PETER B. VRANAS, DIMITRISJ . BERTSIMASa nd AMEDEOR . ODONI *THE MULTI-AIRPORT GROUND-HOLDING PROBLEM IN AIR TRAFFIC CONTROL* Massachusetts Institute of Technology, Cambridge, Massachusetts 7

[16] Calculate distance, bearing and more between Latitude/Longitude points http://www.movable-type.co.uk/scripts/latlong.html 23th July 2015 10

[17] SkyVector Aeronautical Charts https://skyvector.com/ 23th July 2015 xi, 11

[18] Gurobi Optimizer http://www.gurobi.com/products/gurobi-optimizer 23th July 2015 14

[19] Free Route Airspace (FRA) http://www.skybrary.aero/index.php/Free_Route_Airspace_(FRA) 23th July 2015 xi, 14

[20] Fuel economy in aircraft https://en.wikipedia.org/wiki/Fuel_economy_in_aircraft 23th July 2015 21

[21] Holding (aeronautics) https://en.wikipedia.org/wiki/Holding_(aeronautics) 23th July 2015 21

[22] EUROPEAN COMMISSION "ENERGY PRODUCTS AND ELECTRICITY" *"EXCISE DUTY TABLES Part II – Energy products and Electricity"* page 29, January 2015. 22

[23] Air Operations "Part 4 The Operation at High Speed" *"Introduction to the operation of a commercial airplane"* Slide 10, February 2015. xi, 22, 23

[24] Eurocontrol Web Page https://www.eurocontrol.int/ 20th July 2015 51

[25] Examples with awk: A short introduction http://www.linuxfocus.org/English/September1999/article103.html 23th July 2015 52

[26] NEST Modelling Tool http://www.eurocontrol.int/services/nest-modelling-tool 20th July 2015 63

APPENDICES

APPENDIX A. EUROCONTROL FLIGHT DATA DOWNLOAD, AWK SOFTWARE UTILITY AND NEST TESTING

A.1. Eurocontrol ALL_FT+ data file download

One of the first's steps in this project is to obtain the information that will be analyzed by the code developed. The study is focused on the European territory, so the best data source for this purpose is Eurocontrol. Now, the different steps to download data files are going to be explained.

A.1.1. Eurocontrol User Registration and subscription to DDR2/DDR3

To access the information desired, the registration process is completely necessary to be accomplished in order to obtain the credentials. To do so, the OneSky online extranet service offered in the main page of Eurocontrol is visited, as shown in figure A.1 and then complete the self-registration process [24].

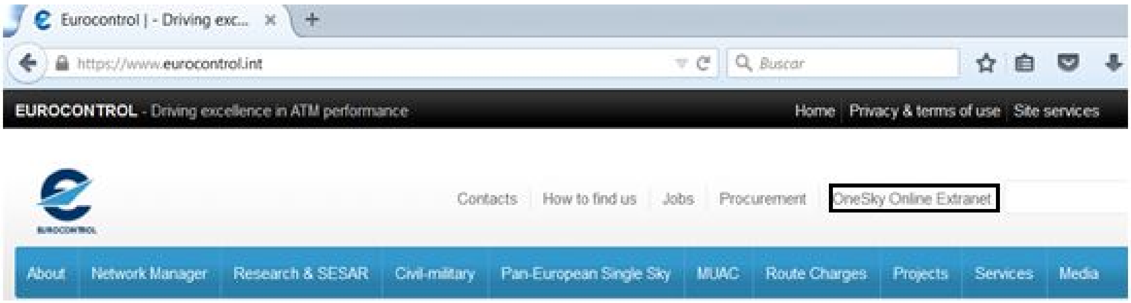


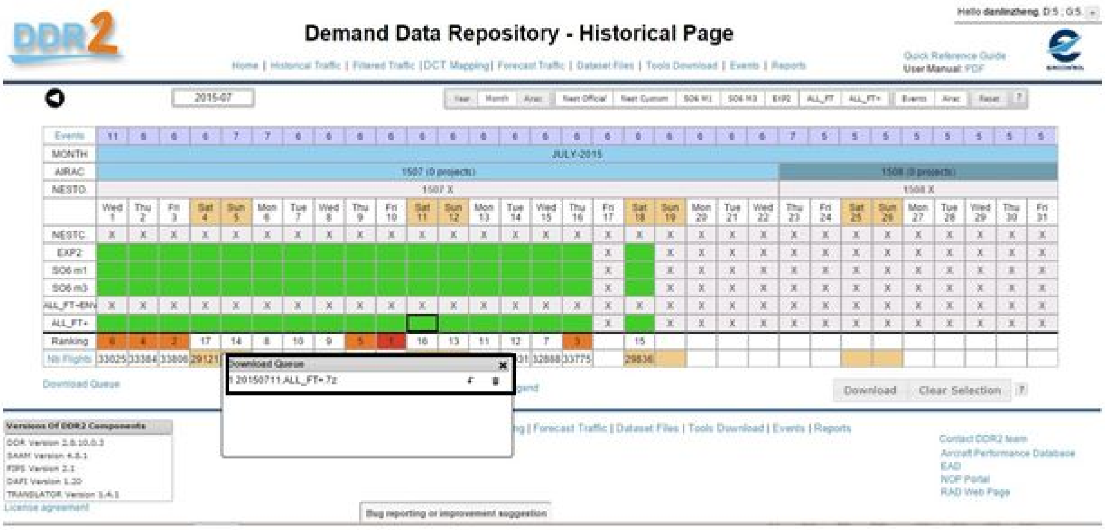
Figure A.1: Eurocontrol main webpage

Once registered and also logged in, to be able to download the historical traffic data, access to the DDR2/3 has to be requested – Demand Data Repository, which is a platform containing daily flight data from the last years. To do so, the option "subscribe for online services" must be selected and then select DDR2. Notice that the request could not be processed instantly, so some days until complete access should be waited.

A.1.2. ALL_FT+ data file download

If the access to Demand Data Repository (DDR2/3) is already available, then a page similar to the figure A.2 should be shown.

As seen in the figure A.2, there are a lot of file formats available (i.e. EXP2, SO6, ALL_FT+...), but the only one needed in this thesis is the ALL_FT+ because it has the most complete flight information needed. Note that it is mandatory to download only the ALL_FT+ format in order to be able to run the AWK script and consequently the Matlab



DDR2

Demand Data Repository - Historical Page

Hello danlinzheng D5 ; G5

Home | Historical Traffic | Filtered Traffic |DCT Mapping| Forecast Traffic | Dataset Files | Tools Download | Events | Reports

Quick Reference Guide
User Manual: PDF

2015-07

Events	11	6	6	6	7	7	6	6	6	6	6	6	6	6	6	6	6	6	6	6	6	6	7	5	5	5	5	5	5	5	5
MONTH	JULY-2015																														
AIRAC	1507 (0 projects)																						1508 (0 projects)								
NESTO	1507 X																						1508 X								
	Wed 1	Thu 2	Fri 3	Sat 4	Sun 5	Mon 6	Tue 7	Wed 8	Thu 9	Fri 10	Sat 11	Sun 12	Mon 13	Tue 14	Wed 15	Thu 16	Fri 17	Sat 18	Sun 19	Mon 20	Tue 21	Wed 22	Thu 23	Fri 24	Sat 25	Sun 26	Mon 27	Tue 28	Wed 29	Thu 30	Fri 31
NESTC	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X
EXP2	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	X	🟩	X	X	X	X	X	X	X	X	X	X	X	X	X
SO6 m1	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	X	🟩	X	X	X	X	X	X	X	X	X	X	X	X	X
SO6 m3	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	X	🟩	X	X	X	X	X	X	X	X	X	X	X	X	X
ALL_FT+ENV	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X
ALL_FT+	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	X	🟩	X	X	X	X	X	X	X	X	X	X	X	X	X
Ranking	6	4	2	17	14	8	10	9	5	1	16	13	11	12	7	3		15													
Nb Flights	33025	33384	33806	29121										01	32888	33775		29836													

Download Queue

Download Queue

1 20150711.ALL_FT+.7z

Versions Of DDR2 Components
DDR Version 2.8.10.0.3
SAAM Version 4.8.1
PIPS Version 2.1
DAFI Version 1.20
TRANSLATOR Version 1.4.1
License agreement

ng | Forecast Traffic | Dataset Files | Tools Download | Events | Reports

Contact DDR2 team
Aircraft Performance Database
EAD
NOP Portal
RAD Web Page

Bug reporting or improvement suggestion

Figure A.2: Demand Data Repository – Historical Page

code to perform the simulations. The information is structured by days, so a specific day should be selected (i.e. 11/07/2015 in figure A.2).

A.2. AWK output data file

In this section, the AWK software utility, the script code to be executed and the final output file desired will be explained in detail.

A.2.1. AWK Utility

In this web page [25] a brief introduction to AWK is available. In this case, the utility of this software is to read the huge All_FT+ file (usually more than 1 GB), line by line. Since there are many data to be filtered, it is very recommendable to make a script code with the filtering information needed and then execute this file in the AWK command. The script input would be the All_FT+ file, and the output should be data files obtained by "print" function from the script, which would then be the input to Matlab Program.

A.2.2. Complete Commented Script

Once the ALL_FT + data file of the day to be analysed is downloaded and the AWK is installed. The next step is to code the script in order to filter only the information needed as the input for the Matlab program.

