IMPROVEMENTS TO THE GLOBAL OCEANIC MODEL AND PERFORMANCE ASSESSMENT OF THE NORTH ATLANTIC ORGANIZED TRACK SYSTEM

Yanqi Liang

Thesis submitted to the Faculty of the Virginia Polytechnic Institute
And State University in partial fulfillment of the requirements for the degree of

Master of Science
In
Civil Engineering

Antonio A. Trani, Chair
Montasir M. Abbas
Linbing Wang

July 14, 2017
Blacksburg, Virginia

Keywords: North Atlantic Organized Track System, Simulation Model, Reduced Separation, Fuel Consumption, Level of Service

IMPROVEMENTS TO GLOBAL OCEANIC MODEL AND PERFORMANCE ASSESSMENT OF NORTH ATLANTIC ORGARNIZED TRACK SYSTEM

Yanqi Liang

ABSTRACT

This thesis presents a performance assessment of flight operations in the North Atlantic Organized Track System (OTS) using the Global Oceanic Model. The main contributions of the study are: a) improvements to the logic of the Global Oceanic Model; b) prediction of benefits among various aircraft separation minima and operational policies to assign flights to tracks in the OTS system; and c) forecast of OTS traffic over North Atlantic from 2020 to 2040. The preliminary results show that a concept of operation with longitudinal separation minima of 15 nm and information of the flight cost matrix provides average fuel savings of 93 kilograms per flight using 2020 traffic. The fuel savings increase to 170 kilograms per flight using traffic levels expected in the year 2040. A new operational track assignment routine is developed and it could save around 40 kilograms per flight compared with the current concept of operations.

The study results show a shortage of capacity of the Organized Track System in the future. The analysis shows that the OTS configuration used today and in 2020 is unable to accommodate the traffic projected in 2040. The analysis concludes that more tracks will be needed to maintain an acceptable level of service.

IMPROVEMENTS TO GLOBAL OCEANIC MODEL AND PERFORMANCE ASSESSMENT OF NORTH ATLANTIC ORGARNIZED TRACK SYSTEM

Yanqi Liang

GENERAL AUDIENCE ABSTRACT

The North Atlantic Organized Track System (OTS) are directional tracks for aircraft to fly between North America and Europe. This thesis presents a performance assessment of flight operations in the North Atlantic using a computer simulation model -- Global Oceanic Model. The main contributions of the study are: a) improvements to the logic of the Global Oceanic Model; b) prediction of benefits among various aircraft separation minima and operational policies to assign flights to tracks in the OTS system; and c) forecast of OTS traffic over Atlantic from 2020 to 2040. The preliminary results show that the predicted average fuel savings in the year 2020 are 93 kilograms per flight when aircraft are separated 15 nm longitudinally and assigned to tracks based on the flight cost matrix. The average fuel savings increase to 170 kilograms per flight using traffic levels expected in the year 2040. Additionally, a new operational track assignment routine is developed and it could save around 40 kilograms per flight compared with the current concept of operations.

In conclusion, the Organized Track System configuration used today may be unable to accommodate the traffic projected in the year 2040. The shortage of capacity of the OTS indicates that more tracks will be needed to maintain an acceptable level of service.

ACKNOWLEGEMENTS

First of all, I would like to express my sincere gratitude to my advisor, Dr. Antonio A. Trani, for his guidance and support during my M.S. degree study. It is a great honor to work with him in the Air Transportation Systems Laboratory. His knowledge and easy-going personality benefited me a lot. His great passion for aviation and willingness to learn new things will influence and motivate me profoundly for the rest of my life.

My sincere gratitude to my committee members: Dr. Montasir M. Abbas and Dr. Linbing Wang for their patience and feedback on my thesis.

I am particularly thankful to Nicolas Hinze and Arman Izadi for their assistance in the research task. It has been a great experience working with these talented people in the development of the Global Oceanic Model.

Last and most importantly, I must express my profound gratitude to my family for their unconditional support and continuous encouragement through my years of study. To my parents, thank you for trusting me all the time and letting me choose my career.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1 Introduction

1.1 Objective of the Study

The objective of the study is to assess the performance of future aviation concepts to operate the North Atlantic Organized Track System (NAT OTS). This study serves as input to a cost benefit analysis conducted by the Federal Aviation Administration (FAA). This study is conducted using computer simulation. The analysis performed has been used by air navigation system providers (stakeholders) to make decisions on how to operate the future NAT OTS. In the future and with the development of new navigation, surveillance and communication systems, the minimum longitudinal separation between flights in the NAT OTS system could be reduced from the current 10 minutes (~80 nautical miles) to 5 minutes (~40 nm) and even 2 minutes (~15 nm). Lateral separation in the OTS will be reduced from 1 degree (~60 nm) to ½ degree (~23 nm). Reduced lateral separation means that more aircraft could obtain their optimal track and cruise flight levels, saving fuel burn and travel time.

To achieve this objective, a discrete time-step simulation model named Global Oceanic Model (GO Model) has been used in the study. The GO Model is an enhanced version of North Atlantic Simulation Model (NATSAM III) developed at the Air Transportation Systems Laboratory. The GO Model can simulate flights using the Organized Track System (OTS) and random flights (i.e., flights outside the OTS) simultaneously applying different separation rules to each group of flights. This study focuses on the performance of OTS flights only.

The study developed and integrated critical updates to the GO Model to improve the realism and reliability of the simulation results. The benefit analysis uses fuel consumption, travel time, and level of service metrics to evaluate various aircraft separation-related scenarios and several demand loads applicable in future years. The analysis provides information to air navigation service providers to improve the future operations in the North Atlantic.

1.2 The North Atlantic Region

1.2.1 Background

The North Atlantic airspace serves flights travelling between North America and Europe. The North Atlantic (NAT) is the busiest oceanic airspace in the world with more than 450,000 flights per year (Federal Aviation Admistration, 2017). The NAT airspace is called High Level Airspace (HLA) and Reduced Vertical Separation Minimum (RVSM) rules apply. Unlike domestic airspace with ground radar coverage, flights in the NAT airspace are not monitored continuously. Tracking flight operations in the NAT relies on discrete communications between the flight crew and ground oceanic controllers. Operations in the NAT follow offshore/oceanic procedures, which apply large minimum separations compared to those used in domestic airspace with ground radar coverage.

In February, 2016, the North Atlantic Minimum Navigational Performance Specifications Airspace (NAT MNPSA) was re-designated as North Atlantic High Level Airspace (NAT HLA). The main difference is that NAT HLA excludes portions of Shanwick Oceanic Control Area (SOTA and BOTA) but includes Bodo Oceanic. It should be noticed that '*approvals issued to operate in the NAT HLA are referred to as 'NAT MNPS' approvals'*(ICAO, 2017). MNPS airspace is established to mitigate the risk of collision resulted from a loss of horizontal separation between aircraft and an agreed target level of safety. The accessible altitude in MNPS airspace is between Flight Level (FL) 285[1] and FL 420. In MNPS airspace, the lateral separation between aircraft is 60 nm or 1 degree. Given the curvature of the earth, 'Gentle Slope Rules' are adopted to slightly adjust the latitude effect but lateral separation never falls under 50.5 nm. The longitudinal separation minima vary from 10 minutes to 15 minutes depending on the aircraft class (jet, propeller aircraft) and the differences in speed between the flights.

[1] Flight level is expressed in hundreds of feet. For example, FL 285 means 28,500 ft.

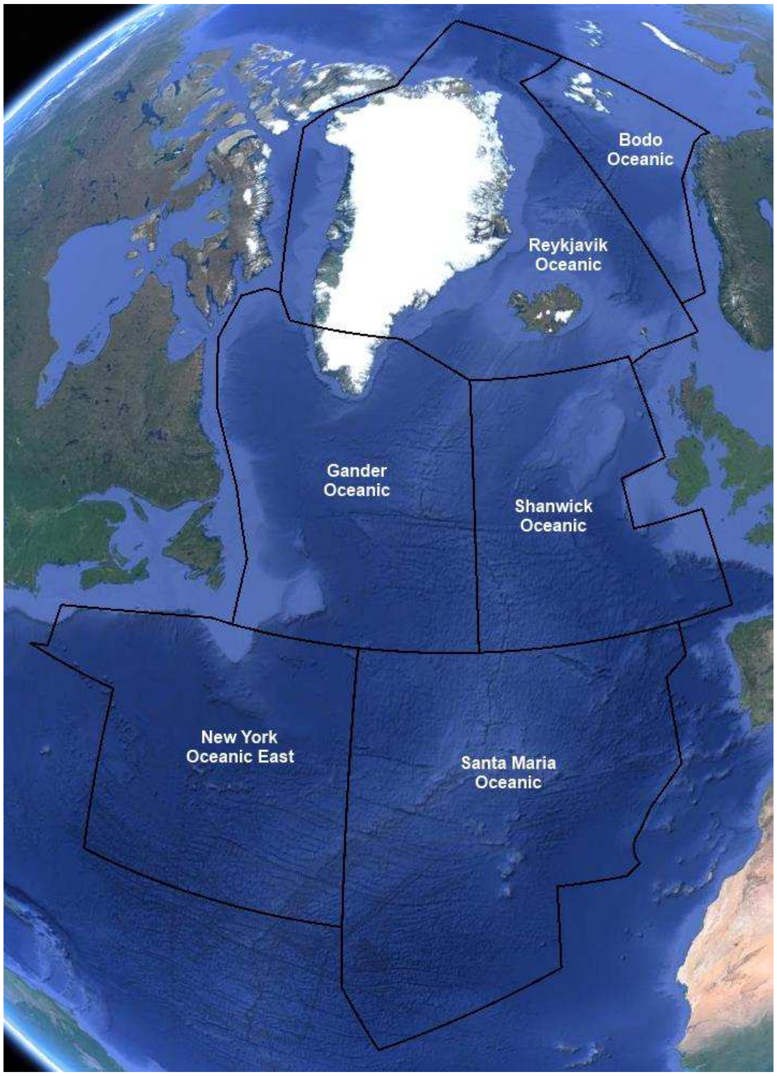


Figure 1-1 The North Atlantic High Level Airspace (NAT HLA).

The boundaries of the NAT HLA include the following oceanic control areas: Reykjavik, Shanwick (excluding SOTA & BOTA), Gander, Santa Maria Oceanic, Bodo Oceanic and the portion of New York East which is north of 27°N. The oceanic control areas are shown in Figure 1-1.

RVSM airspace is established within the confines of MNPS airspace and applied between FL 290 and FL 410. The vertical separation between RVSM-approved aircraft in this airspace is 1000 ft. RVSM allows more aircraft to fly at their optimal altitude, thereby improving fuel efficiency and performance. For aircraft or operator not authorized for RVSM operations or without operable RVSM equipment, the Federal Aviation Administration (FAA) prohibits filing an RVSM equipment/capability qualifier (U.S. Department of Transportation Federal Aviation Administration, 2012). These non RVSM-approved aircraft must fly lower or higher than the RVSM airspace.

1.2.2 The North Atlantic Organized Track System

Due to increasing airline traffic across the North Atlantic and to manage workload for air traffic controllers separating aircraft effectively, the North Atlantic Organized Track System (NAT OTS) was introduced in 1961. Aircraft flying onto fixed track structures are separated effectively by time, altitude and latitude (see Figure 1-2).

The routes and flight levels in the Organized Track System are published twice every day by FAA, NavCanada, the National Air Traffic Services (NATS) and the Joint Aviation Authorities (JAA). Eastbound tracks are valid from 01:00 Coordinated Universal Time (UTC) to 8:00 UTC at 30°W. Westbound tracks are valid from 11:30 UTC to 19:00 UTC at 30°W. This operational concept accommodates traditional airline schedules, with departures from North America to Europe at night allowing passengers to arrive at their destination in the morning. Westbound departures from Europe are scheduled in the morning and late afternoon and arrive in North America early afternoon to late evening. This schedule helps airlines to use aircraft efficiently flying to Europe in the evening and back to North America in the day.

Figure 1-2 Eastbound OTS on June 10th, 2017. Source: http://www.flightradar24.com/.

The OTS is located inside the MNPS and RVSM airspace. The factors considered in the design of the OTS are as follows:

1) Minimum time tracks of the flights and their optimal cruise altitudes.

2) Meteorological conditions such as weather and wind patterns. In general, eastbound tracks take advantage of routing along the jetstream, while westbound tracks try to minimize the effect of headwind.
3) Airline preferences, airspace restrictions, demand from opposite direction and feedback from users.

In December 2015, the first phase of North Atlantic trials to reduce the lateral separation minimum from 60 nm (1°) to 25 nm (1/2°) commenced. This phase includes reduced lateral separations in two core OTS tracks between FL350 and FL390 in the airspace associated with the NAT Region Data Link Mandate. This setting is also referred to Reduced Lateral Separation Minima (RLAT). Only aircraft with the appropriate Required Navigation Performance (RNP4) approval and operating Automatic Dependent Surveillance-Contract (ADS-C) and Controller Pilot Data Link Communications (CPDLC), are permitted to operate on the 1/2 degree spaced tracks (Flight Service Bureau, 2015). Phase 2 of the trial will extend the RLAT concept to all tracks within valid altitude range. The implementation has been delayed and the new date is yet to be published. RLAT reduces the separation among tracks and thereby allowing more aircraft to fly in fuel efficient routes.

In 2011, the longitudinal separation was reduced from ten minutes to five minutes for aircraft equipped with ADS-C and CPDLC (Flight Service Bureau, 2015). By 2020, all aircraft are expected to be equipped with ADS-C, CPDLC and be RNP 4 equipped. In the future, aircraft could be separated by two minutes in trail if they are equipped with satellite-based automatic dependent surveillance-broadcast (SB ADS-B), CPDLC datalink and RNP4.

Reduced vertical separation minima (RVSM) is applied in the North Atlantic for RVSM-approved aircraft with vertical separations of 1,000 ft. between FL290 (29,000 ft.) and FL410 (41,000 ft.), inclusive.

1.2.3 Random Airspace

The use of OTS tracks is not mandatory. Currently about half of NAT flights utilize the OTS (ICAO, 2017). Aircraft may fly on random routes which remain clear of the

OTS or may fly on any route that joins or leaves an outer track of the OTS (ICAO, 2017). Flights outside the hours of OTS operation or flights filing cruise flight levels above published levels during valid OTS times can fly random routes. Random routes are unique to each flight allowing the airline to file a flight plan with the most efficient route for each type of aircraft. This can improve the performance by providing an optimal flight path based on weather conditions and operational efficiency.

One important difference between the OTS and random airspace is the application of different vertical separation rules. In the OTS, flights are separated 1000 ft. vertically within valid flight levels. However, random airspace follows hemispherical rules. Eastbound flights (flying magnetic headings from 0^o to 179^o) fly odd thousand flight levels (i.e., FL 290, 310, 330, 350, 370 and 390) while westbound flights (flying magnetic headings from 180^o to 359^o) use even thousand flight levels (i.e., FL 300, 320, 340, 360, 380, 400).

1.2.4 North Atlantic Data Link Mandate

To operate in the OTS, aircraft should meet requirements of specified tracks and flight levels within NAT system. The requirements have been continuously published in a phased approach in recent years. In February 2013, Phase 1 of the North Atlantic Data Link mandate was implemented. This mandate requires flights to be equipped with a Controller Pilot Data Link (CPDLC) system and Automatic Dependent Surveillance-Contract (ADS-C) to fly in the altitude band FL 360 – 390 inclusive in two core tracks. In February 2015, a Phase 2 mandate was implemented. The new mandate requires the same equipment as Phase 1 and expand the operations to all NAT OTS Tracks in the altitude band FL 350 – 390 inclusive. Phase 2 consists of three stages:

- Phase 2A, started in February 2015 and applies the mandate to FL350 to FL390 on all NAT OTS tracks;
- Phase 2B, will start in December 2017 and apply to FL350-FL390 throughout the ICAO NAT Region;
- Phase 2C, will start in January 2020 and apply to FL290 and above throughout the ICAO NAT Region (ICAO, 2017).

Although a mandate to install satellite-based Automatic Dependent Surveillance – Broadcast (ADS-B) over North Atlantic has not been determined, it is critical to study the potential benefits of operating aircraft with even smaller reduced separation criteria.

Descriptions of equipment used by oceanic flights are explained below.

i. Controller Pilot Data Link Communications (CPDLC)

CPDLC is system that allows oceanic controllers and pilots to communicate with codified clearance/information/request messages. Equipped with CPDLC, the oceanic controller is able to issue flight level assignment, crossing constrains, lateral deviations, route changes and clearances, speed assignments, radio frequency assignments and various requests for information. The pilot can respond to messages, request clearances and declare an emergency. The communication directives are entered into a digital interface in the flight deck (see Figure 1-3).

Figure 1-3 CPDLC Digital Interface in the Flight Deck.
Source: https://en.wikipedia.org/wiki/File:A330_DCDU.jpg.

ii. Automatic Dependent Surveillance-Contract

ADS-C is an agreement between the ground system and the aircraft on what data is transmitted based on an explicit contract. The ADS-C application has four contract types: periodic contract, demand contract, event contract, and mayday message.

Periodic contracts are time-based and can be varied when necessary by Air Traffic Control (ATC) needs. In oceanic areas, periodic messages are normally scheduled every 12-14 minutes.

Demand contract: ATC initiates this contract when the position of all aircraft is needed. Every flight responds on demand.

An Event contract is a contract set up by the ATC that predesignates the aircraft altitude, vertical speed or any of several other different parameters. If the flight deviates from any of these parameters, the crew is notified immediately.

Mayday message: This contract is controlled and initiated by the pilot during emergency circumstances (Vena, 2012).

iii. Automatic Dependent Surveillance-Broadcast

ADS-B is a surveillance technique that relies on aircraft or airport vehicles broadcasting their identity, position and other information derived from on board systems. This signal (ADS-B Out) can be captured for surveillance purposes on the ground (ADS-B Out) or on-board aircraft in order to facilitate airborne traffic situational awareness, spacing, separation and self-separation (ADS-B In). ADS-B data include aircraft horizontal position (latitude/longitude), aircraft barometric altitude, quality indicators, aircraft identification (unique 24-bit aircraft address, aircraft identification, Mode A code (in the case of CS ACNS for "ADS-B Out")), emergency status, special position indicator when selected (SKYbary, 2016).

iv. Required Navigation Performance

RNP is a qualifier that defines the 95 percent navigation accuracy performance that meets a specified value for a particular phase of flight or flight segment. RNP incorporates associated onboard performance-monitoring and alerting features to notify the pilot when the RNP for a particular phase or segment of a flight is not being met (FAA, 2016). RNP also refers to the level of performance for a specific procedure or a specific block of airspace.

RNP 10 requires the aircraft navigation system to calculate its position to within a square with a lateral dimension of 10 nm with the accuracy for at least 95% of total flight time. The accuracy refers to lateral total system error and along-track error. RNP 4 also requires the error of using aircraft navigation system to calculate its position to be less than 4 nm for at least 95% of total flight time (see Figure 1-4).

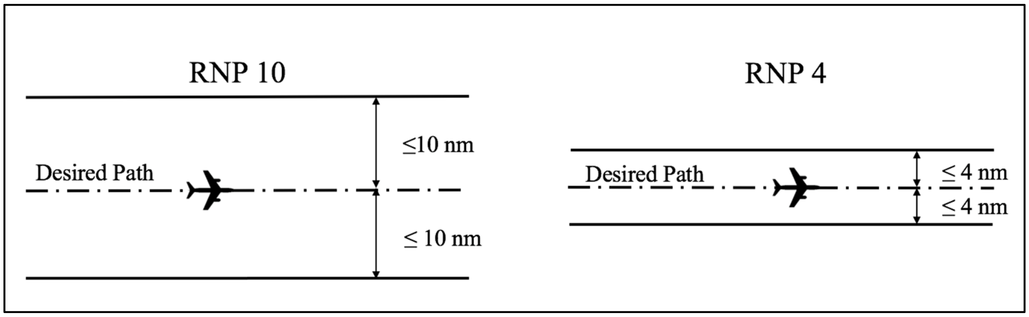


Figure 1-4 Illustration of RNP 10 and RNP 4.

1.3 North Atlantic Traffic Forecast

(ICAO, 2016) approved a North Atlantic Economic, Financial and Forecast Group (NAT EFFG) traffic forecast which is composed of near-term projection for the first 5 years and long-term portion that forecasts 6 to 20 years into the future. The near-term forecast is based on fleet analysis and business plans and projects an annual growth rate of 3.6%. Between 2015 and 2035, NAT traffic is projected to grow 3.2% annually.

