A METHODOLOGY FOR ANALYSIS OF METROPLEX AIR TRAFFIC FLOWS

by

Akshay Belle
A Dissertation
Submitted to the
Graduate Faculty
of
George Mason University
in Partial Fulfillment of
The Requirements for the Degree
of
Doctor of Philosophy
Systems Engineering and Operations Research

Committee:

[illegible]	Dr. Lance Sherry, Dissertation Director
[illegible]	Dr. Massimiliano Albanese, Committee Member
[illegible]	Dr. George Dohonue, Committee Member
[illegible]	Dr. Rajesh Ganesan, Committee Member
[illegible]	Dr. John Shortle, Committee Member
[illegible]	Dr. Ariela Sofer, Department Chair
[illegible]	Dr. Kenneth S. Ball, Dean, Volgenau School of Engineering

Date: 11/20/2013

Fall Semester 2013
George Mason University
Fairfax, VA

A Methodology for Analysis of Metroplex Air Traffic Flows

A Dissertation submitted in partial fulfillment of the requirements for the degree of Doctor of Philosophy at George Mason University

by

Akshay Belle
Master of Science
George Mason University, 2012

Director: Lance Sherry, Associate Professor
Department of Systems Engineering and Operations Research

Fall Semester 2013
George Mason University
Fairfax, VA

DEDICATION

I dedicate this dissertation to my parents, Kuppaswamy Rao Belle and Savitha Belle.

ACKNOWLEDGEMENTS

I am grateful to Dr. Lance Sherry for being my dissertation director. I would like to thank him for helping me stay motivated and focused through the years. Without his expertise, support and guidance this would not have been possible.

I am grateful to my committee members, Dr. George Donohue, Dr. John Shortle, Dr. Rajesh Ganesan and Dr. Massimiliano Albanese, for their insights in helping set the direction of my work. Dr. Donohue, an internationally renowned expert in the field of air transportation research, provided philosophical and technical guidance on the design, implementation and presentation of the results of this research. Dr. Shortle, an expert in Simulation and Queueing theory, assisted with analytics and provided feedback that helped improve the quality of the dissertation. Dr. Ganesan, an expert in Approximate Dynamic Programming, assisted with analytics and provided critical comments that helped shape and improve the dissertation. Dr. Albanese, an expert in Cyber Security and Large Data Analytics, provided an outsider's viewpoint on this research that helped in writing a better description of the problem.

I am grateful to my department chair Prof. Ariela Sofer for having supported me during my first year at Mason. My special thanks to our department staff members Angel Manzo and Josefine Wiecks for making my life as a student so much easier.

I would like to thank Michael Wambsganss and Crown Consulting Inc. for sponsoring the project that set the foundation for this research.

I would like to thank Paula Lewis for her support and feedback.

I would like to thank Dr. Terrance Thompson and Dr. Arash Yousefi for mentoring and motivating me to go ahead and get my PhD.

I would like to thank my friends, Saba, Yimin, Zhenming, Houda and most importantly Vivek for their support and contribution.

Finally, I would like to thank my entire family back home and here for their love and support. I am forever indebted to my dad Kuppaswamy, mom Savitha, my sister Deepali, and most importantly my wife Audarya for always being there for me.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF EQUATIONS

LIST OF ABBREVIATIONS OR SYMBOLS

Air Navigation Service Provider	ANSP
Area Navigation	RNAV
Aviation System Performance Metrics	ASPM
Base of Aircraft Data	BADA
Bureau of Transportation Statistics	BTS
Chicago O'hare International Airport	ORD
Dallas Fort Worth International Airport	DFW
Dallas Love Field airport	DAL
Federal Aviation Administration	FAA
Fort Lauderdale–Hollywood International Airport	FLL
Four Dimension	4D
Instrument Landing System	ILS
Instrument Meteorological Conditions	IMC
John F. Kennedy International Airport	JFK
John Wayne-Orange County Airport	SNA
Joint Planning and Development Office	JPDO
LaGuardia Airport	LGA
Los Angeles International Airport	LAX
Miami International Airport	MIA
Midway International Airport	MDW
Multi-function Control and Display Unit	MCDU
National Airspace System	NAS
National Flight Data Center	NFDC
National Offload Program	NOP
Newark Liberty International Airport	EWR
Performance Based Navigation	PBN
Radius to Fix	RF
Required Navigation Performance	RNP
Required Time of Arrival	RTA
Terminal Radar Approach Control	TRACON
Teterboro Airport	TEB
Traffic flow Management Initiatives	TMI
Van Nuys Airport	VNY
Visual Flight Rules	VFR
Visual Meteorological Conditions	VMC

ABSTRACT

A METHODOLOGY FOR ANALYSIS OF METROPLEX AIR TRAFFIC FLOWS

Akshay Belle, Ph.D.