Here the complete script code is presented below:

```
#########################################################################
#######################Final Degree Thesis#################################
############ AWK SCRIPT FOR ALL_FT+ DATA FILE FILTERING#####################
```

```
##################AUTORS: GUILLEM TARIN AND DANLIN ZHENG########################
#################Technical University of Catalonia (UPC)########################
################################################################################
#################################JULY-2015######################################
################################################################################
################################################################################

#########################BEGIN SCRIPT CODE######################################

function LatLonRad(LATLON) #Function for latitude and longitude conversion from degrees to radians.
{
    LAT=LATLON[1]LATLON[2]+(LATLON[3]LATLON[4])/60+(LATLON[5]LATLON[6])/3600;#Decimal degrees
    if (LATLON[7]== "S")
    {
        LAT=-LAT
    }

    LON=LATLON[8]LATLON[9]LATLON[10]+(LATLON[11]LATLON[12])/60+(LATLON[13]LATLON[14])/3600;

    if (LATLON[15]== "W")
    {
       LON=-LON
    }

    LAT=LAT*pi/180;#Radians
    LON=LON*pi/180;#Radians
}

#Begin the main code.

BEGIN {

# Input variables modifiable by the user:
Airport="LEBL";
Date = "20150524"; #Date of the file you are analysing.
RadiusEntry=500;#KM
RadiusExit=50;#KM
R=6378.1; #KM
pi = atan2(0, -1);


FS = ";" #column separator for input file
OFS = ";" #column separator for out file
}

{
# Input parameters

    rM1=$86;
    rM2=$100;
    rM3=$114;
    OrigID=$1;
    DestID=$2;
    RegistrationMark=$58;
    IfpsId=$7;
    AircraftTypeIcaoId=$5;

    n=split(rM1,M1," ") #Length of the array M1
    g=split(rM2,M2," ") #Length of the array M2
    k=split(rM3,M3," ") #Length of the array M3

    #To get the first common variables:
    split (M1[1], M11, ":")
    split (M2[1], M21, ":")
    split (M3[1], M31, ":")
    split (M1[n], M12, ":")
    split (M2[g], M22, ":")
    split (M3[k], M32, ":")

    split (M12[7],LATLONAirport,"");
    LatLonRad(LATLONAirport)

    LATAirport=LAT;
    LONAirport=LON;

    split (M32[7],LATLONAirportR,"");
    LatLonRad(LATLONAirportR)

    LATAirportR=LAT;
    LONAirportR=LON;

    #Getting information from the departure airport (DEPARTURE FLIGHTS):

    if ((OrigID == Airport) && (DestID != Airport) && M12[5]>50) #The destination and origin airports
    must be different and destination airport must be farer than 50km.
    {
split (M11[1],M111,""); # Split hours
Date2 = M111[1]M111[2]M111[3]M111[4]M111[5]M111[6]M111[7]M111[8]; #Date of the data file
Dif = Date-Date2; #We can have a departure flight on the same day or one day after,
but not one day before.
# If Dif = 0; the flights are in the same day.
# If Dif = -1, one flight corresponds to the day analysed and the other one to the day after.
# If Dif = 1, one flight corresponds to the day analysed and the other one to the day before.

        if (M31[1]== "")#If empty, flight has been cancelled
        {

if((Dif == 0) || (Dif == -1))# We filter the flights according to the date
{
#M31[1] = -1; Flights that have M3 point empty mean that they have been cancelled.
print RegistrationMark,IfpsId,AircraftTypeIcaoId,DestID,M11[1],-1> "ALL_FT_DEP.dat"
}
```

```
        }
        else
        {
if((Dif == 0) || (Dif == -1)) # We filter the flights according to the date
{
            print RegistrationMark,IfpsId,AircraftTypeIcaoId,DestID,M11[1],M31[1]> "ALL_FT_DEP.dat"
}
        }

    }



split (M12[1],M121,""); # Split hours
Date3 = M121[1]M121[2]M121[3]M121[4]M121[5]M121[6]M121[7]M121[8]; #Date of the data file
Dif2 = Date-Date3; #If we are in the same day, the difference will be 0

# Getting information from the arrival airport (ARRIVAL FLIGHTS)

    counterS=1; counterR=1; #To search for two points

    if ((DestID==Airport) && (OrigID != Airport) && (Dif2 == 0) && M12[5]>50)# The destination and
    origin airports must be different, destination airport must be farer than 50km and the day
    studied.
    {
        if (AircraftTypeIcaoId=="A333" || AircraftTypeIcaoId=="A332" || AircraftTypeIcaoId=="A388" ||
        AircraftTypeIcaoId=="B77W" || AircraftTypeIcaoId=="A306" || AircraftTypeIcaoId=="C550" ||
        AircraftTypeIcaoId=="B763" || AircraftTypeIcaoId=="B764" || AircraftTypeIcaoId~/AN/ ||
        AircraftTypeIcaoId=="A380" || AircraftTypeIcaoId=="B747" || AircraftTypeIcaoId=="A340" ||
        AircraftTypeIcaoId=="B777" || AircraftTypeIcaoId=="A350" || AircraftTypeIcaoId~/MD/ ||
        AircraftTypeIcaoId~/Il/ || AircraftTypeIcaoId=="B787" || AircraftTypeIcaoId~/L/ ||
        AircraftTypeIcaoId=="A330" || AircraftTypeIcaoId=="B767" || AircraftTypeIcaoId=="A300" ||
         AircraftTypeIcaoId=="A310" || AircraftTypeIcaoId=="A400" )
        {
           Cat="H";
        }
        else if (AircraftTypeIcaoId=="E120" || AircraftTypeIcaoId=="B738" || AircraftTypeIcaoId=="F900"
        AircraftTypeIcaoId=="GALX" || AircraftTypeIcaoId=="C680" || AircraftTypeIcaoId=="E135" ||
        AircraftTypeIcaoId=="C56X" || AircraftTypeIcaoId=="GLF4" || AircraftTypeIcaoId=="B733" ||
        AircraftTypeIcaoId=="DH8D" || AircraftTypeIcaoId=="B734" || AircraftTypeIcaoId=="C750" ||
        AircraftTypeIcaoId=="B739" || AircraftTypeIcaoId=="B752" || AircraftTypeIcaoId~/H/ ||
        AircraftTypeIcaoId=="B757" || AircraftTypeIcaoId~/TU/ || AircraftTypeIcaoId=="A321" ||
        AircraftTypeIcaoId=="B737" || AircraftTypeIcaoId=="A319" || AircraftTypeIcaoId=="A320" ||
        AircraftTypeIcaoId=="A318" || AircraftTypeIcaoId~/CS/ || AircraftTypeIcaoId=="B717" ||
        AircraftTypeIcaoId=="E190" || AircraftTypeIcaoId=="F100" || AircraftTypeIcaoId=="RJ85" ||
        AircraftTypeIcaoId=="E175" || AircraftTypeIcaoId=="E170" || AircraftTypeIcaoId~/CR/ ||
        AircraftTypeIcaoId=="Q400" || AircraftTypeIcaoId=="E145" || AircraftTypeIcaoId~/AT/ ||
        AircraftTypeIcaoId=="V800" || AircraftTypeIcaoId~/DC/ || AircraftTypeIcaoId=="B377" ||
        AircraftTypeIcaoId=="B707" || AircraftTypeIcaoId=="B720" || AircraftTypeIcaoId=="B727" ||
        AircraftTypeIcaoId=="C880")
        {
           Cat="M";
        }
        else if(AircraftTypeIcaoId=="E55P" || AircraftTypeIcaoId=="SW4" || AircraftTypeIcaoId=="C510" ||
        AircraftTypeIcaoId=="C25A")
        {
           Cat="L";
        }
        else
        {
           Cat="M";
        }
        for (i=1;i<=n;i=i+1) #First loop to look for M1 entry point and exit point
        {
            split (M1[i],subfield,":"); # Split the different data of a point
            split (subfield[7],LATLON,"");
            DistS=subfield[5];
            DistRestS=M12[5]-DistS;#Remaining distance from point to Destination.

            LatLonRad(LATLON)

            deltaLAT=LAT-LATAirport;
            deltaLON=LON-LONAirport;

            a=((sin(deltaLAT/2))^2)+(cos(LATAirport)*cos(LAT)*((sin(deltaLON/2))^2));
            c=2*atan2(sqrt(a),sqrt(1-a));
            DistanceS=R*c; #Km

            if ((DistanceS<=RadiusEntry && counterS==1) || (DistanceS<=RadiusExit && counterS==2))
            #500 Km of radius for entry point and 50km of radius for exit point
            {
                if (DistanceS<=RadiusEntry && counterS==1)
                {
                    DistRestS1=DistRestS;
                    DistanceS1=DistanceS;
                    LATLONS1=subfield[7];
                    TimeS1=subfield[1];
                    FLS1=subfield[4];
                }

                if (DistanceS<=RadiusExit && counterS==2)
                {
                    DistRestS2=DistRestS;
                    DistanceS2=DistanceS;
                    LATLONS2=subfield[7];
                    TimeS2=subfield[1];
                    FLS2=subfield[4];
                }

                if(counterS==2)
                {
split (TimeS1,TimeSHMS,"");
TimeSH=TimeSHMS[9]TimeSHMS[10]+(TimeSHMS[11]TimeSHMS[12])/60+(TimeSHMS[13]TimeSHMS[14])/3600;#Hours
```

```
split (TimeS2,FTimeSHMS,"");
FTimeSH=FTimeSHMS[9]FTimeSHMS[10]+(FTimeSHMS[11]FTimeSHMS[12])/60+(FTimeSHMS[13]FTimeSHMS[14])/3600;
if (FTimeSHMS[9]==0 && TimeSHMS[9]==2)
{
    FTimeSH=FTimeSH+24
}

TimeS1=TimeSHMS[9]TimeSHMS[10]TimeSHMS[11]TimeSHMS[12]TimeSHMS[13]TimeSHMS[14];
TimeS2=FTimeSHMS[9]FTimeSHMS[10]FTimeSHMS[11]FTimeSHMS[12]FTimeSHMS[13]FTimeSHMS[14];

                    MeanSpeedS=(DistRestS1-DistRestS2)/(FTimeSH-TimeSH) #Km/h

                    NTimeH=(DistanceS1-DistanceS2)/MeanSpeedS;#Hours

                    sec=(NTimeH+TimeSH)*3600;#Seconds

                    #convert sec to h:m:s
                    hour=sec/3600;
                    Hour=int(hour);

                    if (Hour<10)
                    {
                        Hour=0Hour
                    }

                    if (Hour==24)
                    {
                        Hour=00
                    }

if (Hour>24)
{
aux=Hour-24;
Hour=0aux;
}

                    mins=( sec / 60 ) % 60 ;
                    Min=int(mins);
                    if (Min<10)
                    {
                        Min=0Min
                    }

                    secs=sec % 60;
                    Sec=int(secs);

                    if (Sec<10)
                    {
                    Sec=0Sec
                    }
                    NETAS=Hour Min Sec; #New Scheduled Exit Point Time in hhmmss

                    if (M22[1]== "") #If empty, scheduled = Regulated flight plan
                    {
                        M22[1] = M12[1];
                    }

                    if (M31[1]== "")#If empty, flight has been cancelled
                    {
                        LATLONR1=-1;
                        TimeR1=-1;
                        DistanceR1=-1;
                        FLR1=-1;
                        NETAR=-1;
                        MeanSpeedR=-1;
                        DistRestR1=-1;
                        LATLONR2=-1;
                        TimeR2=-1;
                        DistanceR2=-1;
                        FLR2=-1;
                        DistRestR2=-1;


print RegistrationMark,IfpsId,AircraftTypeIcaoId,Cat,OrigID,M11[1],-1,M12[1],M22[1],-1,M12[5],-1,
LATLONS1,TimeS1,DistRestS1,DistanceS1,FLS1,LATLONS2,TimeS2,DistRestS2,DistanceS2,FLS2,MeanSpeedS,NETAS,
LATLONR1,TimeR1,DistRestR1,DistanceR1,FLR1,LATLONR2,TimeR2,DistRestR2,DistanceR2,FLR2,MeanSpeedR,NETAR
> "ALL_FT_ARR.dat"

                    }
                    else
                    {

for (i=1;i<=k;i=i+1)#Second loop to look for M3 entry point and exit point once M1 entry point and
exit point are found
{
split (M3[i],subfield,":");
split (subfield[7],LATLON,"");
DistR=subfield[5];
DistRestR=M32[5]-DistR;

LatLonRad(LATLON)

deltaLAT=LAT-LATAirportR;
deltaLON=LON-LONAirportR;

a=((sin(deltaLAT/2))^2)+(cos(LATAirportR)*cos(LAT)*((sin(deltaLON/2))^2));
c=2*atan2(sqrt(a),sqrt(1-a));
DistanceR=R*c; #Km

if ((DistanceR<=RadiusEntry && counterR==1) || (DistanceR<=RadiusExit && counterR==2))#500 Km of
radius for entry point and 50km of radius for exit point
```

```
{
if (DistanceR<=RadiusEntry && counterR==1)
{
DistRestR1=DistRestR;
DistanceR1=DistanceR;
LATLONR1=subfield[7];
TimeR1=subfield[1];
FLR1=subfield[4];
}

if (DistanceR<=RadiusExit && counterR==2)
{
DistRestR2=DistRestR;
DistanceR2=DistanceR;
LATLONR2=subfield[7];
TimeR2=subfield[1];
FLR2=subfield[4];
}

if(counterR==2)
{
split (TimeR1,TimeRHMS,"");
TimeRH=TimeRHMS[9]TimeRHMS[10]+(TimeRHMS[11]TimeRHMS[12])/60+(TimeRHMS[13]TimeRHMS[14])/3600;#Hours
split (TimeR2,FTimeRHMS,"");
FTimeRH=FTimeRHMS[9]FTimeRHMS[10]+(FTimeRHMS[11]FTimeRHMS[12])/60+(FTimeRHMS[13]FTimeRHMS[14])/3600;

if (FTimeRHMS[9]==0 && TimeRHMS[9]==2)
{
   FTimeRH=FTimeRH+24
}
TimeR1=TimeRHMS[9]TimeRHMS[10]TimeRHMS[11]TimeRHMS[12]TimeRHMS[13]TimeRHMS[14];
TimeR2=FTimeRHMS[9]FTimeRHMS[10]FTimeRHMS[11]FTimeRHMS[12]FTimeRHMS[13]FTimeRHMS[14];


MeanSpeedR=(DistRestR1-DistRestR2)/(FTimeRH-TimeRH) #Km/h
NTimeH=(DistanceR1-DistanceR2)/MeanSpeedR;#Hours
sec=(NTimeH+TimeRH)*3600;#Seconds

#convert sec to h:m:s
hour=sec/3600;
Hour=int(hour);

if (Hour<10)
{Hour=0Hour}

if (Hour==24)
{Hour=00}

if (Hour>24)
{
aux=Hour-24;
Hour=0aux;
}

mins=( sec / 60 ) % 60 ;
Min=int(mins);

if (Min<10)
{Min=0Min}

secs=sec % 60;
Sec=int(secs);

if (Sec<10)
{Sec=0Sec}

NETAR=Hour Min Sec; #New Real Exit Point Time in hhmmss

print RegistrationMark,IfpsId,AircraftTypeIcaoId,Cat,OrigID,M11[1],M31[1],M12[1],M22[1],M32[1],M12[5],
M32[5],LATLONS1,TimeS1,DistRestS1,DistanceS1,FLS1,LATLONS2,TimeS2,DistRestS2,DistanceS2,FLS2,MeanSpeedS,
NETAS,LATLONR1,TimeR1,DistRestR1,DistanceR1,FLR1,LATLONR2,TimeR2,DistRestR2,DistanceR2,FLR2,MeanSpeedR,
NETAR > "ALL_FT_ARR.dat"

                            }
                            counterR=counterR+1;
                          }
                         }
                     }
             }
            counterS=counterS+1;
        }
    }
}
}
```