The FAA has developed Matlab source code to execute the North Atlantic traffic forecast. The model generates additional traffic demand at the Origin-Destination (O-D) pair level. Examination of the FAA forecast shows many O-D pairs growing rapidly in the near-term. Such growth is not realistic for the long-term. To generate a traffic demand that matches the 3.2% annual growth rate accepted by ICAO, a modification of the forecast was developed to adjust the individual growth rate of Origin/Destination pairs. A limitation of 7% in the individual annual growth rate for each O-D pair after the year 2030 is assured in this new procedure. Using the assumed 3.2% annual growth rate, it is estimated that the North Atlantic traffic will grow 125% from 2016 to 2040 (see Figure 1-5).

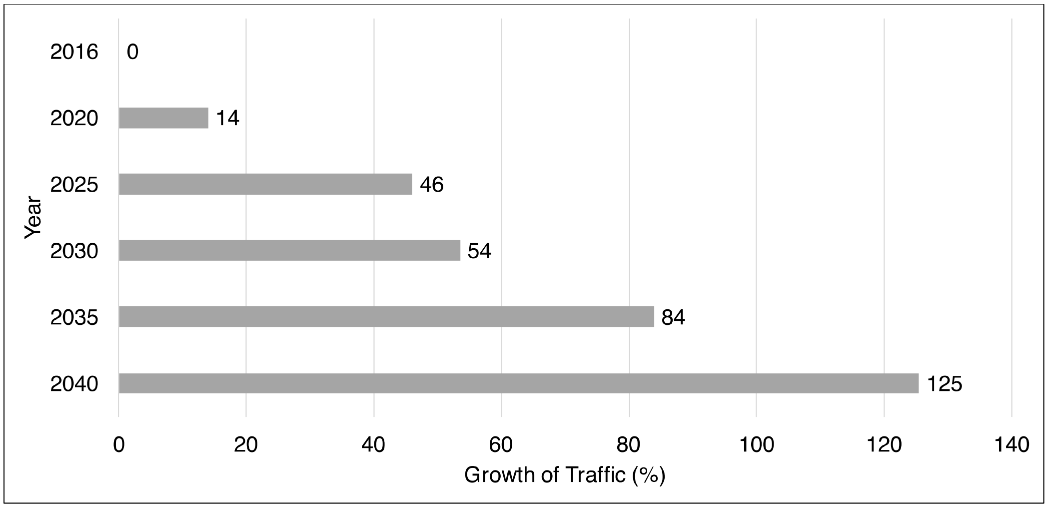


Figure 1-5 North Atlantic Demand Forecast from 2016 to 2040.

CHAPTER 2 The Performance Assessment of Flight Operations in The North Atlantic Organized Track System

2.1 Literature Review

Almira Williams et al. 2006 developed a discrete event simulation model to assess benefits of reduced horizontal and lateral separations in North Atlantic Organized Track System in terms of fuel and time savings, and additional cargo revenue potential (Williams, 2006). The model simulated demand sets in 2005, 2010, 2015 which were generated using actual 2004 flight data as a baseline and five levels of equipage were applied: 0, 25, 50, 75 and 100 percent. The results showed that potential fuel and time savings increase with both equipage and demand levels with a range from 0.05% (when 25% of aircraft are equipped in 2005) to 0.45% (when 100% of aircraft are equipped in 2015). Fuel savings potential was estimated to be 75% higher if the cost of fuel required to carry additional cargo is not included. In addition, equipped flights save on average 1.5-4.6 times more than non-equipped flights.

Norma Campos et al. 2012 studied the cost impact to U.S. commercial operators with the implementation of a data link mandate in the North Atlantic (NAT) Airspace. The relevant data links are FANS-1/A ADSC and CPDLC. The analysis showed that avionics procurement and installation are the largest costs for aircraft operators. The biggest portion of such cost applies to the retrofit of legacy aircraft, which represents highly frequent operations in the NAT and very low equipage levels (Campos, Graham, Grimes, & Joyce, 2013).

Aswin Kumar Gunnam 2012 developed a numerical integration and simulation model called North Atlantic Simulation and Modeling (NATSAM) to study the effects of new strategies in the North Atlantic Organized Track System. The model explored the benefits of several separation minima and different equipage levels. The model estimated the fuel burned for each track and flight level and then it created a cost matrix for aircraft to select optimal track and flight level.

Olga Rodionova et al. 2014 introduced a mathematical model to optimize aircraft trajectory in North Atlantic Airspace. They only applied real eastbound traffic data to a genetic algorithm with several separation standards (10 min for now and 2 min in the future). It is assumed that a flight is constrained to change to higher flight level at waypoint and this climb is instantaneous. The impact of the oceanic wind is also considered. The results showed that the reduction of current separation standard could lead to more optimal trajectories and significantly decrease the total flight time and congestion level in pre-oceanic airspace (Rodionova, 2014).

Tao Li 2014 developed a microscopic discrete event simulation model for air traffic in the Organized Track System over North Atlantic Airspace. Four procedures were applied to improve the flight operations: 1) climb inside the OTS, 2) track switch, 3) Mach number adjustment in OTS entry area, 4) assign multiple flights simultaneously. The results showed that the first procedure (climb inside OTS) produces the most annual fuel benefit (about $8000 per flight) while the second (track switch) has the least annual fuel benefit (about $2000 per flight). He concluded that the first two procedures would benefit all the flights using and the latter two would improve the OTS assignment process (Li, 2014).

Nikolaos Tsikas 2015 developed algorithms to modify the North Atlantic OTS configuration and simulated OTS traffic in several scenarios with different separation minima. The performance metrics include fuel consumption, travel time and level of service. The results showed that the scenario with reduced lateral separation minimum (25 nm) and the reduced longitudinal separation minimum (8 nm) was the most optimal.

2.2 Data Analysis

2.2.1 TFMS Data

Traffic Flow Management System (TFMS, previously ETMS) is a data exchange system for supporting the management and monitoring of national air traffic flows. TFMS processes all available data sources such as flight plan messages, flight plan amendment messages, and departure and arrival messages. The FAA Airspace Lab assembles TFMS flight messages into one record per flight. TFMS is restricted to the subset of flights that fly under Instrument Flight Rules (IFR) and are captured by the FAA enroute computers (FAA, 2014).

The FAA provided the Air Transportation Systems Laboratory at Virginia Tech with demand sets for eight seed days of traffic (see Table 2-1). Each demand set includes flight information one day before and after the seed day. Having demand sets for three consecutive days enables the GO Model to 'warm up' the simulation system and focus on flights on the seed day. This consecutive demand set makes the results more accurate because there is no need to clone the flight demand set.

Table 2-1 Eight TFMS Seed Days.

01/28/2016	03/28/2016	03/31/2016	05/28/2016
06/25/2016	08/16/2016	10/17/2016	11/01/2016

The flight demand sets were provided in text (.txt) format. A number of steps were used to process the analysis of these sets (see Figure 2-1). The first step of the analysis was to import them in Matlab and convert them into binary (.mat) files. The flights were then classified based on the oceanic region they traverse. Four new fields were added to the demand sets to identify the oceanic region they traverse as Pacific, Atlantic, West Atlantic Route System (WATRS) and New York Air Route Traffic Control Center (ZNY). Each flight was marked with either 1 or 0 when it crossed or not the boundaries of each of the preceding areas. Subsequently, the ICAO four letter airport coding was replaced with the corresponding three letter IATA code. This process was essential in order to check for potential missing O-D pairs using the Official Airline Guide (OAG) data. The last step was to plot all flights in order to visually inspect any anomalies (e.g. Non-oceanic flights). Unrealistic flights based on distance, aircraft type or departure and arrival airports were identified and eliminated (Tsikas, 2016).

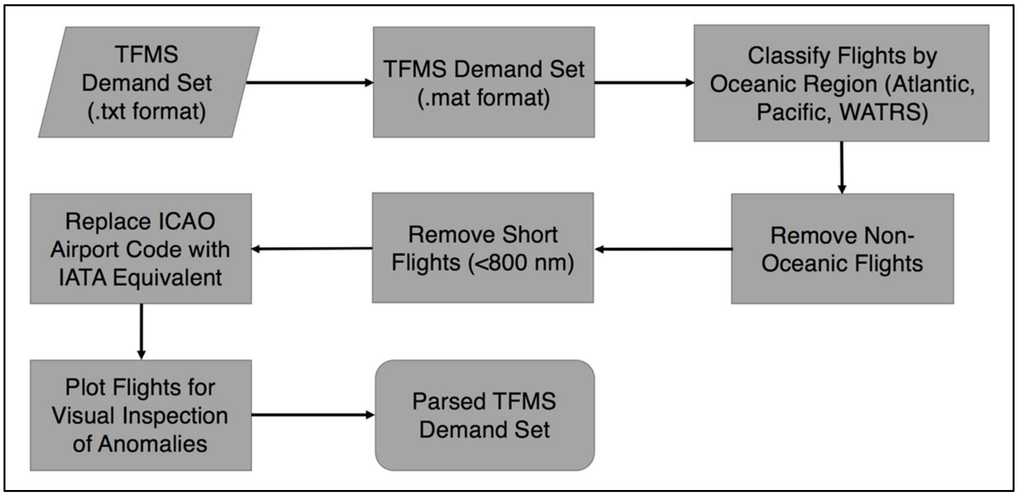


Figure 2-1 Flowchart of TFMS Data Analysis Process.

2.2.2 OAG Data

The Official Airline Guide (OAG) is an air travel intelligence company that provides digital information and applications to the world's airlines, airports, government agencies and travel-related service companies. OAG compiles airline schedules into a database consisting of future and historical flight details for more than 900 airlines and over 4,000 airports. Because OAG demand set is comprehensive, it is used to identify gaps in the TFMS demand set.

(Tsikas, 2016) performed an analysis of TFMS and OAG data and created a procedure to identify gaps in the TFMS data using OAG data in the GO Model. For this study we use the TFMS data provided by FAA.

2.2.3 BADA Data

The Base of Aircraft Data (BADA) is an Aircraft Performance Model (APM) developed and maintained by EUROCONTROL, with the active cooperation of aircraft manufactures and operating airlines. The BADA APM has been developed for simulation studies, and prediction of aircraft trajectories for purposes of Air Traffic Management research and operations (Eurocontrol, 2015).

BADA APM is based on the Total Energy Model[2] approach to aircraft

[2] Total Energy Model equates the rate of work done by forces acting on the aircraft with the rate of increase in potential and kinetic energy. It can be considered as a reduced point-mass model.

performance modelling. It is made of two components – the Model Specifications providing the fundamental equations used to calculate aircraft performance parameters and a dataset containing the aircraft-specific coefficients necessary to perform calculations.

The BADA version employed in this study is EUROCONTROL BADA 3.13 with 42 aircraft types. Each TFMS flight is mapped to a BADA aircraft type in the GO Model. Each aircraft is modeled from takeoff to landing to create realistic flight trajectory based on the Total Energy Model.

2.2.4 NAT OTS Track Data

NAVCanada[3] provided twenty-four daily track configurations of OTS data. Track records include date, direction, track alphabetic name, entry and exit waypoints, and geographical coordinates (See Figure 2-2). This dataset does not include the available flight levels for each track. Flight level data was collected from Blackswan[4] (See Figure 2-3). A comprehensive track record is combined from these two datasets. The resulting records are converted into a binary (. mat) file.

Two directional OTS track configurations are shown in Figure 2-4. Each track is designated by alphabet letters. Eastbound tracks are blue tracks labeled 'S', 'T', 'U', 'V', 'W', 'X', 'Y', 'Z'. Westbound tracks are green tracks labeled 'A', 'B', 'C', 'D', 'E', 'F', 'G', 'H', 'J'. On June 25, 2016, three eastbound tracks 'T', 'U', 'V' are spaced ½ degree and called 'core tracks'.

Date at 30W	Direction	Nat Track	Point of Entry	10W Coordinates	15W Coordinates	20W Coordinates	30W Coordinates	40W Coordinates	50W Coordinates	60W Coordinates	Point of Exit	Point following Oceanic Exit
25Jun2016	W	A	ATSIX			6200N02000W	6300N03000W	6300N04000W	6200N05000W		PIDSO	

Figure 2-2 Example of Track Data from NAVCanada.

[3] NAVCanada is a privately run, not-for-profit corporation that owns and operates Canada's civil air navigation system.

[4] Blackswan.ch is a website that publishes Organized Track System data every day and provides a one-year track information dataset.

Track		Eastbound FLs	Westbound FLs	NAR	EUR RTS	TMI
A	ATSIX 62/20 63/30 63/40 62/50 PIDSO	NIL	310 320 330 350 360 370	NIL	WEST NIL	**177** 242002 EGGXZOZX

Figure 2-3 Example of Track Data Collected from Blackswan on June 25, 2016.

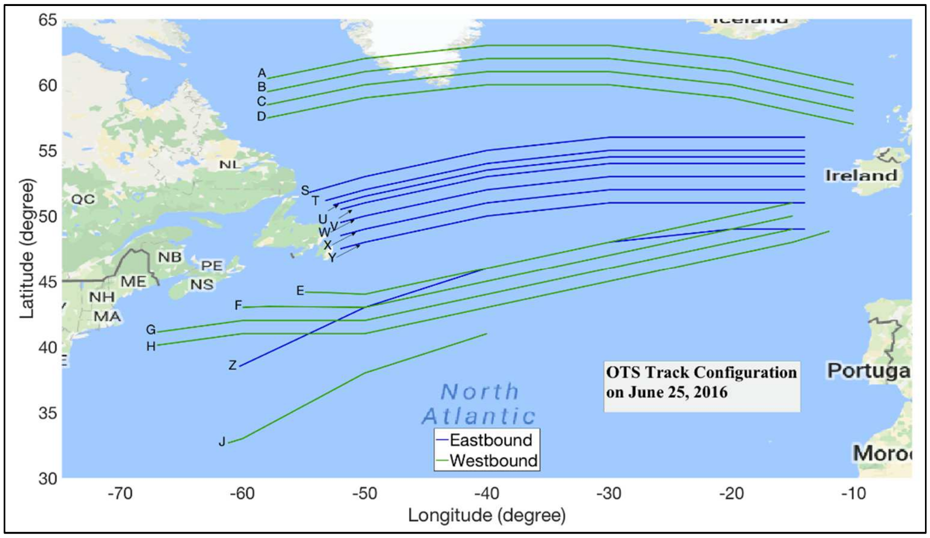


Figure 2-4 Illustration of OTS Track Configuration. Source: NAVCanada.

2.2.5 Wind Data

Wind data used for modeling is collected from the National Center for Atmospheric Research (NCAR) Reanalysis model (NOAA/ESRL/PSD 2014). The reanalysis model was developed by Earth System Research Laboratory Physical Science Division. The data provides the magnitude and direction of the wind every 2.5 degrees latitude and longitude at 17 geopotential heights (Li, 2014).

For eastbound flights, the wind over the North Atlantic is generally tailwind dominantly because the jetstream flows from west to east. Tailwind means the wind is in the same direction as the flight and it could help reduce fuel consumption and travel time. North Atlantic westbound flights normally experience headwinds which are opposite to the direction of flight. Therefore, tracks are configured to maximize the effect of tailwind for eastbound flights and minimize headwind for westbound tracks. A map of wind vector over North Atlantic on June 25, 2016 is shown in Figure 2-5.

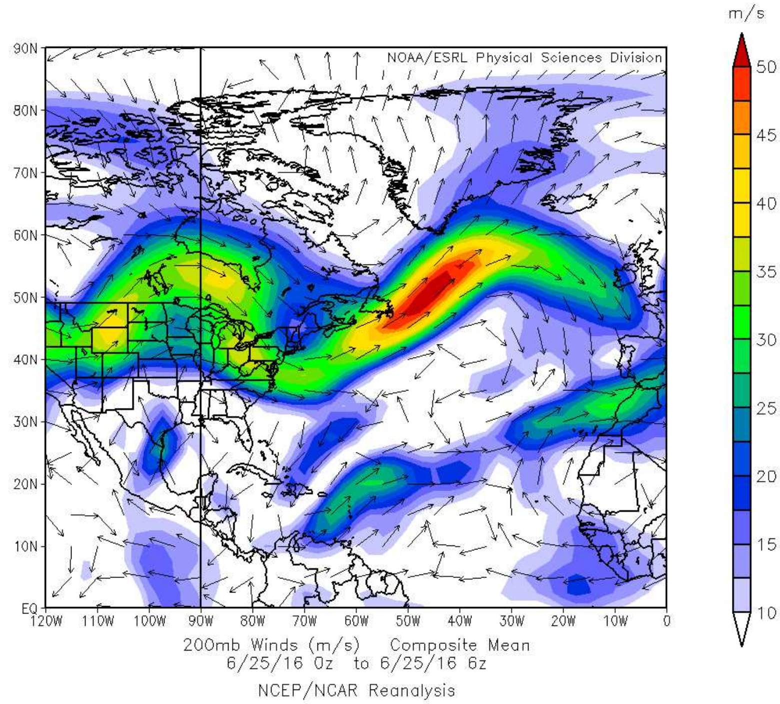


Figure 2-5 Wind Vector at 200 mb (39,000 ft.) on June 25, 2016 from NCAR Model.

2.3 Modeling

2.3.1 Global Oceanic Model

In 2012 Aswin Gunnam developed a numerical integration and simulation model called the North Atlantic Simulation and Modeling (NATSAM) to study the effects of new strategies on NAT system performance. The model simulated individual flight performance to calculate fuel burned for each track and flight level and created a cost matrix for selecting the optimal tracks for each flight.

In 2014 Tao Li built a microscopic discrete-time simulation model to simulate air traffic in the OTS. This model (NATSAM III) was based on NATSAM and improved the simulation by adding limited flight de-conflicting rules. The model added more procedures allowing climbs inside the OTS and switching OTS tracks. The improved logic included Mach number adjustments and simultaneous assignment of multiple flights to tracks.

In 2016 the Air Transportation System Laboratory further enhanced the NATSAM III and it became the Global Oceanic Model. The model is a microscopic simulation tool that offers quick, inexpensive and realistic evaluation on new policies, procedures and technologies proposed to improve flight operations over global oceanic airspace. This model can simulate random flights with user-preferred routes and Organized Track System flights. The current model has eight main modules as shown in Figure 2-6.

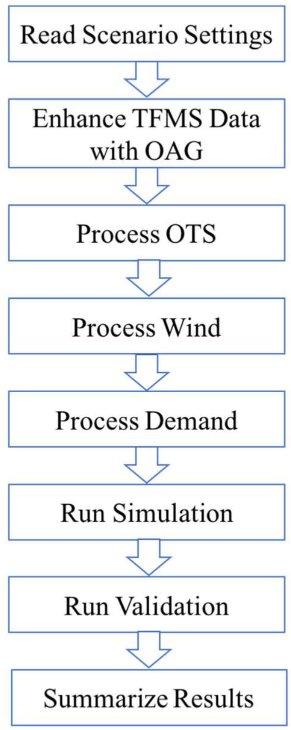


Figure 2-6 Flowchart of Steps Employed in Simplified the Global Oceanic Model Process.

Users can set up their preferred scenario as the first step. Users can define different separation standards, number of climbs inside OTS and track configurations (half degree separated or original track setting). Since the 2016 TFMS data provided by FAA is comprehensive and reliable, there is no need to enhance the model input with OAG data. We developed procedures to read OTS track data from NAVCanada and Blackswan and converted it into a binary (.mat) file. Wind data from pre-analysis model is prepared and used to calculate aircraft ground speed[5]. Processing demand is a critical step which generates a comprehensive flight structure that includes flight information, track assignment and demand schedule.

[5] Ground speed is determined by the sum of aircraft's true airspeed and wind speed; a headwind subtracts from ground speed while a tailwind adds to it.

The model is a time-step simulation that updates the state variables of each flight according to a user-defined time step (normally 5 seconds). The model manages request, handles track and flight level assignment, checks for climb availability and detects conflicts every time step. The output of the model is a flight trajectory and flight state results (e.g. fuel consumption, travel time, etc.). The model validation consists of basic aircraft statistics, a potential conflict violation post-processor and track loading statistics. The output files summarize simulation results into excel file in order to compare statistics across scenarios.

2.3.2 Improvements to Global Oceanic Model

2.3.2.1 OTS Flight Assignment

i. OTS Cost Matrix

Analysis of FAA TFMS data revealed that incomplete information about flight levels requested by each flight. Moreover, track use information is not reliable. One reason for missing track information is some flights travel in the southern part of the Atlantic, yet their requested track is in the northern part of NAT OTS. Furthermore, we found that flight levels for some flights were not realistic. For example, some large aircraft are assigned to high flight level such as FL 380, however, they cannot obtain such a high flight level initially because of the weight.

An approach to address this problem is to use a model that calculated track and flight level assignment based on a cost matrix. The cost matrix is calculated by a set of functions that evaluate numerous flight plans (random and OTS) for each NAT flight in the simulation. At present time, fuel consumption is the only metric used to construct cost matrix. The cost matrix is a table containing combinations of track and flight level fuel costs (see Figure 2-7).