George Mason University, 2013

Dissertation Director: Dr. Lance Sherry

A key determinant of the airspace capacity serving a metropolitan area with multiple airports is the extent of interaction between arrival and departure flows between the airports. The airports for some “metroplexes” are geographically located such that under certain wind and weather conditions, there exist conflicts between the flows. This results in excess costs from ground holding for departures and airborne holding for arrivals.

Advances in aircraft navigation technology (i.e. Performance Based Navigation) have created opportunities to improve arrival flow efficiencies and de-conflict metroplex flows. The adoption of these technologies has been slow and haphazard due to uncertainties in the estimates of the Return-on-Investment (ROI), the need for collaboration and simultaneous equipage across competing stakeholders, and the allocation of benefits to parties that choose not to equip but gain benefits when their competition equips. Together these issues have created a “modernization stalemate.”

The recent availability of high fidelity surveillance track data coupled with aerodynamic models and weather data have created an opportunity to provide detailed Return-on-Investment analysis of metroplex traffic flows that includes the real-world complexities of traffic flows and aircraft trajectories. This type of analysis provides accurate benefits assessment for various flow and equipage configurations.

This dissertation describes a holistic methodology that uses high fidelity surveillance track data coupled with aerodynamic models and weather data to quantify the benefits of existing and proposed concepts-of-operations and technologies that require simultaneous equipage and development of collaborative procedures by multiple stakeholders.

The methodology includes six algorithmic functions: (1) terminal area flow analysis to characterize of flow and track assignment, (2) analysis of the effects of metroplex flow conflict for arrival holding patterns, (3) estimates of the performance metrics (e.g. times, distance and fuel burn) for terminal area flows and holding patterns, (4) estimates of the benefits of PBN approach procedures at an airport, (5) estimates of the benefits of metroplex airspace de-confliction, and (6) estimates of the return on investment for the equipped operator.

A case study analysis of the benefits of the introduction of a Required Navigation Performance (RNP) approach procedure for air traffic arrival flows in the Chicago Terminal Radar Approach Control (TRACON) is described. The analysis showed that the airspace used to service both, the Chicago O'Hare International Airport (ORD) and the Chicago Midway International Airport (MDW) experiences a flow conflict (13C ILS arrivals at MDW and 22L departures at ORD) on an average 1.6% of the time per year.

When the metroplex airspace is de-conflicted by the introduction of an RNP approach for 13C at MDW, the direct airline operating cost per year is reduced on an average by $.04M at MDW and $1.33M at ORD. The savings at MDW are from elimination of holding patterns and the fuel burn saving of a shorter RNP approach over the ILS approach. At ORD the savings are from a reduction in departure delays. The ratio of the total benefits distributed between flights at MDW and ORD is 1:33 in favor of non-equipped ORD departures. This is equivalent to 1:9 per flight ratio in favor of non-equipped ORD departures.

The methodology also enabled the evaluation of the introduction of additional RNP approach procedures to other runways at MDW to improve the benefits for the equipped arrivals to MDW. This has the potential of saving an average 660K gallon per year of fuel for arrivals at MDW. At $3/gallon this amounts to a savings of an additional $1.97M per year.

The methodology also enabled the evaluation of an "optimal runway configuration," based on wind magnitude/direction *and* flow fuel burn efficiency, to further improve the benefits for the equipped arrivals to MDW. This has the potential of saving an average of 890K gallons per year of fuel for arrivals at MDW. At $3/gallon this amounts to a savings of an additional $2.67M per year.

With these accumulated savings, the RNP approach does <u>not</u> yield a positive ROI at MDW. The carrier at MDW will have to perform at least a half million RNP approaches per year throughout its network, saving at least 33 kg of fuel per approach on an average to break-even in 10 years at a discount rate of 5%.

This analysis demonstrates the economics behind the “modernization stalemate.” The equipping airline cannot turn a positive ROI in a reasonable time-frame while the non-equipped, competing airlines (i.e. free-riders) benefit significantly more than the equipping airline. Mandating equipage is inefficient as all aircraft do not need to equip to improve the efficiencies. Government subsidies for equipage and preferential service incentives for equipage must be calibrated to the asymmetric benefits computed by this methodology.