A.2.3. AWK output files format

When the script is already created and operative, it is time to execute it and also generate the output files that will be used when working with Matlab.

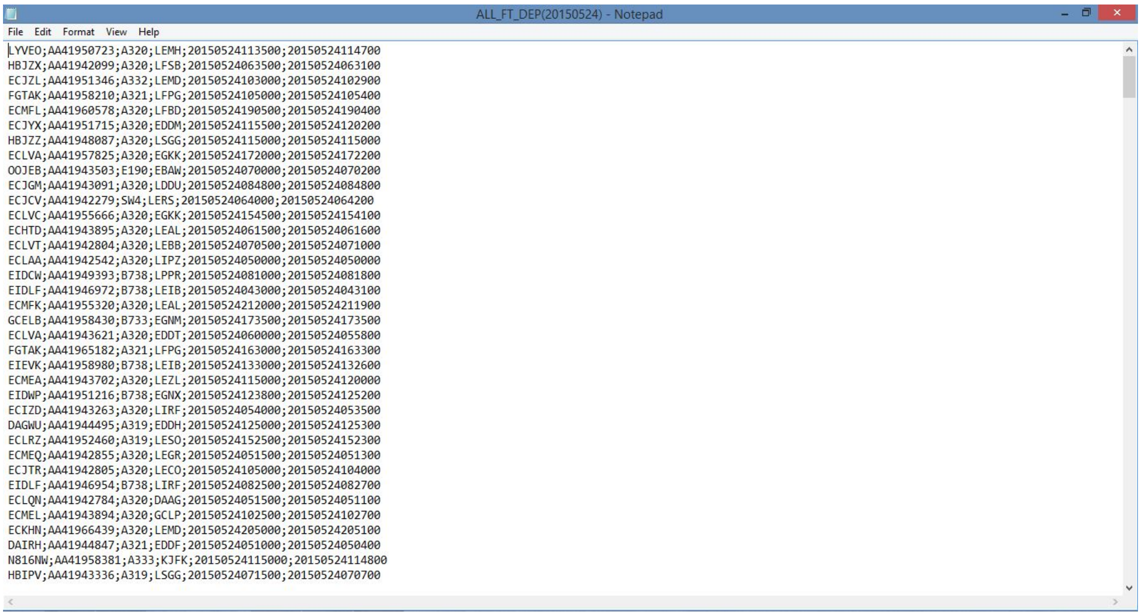
After Eurocontrol data processing using the AWK script, two files with all the information filtered are obtained. In there, it contains information about all the aircraft departing and arriving at LEBL airport during a day selected. In the script the date considered is 24-05-2015. The following and final step is to describe how the output files are structured and the data they contain.

For the departure file[1], it is organized in 6 fields. Below, it is described the 6 fields, their definition, units and an example:

[1] Only useful for the pre-tactical phase optimization model [1], since the script is shared for both thesis.

1. Registration Mark: ”LYVEO”
2. Aircraft ID: ”AA41950723”
3. Aircraft Type: ”A320”
4. Destination Airport ID: ”LEMH”
5. Scheduled date and time of departure [yyyymmddhhmmss]: ”20150524113500”
6. Real date and time of departure [yyyymmddhhmmss] (-1 if cancelled): ”20150524114700”

In the figure A.3 it shows an example of this format.



```
LYVEO;AA41950723;A320;LEMH;20150524113500;20150524114700
HBJZX;AA41942099;A320;LFSB;20150524063500;20150524063100
ECJZL;AA41951346;A332;LEMD;20150524103000;20150524102900
FGTAK;AA41958210;A321;LFPG;20150524105000;20150524105400
ECMFL;AA41960578;A320;LFBD;20150524190500;20150524190400
ECJYX;AA41951715;A320;EDDM;20150524115500;20150524120200
HBJZZ;AA41948087;A320;LSGG;20150524115000;20150524115000
ECLVA;AA41957825;A320;EGKK;20150524172000;20150524172200
OOJEB;AA41943503;E190;EBAW;20150524070000;20150524070200
ECJGM;AA41943091;A320;LDDU;20150524084800;20150524084800
ECJCV;AA41942279;SW4;LERS;20150524064000;20150524064200
ECLVC;AA41955666;A320;EGKK;20150524154500;20150524154100
ECHTD;AA41943895;A320;LEAL;20150524061500;20150524061600
ECLVT;AA41942804;A320;LEBB;20150524070500;20150524071000
ECLAA;AA41942542;A320;LIPZ;20150524050000;20150524050000
EIDCW;AA41949393;B738;LPPR;20150524081000;20150524081800
EIDLF;AA41946972;B738;LEIB;20150524043000;20150524043100
ECMFK;AA41955320;A320;LEAL;20150524212000;20150524211900
GCELB;AA41958430;B733;EGNM;20150524173500;20150524173500
ECLVA;AA41943621;A320;EDDT;20150524060000;20150524055800
FGTAK;AA41965182;A321;LFPG;20150524163000;20150524163300
EIEVK;AA41958980;B738;LEIB;20150524133000;20150524132600
ECMEA;AA41943702;A320;LEZL;20150524115000;20150524120000
EIDWP;AA41951216;B738;EGNX;20150524123800;20150524125200
ECIZD;AA41943263;A320;LIRF;20150524054000;20150524053500
DAGWU;AA41944495;A319;EDDH;20150524125000;20150524125300
ECLRZ;AA41952460;A319;LESO;20150524152500;20150524152300
ECMEQ;AA41942855;A320;LEGR;20150524051500;20150524051300
ECJTR;AA41942805;A320;LECO;20150524105000;20150524104000
EIDLF;AA41946954;B738;LIRF;20150524082500;20150524082700
ECLQN;AA41942784;A320;DAAG;20150524051500;20150524051100
ECMEL;AA41943894;A320;GCLP;20150524102500;20150524102700
ECKHN;AA41966439;A320;LEMD;20150524205000;20150524205100
DAIRH;AA41944847;A321;EDDF;20150524051000;20150524050400
N816NW;AA41958381;A333;KJFK;20150524115000;20150524114800
HBIPV;AA41943336;A319;LSGG;20150524071500;20150524070700
```

Figure A.3: Example of ”ALL_FT_DEP” of 24th May 2015

For the arrival file, it is organized in 36 fields. Below, it is described the 36 fields, their definition, units and an example:

1. Registration Mark: ”ECLGV”
2. Aircraft ID: ”AA41968707”
3. Aircraft Type: ”F2TH”
4. Aircraft Category [”H” (Heavy), ”M” (Medium) or ”L” (Light)]: ”H”
5. Origin Airport ID: ”LEIB”
6. Scheduled Departure date and time [yyyymmddhhmmss]: ”20150524165500”
7. Real Departure date and time [yyyymmddhhmmss] (-1 if cancelled): ”20150524165800”
8. Scheduled Arrival date and time [yyyymmddhhmmss]: ”20150524174449”
9. Regulated Arrival date and time [yyyymmddhhmmss]: ”20150524174449”