Fuel Cost Matrix (kg)		Eastbound Tracks							
		NAT S	NAT T	NAT U	NAT V	NAT W	NAT X	NAT Y	NAT Z
Flight Level (ft.)	40,000	999,999	999,999	999,999	999,999	77,777,777	999,999	999,999	999,999
	39,000	999,999	999,999	999,999	999,999	999,999	999,999	999,999	77,777,777
	38,000	999,999	999,999	999,999	999,999	999,999	999,999	999,999	999,999
	37,000	92,031	91,602	91,217	90,843	90,430	90,066	89,735	77,777,777
	36,000	92,384	91,953	91,570	91,199	90,793	90,438	90,120	77,777,777
	35,000	92,747	92,316	91,935	91,570	91,171	90,824	90,518	77,777,777
	34,000	93,163	92,732	92,355	91,991	77,777,777	91,263	90,970	90,619
	33,000	93,834	93,396	93,007	92,634	77,777,777	91,899	91,601	77,777,777
	32,000	94,580	94,132	93,732	93,349	77,777,777	92,605	92,303	92,178
	31,000	77,777,777	77,777,777	77,777,777	77,777,777	77,777,777	77,777,777	77,777,777	77,777,777

77,777,777: Big Number Assigned to Flight Levels Not Available for Evaluation
999,999 : Big Number Assigned to Flight Levels Unfeasible for This Aircraft

Figure 2-7 Fuel Cost Matrix for a Set of OTS Flight Plans (MMMX-LEMD, Airbus 340-600).

The numbers labeled '77,777,777' in Figure 2-7 indicate the flight is not capable of flying in the track and flight level because of track configuration. The numbers '999,999' are unfeasible track and flight levels due to weight limitation at the entry point of the OTS. Unfeasible track and flight levels would reduce computational time of fuel consumption and travel time. The track and flight level combination with the least fuel cost is considered as the ideal track and flight level for that flight in the simulation. For the values shown in Figure 2-7, the ideal track and flight level is NAT Y at 37,000 ft. with a 'fuel cost' of 89,735 kilograms.

It is important to note that for flights whose optimal flight plan is to fly a random route (non OTS), the fuel consumption matrix includes a random route and OTS cost matrix. If the flight intersects with the OTS over a long distance within the OTS operational hours, it will be assigned the best OTS route.

ii. Assignment Routine

After determination of the best track and flight level for each flight has been made, the decision for actual OTS assignment is made in the simulation 350 nm (~45 min) before the entry point of the Organized Track System. The process begins with a calculation of the maximum operational altitude achievable by the flight. It is possible that the ideal flight level is higher than maximum operation altitude at the decision point because we use average wind conditions in calculation of the cost matrix. In the simulation, wind data is updated every time step and hence this provides a more precise evaluation of the actual mass state of the flight at the assignment point.

Since the traffic in the OTS could be heavy during peak hour conditions, not all the oceanic flights are able to obtain their ideal track and flight level. When a requested track and flight level is not available, air traffic controllers will use three approaches to re-assign the flight. First, air traffic controllers will climb or descend the flight to other flight levels in the same track. Second, they may ask pilot to switch to other tracks. The last approach is to adjust the aircraft Mach number to modify the arrival time and achieve the desired separation at the entry point in the oceanic track.

Based on discussions with NATS UK controllers, their preferred method is to re-assign a flight using the same track but climbing 1,000 ft. or descending up to 3,000 ft. if the requested flight level is not available. If none of them is approved, controllers would consider the requested flight level on another track and repeat the flight level checking process.

When a large aircraft is assigned 3,000 ft. below to its optional altitude, this increases the fuel use considerably. This study attempts to analyze the fuel benefits with different OTS track/flight level assignment procedures compared to the current operational policy.

An alternative concept is to first use a 1,000-ft. climb or a 1,000-ft. descent in the ideal track. If none of these choices is available, then flight levels in other tracks are considered based on the cost matrix.

A second alternative concept of operations is to assign track and flight level totally based on the flight cost matrix. If the ideal track and flight level could not be achieved, then flight levels in other track would be taken into consideration, using the ranked solutions of the cost matrix.

Computer code was developed and integrated into the GO Model to model these new track assignment procedures. The algorithms are called ‘FLs Heuristics’ and ‘Full Cost Matrix Heuristics’. If the ideal track and flight level is not available, ‘FLs Heuristics’ considers other flight levels in the same track. If these options fall, the ideal flight level in other tracks will be considered. The next track is selected from a track sequence, which is generated based on the mean value of fuel consumption for each track.

Consider the fuel cost matrix shown in Figure 2-7 as an example. Excluding large numbers, the track sequence based on mean value of fuel consumption is shown in Table 2-2. To save computational time, tracks that are distant from the ideal track are not considered. Checklists of the three track and flight level assignment concepts of operation are shown in Table 2-3. The ranking order in Table 2-2 and Table 2-3 is from left to right.

Table 2-2 Track Sequence for Cost Matrix Shown in Figure 2-7.

Track Sequence				
NAT Y	NAT W	NAT X	NAT Z	NAT V

Table 2-3 Checklists of The Three Assignment Routines.

Heuristics	Checklist of Track and Flight Level									
FLs_1_3	Fuel (kg)	89,735	90,120	90,518	90,970	90,430	90,793	91,171	90,066	90,438
	Track and Flight Level Sequence	**NAT Y, 37,000**	**NAT Y, 36,000**	**NAT Y, 35,000**	**NAT Y 34,000**	**NAT W, 37,000**	**NAT W, 36,000**	**NAT W, 35,000**	**NAT X, 37,000**	**NAT X, 36,000**
FLs_1_1	Fuel (kg)	89,735	90,120	90,430	90,793	90,006	90,438	90,843	91,199	/
	Track and Flight Level Sequence	**NAT Y, 37,000**	**NAT Y, 36,000**	**NAT W, 37,000**	**NAT W, 36,000**	**NAT X, 37,000**	**NAT X, 36,000**	**NAT V, 37,000**	**NAT V, 36,000**	**/**
Full Cost Matrix	Fuel (kg)	89,735	90,066	90,120	90,430	90,438	90,518	90,619	90,793	90,824
	Track and Flight Level Sequence	**NAT Y, 37,000**	**NAT X, 37,000**	**NAT Y, 36,000**	**NAT W, 37,000**	**NAT X, 36,000**	**NAT Y, 35,000**	**NAT Z, 34,000**	**NAT W, 36,000**	**NAT X, 35,000**

Notes: 1. FLs_1_3 means current routine assigning 1,000 ft. above or up to 3,000 ft. below ideal flight level.

2. FLs_1_1 means assignment of 1,000 ft. above or 1,000 ft. below ideal flight level.

3. Full Cost Matrix only considers fuel cost regardless of the ideal track.

2.3.2.2 Additional Headway Rule

In the OTS assignment, two factors are considered by oceanic controllers: longitudinal separation inside the OTS and the flight path clearance (from aircraft's current position to the entry point of the track). Flights equipped with CPDLC and ADS-C systems and approved to RNP 4 navigation standards can be separated longitudinally by 5 minutes. A Mach number[6] technique is used to adjust the minimum longitudinal separation at the entry track point to ensure conflict free trajectories for the complete track path length. The equations about this adjustment are as follows:

$$\text{minimum longitudinal separation} = \begin{cases} \max(H_{long}+k_{open}(S_L-S_F), H_{long}), \text{for opening case } (S_L \geq S_F) \\ \max(H_{long}+k_{close}(S_L-S_F), H_{long}), \text{for closing case } (S_L < S_F) \end{cases} \quad \textbf{Eq. 2-1}$$

where H_{long} is the standard longitudinal separation (5 minutes for ADS-C equipped aircraft), $k_{open} = -100$ and $k_{close} = -300$, S_L and S_F are the speeds (in Mach number) of the leading and following aircraft, respectively.

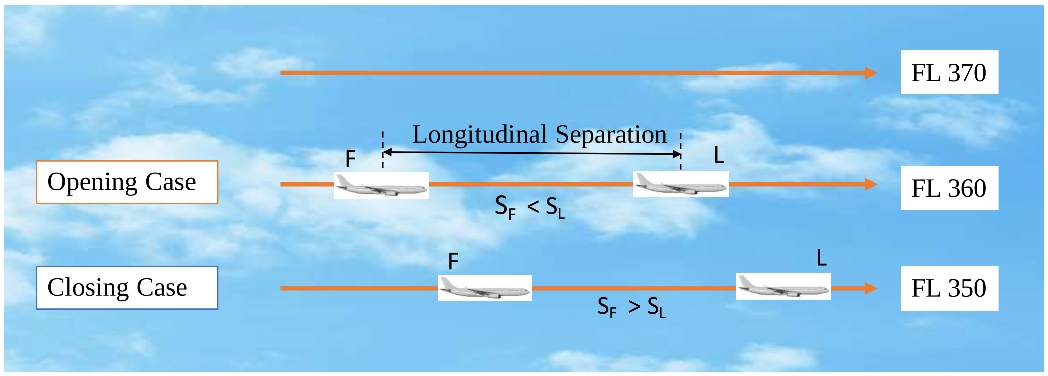


Figure 2-8 Illustration of Opening Case and Closing Case.

There are two cases of aircraft in-trail performance procedures. The opening case occurs when the leading aircraft is faster than the following aircraft. The closing case is a situation when the leading aircraft is slower than the following aircraft, making the separation smaller over time. These cases are shown in Figure 2-8.

The estimation of the minimum longitudinal separation depends on the track length. A long track (e.g. 2,400 nm) requires longer flight times than a shorter one. Equation Eq. 2-1 needs revisions to make sure the separation between two aircraft in the closing case is equal or above the minimum longitudinal separation at the entry point. For example, if the leading aircraft has a 0.81 Mach number and the following has 0.83 Mach number, assume the cruise altitude is 35,000 ft. in a 2,400-nm long track. The

[6] In aviation field, Mach number is the ratio of aircraft's true air speed and the sound speed at the given altitude.

speed of sound at 35,000 ft. is 574 knots. From Eq. 2-1, the minimum longitudinal separation is $\max(5 - 300 \times (0.81 - 0.83), 5) = 11 \ \text{min}$. Now assume the leading aircraft enters the OTS track at time 0 min and the following enters at time 11 min. The times when they exit the 2,400-nm track are 5.16 hrs. and 5.22 hrs., respectively. (See calculations below.)

Leading aircraft's exit time: $0 + \frac{2400}{574 \times 0.81} = 5.16 \ hrs.$

Following aircraft's exit time: $\frac{11}{60} + \frac{2400}{574 \times 0.83} = 5.22 \ hrs.$

The time difference at the exit point is 3.6 minutes which clearly represents a violation of the 5 minute rule.

$$(5.22 - 5.16) \times 60 = 3.6 \ \text{min} < 5 \ \text{min}$$

To address this issue, we use a simple modification by adding to the in-trail (or headway) separation using a simple linear regression model. The coefficients of the linear model are derived externally to the GO Model. The additional headway takes track distance, speed difference and direction into account to enlarge the separation in trail. The equations integrate a 20-second buffer (see Eq. 2-2 and Eq. 2-3).

$$headway_E = (k_1 + k_2 * TD)\left(\frac{\Delta Mach}{0.01}\right), \text{for eastbound tracks} \qquad \textbf{Eq. 2-2}$$

Where $headway_E = east\ additional\ headway\ (min)$, $k_1 = 0.333$, $k_2 = 0.0018$, $TD = track\ distance\ (nm)$, $\Delta Mach = delta\ Mach\ number(dim)$

$$headway_W = (k_3 + k_4 * TD)\left(\frac{\Delta Mach}{0.01}\right), \text{for westbound tracks} \qquad \textbf{Eq. 2-3}$$

Where $headway_W = west\ additional\ headway\ (min)$ $k_3 = 0.333$, $k_4 = 0.0026$, $TD = track\ distance\ (nm)$, $\Delta Mach = delta\ Mach\ number(dim)$

The application of these equations at the entry point satisfies the condition that two aircraft are separated for the complete length of the track. This method is also applied to the climb requests inside the OTS and explained in the next section.

2.3.2.3 Climb Inside OTS

Climbs inside OTS are critical procedures to save fuel in oceanic airspace. (Li, 2014). The NATSAM III contained algorithms to execute climbs inside OTS for fuel efficiency. There are two requirements to process a climb request: an ATC routine to check for conflicts and a pilot request. Pilot requests are controlled by a user-defined parameter. The pilot routine would estimate fuel consumption using the OTS exit as the target waypoint and select the candidate flight level that minimizes fuel to the exit point of the current track. To process a climb request, the ATC routine checks the minimum longitudinal separation at the newly requested flight level and flight levels the flight will traverse.

The method above works well if the climbing aircraft is slower than the leading aircraft or maintains the same Mach number. However, when a climb request involves a closing case (see Figure 2-9), this method cannot guarantee the longitudinal separation in the rest of the track. The additional headway rule described in the previous section is introduced to add the necessary separation in the climb request.

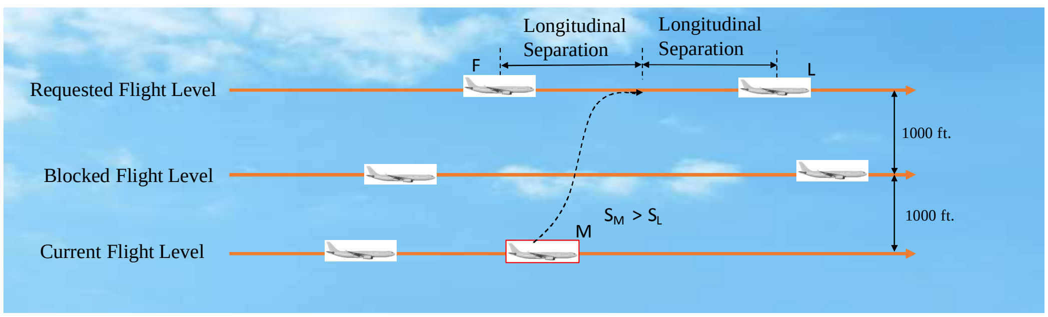


Figure 2-9 Climb Request in Closing Case.

2.3.2.4 Post-Processor of Potential Violation

A Post-Processor is an enhancement to the GO Model created by Arman Izadi, a Ph.D. student in Air Transportation Systems Laboratory at Virginia Tech. The post-processor detects potential violations by verifying the minimum separation distance between flight pairs. It checks the vertical, lateral and longitudinal separations between all the flights and defines the Closest Point of Approach (CPA) for both random and OTS flights based on their equipage level.

The post-processor was used in this analysis to check for potential violations involving OTS flights. The post-processor was also used to check cases of insufficient longitudinal separation in climb requests. The number of potential violations is an important metric to evaluate the accuracy of GO Model.

Figure 2-10 shows an example of a potential separation violation close to the minimum separation standard. This type of violation is called a 'glancing violation' and would normally be handled by the air traffic controller.

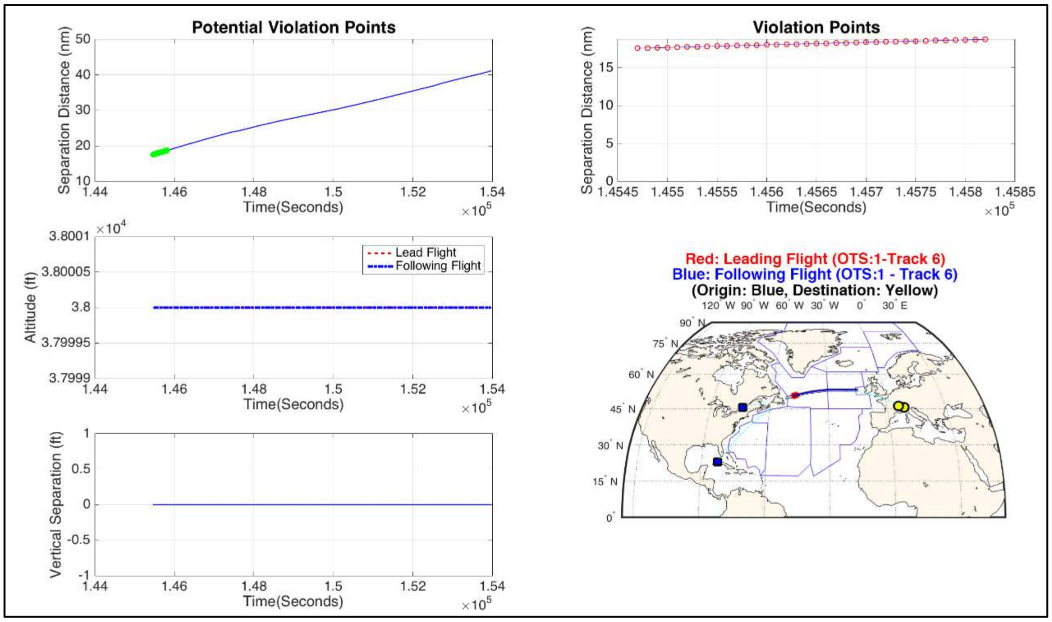


Figure 2-10 Example of 'Glancing' Potential Violation.

2.3.3 Simulation Scenarios

With the development and integration of the algorithms described in previous sections, the GO Model is a more intelligent, realistic, and accurate simulation tool to evaluate the feasibility and benefits of future OTS flight assignment strategies.

Three degrees of freedom are tested in the model: a) changes in track assignment and b) traffic demand, and c) the minimum longitudinal separation between flights. The first degree of freedom means that in the same year, we assess the performance among different strategies to assign flights in the OTS using metrics of fuel consumption and level of service. The second degree of freedom is to forecast air traffic in future years and further evaluate advantages among OTS assignment strategies. A table of simulation scenarios is shown in Table 2-4.

To simulate flights in the years 2020 and 2040, tracks are manually modified to the closest OTS track and spaced ½ degree (see Figure 2-11). The demand in future years are cloned from the June 25, 2016 seed day. The June 25 seed day has a high volume of traffic in the year. The separation standard to each scenario is applied to all aircraft, i.e., we assume all aircraft are equipped with the necessary avionics that could follow the minimum standards.

Table 2-4 Simulation Scenarios Tested.

Scenarios	Seed Day	Separation (Lateral / Longitudinal)	Assignment Routine			Traffic Demand		
			FLs_1_3	FLs_1_1	Full Cost Matrix	Year 2020	Year 2030	Year 2040
Baseline	June 25, 2016	*23 nm / 5 min (~40 nm)	×			×		
Scenario 1	June 25, 2016	23 nm / 2 min (~15 nm)	×			×		
Scenario 2	June 25, 2016	23 nm / 2 min (~15 nm)		×		×		
Scenario 3	June 25, 2016	23 nm / 2 min (~15 nm)			×	×		
Scenario 4	June 25, 2016	23 nm / 5 min (~40 nm)	×					×
Scenario 5	June 25, 2016	23 nm / 2 min (~15 nm)	×					×
Scenario 6	June 25, 2016	23 nm / 2 min (~15 nm)			×			×

Notes: (*) Assume RLatSM/RLongSM implementation by end of 2017 and track spacing is ½ degree.

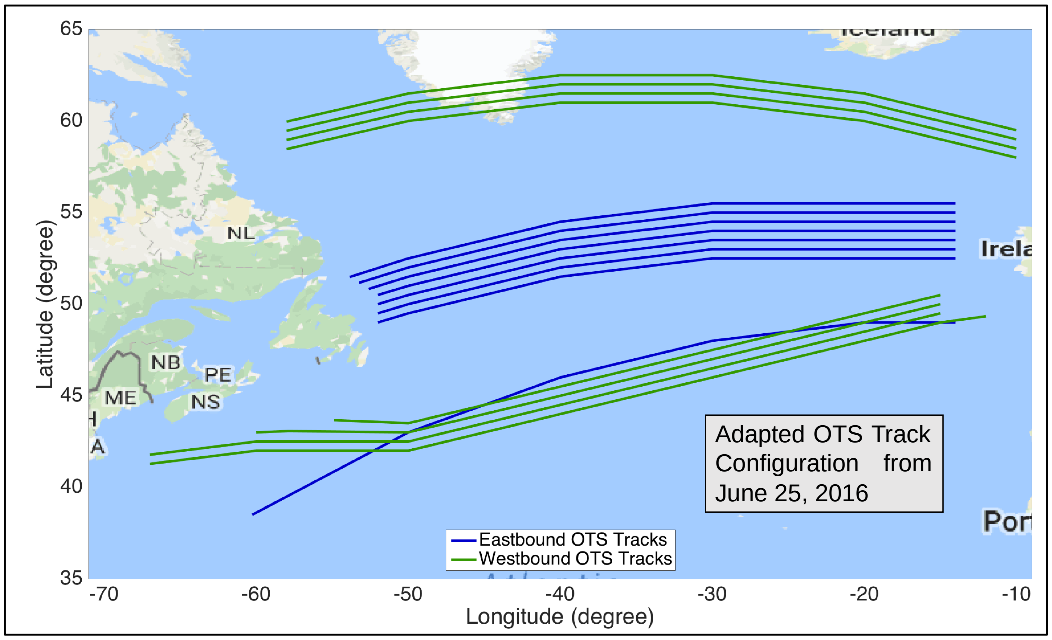


Figure 2-11 OTS Tracks with Reduced Lateral Separation (1/2 Degree Spacing).