1 CHAPTER1: INTRODUCTION

The term Metroplex refers to a system of airports serving a large Metropolitan area (FAA, 2012e). The airports in a metroplex are often in close proximity to each other and can have interdependent arrival and departure procedures (JPDO, 2007). In the United States (U.S) the Federal Aviation Administration (FAA) has identified 21 metroplexes (FAA, 2012e).

Metroplexes are a critical component of the nation's economy and the air transportation system. The 33 ASPM[1] airports at the 21 U.S metroplexes account for more than 48% of the total operations in the NAS's hub[2] airports (FAA, 2012f) . The metropolitan regions these airports serve account for 35% (United States Census Bureau, 2012) of the nation's population (314 million as of 2012) and 44% (U.S. Department of Commerce, 2012) of the gross domestic product (U.S GDP in 2012 was $15.68 trillion).

Given the interconnected nature of the air transportation system, a reduction in capacity at the metroplexes results in delays that propagate through the entire system (DeLaurentis & Ayyalasomayajula, 2010; Laskey, Xu, & Chen, 2012).

A key determinant of the capacity of the metroplex airspace is the extent of interaction between flows of aircraft in the airspace (terminal airspace) surrounding the

[1] The Aviation System Performance Metrics (ASPM) database system currently provides detailed data on flights to and from the ASPM airports (currently 77) (ASPM System Overview, 2012).
[2] U.S airports that have .05% or more of the total passenger boarding per year (FAA, 2012a)

metroplex. In some metroplexes, the geometry of the airports and its procedures is such that under certain wind and weather conditions there exists conflicts between flows that require excessive ground holding for departures at one airport and airborne holding for arrivals at the neighboring airport. This results in a reduction in effective capacity of the metroplex airspace, while increasing the potential for added delays and costs to passengers and airlines.

There are six metroplexes in the U.S which have flow conflicts between neighboring airports due to their close proximity and the interdependent arrival and departure procedure, shown in Table 1 (Clarke et al., 2011)

Table 1: U.S Metroplexes with Interdependent Procedures

Sl.no	Metroplex	Ops per day (Year 2012)	# Major Airports
1	New York	3257	3
2	Chicago	3055	2
3	Los Angeles	2797	4
4	Dallas	2236	2
5	San Francisco	1903	3
6	Miami	1734	2

Recent advancements in aircraft navigation and approach capabilities have created opportunities to improve arrival flow efficiencies and de-conflict metroplex flows.

1.1 Required Navigation Performance Approach and Metroplex De-confliction

Airspace navigation has evolved from point-to-point navigation enabled by conventional ground-based navigation systems, to area navigation (RNAV) enabled by a combination of ground-based navigation, inertial referencing systems, and satellite based navigation system (see Figure 1). Further, the addition of monitoring and altering systems on board the aircraft has enabled the aircraft navigation system to monitor its navigation performance, and to identify for the pilot the level of navigation compliance during an operation. The level of navigation compliance is defined by the Required Navigation Performance (RNP) and depends on the aircraft equipment and the navigation infrastructure (FAA, 2012g). These navigational advancements referred to as Performance Based Navigation (PBN), have enabled the implementation of precise curved path approach procedures in the terminal airspace that improve flow efficiencies and de-conflict metroplex airspace.

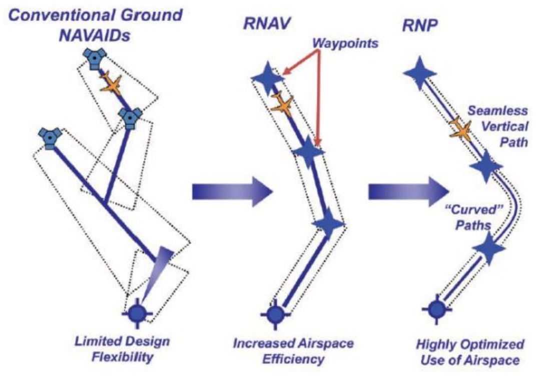

Conventional Ground NAVAIDs
RNAV
Waypoints
RNP
Seamless Vertical Path
"Curved" Paths
Limited Design Flexibility
Increased Airspace Efficiency
Highly Optimized Use of Airspace

Figure 1: Performance Based Navigation (PBN) has enabled the implementation of precise curved path approach procedures in the terminal airspace that improve flow efficiencies and de-conflict metroplex airspace (Source: Ray 2013)

The precise curved path PBN procedure for terminal airspace is called RNP 0.3 approach with Radius to Fix (RF) leg. The "RNP 0.3" is the level of performance required for the approach i.e., the aircraft are required to maintain centerline within 0.3 nautical miles (NM) 95 percent of the time and twice the RNP value, or 0.6 NM, 99.999 percent of the time, and the RF leg refers to the curved path between two fixes (see Figure 2). Using the RNP 0.3 approach with RF leg, aircraft are contained along a precise curved path, allowing safe navigation near high terrain, obstacles and airspace occupied by other flows of air traffic (Ray, 2013).