10. Real Arrival date and time [yyyymmddhhmmss] (-1 if cancelled): ”20150524173459”

11. Scheduled Total Route Distance [km]: ”391”

12. Real Total Route Distance [km] (-1 if cancelled): ”345”

13. Scheduled Entry Point Latitude and Longitude ¡500km of radius[ddmmss]: ”385222N0012223E”

14. Scheduled Time Entry Point [hhmmss]: ”165500”

15. Scheduled Remaining distance from Entry Point in trajectory [km]: ”391”

16. Scheduled Distance to destination airport at Entry Point (straight line distance) [km]: ”276.456”

17. Scheduled Flight Level at Entry Point [FL]: ”0”

18. Scheduled Exit Point Latitude and Longitude ¡50km of radius[ddmmss]: ”405514N0020517E”

19. Scheduled Time Exit Point [hhmmss]: ”173022”

20. Scheduled Remaining distance from Exit Point in trajectory [km]: ”78”

21. Scheduled Distance to destination airport at Exit Point (straight line distance) [km]: ”41.9066”

22. Scheduled Flight Level at Exit Point [FL]: ”100”

23. Scheduled Mean Speed from Entry Point to Exit Point [km/h]: ”531.012”

24. New Scheduled Arrival Time considering straight line trajectory [hhmmss]: ”172130”

25. Real Entry Point Latitude and Longitude ¡500km of radius[ddmmss] (-1 if cancelled): ”385222N0012223E”

26. Real Time Entry Point [hhmmss] (-1 if cancelled): ”165800”

27. Real Remaining distance from Entry Point in trajectory [km] (-1 if cancelled): ”345”

28. Real Distance to destination airport at Entry Point (straight line distance) [km] (-1 if cancelled): ”276.456”

29. Real Flight Level at Entry Point [FL] (-1 if cancelled): ”0”

30. Real Exit Point Latitude and Longitude ¡50km of radius[ddmmss] (-1 if cancelled): ”405128N0020235E”

31. Real Time Exit Point [hhmmss] (-1 if cancelled): ”172034”

32. Real Remaining distance from Exit Point in trajectory [km] (-1 if cancelled): ”87”

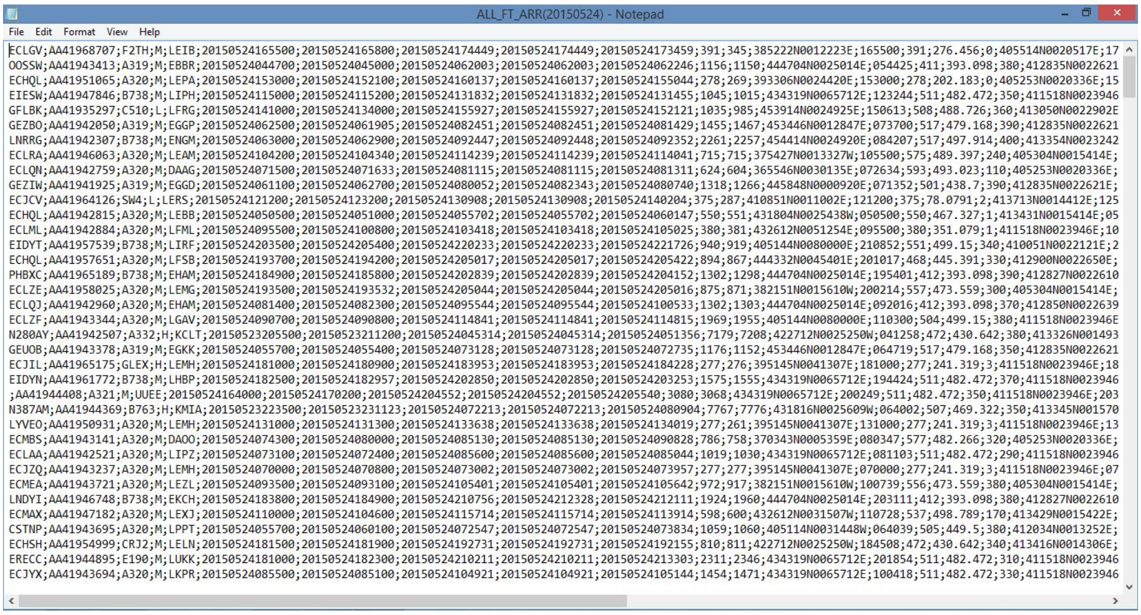
33. Real Distance to destination airport at Exit Point (straight line distance) [km] (-1 if cancelled): ”48.9765”

34. Real Flight Level at Exit Point [FL] (-1 if cancelled): ”110”

35. Real Mean Speed from Entry Point to Exit Point [km/h] (-1 if cancelled): ”685.97”

36. New Real Arrival Time considering straight line trajectory [hhmmss] (-1 if cancelled): ”171753”

In the figure A.4 it shows an example of this format.



```
ECLGV;AA41968707;F2TH;M;LEIB;20150524165500;20150524165800;20150524174449;20150524174449;20150524173459;391;345;385222N0012223E;165500;391;276.456;0;405514N0020517E;17
OOSSW;AA41943413;A319;M;EBBR;20150524044700;20150524045000;20150524062003;20150524062003;20150524062246;1156;1150;444704N0025014E;054425;411;393.098;380;412835N0022621
ECHQL;AA41951065;A320;M;LEPA;20150524153000;20150524152100;20150524160137;20150524160137;20150524155044;278;269;393306N0024420E;153000;278;202.183;0;405253N0020336E;15
EIESW;AA41947846;B738;M;LIPH;20150524115000;20150524115200;20150524131832;20150524131832;20150524131455;1045;1015;434319N0065712E;123244;511;482.472;350;411518N0023946
GFLBK;AA41935297;C510;L;LFRG;20150524141000;20150524134000;20150524155927;20150524155927;20150524152121;1035;985;453914N0024925E;150613;508;488.726;360;413050N0022902E
GEZBO;AA41942050;A319;M;EGGP;20150524062500;20150524061905;20150524082451;20150524082451;20150524081429;1455;1467;453446N0012847E;073700;517;479.168;390;412835N0022621
LNRRG;AA41942307;B738;M;ENGM;20150524063000;20150524062900;20150524092447;20150524092448;20150524092352;2261;2257;454414N0024920E;084207;517;497.914;400;413354N0023242
ECLRA;AA41946063;A320;M;LEAM;20150524104200;20150524104340;20150524114239;20150524114239;20150524114041;715;715;375427N0013327W;105500;575;489.397;240;405304N0015414E;
ECLQN;AA41942759;A320;M;DAAG;20150524071500;20150524071633;20150524081115;20150524081115;20150524081311;624;604;365546N0030135E;072634;593;493.023;110;405253N0020336E;
GEZIW;AA41941925;A319;M;EGGD;20150524061100;20150524062700;20150524080052;20150524082343;20150524080740;1318;1266;445848N0000920E;071352;501;438.7;390;412835N0022621E;
ECJCV;AA41964126;SW4;L;LERS;20150524121200;20150524123200;20150524130908;20150524130908;20150524140204;375;287;410851N0011002E;121200;375;78.0791;2;413713N0014412E;125
ECHQL;AA41942815;A320;M;LEBB;20150524050500;20150524051000;20150524055702;20150524055702;20150524060147;550;551;431804N0025438W;050500;550;467.327;1;413431N0015414E;05
ECLML;AA41942884;A320;M;LFML;20150524095500;20150524100800;20150524103418;20150524103418;20150524105025;380;381;432612N0051254E;095500;380;351.079;1;411518N0023946E;10
EIDYT;AA41957539;B738;M;LIRF;20150524203500;20150524205400;20150524220233;20150524220233;20150524221726;940;919;405144N0080000E;210852;551;499.15;340;410051N0022121E;2
ECHQL;AA41957651;A320;M;LFSB;20150524193700;20150524194200;20150524205017;20150524205017;20150524205422;894;867;444332N0045401E;201017;468;445.391;330;412900N0022650E;
PHBXC;AA41965189;B738;M;EHAM;20150524184900;20150524185800;20150524202839;20150524202839;20150524204152;1302;1298;444704N0025014E;195401;412;393.098;390;412827N0022610
ECLZE;AA41958025;A320;M;LEMG;20150524193500;20150524193532;20150524205044;20150524205044;20150524205016;875;871;382151N0015610W;200214;557;473.559;300;405304N0015414E;
ECLQJ;AA41942960;A320;M;EHAM;20150524081400;20150524082300;20150524095544;20150524095544;20150524100533;1302;1303;444704N0025014E;092016;412;393.098;370;412850N0022639
ECLZF;AA41943344;A320;M;LGAV;20150524090700;20150524090800;20150524114841;20150524114841;20150524114815;1969;1955;405144N0080000E;110300;504;499.15;380;411518N0023946E
N280AY;AA41942507;A332;H;KCLT;20150523205500;20150523211200;20150524045314;20150524045314;20150524051356;7179;7208;422712N0025250W;041258;472;430.642;380;413326N001493
GEUOB;AA41943378;A319;M;EGKK;20150524055700;20150524055400;20150524073128;20150524073128;20150524072735;1176;1152;453446N0012847E;064719;517;479.168;350;412835N0022621
ECJIL;AA41965175;GLEX;H;LEMH;20150524181000;20150524180900;20150524183953;20150524183953;20150524184228;277;276;395145N0041307E;181000;277;241.319;3;411518N0023946E;18
EIDYN;AA41961772;B738;M;LHBP;20150524182500;20150524182957;20150524202850;20150524202850;20150524203253;1575;1555;434319N0065712E;194424;511;482.472;370;411518N0023946
;AA41944408;A321;M;UUEE;20150524164000;20150524170200;20150524204552;20150524204552;20150524205540;3080;3068;434319N0065712E;200249;511;482.472;350;411518N0023946E;203
N387AM;AA41944369;B763;H;KMIA;20150523223500;20150523231123;20150524072213;20150524072213;20150524080904;7767;7776;431816N0025609W;064002;507;469.322;350;413345N001570
LYVEO;AA41950931;A320;M;LEMH;20150524131000;20150524131300;20150524133638;20150524133638;20150524134019;277;261;395145N0041307E;131000;277;241.319;3;411518N0023946E;13
ECMBS;AA41943141;A320;M;DAOO;20150524074300;20150524080000;20150524085130;20150524085130;20150524090828;786;758;370343N0005359E;080347;577;482.266;320;405253N0020336E;
ECLAA;AA41942521;A320;M;LIPZ;20150524073100;20150524072400;20150524085600;20150524085600;20150524085044;1019;1030;434319N0065712E;081103;511;482.472;290;411518N0023946
ECJZQ;AA41943237;A320;M;LEMH;20150524070000;20150524070800;20150524073002;20150524073002;20150524073957;277;277;395145N0041307E;070000;277;241.319;3;411518N0023946E;07
ECMEA;AA41943721;A320;M;LEZL;20150524093500;20150524093100;20150524105401;20150524105401;20150524105642;972;917;382151N0015610W;100739;556;473.559;380;405304N0015414E;
LNDYI;AA41946748;B738;M;EKCH;20150524183800;20150524184900;20150524210756;20150524212328;20150524212111;1924;1960;444704N0025014E;203111;412;393.098;380;412827N0022610
ECMAX;AA41947182;A320;M;LEXJ;20150524110000;20150524104600;20150524115714;20150524115714;20150524113914;598;600;432612N0031507W;110728;537;498.789;170;413429N0015422E;
CSTNP;AA41943695;A320;M;LPPT;20150524055700;20150524060100;20150524072547;20150524072547;20150524073834;1059;1060;405114N0031448W;064039;505;449.5;380;412034N0013252E;
ECHSH;AA41954999;CRJ2;M;LELN;20150524181500;20150524181900;20150524192731;20150524192731;20150524192155;810;811;422712N0025250W;184508;472;430.642;340;413416N0014306E;
ERECC;AA41944895;E190;M;LUKK;20150524181000;20150524182300;20150524210211;20150524210211;20150524213303;2311;2346;434319N0065712E;201854;511;482.472;310;411518N0023946
ECJYX;AA41943694;A320;M;LKPR;20150524085500;20150524085100;20150524104921;20150524104921;20150524105144;1454;1471;434319N0065712E;100418;511;482.472;330;411518N0023946
```

Figure A.4: Example of ”ALL_FT_ARR” of 24th May 2015

With the files’ format defined, these files are ready to be read by Matlab to start with the optimization process.

A.3. NEST download and testing

After reading the ALL_FT+ file with the AWK script, two files corresponding to departures and arrivals at LEBL during the studied day are going to be obtained thanks to the filtering process. However, the confidence of these filtered data should be ensured. To carry out a validation process, the information coming from the script will be compared with the information provided by a Eurocontrol ATM software called NEST.

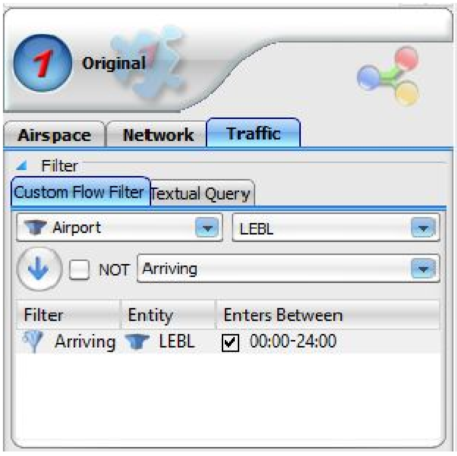
To download and install it, the instructions given in the access conditions lash in [26] should be followed.