Baseline

Baseline scenario uses a traffic demand in 2020 extrapolated from the June 25, 2016 seed day. The baseline separation is 23 nm laterally and 40 nm longitudinally. The baseline case considers the current OTS assignment concept where aircraft are assigned one flight level above or up to three flight levels below if the optimal cruise flight level is not available.

Scenario 1

Scenario 1 assumes the same assignment routine and demand year as the baseline with the addition of satellite ADS-B avionics. The satellite ADS-B may allow a longitudinal separation of 15 nm. The setting aims to compare the benefit between avionics ADS-C and satellite ADS-B in 2020.

Scenarios 2 and 3

Scenario 2 and 3 assume the same avionics equipage of satellite ADS-B and year 2020 demand but they vary the OTS assignment concept of operation. Scenario 2 assigns aircraft to one flight level above or below if the optimal cruise flight level is not available. Scenario 3 assigns the next best flight level derived from the cost matrix. Scenarios 1, 2, 3 compare fuel and level of service benefits across different assignment concepts of operation.

Scenario 4

Scenario 4 is similar to the baseline case but uses a higher demand representative of year 2040. The minimum longitudinal separation is 40 nm and the OTS track assignment concept is FLs_1_3. The demand in this scenario is generated using the June 25, 2016 seed day but the number of flights is nearly twice that in 2020.

Scenarios 5 and 6

Scenario 5 is to estimate the integrated benefit as it uses the 15-nm longitudinal separation standard. It applies the OTS track assignment concept of FLs_1_3. Scenario 6 assumes the same equipage level but operates Full Cost Matrix assignment concept of operation. The purpose of these settings is to compare the benefits among Scenarios 4, 5, 6 for 2040 and those obtained with lower traffic loads in the year 2020.

2.4 Results

2.4.1 System Running Time

The seven scenarios presented in Table 2-4 were simulated using Matlab version 2016b in a 64-bit desktop computer. The simulation time increases with the growing flight demand. Table 2-5 shows an example of the execution time for each section of the model using year 2020 demand.

Table 2-5 Global Oceanic Model Execution Time per Section.

GO Model Section	Execution Time (min)
Enhance TFMS Data with OAG	1.00
OTS Processor	0.14
Wind Processor	0.08
Demand Processor	318.24
Flight Operations Simulator	51.16
Validation	13.79
Summarize Results	0.14
Total	384.55

The most time-consuming process in the GO Model is ‘Demand Processor’ since it includes the generation of wind optimal flight plans for each flight. The model runs a demand for three consecutive days. The first day is used to ‘warm up’ the system and the final day to assure that the flights in the middle day are complete. Statistics are collected for the middle day.

2.4.2 Fuel Consumption

Several metrics are used to assess the performance of the system. These include: a) average fuel consumption, b) average travel time and c) level of service metrics. The comparison of metrics is done in two parts. One is to compare scenarios in terms of separation minima and assignment routine in the same year. The other is to compare the differences of benefits across multiple years.

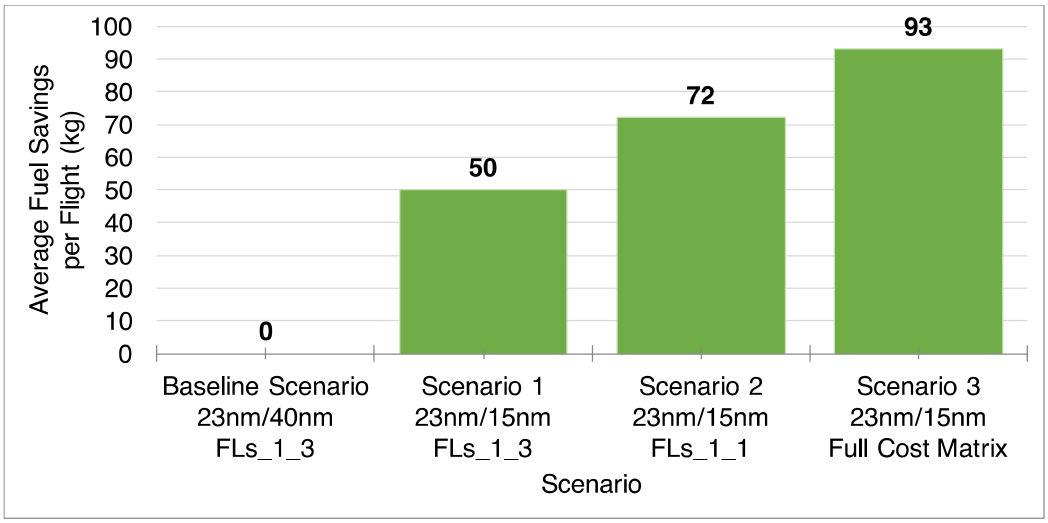


Figure 2-12 Potential Fuel Benefits for OTS Flights in 2020.

Figure 2-12 presents the fuel savings of Scenarios 1, 2, 3 compared to Baseline Scenario in 2020. With the same assignment routine, the reduced longitudinal separation of 15 nm could save on average 50 kilograms of fuel per flight. To better assign OTS flights at entry point of the system, the routine of assigning 1,000 ft. below has an average benefit of 22 kilograms per flight more than the one of assigning up to 3,000 ft. below. Using the Full Cost Matrix OTS assignment concept of operations, a savings of 93 kilograms per flight is observed compared to the baseline scenario.

The conversion of fuel consumption to U.S. dollar values was performed based on the current world average jet fuel price that is equal to $435.9/metric tons (IATA, 2017). Figure 2-13 presents the annual fuel benefit per scenario compared to the baseline scenario in 2020. For a demand of 996 flights in 2020, the annual fuel benefit of the Full Cost Matrix OTS assignment concept of operations is 14.74 million dollars.

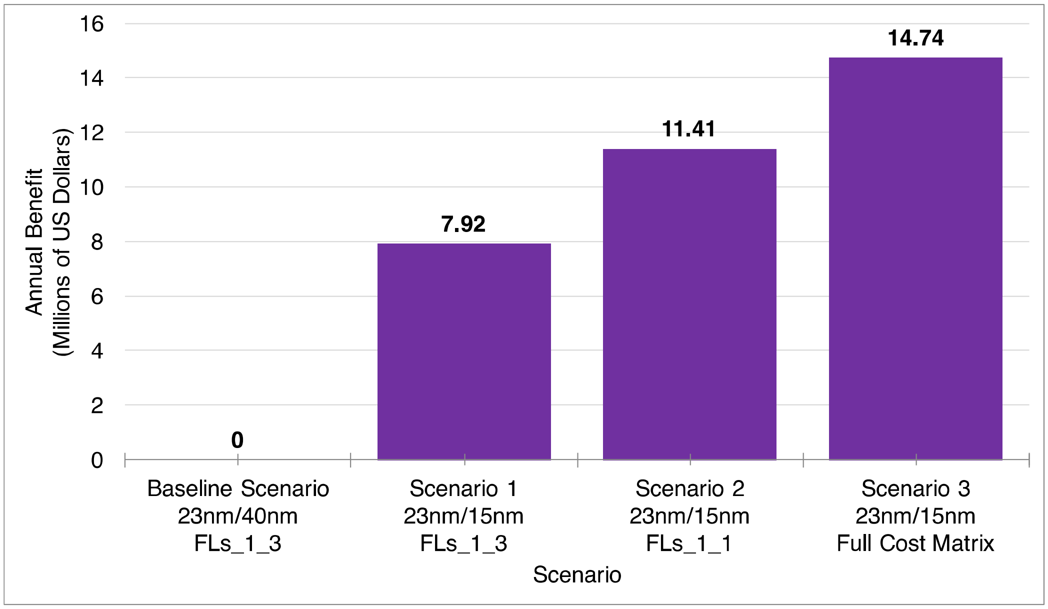


Figure 2-13 Annual Fuel Benefit for OTS Flights in 2020.

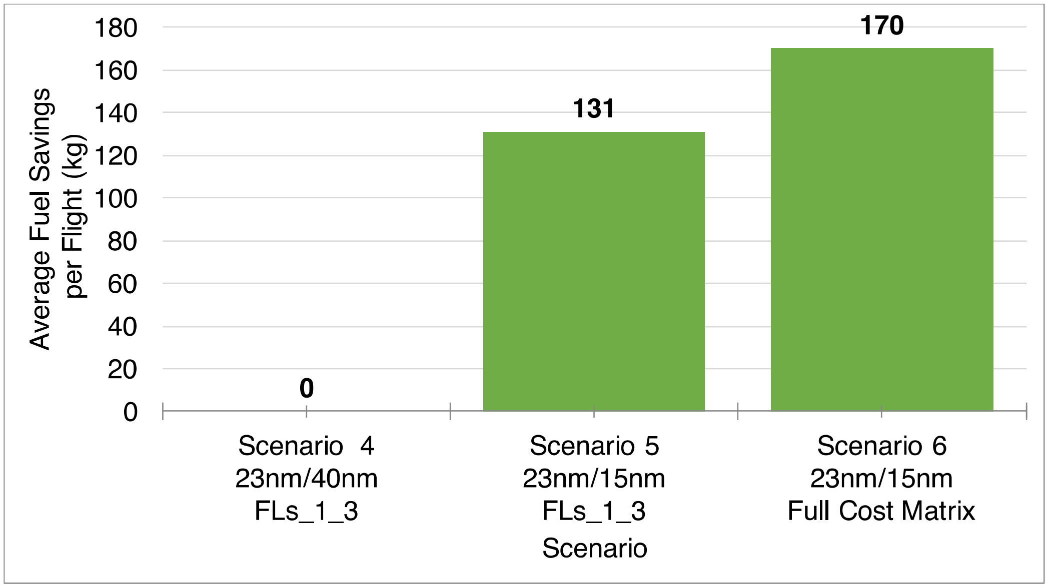


Figure 2-14 Potential Fuel Benefits for OTS Flights in 2040.

Scenario 4 could be considered as the 'future' baseline scenario in 2040. Scenario 6 is defined as the optimal with reduced longitudinal separation and Full Cost Matrix assignment concept of operations. The difference between these two scenarios are shown in Figure 2-14. With a higher demand in 2040, the benefit of using reduced longitudinal separation and Full Cost Matrix assignment concept of operation increases to 170 kilograms per flight. The annual fuel benefit in 2040 with 1855 flights flying in OTS will increase to 50.17 million dollars using Full Cost Matrix assignment concept of operation (see Figure 2-15). It also indicates that the usage of ADS-B applying 15 nm separation could save 131 kilograms of fuel per flight in 2040. The application of reduced longitudinal separation down to 15 nm will save 38.66 million dollars annually in 2040.

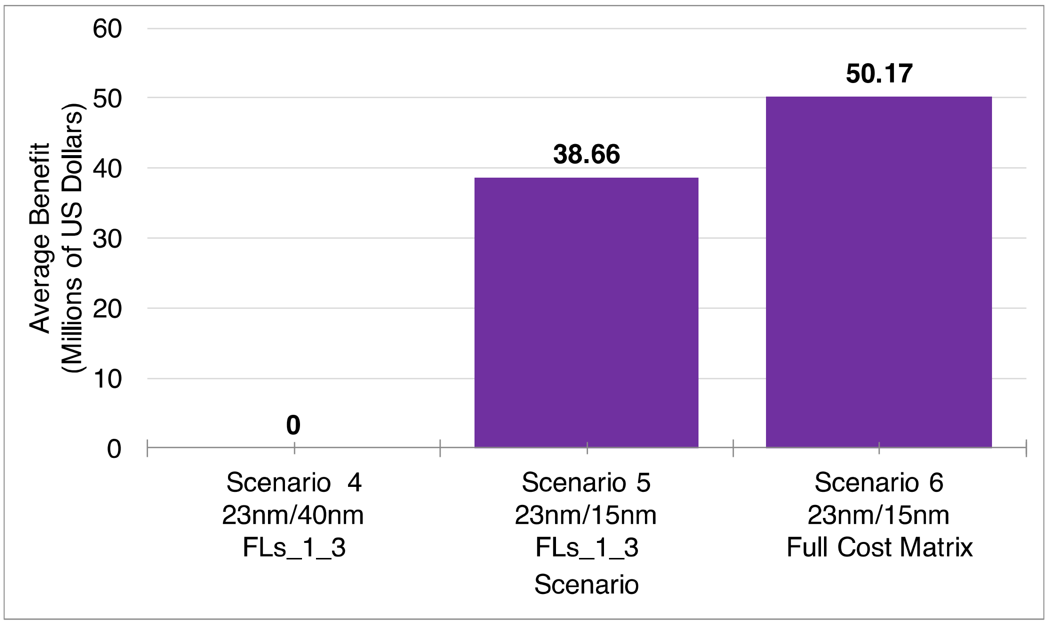


Figure 2-15 Annual Fuel Benefit for OTS Flights in 2040.

2.4.3 Travel Time

The results of the travel time metric are presented in this section. First, the model simulates the whole trajectory of flights flying across North Atlantic airspace. However, the model does not account for the delays near origin and destination airport since this study focuses on assessing the performance of NAT Organized Track System.

The results show that the baseline scenario has the least average travel time than the other scenarios for year 2020. Figure 2-16 presents the travel time differences of Scenarios 1, 2, 3 compared to Baseline Scenario. The explanation of this trend is that with smaller separation minimum, it is more possible for aircraft to execute more step-climbs. The strength of the jetstream is less at higher altitudes. If more aircraft climb, they have less advantage of a tailwind. The climbing performance of scenarios in year 2020 is shown in Figure 2-17. The baseline scenario has the lowest climb performance. All other assignment concepts provide improvements to the percentage of aircraft that climb.

In year 2040, Scenario 4 has the least average travel time than Scenario 5 and 6 (see Figure 2-18). The average travel time difference between Scenario 4 and 5 (0.27 min) in year 2040 is larger than the one between the Baseline Scenario and Scenario 1 (0.15 min). This can be attributed to the climb performance difference between Scenario 4 and 5 in year 2040 (46%, see Figure 2-19) is much higher than the one between Baseline Scenario and Scenario 1 in 2020 (29%). The higher the altitude is, the less the strength of the jetstream is.

It is noted that the climb performance in 2040 decreases about 20% from the scenarios with same separation minima and assignment concepts for 2020 OTS traffic.

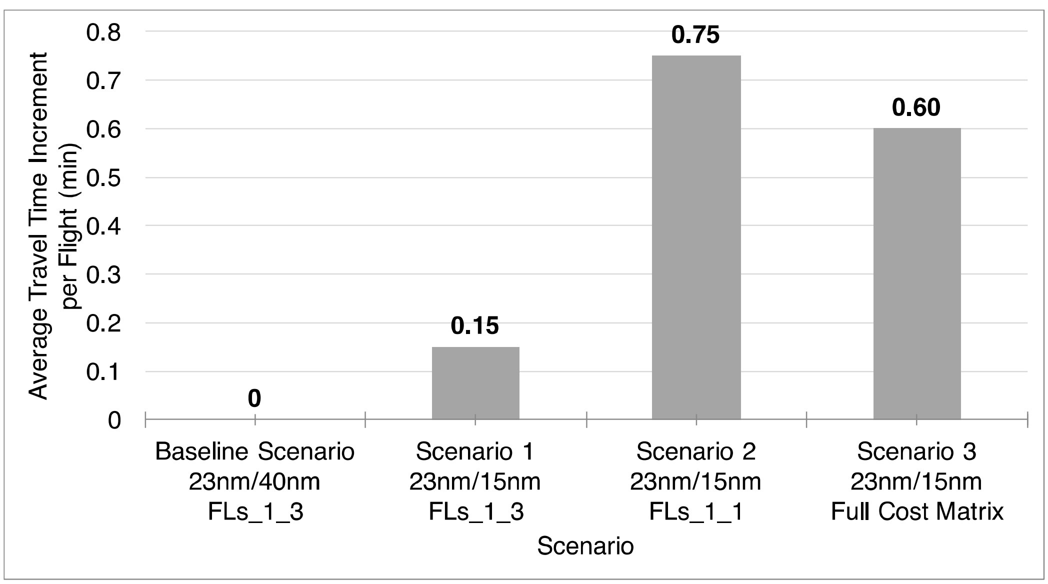


Figure 2-16 Travel Time Differences for OTS Flights in 2020.

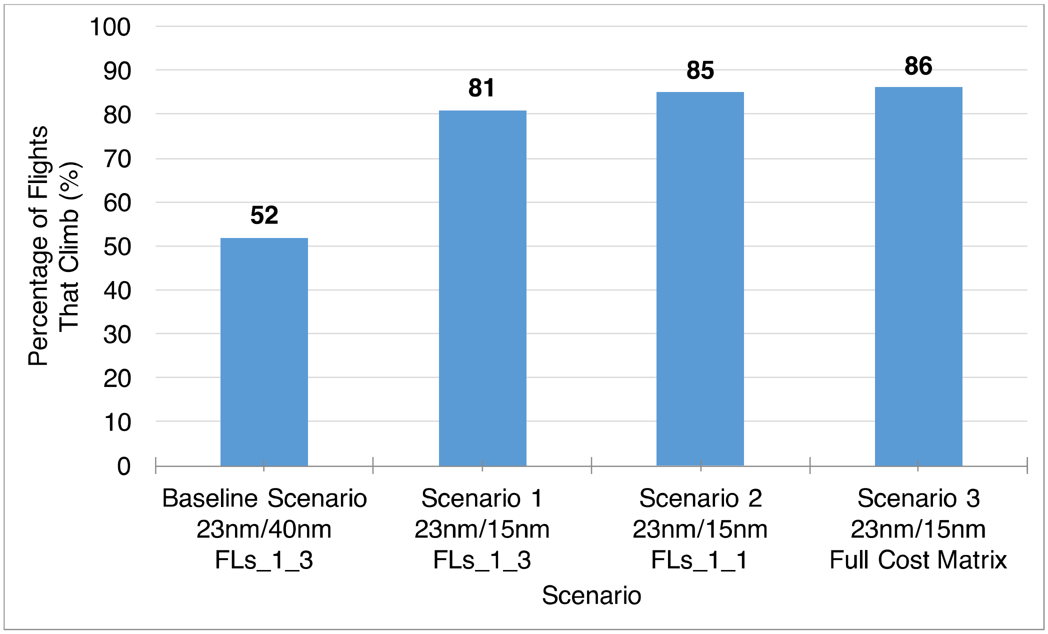


Figure 2-17 Climbing Performance for OTS Flights in 2020.

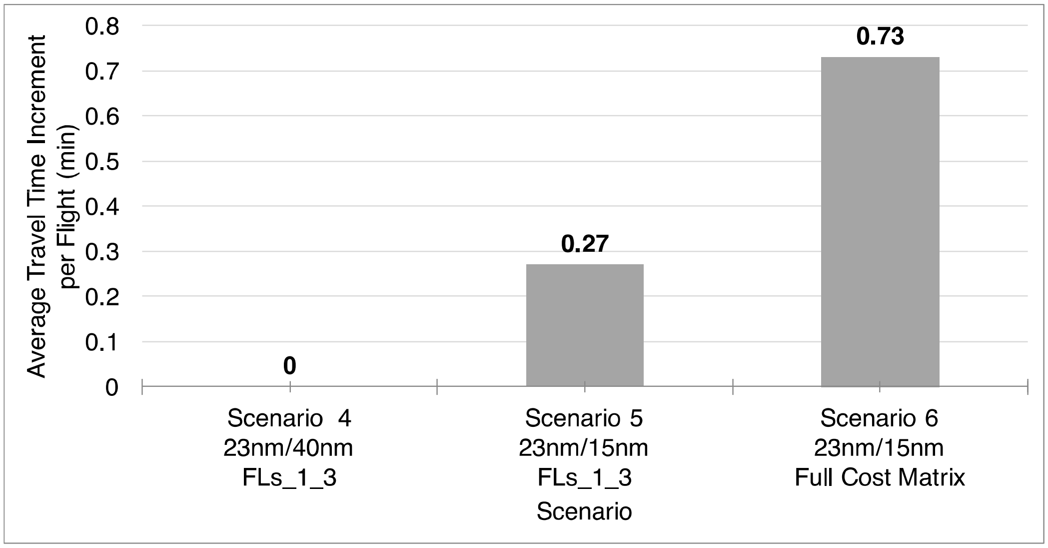


Figure 2-18 Travel Time Differences for OTS Flights in 2040.

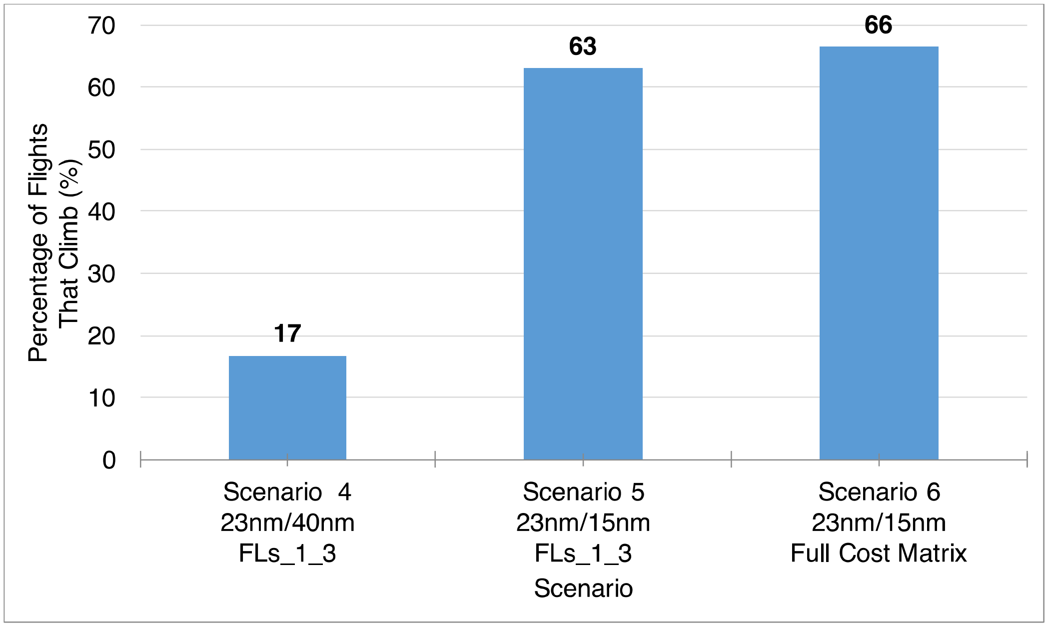


Figure 2-19 Climbing Performance for OTS Flights in 2040.

2.4.4 Level of Service

The level of service in the OTS is defined as the percent of flights that obtain their optimal requested track and flight level.

As explained in Section 2.3.2.1, a maximum operational altitude is calculated based on aircraft weight. In general, aircraft requests to fly at the ideal flight level is based on cost matrix. However, the ideal flight level is generally higher than maximum operational altitude because the calculations of wind are different in the calculation of cost matrix and the simulation. We consider aircraft obtain their requested flight level if the assigned flight level is equal to maximum operational altitude, or it is equal or higher than ideal flight level (in the case ideal flight level is lower than maximum operational altitude).

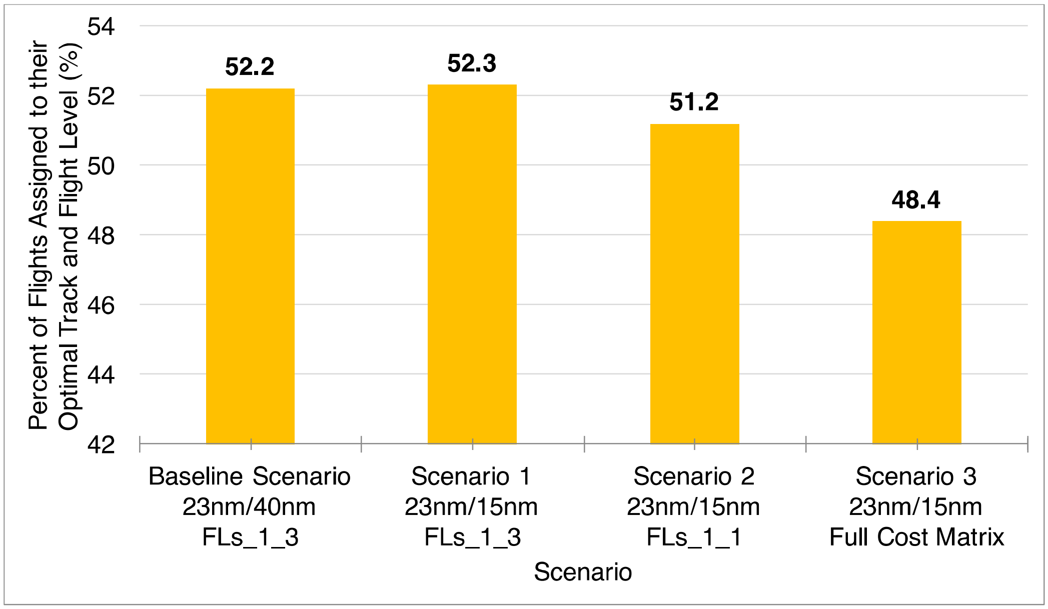


Figure 2-20 Level of Service of Scenarios for OTS Flights in 2020.

Figure 2-20 illustrates the effect of OTS assignment concept of operation in the Level of Service metric. The Full Cost Matrix scenario has the lowest percentage of flights assigned to the optimal flight level in ideal track since this scenario assigns aircraft to higher flight level regardless of the ideal track. In other words, the Level of Service is sacrificed for fuel savings. It also indicates a large number of flights that are assigned to alternative tracks. The number of flights assigned to their optimal flight level in ideal track in scenario 'FLs_1_1' is less than that in 'FLs_1_3'. It is because flights in 'FLs_1_1' scenario are more likely to be assigned to an adjacent track and at the optimal flight level.

The Level of Service is not improved with reduced separation minimum down to 15 nm in 2020. However, 3.4% more flights (see Figure 2-21) with reduced separation achieved their requested flight levels in 2040. The reason behind this is the increment in flight demand. Although all scenarios have lower Level of Service in 2040 than 2020, Scenario 5 with reduced longitudinal separation (15 nm) gains an advantage over Scenario 4 with 40-nm separation minimum in 2040.

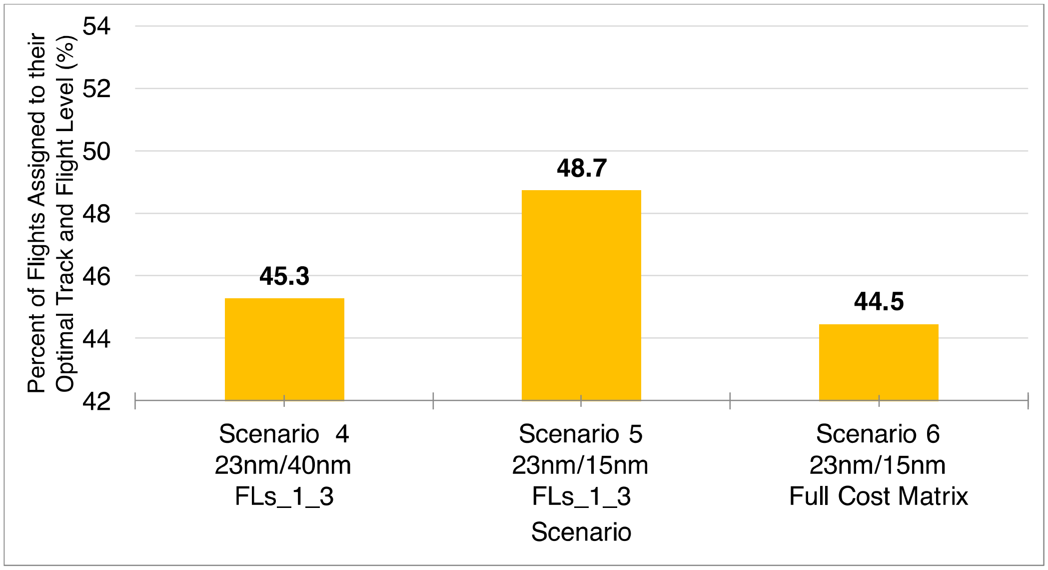


Figure 2-21 Level of Service of Scenarios for OTS Flights in 2040.

A similar analysis was performed for flights that were not assigned the flight levels that they requested. In the 'FLs Heuristics' assignment routine, priority was given to flights levels bounded by the 1000-3000 foot range of the heuristic. If these flight levels were still unavailable, flights could descend 5,000 ft. below their optimal flight level before changing track.

The analysis shows that the 'FLs_1_1' and the Full Cost Matrix assignment heuristic algorithms have a better performance in assigning flight levels assignments in 2020 (see Figure 2-22). Using the Full Cost Matrix assignment, more than 80% flights could achieve their requested flight level at the entry of OTS tracks. In addition, the flight level assignment routines between usage of ADS-C and ADS-B do not have a significant difference, which correlates with the performance of Level of Service in Figure 2-20. In 2040, all scenarios have a slight reduction in the percentage of 'Achieved Requested FL' (see Figure 2-23). The 'Achieve Requested FL' means that aircraft obtain their requested optimal flight level at the entry of OTS tracks. The performance of the 'Full Cost Matrix' scenario shows a 20% improvement over other track assignment algorithms.

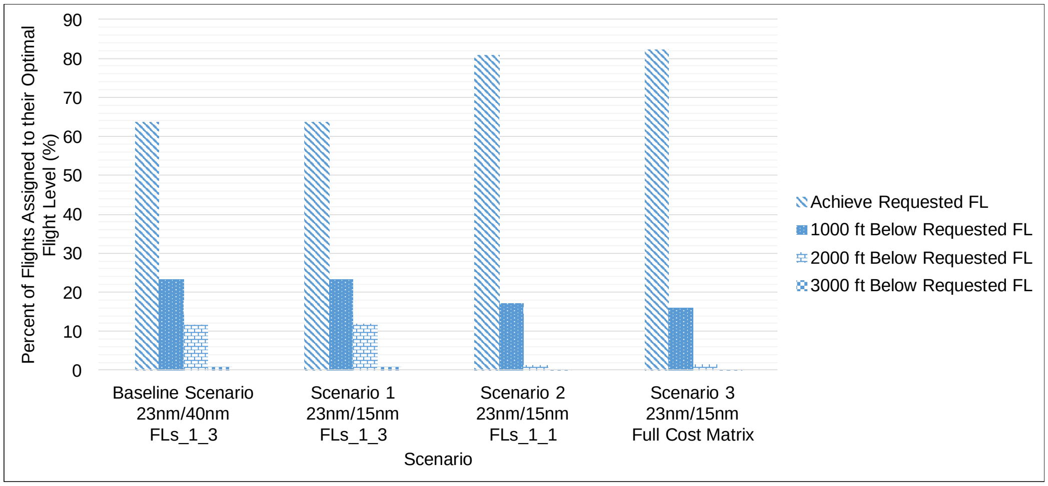


Figure 2-22 Flight Level Assignment Among Scenarios in 2020.

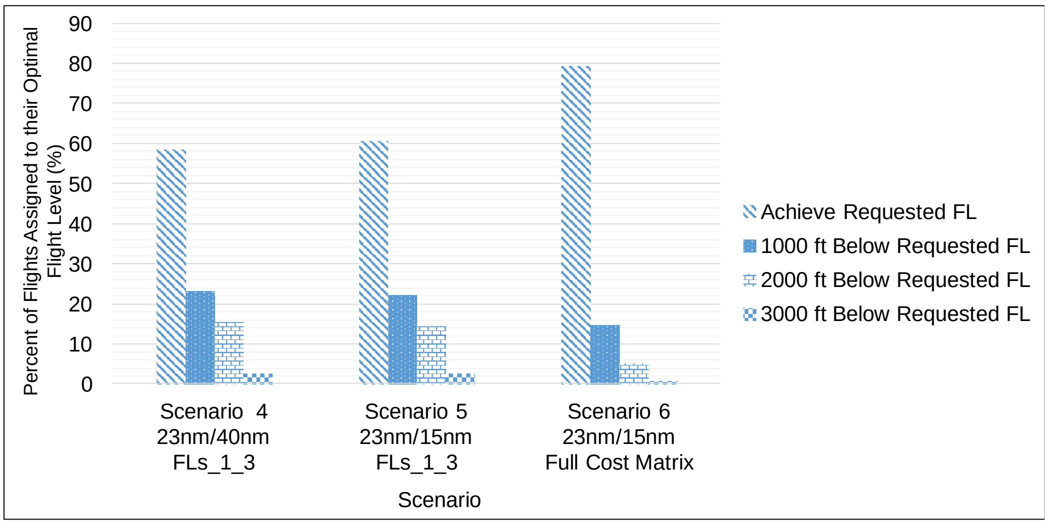


Figure 2-23 Flight Level Assignment Among Scenarios in 2040.

Another analysis is performed to assess the flight level assignment for a specific aircraft Boeing 767-300. Figure 2-24 presents the flight level assignment of Boeing 767-300 in scenario 1 (using FLs_1_3 and 23 nm/15 nm separation). Figure 2-25 shows the flight level assignment for the same aircraft type while applying the Full Cost Matrix assignment concept of operation. The Full Cost Matrix assignment concept of operation improves the aircraft altitude assignment by 132 ft. on average above the current assignment concept of operation.

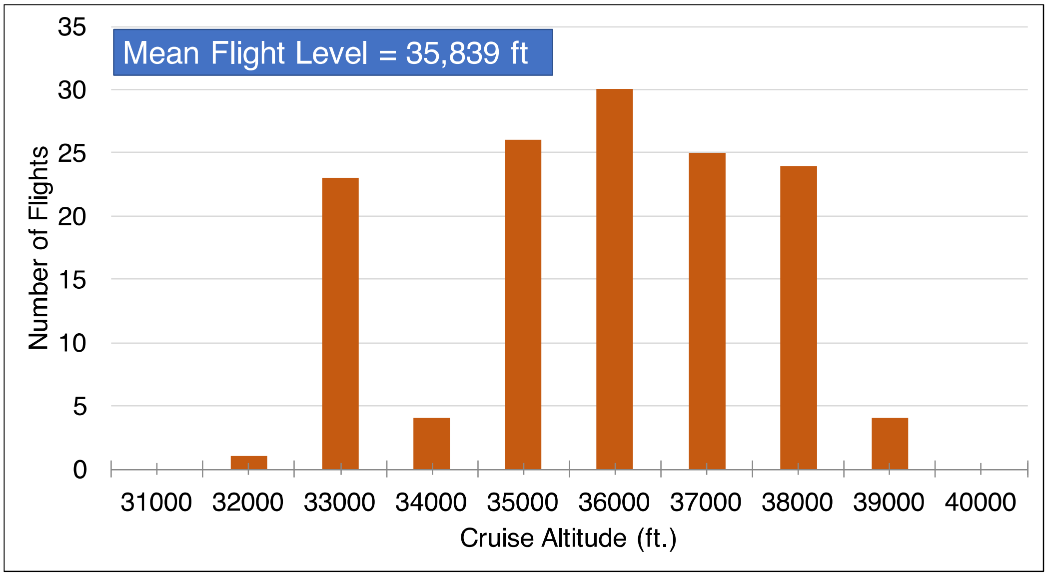


Figure 2-24 Flight Level Assignment of Boeing 767-300 Using Assignment Routine FLs_1_3 and Separation 23 nm/15 nm.

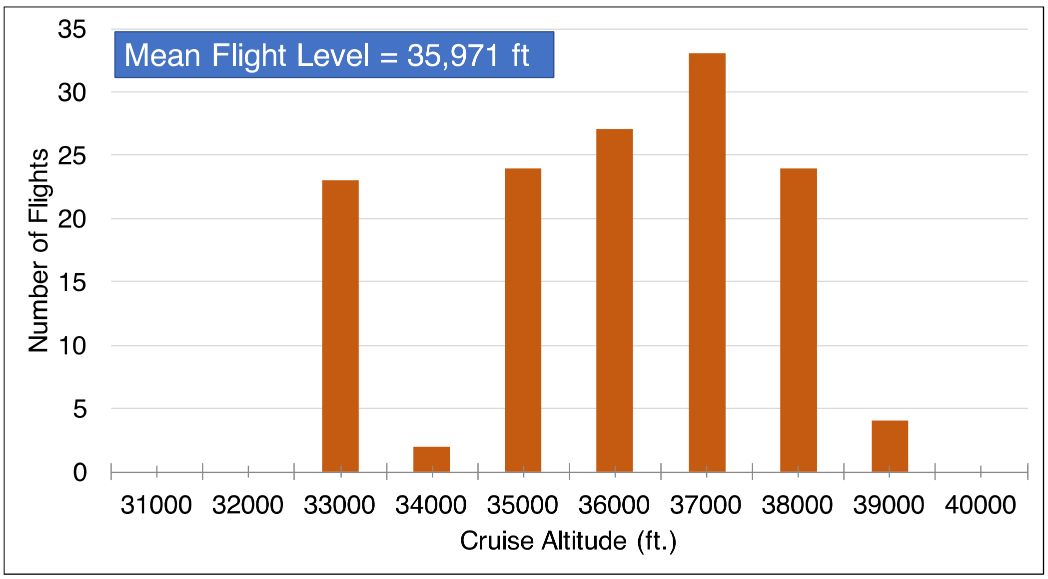


Figure 2-25 Flight Level Assignment of Boeing 767-300 Using Assignment Routine Full Cost Matrix and Separation 23 nm/15 nm.

CHAPTER 3 Validation

3.1 Basic Aircraft Performance

This section presents a simple validation of the aircraft performance in the GO Model. Figure 3-1 and Figure 3-2 present the performance of Boeing 767-300 from the GO Model and actual airline data, respectively. Figure 3-3 and Figure 3-4 show the performance of Airbus 330-300 from the GO Model and airline data, respectively. Both aircraft are twin engine, wide body aircraft. The relation between route distance and fuel consumption is linearly increasing. Compared with the actual fuel burn data provided by A4A and IATA, there is about 2,000 kilograms' fuel burn discrepancy between the GO Model and actual airline data for the Boeing 767-300 aircraft. And the discrepancy for Airbus 330-300 is 2,500 kilograms. The reasons for the discrepancies are as follows:

1) The GO Model does not account for terminal area affects which add to the fuel burn for busy Origin Destination pairs.
2) The GO Model does not account for taxi-in and out of the airports.
3) Some of the non-linear effects of real aircraft aerodynamics are not captured. For example, BADA 3.13 uses a parabolic drag polar. In real aircraft operations, the high-speed drag rise is beyond parabolic.
4) The airline data consists of 79 days while the model simulates three consecutive days' flight operations using the same track and wind data. As the track and wind data may vary greatly day by day, the variety would result in the different fuel consumption for the same Origin Destination pair.

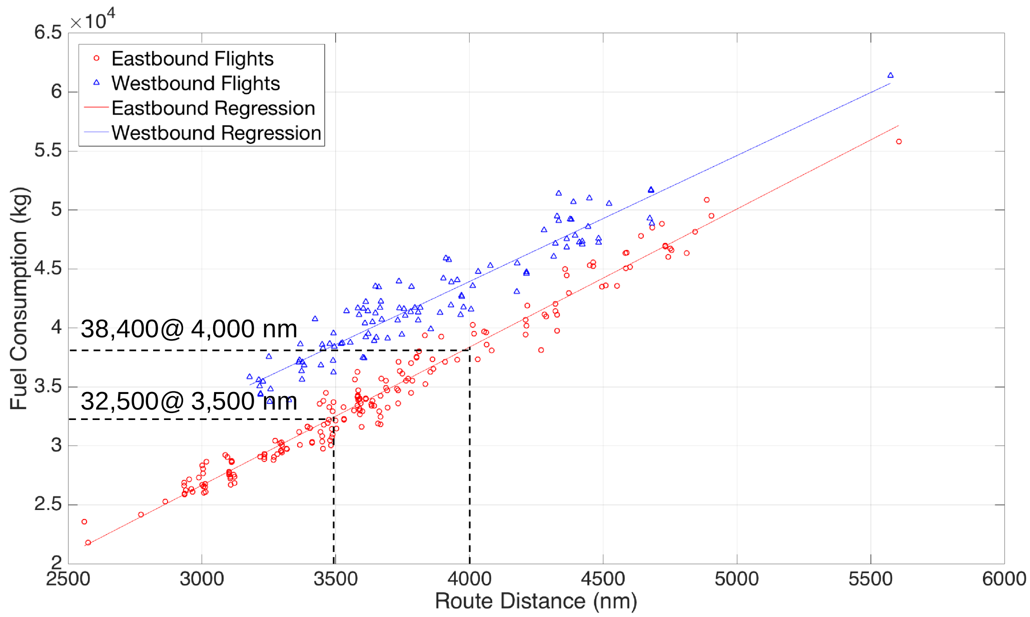


Figure 3-1 Performance of Boeing 767-300 Aircraft from the GO Model.