First of all, the data corresponding to the day desired to be analysed should be downloaded. To do so, an AIRAC scenario must be downloaded, which is a file containing flight data during a determined time period, usually 1 month.

The next step is to compare the information coming from the AWK script with NEST in order to verify that the information filtered is correct. This is achieved by checking if the data from both sources match perfectly, which will mean that the AWK script works properly.

The first step is to select the AIRAC day to be studied and then apply filters if necessary. Note that the AIRAC file contains information about all the flights performed in the day selected around all the European territory. Therefore, the filters will be a must in order to limit the given information. Following that way, the left side of the NEST software is used to apply the filters desired.

In our case, select only the flights corresponding to the day studied in Barcelona airport (LEBL) and after that, filter them by arrivals and departures in order to compare them with the two output files obtained from the script. See figures A.5 and A.6.

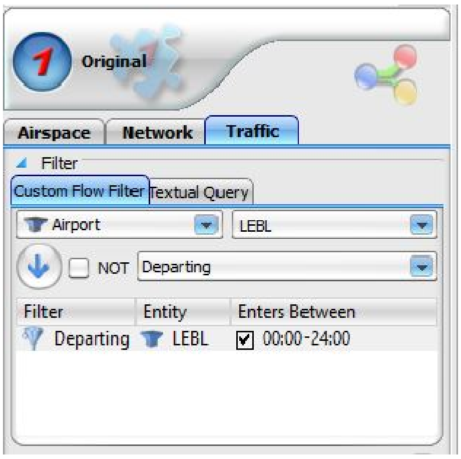


Airport | LEBL

□ NOT Arriving

Filter	Entity	Enters Between
Arriving	LEBL	☑ 00:00-24:00

Figure A.5: Filtering AIRAC data by LEBL arriving flights



Airport | LEBL

□ NOT Departing

Filter	Entity	Enters Between
Departing	LEBL	☑ 00:00-24:00

Figure A.6: Filtering AIRAC data by LEBL departing flights

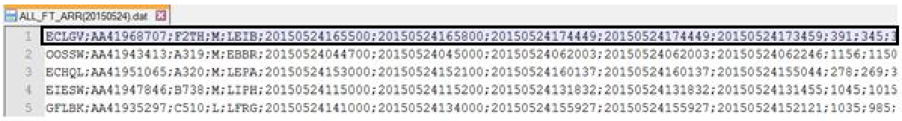
Now, the flights have to be checked. To do so, one arrival flight and one departure flight are going to be verified to ensure the entire process. At this point two tables (see tables A.1 and A.2) summarizing the information to be checked are shown for arrival and departure flights, comparing their corresponding flight information in the arrival and departure files by AWK (see figures A.7 and A.9) and NEST (see figures A.8 and A.10).

Checking the arrival flight:

Date	File	Flight ID	Scheduled Arrival Time	Origin	Destination	Validation (Ok / Wrong)
24/05/2015	Arrival	AA41968707	17:44:49	LEIB	LEBL	OK

Table A.1: Arriving flight information

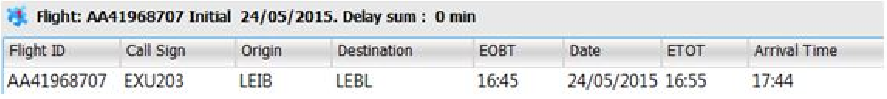
AWK script Arrival File:



```
ECLGV;AA41968707;F2TH;M;LEIB;20150524165500;20150524165800;20150524174449;20150524174449;20150524173459;391;345;
OOSSW;AA41943413;A319;M;EBBR;20150524044700;20150524045000;20150524062003;20150524062003;20150524062246;1156;1150
ECHQL;AA41951065;A320;M;LEPA;20150524153000;20150524152100;20150524160137;20150524160137;20150524155044;278;269;3
EIESW;AA41947846;B738;M;LIPH;20150524115000;20150524115200;20150524131832;20150524131832;20150524131455;1045;1015
GFLBK;AA41935297;C510;L;LFRG;20150524141000;20150524134000;20150524155927;20150524155927;20150524152121;1035;985;
```

Figure A.7: Arrival file obtained from the AWK script

NEST Arrival data:



Flight: AA41968707 Initial 24/05/2015. Delay sum : 0 min

Flight ID	Call Sign	Origin	Destination	EOBT	Date	ETOT	Arrival Time
AA41968707	EXU203	LEIB	LEBL	16:45	24/05/2015	16:55	17:44

Figure A.8: Arrival information obtained from NEST

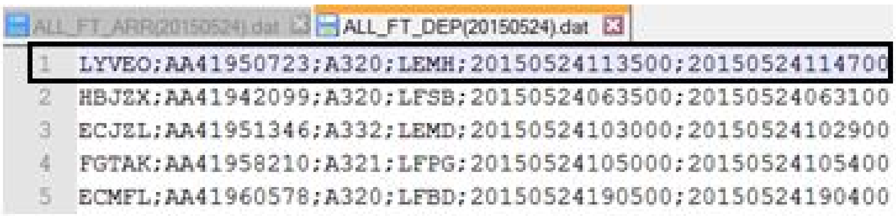
All the information specified in table A.1 (Flight ID, Scheduled Arrival Time. . .) agrees in both sources, so the statement that all the filtered arrival data coming from the AWK script is correct can be validated and consequently that the script code works properly.

Checking the departure flight:

Date	File	Flight ID	Scheduled Arrival Time	Origin	Destination	Validation (Ok / Wrong)
24/05/2015	Departure	AA41950723	11:35:00	LEBL	LEMH	OK

Table A.2: Departing flight information

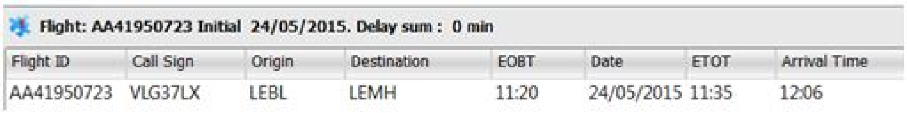
AWK script Departure file:



```
LYVEO;AA41950723;A320;LEMH;20150524113500;20150524114700
HBJZX;AA41942099;A320;LFSB;20150524063500;20150524063100
ECJZL;AA41951346;A332;LEMD;20150524103000;20150524102900
FGTAK;AA41958210;A321;LFPG;20150524105000;20150524105400
ECMFL;AA41960578;A320;LFBD;20150524190500;20150524190400
```

Figure A.9: Departure file obtained from the AWK script

NEST Departure data:



Flight: AA41950723 Initial 24/05/2015. Delay sum : 0 min

Flight ID	Call Sign	Origin	Destination	EOBT	Date	ETOT	Arrival Time
AA41950723	VLG37LX	LEBL	LEMH	11:20	24/05/2015	11:35	12:06

Figure A.10: Departure information obtained from NEST

In the same way, it is possible to state that the departure file is also generated correctly because the data matches perfectly between both sources.

APPENDIX B. MATLAB MAIN CODE AND OUTPUT FILES FORMAT

When expecting to present a final degree thesis in an engineering career, some programming software are used frequently, in this case, the main program used to code the AHP model is Matlab.

The Matlab programming software is available for all students in UPC, the only requirement is to be connected to a UPC IP Address.

B.1. Complete commented Matlab main code of Tactical AHP

Here the complete Matlab main code is presented below (standard parameters' values are considered):

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%% Final Degree Thesis %%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%% Matlab Code for AHP Tactical Phase Optimization %%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%% Autor: Danlin Zheng  %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%% Technical University of Catalonia (UPC) %%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%% July-2015 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%%%%%%%%%%%%%%%%%%%%%%%% BEGIN MAIN CODE %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

clc;
clear all;
close all;

%% Load data file

[DoubleFlightData,StringFlightData]= ReadData(’ALL_FT_ARR(20150711).dat’,
’Delay_table_20150711_aar_20.dat’);

%% Capacity Parameters

AAR = 20; %Airport Acceptance Rate (each hour)
S=180;% capacity is given every S seconds (1 minute = 60 seconds)
C= floor(AAR/3600*S); % capacity: C aircraft every S seconds
(how many aircraft the airport can accept each minute)

%% Remove cancelled flights for tactical phase

z=1;
while  z<=length(DoubleFlightData.RealDepTime)
```

```
     if (DoubleFlightData.RealDepTime(z)== -1)

        StringFlightData(z) = [];
        DoubleFlightData.SchDepTime(z) = [];
        DoubleFlightData.RealDepTime(z) = [];
        DoubleFlightData.SchArrTime(z) = [];
        DoubleFlightData.RegArrTime(z) = [];
        DoubleFlightData.RealArrTime(z) = [];
        DoubleFlightData.SchTotalDist(z) = [];
        DoubleFlightData.RealTotalDist(z) = [];
        DoubleFlightData.SchTimeEntry(z) = [];
        DoubleFlightData.SchRemainDistTrajEntry(z) = [];
        DoubleFlightData.SchDistanceStraightEntry(z) = [];
        DoubleFlightData.SchFLEntry(z) = [];
        DoubleFlightData.SchTimeExit(z) = [];
        DoubleFlightData.SchRemainDistTrajExit(z) = [];
        DoubleFlightData.SchDistanceStraightExit(z) = [];
        DoubleFlightData.SchFLExit(z) = [];
        DoubleFlightData.SchMeanspeed(z) = [];
        DoubleFlightData.SchNExitT(z) = [];
        DoubleFlightData.RealTimeEntry(z) = [];
        DoubleFlightData.RealRemainDistTrajEntry(z) = [];
        DoubleFlightData.RealDistanceStraightEntry(z) = [];
        DoubleFlightData.RealFLEntry(z) = [];
        DoubleFlightData.RealTimeExit(z) = [];
        DoubleFlightData.RealRemainDistTrajExit(z) = [];
        DoubleFlightData.RealDistanceStraightExit(z) = [];
        DoubleFlightData.RealFLExit(z) = [];
        DoubleFlightData.RealMeanspeed(z) = [];
        DoubleFlightData.RealNExitT(z) = [];
        DoubleFlightData.DelaysPre(z) = [];

        z=1;
    else
        z=z+1;
    end
end

%% Converting needed times to seconds

for i=1:length(DoubleFlightData.SchArrTime)

    SchArrTime_str = num2str(DoubleFlightData.SchArrTime(i));
    %Schedule Arrival Time (convert double to string)

    if(length(SchArrTime_str(1,:))>6)
            SchArrTime_str = strcat(SchArrTime_str(1,9),SchArrTime_str(1,10),SchArrTime_str(1,11),
            SchArrTime_str(1,12),SchArrTime_str(1,13),SchArrTime_str(1,14));
    end
    SchArrTime_dbl = str2double(SchArrTime_str);
     SchETA(i) = SchArrTime_dbl;
```

```
end

[SchETA_seconds] = minutes2seconds(SchETA);
[SchTimeEntry_seconds] = minutes2seconds(DoubleFlightData.SchTimeEntry);
[SchNExitT_seconds] = minutes2seconds(DoubleFlightData.SchNExitT);
[SchTimeExit_seconds] = minutes2seconds(DoubleFlightData.SchTimeExit);
SchDistance=DoubleFlightData.SchTotalDist;

%% Modifying Exit Point Times (Later New Scheduled Time to be as Scheduled Exit Times)

for i=1:length(SchNExitT_seconds)
    if(SchNExitT_seconds(i)>SchTimeExit_seconds(i))
        SchNExitT_seconds(i)=SchTimeExit_seconds(i);
    end
end