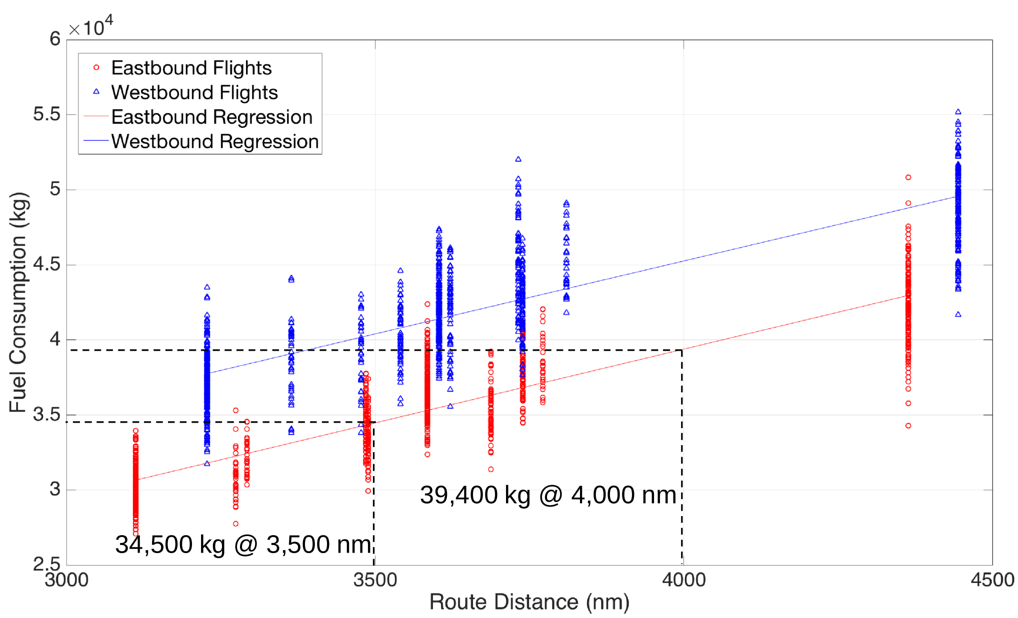


Figure 3-2 Actual Performance of Boeing 767-300 Aircraft from Airline Data (A4A).

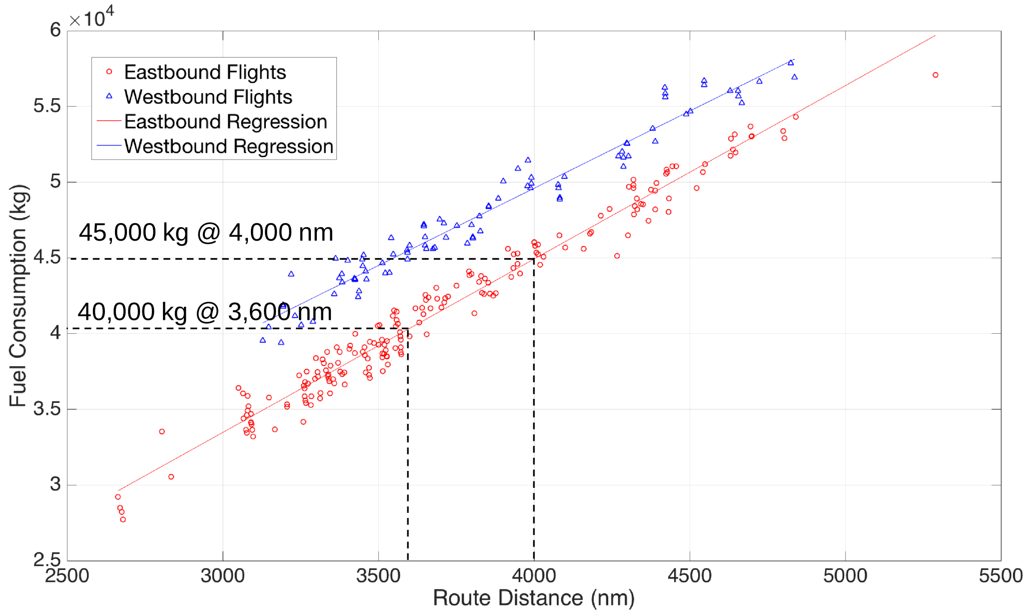


Figure 3-3 Performance of Airbus 330-300 Aircraft from the GO Model.

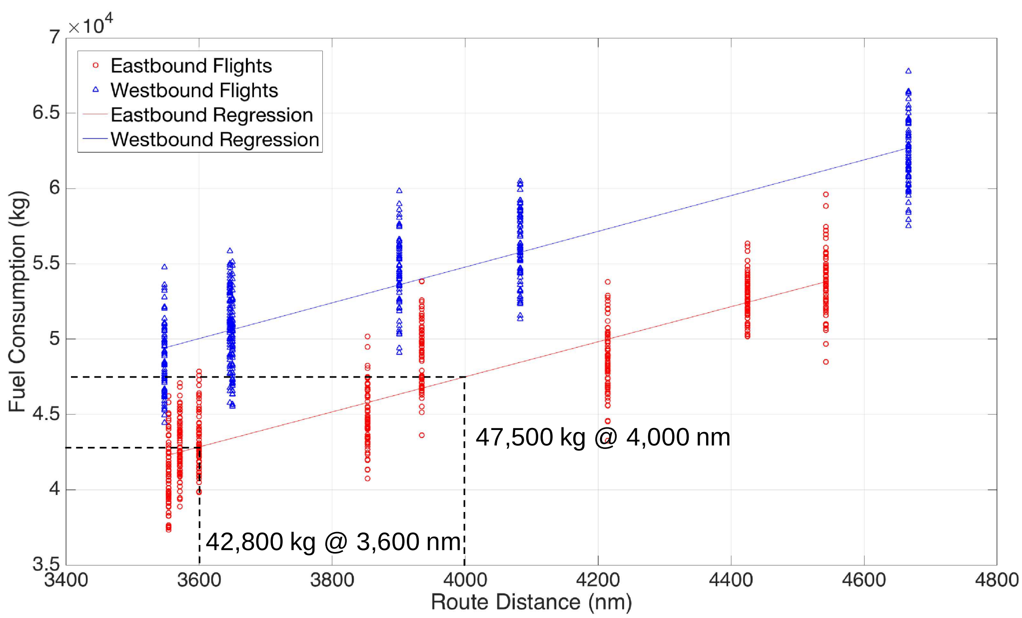


Figure 3-4 Actual Performance of Airbus 330-300 Aircraft from Airline Data (A4A).

3.2 Track Loading and Number of Conflicts

The track loading is another metric to validate the model. As seen in Figure 3-5 and Figure 3-6, the numbers of flights in most tracks double in 2040. With a higher utilization rate, a higher number of potential violations occur. Considering the lower Level of Service in 2040, we may conclude that the current track system will not be able to handle well such a large number of flights in the future.

On the other hand, after many improvements, the percentages of potential violations are less than 4%, which is an acceptable system error rate. It indicates the GO Model is reliable to simulate operational procedures to generate results.

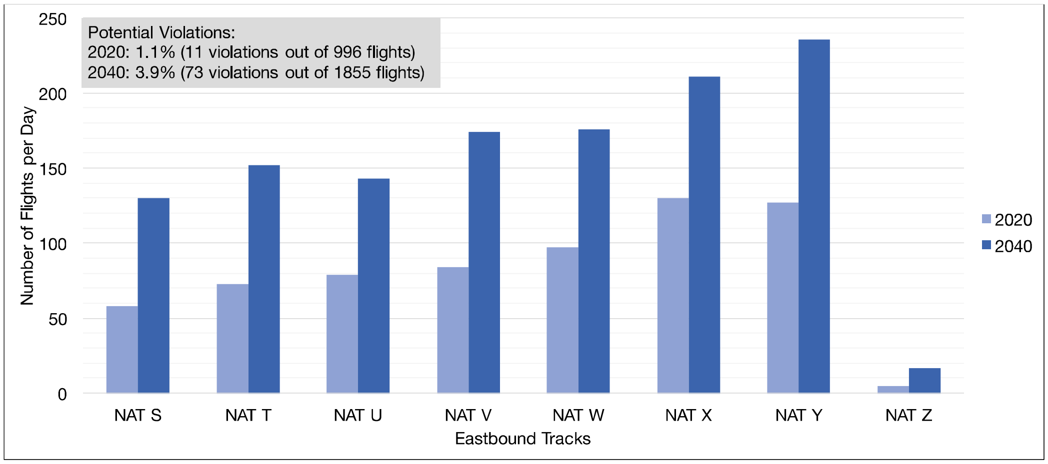


Figure 3-5 Eastbound Track Loading for OTS Flights in 2020 and 2040.

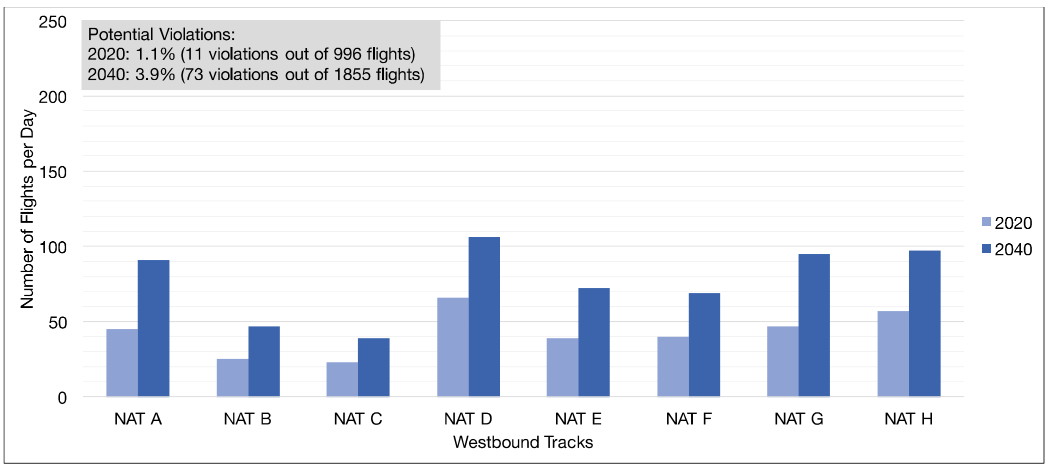


Figure 3-6 Westbound Track Loading for OTS Flights in 2020 and 2040.

CHAPTER 4 Conclusions

This thesis developed and integrated important improvements to the Global Oceanic Model. The modifications to OTS assignment algorithms, improvements to headways logic for in-trail separation, and validation of results presented make the GO Model more realistic and accurate for decision making. The current version of GO Model is able to simulate OTS flights with an error rate less than 4%.

The thesis assesses the near-term and long-term operational performance of the North Atlantic Organized Track System using several metrics that include: fuel consumption, travel time and level of service. All scenarios simulated adopt the same OTS configuration, in which tracks are laterally separated by ½ degree (~23 nm). Improved communication and surveillance performance of aircraft avionic systems could make possible reductions in longitudinal separation from 40 nm to 15 nm in the future. In 2020, the fuel savings of reduced separation minima (~15 nm) are estimated to be 50 kilograms per flight. This number will increase to 131 kilograms per flight in the demand increases by 125%. The results presented suggest the benefits of improved surveillance to accommodate a large number of flights. However, as the demand increases by 125% in 2040, the percent of aircraft assigned to their optimal track and flight level decreases by 5% and the percentage of potential violations increases by 2.8%. This indicates that the current OTS configuration could not accommodate the future demand and more tracks may be needed.

An analysis of OTS assignment concept of operations has been studied to understand possible benefit by changing current operational procedures. The results show that Full Cost Matrix assignment concept of operations performs better than the current method of assigning aircraft to climb 1,000 ft. or descend up to 3,000 ft. The fuel savings of the advanced concept is about 40 kilograms per flight.

CHAPTER 5 Recommendations

The Global Oceanic Model has been improved to help estimate fuel and level of service benefits in the NAT OTS derived from improved operational procedures. More validation metrics have been added to the model to assist researchers and decision makers to interpret the GO Model results.

All scenarios in this thesis use the same OTS configuration and we conclude that this configuration could not accommodate the large flight demand expected in the year 2040. With a larger demand in 2040, the validation results of GO Model present more potential violations. This could have safety implications in the operation of the NAT OTS. Future air navigation service providers may focus on exploring an optimal OTS configuration in terms of wind, weather and demand set.

The concept of operations using the Full Cost Matrix assignment algorithm has considerable benefits. However, there are technical requirements for its implementation, such as a better information sharing between aircraft and the ATC system. The Full Cost Matrix assignment routine may increase the workload of controllers since more aircraft will switch tracks to obtain a higher flight level in order to save fuel. More communications between air traffic controllers and pilots may be needed to know about their preferences. A full disclosure of the cost matrix to aircraft will be needed to allow a more advanced assignment concept of operations.

REFERENCES:

Campos, N. V., Graham, T., Grimes, R., & Joyce, K. (2013). North Atlantic Data Link Mandate: Cost Impact to U.S. Commercial Operators. In *TRB 2013 Annual Meeting*.

Eurocontrol. (2015). *Base of Aircraft Data*. Retrieved from http://www.eurocontrol.int/publications/base-aircraft-data-bada-factsheet

FAA. (2014). Traffic Flow Management System (TFMS). Retrieved from http://aspmhelp.faa.gov/index.php/Traffic_Flow_Management_System_(TFMS)

FAA. (2016). Advisory Circular 90-101A: Approval Guidance for RNP Procedures with AR. https://doi.org/AFS-800 AC 91-97

Federal Aviation Admistration. (2017). North Atlantic Resource Guide for U.S. Operators.

IATA. (2017). Fuel Price Analysis. Retrieved from http://www.iata.org/publications/economics/fuel-monitor/Pages/price-analysis.aspx

ICAO. (2016). *North Atlantic FIR Traffic Forecast*.

ICAO. (2017). North Atlantic Operations and Airspace Manual.

Li, T. (2014). *General Aviation Demand Forecasting Models and A Microscopic North Atlantic Air Traffic Simulation Model*. Virginia Polytechnic Institute and State University.

Rodionova, O., Sbihi, M., Delahaye, D., & Mongeau, M. (2014). North Atlantic Aircraft Trajectory Optimization. *IEEE Transactions on Intelligent Transportation Systems*, *15*(5), 2202–2212. https://doi.org/10.1109/TITS.2014.2312315

SKYbary. (2016). Automatic Dependent Surveillance Broadcast (ADS-B). Retrieved from https://www.skybrary.aero/index.php/Automatic_Dependent_Surveillance_Broadcast_(ADS-B)

The Flight Service Bureau. (2015). *NAT Track Structure Changes NOV 2015*.

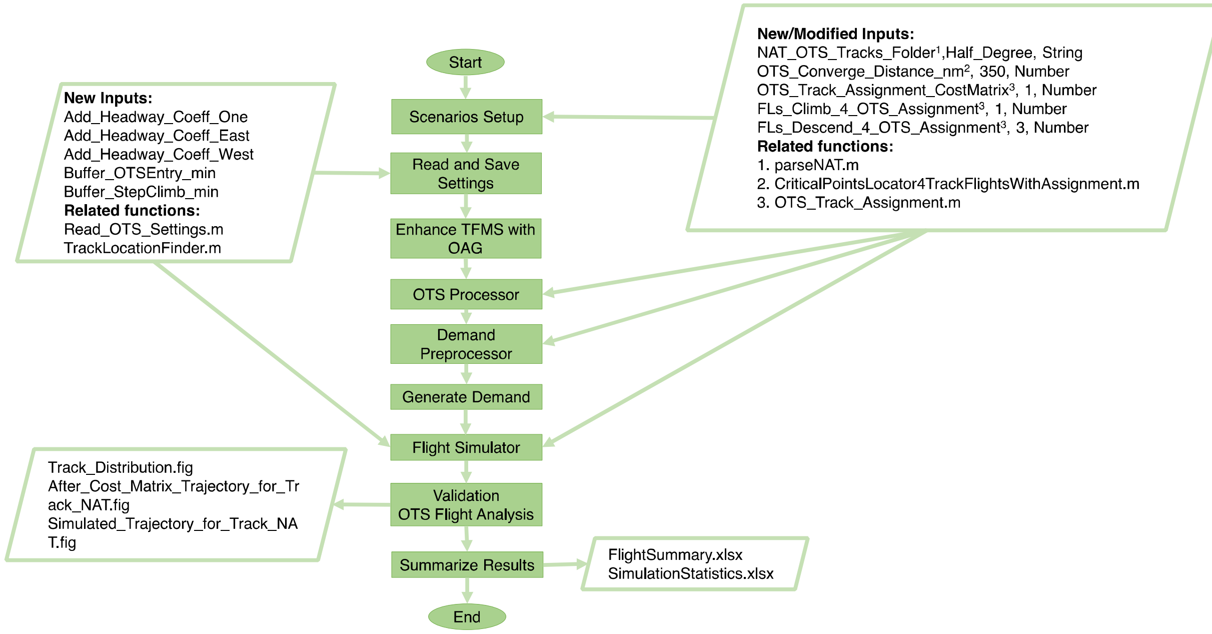
Tsikas, N. K. (2016). *Performance Assessment of Operations in The North Atlantic Organized Track System and Chicago O' Hare International Airport Noise Study*. Virginia Polytechnic Institute And State University.

U.S. Department of Transportation Federal Aviation Administration. (2012). *Flight Planning Responsibilities When Conducting Reduced Vertical Separation Minimum (RVSM) Operations*. Washington, DC.

Vena, J. (2012). Understanding FANS: ADS-C & CPDLC.

Williams, A. R., & Greenfield, I. (2006). Benefits Assessment of Reduced Separations in North Atlantic Organized Track System Read. *6th AIAA Aviation Technology, Integration and Operations Conference (ATIO)*.

APPENDIX A – Flowchart of Workflow



APPENDIX B – Source Code

The relevant codes include Read_OTS_Settings.m, TrackLocationFinder.m, OTS_Track_Assignment.m, OTS_Flights_Analysis.m, OTS_Climb_LOS_Analyzer.m.