%% Create Noise for Pretactical Delays

for p=1:length(DoubleFlightData.DelaysPre)

        neg=round(879/2115,4);
        x=rand(1,1);
        if (x<=neg)
            Z=gamrnd(1.220,5.533);
            NoisePre(p)=round(Z)*-1;
        else
            Z=gamrnd(1.195,8.408);
            NoisePre(p)=round(Z);
        end

end


%% Sum of the delay and noise from pretactical optimization

for i=1:length(DoubleFlightData.DelaysPre)

for n=1:length(DoubleFlightData.DelaysPre)
tf=strcmp(char(StringFlightData(i).FlightIDDelay),char(StringFlightData(n).FlightID));
if (tf==true)
    SchETA_seconds(n)=SchETA_seconds(n)+DoubleFlightData.DelaysPre(i)*60+NoisePre(i)*60;
    SchTimeEntry_seconds(n) = SchTimeEntry_seconds(n)+ DoubleFlightData.DelaysPre(i)*60+NoisePre(i)*60;
    SchNExitT_seconds(n) = SchNExitT_seconds(n)+ DoubleFlightData.DelaysPre(i)*60+NoisePre(i)*60;
    SchTimeExit_seconds(n) = SchTimeExit_seconds(n)+ DoubleFlightData.DelaysPre(i)*60+NoisePre(i)*60;
end

end

end
```

```
%% Modifying Scheduled Entry Point, Exit Point and New Exit Point Times
(Later times than Scheduled ETA to -24h)

for w=1:length(SchETA_seconds)

    if (SchTimeEntry_seconds(w)>SchETA_seconds(w))
        SchTimeEntry_seconds(w)= SchTimeEntry_seconds(w)-86400;
    end
    if (SchNExitT_seconds(w)>SchETA_seconds(w))
        SchNExitT_seconds(w)= SchNExitT_seconds(w)-86400;
    end
    if (SchTimeExit_seconds(w)>SchETA_seconds(w))
        SchTimeExit_seconds(w)= SchTimeExit_seconds(w)-86400;
    end

end



%% RBS using Scheduled Exit Time

interval_RBS=60;
time_arrival_RBS=(min(SchNExitT_seconds):interval_RBS:max(SchNExitT_seconds)+86400);
[FlightIDRBS,RBSDelay,RBSRealExitTime,SchNExitTRBS,SchExitTRBS,CostRBS] =
RBS(StringFlightData,SchNExitT_seconds,SchTimeExit_seconds, time_arrival_RBS,S,SchDistance);


%% Order from earliest to latest flights respect Entry Point Time

[SchTimeEntry_seconds_ordered,index] = sort(SchTimeEntry_seconds); %Order regarding Entry Point Time

%Ordering FlightID, Scheduled New Exit Time, Scheduled Exit Time, Scheduled
%Distance and Category.


for i=1:length(index)

    FlightID_ordered(i) = cellstr(StringFlightData(index(i)).FlightID);
    SchNExitT_seconds_ordered(i) = SchNExitT_seconds(index(i));
    SchTimeExit_seconds_ordered(i) = SchTimeExit_seconds(index(i));
    SchDistance_ordered(i)=SchDistance(index(i));
    Category_ordered(i) = cellstr(StringFlightData(index(i)).Category);

end

Category=Category_ordered’;
FlightID=FlightID_ordered’;
SchTimeEntry_seconds=SchTimeEntry_seconds_ordered;
SchNExitT_seconds=SchNExitT_seconds_ordered’;
SchTimeExit_seconds=SchTimeExit_seconds_ordered’;
SchDistance=SchDistance_ordered’;
```

```
clear Category_ordered; clear FlightID_ordered; clear SchTimeEntry_seconds_ordered;
clear SchNExitT_seconds_ordered; clear SchTimeExit_seconds_ordered; clear SchDistance_ordered;
clear index;


%% Filtering to check the simulation

%Only filter if needed, by default this filter will save all flights of the
%day.
[FlightID,SchTimeEntry,SchNExitT,SchTimeExit,Category,SchDistance] =
filter_flights(FlightID,SchTimeEntry_seconds,SchNExitT_seconds,SchTimeExit_seconds,Category,SchDistance);

%% Optimization Loop

%First loop parameters
Result=[];
Categorynew=[];
SchDistancenew=[];
SchNExitTnew=[];
SchTimeExitnew=[];
FlightIDnew=[];
FlightIDFinal=[];
CategoryFinal=[];
CostFinal=[];
SchDistanceFinal=[];
DelayFinal=[];
HoldingFinal=[];
RealExitTimeFinal=[];
SchEntryFinal=[];
SchNExitTimeFinal=[];
SchTimeEntrynew=[];
SchTimeExitFinal=[];
InvalidSlots=[];
count=1;
for i=1:length(SchTimeEntry)
%% Updating parameters

actualtime=SchTimeEntry(i);

if (i<=length(SchNExitT))
    Categorynew=[Categorynew;Category(i)];
    SchDistancenew=[SchDistancenew;SchDistance(i)];
    SchNExitTnew=[SchNExitTnew;SchNExitT(i)];
    SchTimeExitnew=[SchTimeExitnew;SchTimeExit(i)];
    FlightIDnew=[FlightIDnew;FlightID(i)];
    SchTimeEntrynew=[SchTimeEntrynew; SchTimeEntry(i)];
    z=1;
    while  z<=length(SchNExitTnew)
```

```
        if (actualtime>=SchNExitTnew(z))
            FlightIDFinal=[FlightIDFinal;FlightIDnew(z)];
            CostFinal=[CostFinal;CostF(z)];
            CategoryFinal=[CategoryFinal;Categorynew(z)];
            SchDistanceFinal=[SchDistanceFinal;SchDistancenew(z)];
            DelayFinal=[DelayFinal; Delay(z)];
            HoldingFinal=[HoldingFinal;Holding(z)];
            RealExitTimeFinal=[RealExitTimeFinal;RealExitTime(z)];
            SchEntryFinal=[SchEntryFinal; SchTimeEntrynew(z)];
            SchNExitTimeFinal=[SchNExitTimeFinal;SchNExitTnew(z)];
            SchTimeExitFinal=[SchTimeExitFinal;SchTimeExitnew(z)];
            InvalidSlots(count)=Delay(z)*60+SchTimeExitnew(z);% in seconds
            Delay(z)=[];
            Holding(z)=[];
            CostF(z)=[];
            RealExitTime(z)=[];
            FlightIDnew(z) = [];
            SchNExitTnew(z) = [];
            SchTimeExitnew(z) = [];
            SchTimeEntrynew(z)=[];
            Categorynew(z)=[];
            SchDistancenew(z)=[];
            z=1;
            count=count+1;
        else
            z=z+1;
        end
    end

end

%% Problem parameters


%Seconds of the time interval (1 min in this case)
interval = 60;
start=min(SchNExitTnew);
finish=max(SchTimeExitnew)+1800;
time_arrival=(start:interval:finish);
ExitHold=SchTimeExitnew+1800;

cost_parameter = ones(length(SchNExitTnew));

% Non unitary cost parameter: fuel consumtion at holding speed

for c=1:length(SchDistancenew)

    if SchDistancenew(c)<=1900 %km

        tfl=strcmp(char(Categorynew(c)),'L');
        if (tfl==true)
            cost_parameter(c)=0.02408*388.92*0.33*168; %Kerosene

        end

        tfm=strcmp(char(Categorynew(c)),’M’);
        if (tfm==true)

            cost_parameter(c)=0.02408*438.92*0.33*168;

        end

        tfh=strcmp(char(Categorynew(c)),’H’);
        if (tfh==true)

            cost_parameter(c)=0.02408*490.78*0.33*168;

        end

    end

    if SchDistancenew(c)>1900 &&  SchDistancenew(c)<=5600

        tfl=strcmp(char(Categorynew(c)),’L’);
        if (tfl==true)

            cost_parameter(c)=0.02772*388.92*0.33*260;

        end

         tfm=strcmp(char(Categorynew(c)),’M’);
        if (tfm==true)

            cost_parameter(c)=0.02772*438.92*0.33*260;

        end

        tfh=strcmp(char(Categorynew(c)),’H’);
        if (tfh==true)

            cost_parameter(c)=0.02772*490.78*0.33*260;

        end

    end

    if SchDistancenew(c)>5600

        tfl=strcmp(char(Categorynew(c)),’L’);
        if (tfl==true)
```

```
            cost_parameter(c)=0.02948*388.92*0.33*346;

        end

        tfm=strcmp(char(Categorynew(c)),’M’);
        if (tfm==true)

            cost_parameter(c)=0.02948*438.92*0.33*346;

        end

        tfh=strcmp(char(Categorynew(c)),’H’);
        if (tfh==true)

            cost_parameter(c)=0.02948*490.78*0.33*346;

        end

    end


end


% Linearity parameter
epsilon = 0; % cost factor: 0-> linear cost, 1-> cuadratic cost


%% Create the constraint matrices

%Initializing Parameters
roweq=length(SchNExitTnew);
rowineq=size(time_arrival,2)-S/60+1+length(SchNExitTnew);
column=size(time_arrival,2)*length(SchNExitTnew);
eqcondition=zeros(roweq,column);
ineqcondition=zeros(rowineq,column);
slots = 0;
f=[];
Cost=[];
for n = 1:length(SchNExitTnew)

    for j = 1:size(time_arrival,2)

        eqcondition(n,j+slots) = 1;             % Unicity of arrival constraint

        if j<=size(time_arrival,2)-S/60+1

            for r=1:(S/60)

                ineqcondition(j,j+slots+r-1) = 1;   % Capacity constraints
                if (time_arrival(j+r-1)-SchNExitTnew(n)<0)

                    f1=0;

                else

                    f1=time_arrival(j+r-1)-SchNExitTnew(n);

                end

                f(j+slots+r-1) = f1; %Seconds
                D=f1+SchNExitTnew(n)-SchTimeExitnew(n);%in seconds
                Cost(j+slots+r-1)=(cost_parameter(n)*(D/3600))^(1+epsilon); %Euros
             end

        end

        if (i>1)
            b=1;
            found=false;
            while (b<=length(InvalidSlots) && found==false)

                if (time_arrival(j)<=(InvalidSlots(b)-S) || time_arrival(j)>=(InvalidSlots(b)+S))

                    %Nothing to do

                else

                    found=true; % True if this slots is not available due to capacity constraints.

                end
                b=b+1;
            end
            if SchNExitTnew(n)<=(time_arrival(j)) && ExitHold(n)>=time_arrival(j) && found==false

                     %Nothing to do

            else

                    ineqcondition(n+size(time_arrival,2)-S/60+1,j+slots) = 1;
                    % Avoid arriving before minimum exit time and 30 min after scheduled exit time.

             end
        else
            if SchNExitTnew(n)<=(time_arrival(j)) && ExitHold(n)>=time_arrival(j)

                     %Nothing to do

            else

                    ineqcondition(n+size(time_arrival,2)-S/60+1,j+slots) = 1;
```

```
                    % Avoid arriving before minimum exit time and 30 min after scheduled exit time.

            end

        end

    end

    slots = slots + size(time_arrival,2) ;
end

%Convert "f" from seconds to minutes

f=f/60;
%% Gurobi needed parameters

neq = size(eqcondition,1);
nineq = size(ineqcondition,1);
nvar = size(eqcondition,2);
Aeq = eqcondition; % Represents de slots that are available for each aircraft
Aineq = ineqcondition;
bineq=[];
for v=1:nineq
    if v<=nineq-neq

        bineq(v,1) = 1;% Capacity constraints

    else

        bineq(v,1) = 0;% time and holding constraints

    end
end
beq = ones(neq,1);          % Unicity of arrival constraint (the aircraft arrives only once)