Read_OTS_Settings.m

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Read_OTS_Settings.m
%
% This function is used to read in OTS
% Settings (separation rules) for use by the GOM model
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function Read_OTS_Settings(Install_Dir, Project_Dir, Case_Folder, Case_Name, Scenario_Name)

if (ismac == 1)
    SLASH = '/';
elseif (ismac == 0)
    SLASH = '\';
end

Output_Dir = [Project_Dir, SLASH, 'GOM', SLASH, 'Output', SLASH, Case_Folder, SLASH,
Case_Name, SLASH, 'Settings', SLASH];

% ---------------------
% Load Scenario Settings
% ---------------------
load ([Output_Dir, 'Scenario_Settings.mat']);

OTS_Settings_Dir = [Install_Dir, SLASH, 'GOM', SLASH, 'Scenarios', SLASH, 'OTS_Settings',
SLASH];

% Exit if OTS is not used
if Scenario_Settings.NAT_OTS == 0
    return;
end

% ---------------------
% Save OTS settings in struct array
% ---------------------
OTS_Settings = struct([]);

[num,~,~] = xlsread([OTS_Settings_Dir, Scenario_Settings.OTS_Settings, '.xlsx']);

Number_Of_Equipage_Levels = size(num,1);

for el = 1:Number_Of_Equipage_Levels
```

```
    OTS_Settings(el).Equipage_Level                  = num(el,1);
    OTS_Settings(el).OTS_Equipped                    = num(el,2);
    OTS_Settings(el).Longitudinal_Separation_min = num(el,3);
    OTS_Settings(el).Longitudinal_Separation_nm   = num(el,4);
    OTS_Settings(el).Longitudinal_Separation_deg =
nm2deg(OTS_Settings(el).Longitudinal_Separation_nm);
    OTS_Settings(el).Lateral_Separation_min        = num(el,5);
    OTS_Settings(el).Lateral_Separation_nm         = num(el,6);
    OTS_Settings(el).Lateral_Separation_deg        =
nm2deg(OTS_Settings(el).Lateral_Separation_nm);
    OTS_Settings(el).Vertical_Separation_ft        = num(el,7);
    OTS_Settings(el).Max_Cruise_FL                   = num(el,8);
    OTS_Settings(el).Add_Headway_Coeff_One                = num(el,9);
    OTS_Settings(el).Add_Headway_Coeff_East              = num(el,10);
    OTS_Settings(el).Add_Headway_Coeff_West              = num(el,11);
    OTS_Settings(el).Buffer_OTSEntry_min                 = num(el,12);
    OTS_Settings(el).Buffer_StepClimb_min                = num(el,13);

end % for el = 1:Number_Of_Equipage_Levels

% ---------------------
% Saving to disk
% ---------------------
save ([Output_Dir, 'OTS_Settings.mat'], 'OTS_Settings')

Output_Dir_Excel = [Output_Dir, SLASH];
if exist(Output_Dir_Excel, 'dir') == 0
    mkdir(Output_Dir_Excel);
end

FileName = [OTS_Settings_Dir, Scenario_Settings.OTS_Settings, '.xlsx'];
copyfile(FileName, [Output_Dir_Excel, 'OTS_Separation_Standards.xlsx']);

return;
```

TrackLocationFinder.m

```
function Position = TrackLocationFinder(ListOfPreviousFlight, ListOfPreviousFlightOTSRequestInfo, Default_Seperation, ...
    TargetFlightIndex, ThisEntryTime, ThisSpeedMach, RequiredFL, RequiredTrack, RLongIndicator, StepTime,Direction,TrackDist)

global OTS_Settings

% StepTime is in seconds

kOpen = -100;% factor used to calculate long seperation for opening case
kClose = -300;% factor used to calculate long seperation for closing case

ThisEntryTimeMin = (ThisEntryTime*StepTime)/60; % convert from simulation steps to mins
TargetFlightEntryTime = ListOfPreviousFlightOTSRequestInfo(TargetFlightIndex).EstimatedArrivalTime(RequiredFL,RequiredTrack);
TargetFlightEntryTimeMin = (TargetFlightEntryTime*StepTime)/60; % convert from simulation steps to mins
TargetSpeedMach = ListOfPreviousFlight(TargetFlightIndex).CruiseSpeedMachNum;
SpeedMachDiff_This_Follow_Target = TargetSpeedMach - ThisSpeedMach;
SpeedMachDiff_Target_Follow_This = -1*SpeedMachDiff_This_Follow_Target ;

%% New Approach, New Headway Rules (Considering different directions and track distances)
% ----------------------
% New Headway Rule Coefficients
% ----------------------
% k1 = 0.333
% k2 = 0.0018 (eastern flights) k2 = 0.0026 (western flights)
% Reference: Slide: New Headway Rules for the GOM Model A. Trani June 2, 2016

if Direction == 1 %Eastbound
    add_headway_min = (OTS_Settings(1).Add_Headway_Coeff_One + OTS_Settings(1).Add_Headway_Coeff_East*TrackDist)*(abs(SpeedMachDiff_This_Follow_Target)/0.01); % 0.01 is used to calculate the multiple of Mach Difference
elseif Direction == 2 %Westbound
    add_headway_min = (OTS_Settings(1).Add_Headway_Coeff_One + OTS_Settings(1).Add_Headway_Coeff_West*TrackDist)*(abs(SpeedMachDiff_This_Follow_Target)/0.01); % 0.01 is used to calculate the multiple of Mach Difference
end
```

```
RLong_Separation = OTS_Settings(end).Longitudinal_Separation_min; % use the lowest separation
minimum. To make sure the Intail_Time below is bigger than this number.

if SpeedMachDiff_This_Follow_Target >= 0
    % Add Buffer time to default separation. Default separation the bigger separation between target
and subject aircraft
    InTail_Time_ThisfollowsTarget = max(Default_Seperation +
kOpen*SpeedMachDiff_This_Follow_Target +
OTS_Settings(1).Buffer_OTSEntry_min,RLong_Separation); % 2 min injection check time
    InTail_Time_TargetfollowsThis = Default_Seperation +
kClose*SpeedMachDiff_Target_Follow_This + OTS_Settings(1).Buffer_OTSEntry_min;
else
    InTail_Time_ThisfollowsTarget = Default_Seperation +
kClose*SpeedMachDiff_This_Follow_Target + OTS_Settings(1).Buffer_OTSEntry_min +
add_headway_min;
    InTail_Time_TargetfollowsThis = max(Default_Seperation +
kOpen*SpeedMachDiff_Target_Follow_This +
OTS_Settings(1).Buffer_OTSEntry_min,RLong_Separation);
end % if SpeedMachDiff_This_Follow_Target >= 0

if ((ThisEntryTimeMin-TargetFlightEntryTimeMin) >= InTail_Time_ThisfollowsTarget)

    Position = TargetFlightIndex+1;

elseif (TargetFlightEntryTimeMin-ThisEntryTimeMin) >= InTail_Time_TargetfollowsThis

    if TargetFlightIndex == 1
        Position = TargetFlightIndex;
    else
        Position = TrackLocationFinder(ListOfPreviousFlight, ListOfPreviousFlightOTSRequestInfo,
Default_Seperation, ...
            TargetFlightIndex - 1, ThisEntryTime, ThisSpeedMach, RequiredFL, RequiredTrack,
RLongIndicator, StepTime,Direction,TrackDist);
    end % if TargetFlightIndex == 1

else
    Position = -1;
end % if ((ThisEntryTimeMin-TargetFlightEntryTimeMin) >= InTail_Time_ThisfollowsTarget)

return
```

OTS_Track_Assignment.m

```
% ----------------------
% This function assigns flights to OTS tracks
% ----------------------
function [Flight, FlightWindProfile, Counter4TrackRequest, TrackRequest] = ...
    OTS_Track_Assignment(Flight, FlightOTSRequestInfo, FlightWindProfile, RequestTrackFL,
TrackSettings, TrackWind, NoOfFLInOTS, aircraftIndex, Counter4TrackRequest, TrackRequest,
flightLevelsNAT, TotFlights,trackArray)

% ---------------------
% Define Global Constants. IMPORTANT: These variables should never change
% ---------------------
global AltInterval_ft Constants badaForOceanic Scenario_Settings EastTrackWind WestTrackWind
flight_cost flight_cost_Internal_ID

List_of_Valid_Tracks_4_This_Flight     =
FlightOTSRequestInfo(aircraftIndex).EstimatedTrackSequence;
%Number_of_Valid_Tracks_4_This_Flight = length(List_of_Valid_Tracks_4_This_Flight);

% Check the maximum operational altitude at the estimated arrival time,
% assume the flight would be assigned its Ideal track and FL
fuelFlow =
Aerodynamic(Flight(aircraftIndex).AirDensityAtFL,Flight(aircraftIndex).CruiseSpeed_mps,Flight(airc
raftIndex).CurrentMass,Flight(aircraftIndex).CruiseSpeed_Knots,Flight(aircraftIndex).BADA_Index);
% unit: kg/min
EstimatedTripTime_min =
Scenario_Settings.StepTime_sec*(FlightOTSRequestInfo(aircraftIndex).EstimatedArrivalTime(Flight(
aircraftIndex).IdealTrackFL(2),Flight(aircraftIndex).IdealTrackFL(1)) -
FlightOTSRequestInfo(aircraftIndex).RequestSentTime)/60; % convert sec to min
ExpectedWeightLoss_kg = fuelFlow*EstimatedTripTime_min; % unit:kg
maxOperationalAltitude =
min(interp1(badaForOceanic(Flight(aircraftIndex).BADA_Index).operationalAltMass,...
    badaForOceanic(Flight(aircraftIndex).BADA_Index).maxOperationalAltitude,Flight(aircraftIndex).
CurrentMass -
ExpectedWeightLoss_kg),badaForOceanic(Flight(aircraftIndex).BADA_Index).maxAltitude);

maxOperationalAltitude =
min(maxOperationalAltitude,Scenario_Settings.MaximumAlt4OceanicAirspace);

[~,maxOperationalAltitude_FL_Index] = min(abs(floor(maxOperationalAltitude/1000)*1000 -
Constants.FlightLevels4OTSAssignment));
% Begin cheking from the 1st track (Optimal) to the last available for this flight
```

```
[~,Flight_Current_FL_Index] = min(abs(Flight(aircraftIndex).CurrentAltitude(1) -
Constants.FlightLevels4OTSAssignment));

if Scenario_Settings.OTS_Track_Assignment_Cost_Matrix == 1 % Assign Track and FL only based
on cost matrix

    ListOfTrackFL = OTS_Track_FL_Finder_Full_Cost_Matrix(flight_cost(flight_cost_Internal_ID
==
Flight(aircraftIndex).Internal_ID),RequestTrackFL,List_of_Valid_Tracks_4_This_Flight,maxOperatio
nalAltitude_FL_Index,trackArray,flightLevelsNAT);

else    % Assign Track and FL only based on Fl_above and FL_below cuurent FL first, then cost
matrix

    ListOfTrackFL = OTS_Track_FL_Finder(flight_cost(flight_cost_Internal_ID ==
Flight(aircraftIndex).Internal_ID),RequestTrackFL,List_of_Valid_Tracks_4_This_Flight,maxOperatio
nalAltitude_FL_Index,trackArray,flightLevelsNAT);

end
Flight(aircraftIndex).maxOperationalAltitude_FL_Index = maxOperationalAltitude_FL_Index;
Flight(aircraftIndex).ListOfTrackFL = ListOfTrackFL;

NoOfOptions = length(ListOfTrackFL(:,1));

for OptionIndex = 1:NoOfOptions

    Track_And_FL_Under_Consideration = ListOfTrackFL(OptionIndex,:);
    if
TrackSettings(Track_And_FL_Under_Consideration(2),Track_And_FL_Under_Consideration(1))<=0
|| ((TrackSettings(Track_And_FL_Under_Consideration(2),Track_And_FL_Under_Consideration(1))
==1) && (Flight(aircraftIndex).EquipageIndex==0))
        continue;
    end

    %...Check If This Is The First Flight To Enter This Airway
    if
Counter4TrackRequest(Track_And_FL_Under_Consideration(2),Track_And_FL_Under_Consideratio
n(1)) == 0

        Counter4TrackRequest(Track_And_FL_Under_Consideration(2),Track_And_FL_Under_Consi
deration(1)) =
```

```
Counter4TrackRequest(Track_And_FL_Under_Consideration(2),Track_And_FL_Under_Consideratio
n(1))   +1;

        %...Initialize A Vector For This Airway In Order To Keep Record Of Flights That Traversed It
        TrackRequest{Track_And_FL_Under_Consideration(2),Track_And_FL_Under_Consideration(
1)} = zeros(round(TotFlights/2),1);

        %...Keep Record Of Flight ID That Traversed This Airway
        TrackRequest{Track_And_FL_Under_Consideration(2),Track_And_FL_Under_Consideration(
1)}(Counter4TrackRequest(Track_And_FL_Under_Consideration(2),Track_And_FL_Under_Consider
ation(1))) = aircraftIndex ;

        %...Update Flight Structure
        Flight(aircraftIndex).AssignedTrack = Track_And_FL_Under_Consideration(1);
        Flight(aircraftIndex).AssignedFL      = Track_And_FL_Under_Consideration(2);
        Flight(aircraftIndex).OTSRequest      = 0;
        break;
        %...If This Is Not The First Flight To Enter This Airway

    else % if
Counter4TrackRequest(Track_And_FL_Under_Consideration(2),Track_And_FL_Under_Consideratio
n(1)) == 0

        for FirstWaitingIndex =
1:Counter4TrackRequest(Track_And_FL_Under_Consideration(2),Track_And_FL_Under_Considerati
on(1))
            RequestedAircraftIndex =
TrackRequest{Track_And_FL_Under_Consideration(2),Track_And_FL_Under_Consideration(1)}(Fir
stWaitingIndex,1);
            if Flight(RequestedAircraftIndex).InOTSIndicator ~=1
                break;
            end
        end

        ListOfPreviousFlightIndex =
TrackRequest{Track_And_FL_Under_Consideration(2),Track_And_FL_Under_Consideration(1)}(Fir
stWaitingIndex :Counter4TrackRequest(Track_And_FL_Under_Consideration(2),Track_And_FL_Und
er_Consideration(1)),1);
        ListOfPreviousFlight         = Flight(ListOfPreviousFlightIndex);
        ListOfPreviousFlightOTSRequestInfo = FlightOTSRequestInfo(ListOfPreviousFlightIndex);
        TargetFlightIndex = length(ListOfPreviousFlight);
        ThisSpeedMach        = Flight(aircraftIndex).CruiseSpeedMachNum;
```

```
ThisEntryTime =
FlightOTSRequestInfo(aircraftIndex).EstimatedArrivalTime(Flight_Current_FL_Index,Track_And_FL_Under_Consideration(1));

Previous_Acft_In_This_Airway_OTS_Equipped =
Flight(ListOfPreviousFlightIndex(TargetFlightIndex)).OTS.Equipped;
This_Acft_OTS_Equipped = Flight(aircraftIndex).OTS.Equipped;

RLongIndicator = Constants.RLongIndicator4ExclusiveTracks &...
    TrackSettings(Track_And_FL_Under_Consideration(2),Track_And_FL_Under_Consideration(1)) ~= 0 & ...
    This_Acft_OTS_Equipped * Previous_Acft_In_This_Airway_OTS_Equipped;

TrackDist = TrackWind{Track_And_FL_Under_Consideration(1),3};

Default_Seperation =
max(Flight(aircraftIndex).OTS.Longitudinal_Separation_min, Flight(ListOfPreviousFlightIndex(TargetFlightIndex)).OTS.Longitudinal_Separation_min);


Position = TrackLocationFinder(ListOfPreviousFlight, ListOfPreviousFlightOTSRequestInfo, Default_Seperation, ...
    TargetFlightIndex, ThisEntryTime, ThisSpeedMach,
Track_And_FL_Under_Consideration(2), Track_And_FL_Under_Consideration(1), RLongIndicator,
Constants.StepTime,Flight(aircraftIndex).Dir,TrackDist);

if Position ~= -1

    TrackRequest{Track_And_FL_Under_Consideration(2),Track_And_FL_Under_Consideration(1)}(FirstWaitingIndex +
Position:Counter4TrackRequest(Track_And_FL_Under_Consideration(2),Track_And_FL_Under_Consideration(1))+1) = ...
        TrackRequest{Track_And_FL_Under_Consideration(2),Track_And_FL_Under_Consideration(1)}(FirstWaitingIndex + Position-
1:Counter4TrackRequest(Track_And_FL_Under_Consideration(2),Track_And_FL_Under_Consideration(1)));
    TrackRequest{Track_And_FL_Under_Consideration(2),Track_And_FL_Under_Consideration(1)}(FirstWaitingIndex + Position-1) = aircraftIndex ;
    Counter4TrackRequest(Track_And_FL_Under_Consideration(2),Track_And_FL_Under_Consideration(1)) =
Counter4TrackRequest(Track_And_FL_Under_Consideration(2),Track_And_FL_Under_Consideration(1))+1;
```

```
            Flight(aircraftIndex).AssignedTrack = Track_And_FL_Under_Consideration(1);
            Flight(aircraftIndex).AssignedFL      = Track_And_FL_Under_Consideration(2);
            Flight(aircraftIndex).OTSRequest      = 0;
            break;

        end % if Position ~= -1

    end % if
Counter4TrackRequest(Track_And_FL_Under_Consideration(2),Track_And_FL_Under_Consideratio
n(1)) == 0
end
%end % for FL_Under_Consideration =
Maximum_OTS_FL_4_This_Flight:Minimum_OTS_FL_4_This_Flight

if Flight(aircraftIndex).OTSRequest == 0

    if Flight(aircraftIndex).AssignedTrack ~= RequestTrackFL(1) % Assigned OTS Track and
Requested OTS Track are not the same

          Flight(aircraftIndex,1).CurrentTrack = Flight(aircraftIndex).AssignedTrack;
        % Make sure Current CriticalPoint is set to new converge point
        Flight(aircraftIndex).CriticalPointsIndex(1) =
Flight(aircraftIndex).CriticalPoints.AllOTSTracks(Flight(aircraftIndex,1).CurrentTrack).IndexOfOTSE
ntryPoint-1;
        Flight(aircraftIndex).DistBetweenCriticalPoints_deg =
Flight(aircraftIndex).CriticalPoints.AllOTSTracks(Flight(aircraftIndex,1).CurrentTrack).DistBetweenC
riticalPoints_deg;
        Flight(aircraftIndex).CriticalPointsFLLowerIndex =
Flight(aircraftIndex).CriticalPoints.AllOTSTracks(Flight(aircraftIndex,1).CurrentTrack).CriticalPoints
FLLowerIndex;
        Flight(aircraftIndex).DistTravelledBeforeOTS_deg =
Flight(aircraftIndex).CriticalPoints.AllOTSTracks(Flight(aircraftIndex,1).CurrentTrack).DistTravelled
BeforeOTS_deg;
        Flight(aircraftIndex).DistTravelledInsideOTS_deg =
Flight(aircraftIndex).CriticalPoints.AllOTSTracks(Flight(aircraftIndex,1).CurrentTrack).DistTravelledI
nsideOTS_deg;

        Flight(aircraftIndex).CriticalPoints =
Flight(aircraftIndex).CriticalPoints.AllOTSTracks(Flight(aircraftIndex,1).CurrentTrack).CriticalPoints;
        InOTSIndicator_One_Index = find(Flight(aircraftIndex).CriticalPoints.InOTSIndicator);

        Flight(aircraftIndex).FirstCriticalPointInOTS = InOTSIndicator_One_Index(1) - 1;
```

```
        Flight(aircraftIndex).LastCriticalPointInOTS = InOTSIndicator_One_Index(end);

        Flight(aircraftIndex).CriticalPointsFLLowerIndex(Flight(aircraftIndex).FirstCriticalPointInOTS
:Flight(aircraftIndex).LastCriticalPointInOTS,1) = Flight(aircraftIndex).AssignedFL;

        FlightWindProfile(aircraftIndex).WindMatrixBetweenCriticalPoints = ...
            Wind_Profile_Calculator(Flight(aircraftIndex), EastTrackWind, WestTrackWind,
Constants.DistInterval_deg, AltInterval_ft);


        % Make sure Last CriticalPoint is set.
        Flight(aircraftIndex).CriticalPointsIndex(2) =
length(Flight(aircraftIndex).CriticalPoints.InOTSIndicator);

    end
    if Flight(aircraftIndex).AssignedFL ~= RequestTrackFL(2) % Assigned OTS FL and Requested
OTS FL are not the same

        Flight(aircraftIndex).CriticalPoints.FlightLevel_ft(Flight(aircraftIndex).FirstCriticalPointInOTS
- 1:Flight(aircraftIndex).LastCriticalPointInOTS) = ...
            flightLevelsNAT(Flight(aircraftIndex).AssignedFL,1); % Alternate the flight level of
Converge Point to be AssignedFL. In Demand Processor, it is IdealFL.

        if
Flight(aircraftIndex).CriticalPoints.FlightLevel_ft(Flight(aircraftIndex).FirstCriticalPointInOTS) >
Flight(aircraftIndex).CurrentAltitude(1)
            Flight(aircraftIndex).RequiredActionAtCriticalPoint = 1;
        elseif
Flight(aircraftIndex).CriticalPoints.FlightLevel_ft(Flight(aircraftIndex).FirstCriticalPointInOTS) <
Flight(aircraftIndex).CurrentAltitude(1)
            Flight(aircraftIndex).RequiredActionAtCriticalPoint = -1;
        else
            Flight(aircraftIndex).RequiredActionAtCriticalPoint = 0;
        end% if Flight(aircraftIndex).CriticalPoints(Flight(aircraftIndex).FirstCriticalPointInOTS,3) >
Flight(aircraftIndex).CurrentAltitude(1)
        Flight(aircraftIndex).CriticalPointsFLLowerIndex(Flight(aircraftIndex).FirstCriticalPointInOTS
:Flight(aircraftIndex).LastCriticalPointInOTS,1) = Flight(aircraftIndex).AssignedFL;
        Flight(aircraftIndex).CurrentOTSFLIndex = Flight(aircraftIndex).AssignedFL;

    end % if Flight(aircraftIndex).AssignedTrack ~= RequestTrackFL(1)
```

```
    Flight(aircraftIndex).CurrentTrack = Flight(aircraftIndex).AssignedTrack;
    Flight(aircraftIndex).CurrentOTSFLIndex = Flight(aircraftIndex).AssignedFL;
end % if Flight(aircraftIndex).OTSRequest == 0
return;
```

OTS_Flights_Analysis.m

```
function OTS_Flights_Analysis(Install_Dir, Project_Dir, Case_Folder, Case_Name)

if (ismac == 1)
    SLASH = '/';
elseif (ismac == 0)
    SLASH = '\';
end

Output_Dir = [Project_Dir, SLASH, 'GOM', SLASH, 'Output', SLASH, Case_Folder, SLASH,
Case_Name, SLASH];

% Load relevant data into memory
load ([Output_Dir, 'Settings', SLASH, 'Scenario_Settings.mat']);

if Scenario_Settings.NAT_OTS == 0
    return;
end

load ([Output_Dir, 'Settings', SLASH, 'Region_Settings.mat']);

Date = Scenario_Settings.Date;

% Load CostMatrix
RLatSM = 0; % Scenario_Settings.NAT_OTS_RLatSM;
if RLatSM == 1
    OTS_Configuration = 'RLatSM';
else
    OTS_Configuration = 'Basic';
end
load ([Output_Dir, 'CostMatrix', SLASH, OTS_Configuration, SLASH, 'flight_cost_',
Scenario_Settings.Date]);
if exist('flight_cost_ser', 'var') == 1 % Detect if outputs are serialized
    flight_cost = getArrayFromByteStream(flight_cost_ser); clear flight_cost_ser;
end

% Load Demand
load ([Output_Dir, 'Demand', SLASH, 'Demand_',Scenario_Settings.Date]);
Demand_After_Cost_Matrix = Demand; clear Demand;

load ([Output_Dir, 'Demand', SLASH, 'Demand_GOM_2_NATSAMii_',Scenario_Settings.Date]);
Demand_TFMS = Demand_WithBADA; clear Demand_WithBADA;
```

```
% Load OTS data
load ([Output_Dir, 'OTS', SLASH, 'NAT_OTS_',Scenario_Settings.Date]);
load ([Output_Dir, 'Settings', SLASH, 'OTS_Settings.mat']);

Number_Of_Tracks = length(NAT_OTS.track);

% ----------------------------------
% Loading Results
% ----------------------------------
% The day before and after simualtion needs to be added, because the files
% are split by departure date.
day_before_demand_day = datestr(datenum(num2str(Scenario_Settings.Date),'yyyymmdd') -
datenum('01010000','mmddyyyy'), 'yyyymmdd');
demand_day = num2str(Scenario_Settings.Date);
day_after_demand_day = datestr(datenum(num2str(Scenario_Settings.Date),'yyyymmdd') +
datenum('01010000','mmddyyyy'),'yyyymmdd');
DateVector = {day_before_demand_day, demand_day, day_after_demand_day};

Number_Of_Dates = length(DateVector);

% Load Simulation Results
Flight = [];
FlightTrajectory = [];

for day = 1:Number_Of_Dates

    Current_Date = DateVector{day};

    Flight_Filename           = [Output_Dir, 'Flight_Simulator', SLASH, 'Flight_Results_',
Current_Date, '.mat'];
    FlightTrajectory_Filename = [Output_Dir, 'Flight_Simulator', SLASH, 'FlightTrajectory_Results_',
Current_Date, '.mat'];

    if exist(Flight_Filename, 'file') == 0
        continue;
    end

    Flight_Results            = load(Flight_Filename);
    FlightTrajectory_Results = load(FlightTrajectory_Filename);

    % The outputs are serialized in order to improve save performance and reduce .mat file size on disk
```

```
    % To deserialize the outputs, simply use: Flight = getArrayFromByteStream(Flight_ser);
    % More info: http://undocumentedmatlab.com/blog/serializing-deserializing-matlab-data
    if isfield(Flight_Results, 'Flight_ser') % Detect if outputs are serialized
        Flight           = [Flight; getArrayFromByteStream(Flight_Results.Flight_ser)];
        FlightTrajectory = [FlightTrajectory;
getArrayFromByteStream(FlightTrajectory_Results.FlightTrajectory_ser)];
    else
        Flight           = [Flight; Flight_Results.Flight];
        FlightTrajectory = [FlightTrajectory; FlightTrajectory_Results.FlightTrajectory];
    end

    clear Flight_Results;
    clear FlightTrajectory_Results;
end % for day = 1:Number_Of_Dates

% Number Of Flights
Number_Of_Flights = length(Flight);

% Save Dir
Save_Dir = [Output_Dir, 'OTS_Flights_Analysis', SLASH];
mkdir(Save_Dir);

% ----------------------------------
% Analyze flight_cost
% ----------------------------------
GreenColor = [0.47,0.67,0.19];
track_sum = zeros(Number_Of_Tracks,1);

for track_id = 1 : Number_Of_Tracks

    Track_Name = strcat('NAT',NAT_OTS.track(track_id));

    for i = 1 : length(flight_cost)

        if strcmp(flight_cost(i).Filed_Track,Track_Name)
            track_sum(track_id,1) = track_sum(track_id,1) + 1;
        end

    end

end
```

```
sum_raw = sum(track_sum);

figure
bar(track_sum,'FaceColor',GreenColor)
title('Filed Track Distribution - Source: flight cost','FontSize',40)
xl = xlim();
xTickLocations = linspace(xl(1)+1,xl(2)-1,Number_Of_Tracks);
set(gca,'XTick',xTickLocations);
set(gca,'XTickLabel',NAT_OTS.track);
set(gcf,'Color','white');
grid

savefig([Save_Dir,'Filed_Track_Distribution_on_',num2str(Date),' - flight_cost'])
close;

% ---------------------------------
% Analyze Demand after parsing TFMS data and Plot Waypoints in Demand and Track
% ---------------------------------
Number_Of_Demand = length(Demand_TFMS);

Eastflow_Color = [0 0 0.800000011920929];
Westflow_Color = [0.200000002980232 0.600000023841858 0];
BlueColor = [0.3,0.75,0.93];
RedColor = [0.64,0.08,0.18];

Demand_track_sum = zeros(Number_Of_Tracks,1);

for track_id = 1 : Number_Of_Tracks

    Track_Name = strcat('NAT',NAT_OTS.track(track_id));

    track_index = [];

    figure
    latlim = [0 110];
    lonlim = [-125 50];

    axesm('robinson','MapLatLimit',latlim,'MapLonLimit',lonlim,'Frame','on','Grid','on','MeridianLabel'
,'on','ParallelLabel','on')
    axis off
    setm(gca,'MLabelLocation',30)
    landareas = shaperead('landareas.shp','UseGeoCoords', true);
```

```
    geoshow (landareas, 'FaceColor', [1 0.97 0.92]);
    set(gcf, 'Position', get(0,'Screensize'));

    for demand_id = 1 :   Number_Of_Demand

        if strcmp(Demand_TFMS(demand_id).Filed_Track,Track_Name) &&
Demand_TFMS(demand_id).act_date == str2double(Date)

            plotm(Demand_TFMS(demand_id).TFMS_FP_Lat,Demand_TFMS(demand_id).TFMS_FP
_Lon)
            track_index = [track_index;demand_id];
            %               plotm(Demand_TFMS(demand_id).Simulation_FP_Lat,Demand_TFMS(dem
and_id).Simulation_FP_Lon,'g')
            Demand_track_sum(track_id,1) = Demand_track_sum(track_id,1) + 1;

        end
    end

    if strcmp(NAT_OTS.Direction(track_id),'E')
        plotm(NAT_OTS.lat{track_id},NAT_OTS.lon{track_id},'Color',Eastflow_Color,'LineWidth',5
)
    else
        plotm(NAT_OTS.lat{track_id},NAT_OTS.lon{track_id},'Color',Westflow_Color,'LineWidth',
5)
    end

    title(['Flights Filed Track ',char(Track_Name)],'FontSize',40)
    set(gcf,'Color','white');
    savefig([Save_Dir, 'TFMS_Field10_Trajectory_for_Track_',char(Track_Name),'.fig']);
    close;

    Demand_Track_List(track_id).Track_Index = track_index;

end

sum_Demand = sum(Demand_track_sum);

figure
bar(Demand_track_sum,'FaceColor',BlueColor)
title(['Filed Track Distribution on ',num2str(Date),' Total: ',num2str(sum_Demand),' - Source: TFMS
(Field 10)'],'FontSize',40)
xl = xlim();
```

```
xTickLocations = linspace(xl(1)+1,xl(2)-1,Number_Of_Tracks);
set(gca,'XTick',xTickLocations);
set(gca,'XTickLabel',NAT_OTS.track);
set(gcf,'Color','white');
grid

savefig([Save_Dir,'TFMS_Field10_Filed_Track_Distribution_on_', num2str(Date)])
close;

% ---------------------------------
% Analyze Demand after Cost Matrix and Plot Waypoints in Demand and Track
% ---------------------------------
Number_Of_Demand = length(Demand_After_Cost_Matrix);

Eastflow_Color = [0 0 0.800000011920929];
Westflow_Color = [0.200000002980232 0.600000023841858 0];
BlueColor = [0.3,0.75,0.93];
RedColor = [0.64,0.08,0.18];

Demand_track_sum = zeros(Number_Of_Tracks,1);

for track_id = 1 : Number_Of_Tracks

    Track_Name = strcat('NAT',NAT_OTS.track(track_id));

    track_index = [];

    figure
    latlim = [0 110];
    lonlim = [-125 50];

    axesm('robinson','MapLatLimit',latlim,'MapLonLimit',lonlim,'Frame','on','Grid','on','MeridianLabel'
,'on','ParallelLabel','on')
    axis off
    setm(gca,'MLabelLocation',30)
    landareas = shaperead('landareas.shp','UseGeoCoords', true);
    geoshow (landareas, 'FaceColor', [1 0.97 0.92]);
    set(gcf, 'Position', get(0,'Screensize'));

    for demand_id = 1 :   Number_Of_Demand
```

```
        if strcmp(Demand_After_Cost_Matrix(demand_id).Requested_Track,Track_Name) &&
Demand_After_Cost_Matrix(demand_id).act_date == str2double(Date)

            plotm(Demand_After_Cost_Matrix(demand_id).TFMS_FP_Lat,Demand_After_Cost_Matri
x(demand_id).TFMS_FP_Lon)
            track_index = [track_index;demand_id];
            %           plotm(Demand_After_Cost_Matrix(demand_id).Simulation_FP_Lat,Demand
_After_Cost_Matrix(demand_id).Simulation_FP_Lon,'g')
            Demand_track_sum(track_id,1) = Demand_track_sum(track_id,1) + 1;

        end
    end

    if strcmp(NAT_OTS.Direction(track_id),'E')
        plotm(NAT_OTS.lat{track_id},NAT_OTS.lon{track_id},'Color',Eastflow_Color,'LineWidth',5
)
    else
        plotm(NAT_OTS.lat{track_id},NAT_OTS.lon{track_id},'Color',Westflow_Color,'LineWidth',
5)
    end

    title(['Flights Requested Track ',char(Track_Name)],'FontSize',40)
    set(gcf,'Color','white');
    savefig([Save_Dir, 'After_Cost_Matrix_Trajectory_for_Track_',char(Track_Name),'.fig']);
    close;

    Demand_Track_List(track_id).Track_Index = track_index;

end

sum_Demand = sum(Demand_track_sum);

figure
bar(Demand_track_sum,'FaceColor',BlueColor)
title(['Requested Track Distribution on ',num2str(Date),' Total: ',num2str(sum_Demand),' - Source:
After Cost Matrix'],'FontSize',40)
xl = xlim();
xTickLocations = linspace(xl(1)+1,xl(2)-1,Number_Of_Tracks);
set(gca,'XTick',xTickLocations);
set(gca,'XTickLabel',NAT_OTS.track);
set(gcf,'Color','white');
grid
```

```
savefig([Save_Dir,'After_Cost_Matrix_Requested_Track_Distribution_on_', num2str(Date)])
close;

% ---------------------------------
% Analyze Simulated Output and Plot Waypoints in Demand and Track
% ---------------------------------
result_track_sum = zeros(length(NAT_OTS.track),1);
Number_Of_EastTracks = length(find(strcmp(NAT_OTS.Direction,'E')));

flight_cost_Internal_ID = [flight_cost.Internal_ID];
Flight_Internal_ID        = [Flight.Internal_ID];

[~, flight_cost_index] = ismember(Flight_Internal_ID, flight_cost_Internal_ID);

for track_id = 1 : Number_Of_Tracks

    Track_Name = strcat('NAT',NAT_OTS.track(track_id));

    track_index = [];

    figure
    latlim = [0 110];
    lonlim = [-125 50];

    axesm('robinson','MapLatLimit',latlim,'MapLonLimit',lonlim,'Frame','on','Grid','on','MeridianLabel'
,'on','ParallelLabel','on')
    axis off
    setm(gca,'MLabelLocation',30)
    landareas = shaperead('landareas.shp','UseGeoCoords', true);
    geoshow (landareas, 'FaceColor', [1 0.97 0.92]);
    set(gcf, 'Position', get(0,'Screensize'));

    % Pre-plot OTS directional configuration
    if strcmp(NAT_OTS.Direction(track_id),'E')
        track_Direction = 1;
        % Plot Eastbound Tracks
        for tr_id = 1 : Number_Of_EastTracks
            hLine1 =
plotm(NAT_OTS.lat{tr_id},NAT_OTS.lon{tr_id},'Color',Eastflow_Color,'LineWidth',3);
        end
```

```
        hLine2 = plotm(NAT_OTS.lat{track_id},NAT_OTS.lon{track_id},'Color','r','LineWidth',3);
        title(['Simulated Flights In Eastbound Track ',char(Track_Name)],'FontSize',40)
        legend([hLine1,hLine2],{'Eastbound Tracks',['Track
',char(Track_Name)]},'FontSize',14,'Location','south','Orientation','vertical');
    else
        track_Direction = 2;
        % Plot Westbound Tracks
        for tr_id = Number_Of_EastTracks+1 : Number_Of_Tracks
            hLine3 =
plotm(NAT_OTS.lat{tr_id},NAT_OTS.lon{tr_id},'Color',Westflow_Color,'LineWidth',3);
        end

        hLine4 = plotm(NAT_OTS.lat{track_id},NAT_OTS.lon{track_id},'Color','r','LineWidth',3);
        title(['Simulated Flights In Westbound Track ',char(Track_Name)],'FontSize',40)
        legend([hLine3,hLine4],{'Westbound Tracks',['Track
',char(Track_Name)]},'FontSize',14,'Location','south','Orientation','vertical');

        track_id = track_id - Number_Of_EastTracks; % Subtract Number_Of_EastTracks because the
index in NAT_OTS.track is different from AssignedTrack's

    end

    for flight_id = 1 : Number_Of_Flights

        if isempty(Flight(flight_id).IdealTrackFL)
            continue;
        else
            if Flight(flight_id).AssignedTrack == track_id && Flight(flight_id).Dir == track_Direction
&& strcmp(Flight(flight_id).Departure_Date,Date)

                plotm(FlightTrajectory(flight_id).Latitude_deg,FlightTrajectory(flight_id).Longitude_d
eg)

                plotm(flight_cost(flight_cost_index(flight_id)).Simulation_FP_Lat,
flight_cost(flight_cost_index(flight_id)).Simulation_FP_Lon, '-k');

                track_index = [track_index;flight_id];

                if track_Direction == 1
                    result_track_sum(track_id,1) = result_track_sum(track_id,1) + 1;
                else
```

```
                    result_track_sum(track_id + Number_Of_EastTracks,1) = result_track_sum(track_id
+ Number_Of_EastTracks,1) + 1;
                end

            end
        end
    end



    set(gcf,'Color','white');
    savefig([Save_Dir, 'Simulated_Trajectories_for_Track_',char(Track_Name),'.fig']);
    close;

    if track_Direction == 1
        Result_Track_List(track_id).Track_Index = track_index;
    else
        Result_Track_List(track_id + Number_Of_EastTracks).Track_Index = track_index;
    end

end

sum_result = sum(result_track_sum);

figure
bar(result_track_sum,'FaceColor',RedColor)
title(['Simulated Track Distribution on ',num2str(Date),' Total: ',num2str(sum_result),' - Source:
Flight'],'FontSize',40)
xl = xlim();
xTickLocations = linspace(xl(1)+1,xl(2)-1,Number_Of_Tracks);
set(gca,'XTick',xTickLocations);
set(gca,'XTickLabel',NAT_OTS.track);
set(gcf,'Color','white');
grid

savefig([Save_Dir,'Simulated_Track_Distribution_on_',num2str(Date),' - Flight'])
close;

return
```

OTS_Climb_LOS_Analyzer.m

```
function [Number_Of_Climb_Flights,Number_Of_Unassigned_Flights,LOS_Track,LOS_TrackFL] =
OTS_Climb_LOS_Analyzer(Output_Dir,Current_Date)
%% Define Directories

if (ismac == 1)
    SLASH = '/';
elseif (ismac == 0)
    SLASH = '\';
end

%% Load relevant data into memory
load ([Output_Dir, 'Settings', SLASH, 'Scenario_Settings.mat']);

if Scenario_Settings.NAT_OTS == 0
   Number_Of_Climb_Flights        = [];
   Number_Of_Unassigned_Flights = [];
   LOS_Track     = [];
   LOS_TrackFL = [];
   return;
end

load ([Output_Dir, 'Settings', SLASH, 'Region_Settings.mat']);
load ([Output_Dir, 'OTS', SLASH, 'NAT_OTS_',Scenario_Settings.Date,'.mat']);
load ([Output_Dir, 'Settings', SLASH, 'OTS_Settings.mat']);

%% Loading Results

Flight_Results               = load([Output_Dir, 'Flight_Simulator', SLASH, 'Flight_Results_',
Current_Date, '.mat']);
FlightOTSRequestInfo_Results = load([Output_Dir, 'Flight_Simulator', SLASH,
'FlightOTSRequestInfo_Results_', Current_Date, '.mat']);

% The outputs are serialized in order to improve save performance and reduce .mat file size on disk
% To deserialize the outputs, simply use: Flight = getArrayFromByteStream(Flight_ser);
% More info: http://undocumentedmatlab.com/blog/serializing-deserializing-matlab-data
if isfield(Flight_Results, 'Flight_ser') % Detect if outputs are serialized
    Flight               = getArrayFromByteStream(Flight_Results.Flight_ser);
    FlightOTSRequestInfo =
getArrayFromByteStream(FlightOTSRequestInfo_Results.FlightOTSRequestInfo_ser);
end
```

```
clear Flight_Results;
clear FlightOTSRequestInfo_Results

% Extract OTS Flight
OTSFlightIndex = [Flight.OTS_Flight]';
OTSFlight = Flight(OTSFlightIndex == 1);

%% Count number of flights that climb

[Number_Of_Flights,~] = size(Flight);

AprrovedClimbPerFlightCount = [Flight.ApprovedClimbRequestInsideOTS]';

ClimbFlight = Flight(AprrovedClimbPerFlightCount > 0); % Generate a structure consisting of climb
flight
Number_Of_Climb_Flights = length(ClimbFlight);
ClimbPercent = 100*Number_Of_Climb_Flights/length(OTSFlight);

% disp(['The number of flights that climb is ',num2str(Number_Of_Climb_Flights),'. The percentage is
',num2str(ClimbPercent),'%.'])
%% Count number of flights that are not assigned a Track or FL (Its Flight.OTSRequest is 0.5)

UnassignedFlightsIndex = [Flight.OTSRequest]';
UnassignedFlights = Flight(UnassignedFlightsIndex == 0.5);
Number_Of_Unassigned_Flights = length(UnassignedFlights);
UnassignedPercent = Number_Of_Unassigned_Flights/length(OTSFlight);

% disp(['The number of unassigned flights is ',num2str(Number_Of_Unassigned_Flights),'. The
percentage is ',num2str(UnassignedPercent),'%.'])

%% Calculate Level Of Service

% Calculate Level Of Service for only Track Assignment (See if a flight is assigned Requested Track)
LOSTrackIndex = [];

% Calculate Level Of Service for Track and Flight Level Assignment (See if a flight is assigned both
Ideal Track and Ideal FL)
% By definition, we consider flight levels higher than Ideal one as requested FL.
LOSIndex = [];
FL_Diff = zeros(Number_Of_Flights,6);

for fl = 1 : Number_Of_Flights
```

```
    if Flight(fl).OTS_Flight == 1

        RequestedTrack = FlightOTSRequestInfo(fl).EstimatedTrackSequence(1);
        IdealFL = Flight(fl).IdealTrackFL(2);
        maxOperFL = Flight(fl).maxOperationalAltitude_FL_Index;
        AssignedTrack = Flight(fl).AssignedTrack;
        AssignedFL = Flight(fl).AssignedFL;
        FL_Diff(fl,:) = [fl,IdealFL,maxOperFL,AssignedFL,(IdealFL - AssignedFL),(maxOperFL -
AssignedFL)];

        if RequestedTrack == AssignedTrack
            LOSTrackIndex = [LOSTrackIndex;fl];
        end
        % if Assigned FL is higher than Ideal FL, we consider it as Requested FL
        % Note: Higher flight level has lower index
        if RequestedTrack == AssignedTrack &&   (AssignedFL   == maxOperFL || AssignedFL <=
IdealFL)
            LOSIndex = [LOSIndex;fl];
        end

    end

end

LOS_Track = length(LOSTrackIndex);
LOS_TrackFL = length(LOSIndex);

LOSTrackPercent = 100 * LOS_Track/length(OTSFlight);
LOSPercent = 100 * LOS_TrackFL/length(OTSFlight);
```