%% solve with Gurobi (fastest)
tic
A=[Aeq;Aineq];
Sens=[];

for s=1:size(bineq,1)+size(beq,1)

    if s<=neq

        Sens=[Sens,’=’];
    else
        if s<=nineq

           Sens=[Sens,’<’];

        else
```

```
            Sens=[Sens,'='];

        end
    end

end

b=[beq;bineq];
model.A=sparse(A);
model.obj=Cost;
model.rhs=b;
model.vtype='B';
model.sense=Sens;
result = gurobi(model);
X=result.x;
cost=result.objval;
toc

%% Post-process of results
Holding=[];
CostF=[];
Res=[];
Res=find(X);
for h=1:size(Res,1)

    Holding(h)=f(Res(h));
    CostF(h)=Cost(Res(h));

end
Holding=Holding';
CostF=CostF';
HoldingTotal=sum(Holding); %In minutes
CostTotal=sum(CostF);
Holding=Holding*60;%Minutes to seconds
Delay=[];
clear RealExitTime;
for q=1:length(Holding)

        Delay(q)=((Holding(q)+SchNExitTnew(q))-SchTimeExitnew(q));%in seconds
        RealExitTime(q)=cellstr(sec2hms(Delay(q)+SchTimeExitnew(q)));%in hhmmss
end

Delay=Delay/60; %In minutes
Holding=Holding/60;%Seconds to minutes

if (i==length(SchTimeEntry))
    for p=1:length(Delay)
        FlightIDFinal=[FlightIDFinal; FlightIDnew(p)];
        HoldingFinal=[ HoldingFinal;Holding(p)];
        CostFinal=[CostFinal;CostF(p)];
        CategoryFinal=[CategoryFinal;Categorynew(p)];
        SchDistanceFinal=[SchDistanceFinal;SchDistancenew(p)];
        DelayFinal=[DelayFinal; Delay(p)];
        RealExitTimeFinal=[RealExitTimeFinal;RealExitTime(p)];
        SchEntryFinal=[SchEntryFinal; SchTimeEntrynew(p)];
        SchNExitTimeFinal=[SchNExitTimeFinal;SchNExitTnew(p)];
        SchTimeExitFinal=[SchTimeExitFinal;SchTimeExitnew(p)];
    end

end


Delay=Delay’;
DelayTotal=sum(Delay); %In minutes


Result=[];
Result=struct(’Flight_ID’,char(FlightIDnew),’Category’,char(Categorynew),’SchTotalDistance’,
SchDistancenew,’Holding’,Holding,’Total_Holding’,HoldingTotal,’Delay’,Delay,’RealExitTime’,
char(RealExitTime),’Cost’,CostF,’Total_Delay’,DelayTotal,’Total_Cost’,cost,’TotalCost’,CostTotal);
Results(i)=Result;




end

%Optimization loop end

%% Results Representation
s=0;
for i=1:length(Results)


    for n=1:length(Results(i).Holding)

        if (n==length(Results(i).Holding))

            ID=cellstr(Results(i).Flight_ID);
            FlightID(s+n)=ID(n);
            Cat=cellstr(Results(i).Category);
            Category(s+n)=Cat(n);
            SchDistanceF(s+n)=Results(i).SchTotalDistance(n);
            HoldingF(s+n)=Results(i).Holding(n);
            DelayF(s+n)=round(Results(i).Delay(n),2);
            CostF(s+n)=round(Results(i).Cost(n),2);
            RealTime=cellstr(Results(i).RealExitTime);
            RealExitTimeF(s+n)=RealTime(n);
            Total_Holding(s+n)=Results(i).Total_Holding;
            Total_Delay(s+n)=round(Results(i).Total_Delay,2);
            Total_Cost(s+n)=round(Results(i).Total_Cost,2);
            TotalCost(s+n)=round(Results(i).TotalCost,2);

        else
            ID=cellstr(Results(i).Flight_ID);
            FlightID(s+n)=ID(n);
            Cat=cellstr(Results(i).Category);
            Category(s+n)=Cat(n);
            SchDistanceF(s+n)=Results(i).SchTotalDistance(n);
            HoldingF(s+n)=Results(i).Holding(n);
            DelayF(s+n)=Results(i).Delay(n);
            CostF(s+n)=Results(i).Cost(n);
            RealTime=cellstr(Results(i).RealExitTime);
            RealExitTimeF(s+n)=RealTime(n);
            Total_Holding(s+n)=0;
            Total_Delay(s+n)=0;
            Total_Cost(s+n)=0;
            TotalCost(s+n)=0;

        end

    end

    s=s+length(Results(i).Holding);


end

SchDistanceF=SchDistanceF';
Total_Holding=Total_Holding';
Total_Delay=Total_Delay';
Total_Cost=Total_Cost';
TotalCost=TotalCost';
DelayF=DelayF';
CostF=round(CostF,2);
if size(CostF,2)>1
    CostF=CostF';
end
HoldingF=HoldingF';
RealExitTimeF=RealExitTimeF';
T = table (FlightID, Category, SchDistanceF,HoldingF, Total_Holding,DelayF,RealExitTimeF,CostF,
Total_Delay,Total_Cost,TotalCost);

writetable(T,'Results.dat','Delimiter','\t', 'WriteRowNames', true); %% Results Table

% Order Respect new Exit Point Time

[SchNExitTimeFinalordered,index] = sort(SchNExitTimeFinal);

for i=1:length(index)
    CategoryFinalordered(i)=CategoryFinal(index(i));
    SchDistanceFinalordered(i)=SchDistanceFinal(index(i));
    HoldingFinalordered(i)=HoldingFinal(index(i));
    CostFinalordered(i)=CostFinal(index(i));
    DelayFinalordered(i)=DelayFinal(index(i));
    RealExitTimeFinalordered(i)=RealExitTimeFinal(index(i));
    FlightIDFinalordered(i)=FlightIDFinal(index(i));
    SchEntryFinalordered(i)=SchEntryFinal(index(i));
    SchTimeExitFinalordered(i)=SchTimeExitFinal(index(i));
end
HoldingFinal=HoldingFinalordered’;
DelayFinal=DelayFinalordered’;
CostFinal=CostFinalordered’;
CostFinal=round(CostFinal,2);
RealExitTimeFinal=RealExitTimeFinalordered’;
FlightIDFinal=FlightIDFinalordered’;
CategoryFinal=CategoryFinalordered’;
SchDistanceFinal=SchDistanceFinalordered’;

for l=1:length(DelayFinal)

    SchEntryFinalhms(l)=cellstr(sec2hms(SchEntryFinalordered(l)));
    SchNExitTimeFinalhms(l)=cellstr(sec2hms(SchNExitTimeFinalordered(l)));
    SchTimeExitFinalhms(l)=cellstr(sec2hms(SchTimeExitFinalordered(l)));

end

SchEntryFinalhms=SchEntryFinalhms’;
SchNExitTimeFinalhms=SchNExitTimeFinalhms’;
SchTimeExitFinalhms=SchTimeExitFinalhms’;


T2=table(FlightIDFinal,CategoryFinal,SchDistanceFinal,SchEntryFinalhms,SchNExitTimeFinalhms,
SchTimeExitFinalhms,HoldingFinal,DelayFinal,CostFinal,RealExitTimeFinal,FlightIDRBS,SchNExitTRBS,
SchExitTRBS,RBSDelay,CostRBS,RBSRealExitTime);

writetable(T2,’ResultsFinal.dat’,’Delimiter’,’\t’, ’WriteRowNames’, true); %Final Results table

clear SchNExitTimeFinalordered;clear CategoryFinalordered; clear SchDistanceFinalordered;
clear HoldingFinalordered; clear CostFinalordered; clear DelayFinalordered;
clear RealExitTimeFinalordered; clear FlightIDFinalordered; clear SchEntryFinalordered;
clear SchTimeExitFinalordered; clear index;


%%%%%%%%%%%%%%%%%%%%%%%% END MAIN CODE %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

B.2. Matlab output files format

When the Matlab code is already created and operative, it is time to execute it and also generate the output files that will be used when analysing the results.

After the whole simulation process, two files with all the information desired are obtained. In there, it contains information about all the delays and other parameters assigned to the aircraft arriving at LEBL airport during a day selected. In the example the date considered is 11-07-2015. The following and final step is to describe how the output files are structured and the data they contain.

The Table ”Results” is organized in 11 fields. Below, the 11 fields, their definition, units and an example are described:

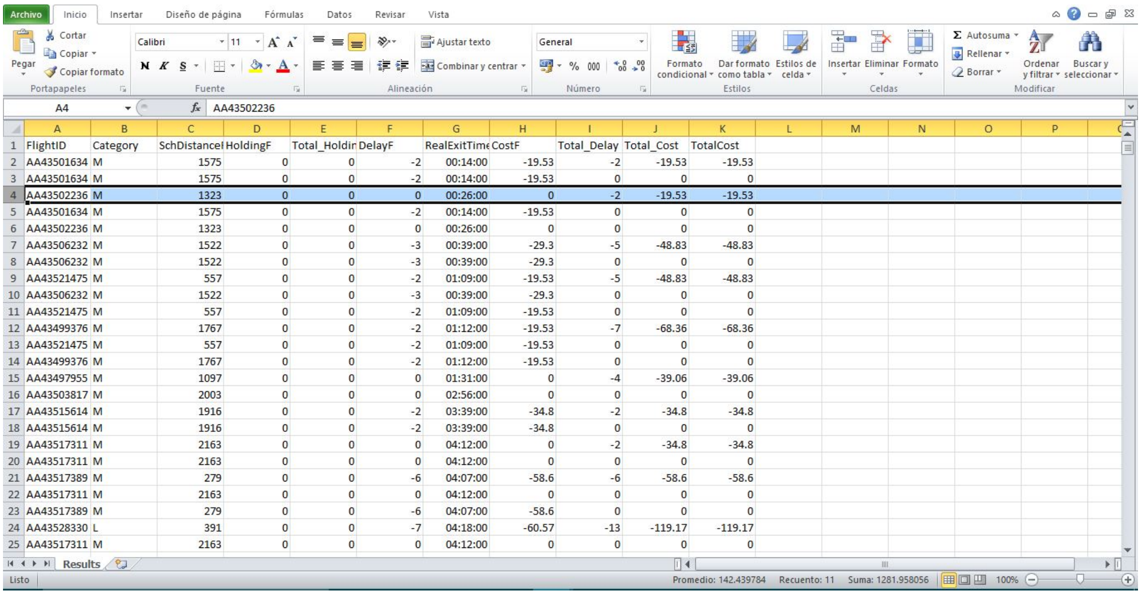
1. Aircraft ID: ”AA43502236”
2. Aircraft Category [”H” (Heavy), ”M” (Medium) or ”L” (Light)]: ”M”
3. Scheduled Total Trajectory Distance [km]: ”1323”
4. Holding Assigned [minutes]: ”0”
5. Total Holding of the iteration [minutes] (only fulfilled in the column of the last flight of that iteration): ”0”
6. Delay Assigned [minutes]: ”0”
7. Real Exit Time [hhmmss]: ”00:26:00”
8. Cost Assigned [€]: ”0”
9. Total Delay of the iteration [minutes] (only fulfilled in the column of the last flight of that iteration): ”-2”
10. Total Cost of the iteration [€] (only fulfilled in the column of the last flight of that iteration) From Gurobi Optimizer: ”-19.53”
11. Total Cost of the iteration [€] (only fulfilled in the column of the last flight of that iteration) From AHP manually: ”-19.53”

The last two columns of Total Cost are useful to check if the Cost is properly calculated. These two parameters should have the same values.

In the figure B.1 it shows an example of this format.

The Table ”ResultsFinal” is organized in 16 fields. Below, the 16 fields, their definition, units and an example are described:

1. Aircraft ID: ”AA43502236”
2. Aircraft Category [”H” (Heavy), ”M” (Medium) or ”L” (Light)]: ”M”
3. Scheduled Total Trajectory Distance [km]: ”1323”



A4 AA43502236

	A	B	C	D	E	F	G	H	I	J	K
1	FlightID	Category	SchDistanceI	HoldingF	Total_Holdir	DelayF	RealExitTime	CostF	Total_Delay	Total_Cost	TotalCost
2	AA43501634	M	1575	0	0	-2	00:14:00	-19.53	-2	-19.53	-19.53
3	AA43501634	M	1575	0	0	-2	00:14:00	-19.53	0	0	0
4	AA43502236	M	1323	0	0	0	00:26:00	0	-2	-19.53	-19.53
5	AA43501634	M	1575	0	0	-2	00:14:00	-19.53	0	0	0
6	AA43502236	M	1323	0	0	0	00:26:00	0	0	0	0
7	AA43506232	M	1522	0	0	-3	00:39:00	-29.3	-5	-48.83	-48.83
8	AA43506232	M	1522	0	0	-3	00:39:00	-29.3	0	0	0
9	AA43521475	M	557	0	0	-2	01:09:00	-19.53	-5	-48.83	-48.83
10	AA43506232	M	1522	0	0	-3	00:39:00	-29.3	0	0	0
11	AA43521475	M	557	0	0	-2	01:09:00	-19.53	0	0	0
12	AA43499376	M	1767	0	0	-2	01:12:00	-19.53	-7	-68.36	-68.36
13	AA43521475	M	557	0	0	-2	01:09:00	-19.53	0	0	0
14	AA43499376	M	1767	0	0	-2	01:12:00	-19.53	0	0	0
15	AA43497955	M	1097	0	0	0	01:31:00	0	-4	-39.06	-39.06
16	AA43503817	M	2003	0	0	0	02:56:00	0	0	0	0
17	AA43515614	M	1916	0	0	-2	03:39:00	-34.8	-2	-34.8	-34.8
18	AA43515614	M	1916	0	0	-2	03:39:00	-34.8	0	0	0
19	AA43517311	M	2163	0	0	0	04:12:00	0	-2	-34.8	-34.8
20	AA43517311	M	2163	0	0	0	04:12:00	0	0	0	0
21	AA43517389	M	279	0	0	-6	04:07:00	-58.6	-6	-58.6	-58.6
22	AA43517311	M	2163	0	0	0	04:12:00	0	0	0	0
23	AA43517389	M	279	0	0	-6	04:07:00	-58.6	0	0	0
24	AA43528330	L	391	0	0	-7	04:18:00	-60.57	-13	-119.17	-119.17
25	AA43517311	M	2163	0	0	0	04:12:00	0	0	0	0

Promedio: 142.439784 Recuento: 11 Suma: 1281.958056

Figure B.1: Example of Results Table of 11th July 2015

4. Scheduled Entry Point Time [hhmmss]:"-01:50:00"

5. New Exit Point Time [hhmmss]: "00:26:00"

6. Scheduled Exit Point Time [hhmmss]: "00:26:00"

7. Final Holding Assigned [minutes] From AHP: "0"

8. Final Delay Assigned [minutes] From AHP: "0"

9. Final Cost [€]: "0"

10. Real Exit Time [hhmmss] From AHP: "00:26:00"

11. Aircraft ID From RBS: "AA43502236"

12. New Exit Point Time [hhmmss] From RBS: "00:26:00"

13. Scheduled Exit Point Time [hhmmss] From RBS: "00:26:00"

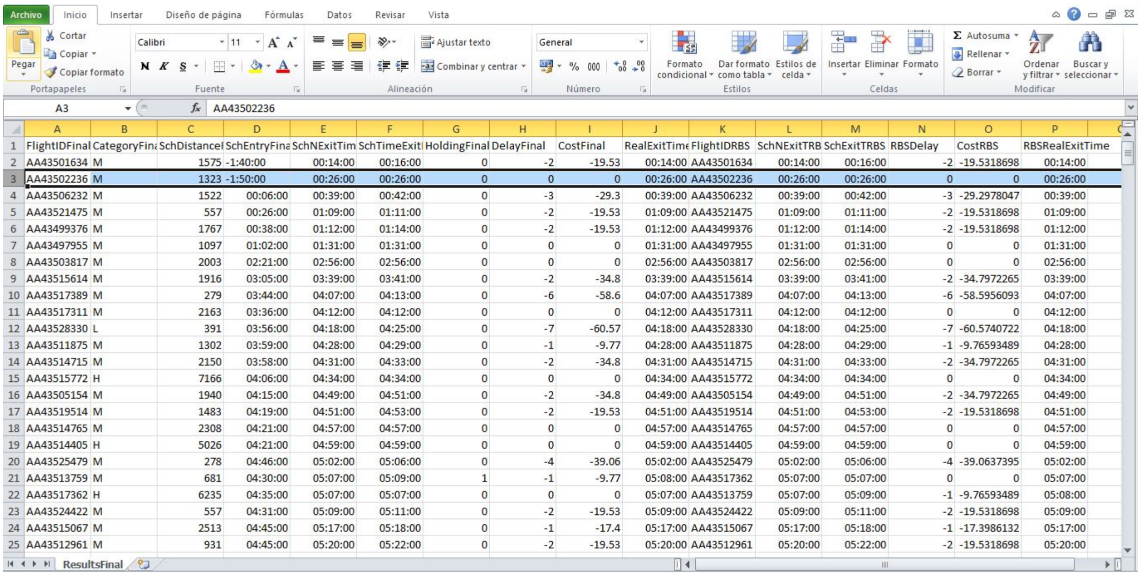
14. Final Delay Assigned [minutes] From RBS: "0"

15. Final Cost Assigned [€] From RBS: "0"

16. Real Exit Time [hhmmss] From RBS: "00:26:00"

In the figure B.2 it shows an example of this format.

With the files' format defined, these files are ready to be analysed in the corresponding chapter.



A3 AA43502236

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P
1	FlightIDFinal	CategoryFina	SchDistanceI	SchEntryFina	SchNExitTim	SchTimeExitI	HoldingFinal	DelayFinal	CostFinal	RealExitTime	FlightIDRBS	SchNExitTRB	SchExitTRBS	RBSDelay	CostRBS	RBSRealExitTime
2	AA43501634	M	1575	-1:40:00	00:14:00	00:16:00	0	-2	-19.53	00:14:00	AA43501634	00:14:00	00:16:00	-2	-19.5318698	00:14:00
3	AA43502236	M	1323	-1:50:00	00:26:00	00:26:00	0	0	0	00:26:00	AA43502236	00:26:00	00:26:00	0	0	00:26:00
4	AA43506232	M	1522	00:06:00	00:39:00	00:42:00	0	-3	-29.3	00:39:00	AA43506232	00:39:00	00:42:00	-3	-29.2978047	00:39:00
5	AA43521475	M	557	00:26:00	01:09:00	01:11:00	0	-2	-19.53	01:09:00	AA43521475	01:09:00	01:11:00	-2	-19.5318698	01:09:00
6	AA43499376	M	1767	00:38:00	01:12:00	01:14:00	0	-2	-19.53	01:12:00	AA43499376	01:12:00	01:14:00	-2	-19.5318698	01:12:00
7	AA43497955	M	1097	01:02:00	01:31:00	01:31:00	0	0	0	01:31:00	AA43497955	01:31:00	01:31:00	0	0	01:31:00
8	AA43503817	M	2003	02:21:00	02:56:00	02:56:00	0	0	0	02:56:00	AA43503817	02:56:00	02:56:00	0	0	02:56:00
9	AA43515614	M	1916	03:05:00	03:39:00	03:41:00	0	-2	-34.8	03:39:00	AA43515614	03:39:00	03:41:00	-2	-34.7972265	03:39:00
10	AA43517389	M	279	03:44:00	04:07:00	04:13:00	0	-6	-58.6	04:07:00	AA43517389	04:07:00	04:13:00	-6	-58.5956093	04:07:00
11	AA43517311	M	2163	03:36:00	04:12:00	04:12:00	0	0	0	04:12:00	AA43517311	04:12:00	04:12:00	0	0	04:12:00
12	AA43528330	L	391	03:56:00	04:18:00	04:25:00	0	-7	-60.57	04:18:00	AA43528330	04:18:00	04:25:00	-7	-60.5740722	04:18:00
13	AA43511875	M	1302	03:59:00	04:28:00	04:29:00	0	-1	-9.77	04:28:00	AA43511875	04:28:00	04:29:00	-1	-9.76593489	04:28:00
14	AA43514715	M	2150	03:58:00	04:31:00	04:33:00	0	-2	-34.8	04:31:00	AA43514715	04:31:00	04:33:00	-2	-34.7972265	04:31:00
15	AA43515772	H	7166	04:06:00	04:34:00	04:34:00	0	0	0	04:34:00	AA43515772	04:34:00	04:34:00	0	0	04:34:00
16	AA43505154	M	1940	04:15:00	04:49:00	04:51:00	0	-2	-34.8	04:49:00	AA43505154	04:49:00	04:51:00	-2	-34.7972265	04:49:00
17	AA43519514	M	1483	04:19:00	04:51:00	04:53:00	0	-2	-19.53	04:51:00	AA43519514	04:51:00	04:53:00	-2	-19.5318698	04:51:00
18	AA43514765	M	2308	04:21:00	04:57:00	04:57:00	0	0	0	04:57:00	AA43514765	04:57:00	04:57:00	0	0	04:57:00
19	AA43514405	H	5026	04:21:00	04:59:00	04:59:00	0	0	0	04:59:00	AA43514405	04:59:00	04:59:00	0	0	04:59:00
20	AA43525479	M	278	04:46:00	05:02:00	05:06:00	0	-4	-39.06	05:02:00	AA43525479	05:02:00	05:06:00	-4	-39.0637395	05:02:00
21	AA43513759	M	681	04:30:00	05:07:00	05:09:00	1	-1	-9.77	05:08:00	AA43517362	05:07:00	05:07:00	0	0	05:07:00
22	AA43517362	H	6235	04:35:00	05:07:00	05:07:00	0	0	0	05:07:00	AA43513759	05:07:00	05:09:00	-1	-9.76593489	05:08:00
23	AA43524422	M	557	04:31:00	05:09:00	05:11:00	0	-2	-19.53	05:09:00	AA43524422	05:09:00	05:11:00	-2	-19.5318698	05:09:00
24	AA43515067	M	2513	04:45:00	05:17:00	05:18:00	0	-1	-17.4	05:17:00	AA43515067	05:17:00	05:18:00	-1	-17.3986132	05:17:00
25	AA43512961	M	931	04:45:00	05:20:00	05:22:00	0	-2	-19.53	05:20:00	AA43512961	05:20:00	05:22:00	-2	-19.5318698	05:20:00

Figure B.2: Example of ResultsFinal Table of 11th July 2015