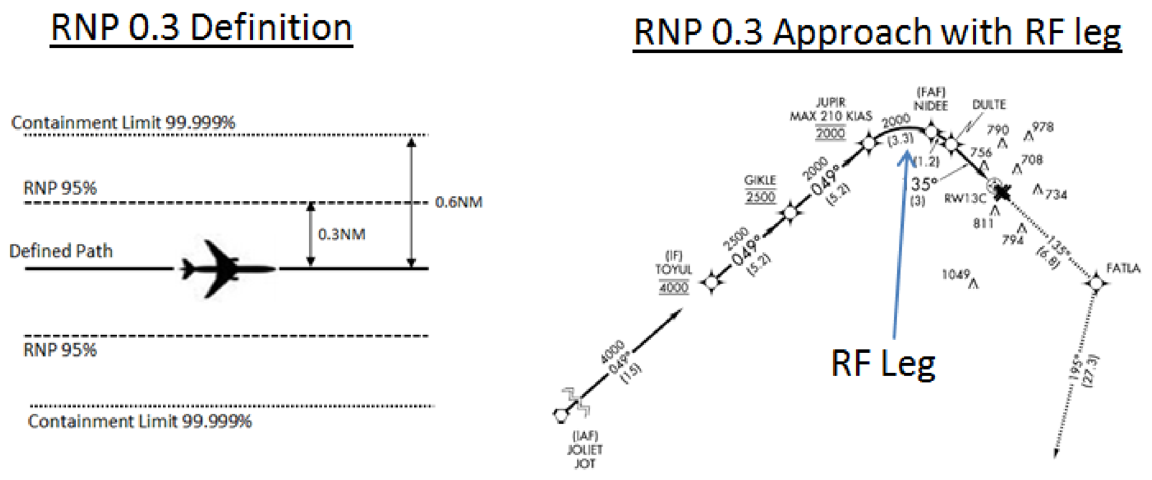


Figure 2: Using the RNP 0.3 approach with RF leg, aircraft are contained along a precise curved path, allowing safe navigation near high terrain, obstacles and airspace occupied by other flows of air traffic

The RNP approach was first deployed in 1996 at Juneau airport in Alaska by Alaska Airlines to improve access and schedule reliability (FAA, 2009). The approach at Juneau during bad weather using conventional ground-based instrument landing system

(ILS) approach (requiring long unobstructed approach path) was not possible due to the tightly encircled mountains. The RNP approach allowed aircraft to navigate with increased precision around the high terrain and to the final approach of the runway. Since then, RNP approach has been deployed world-wide to improve access and schedule reliability to airports in mountainous regions affected by bad weather (details in section 2.5)

In recent years, the use of RNP approach capability has been extended to de-conflicting metroplex airspace. The airspace is de-conflicted by using the curved path RNP approach to make the final approach on to the runway shorter compared to the conventional ILS approach (see Figure 3). This separates the flow of aircraft to one airport away from the flows arriving or departing from a nearby airport.

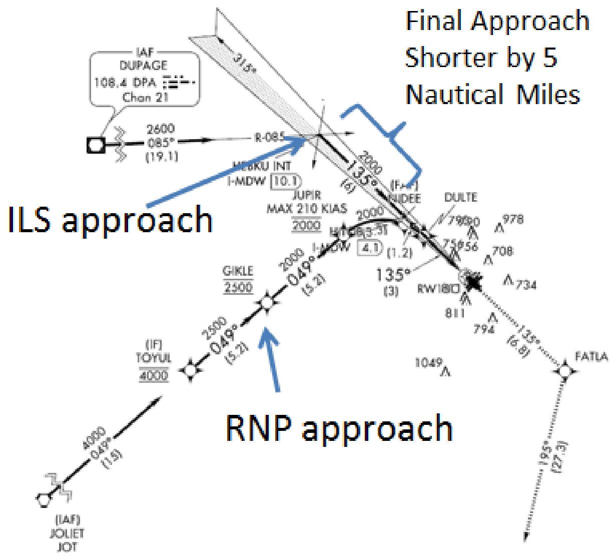


Figure 3: RNP approach "cuts-the-corner" on the final approach to de-conflict terminal area airspace.

The de-confliction of airspace enables the airports at the metroplex to maintain capacity in the event of conditions (low visibility and winds from certain direction) that would otherwise cause the metroplex flow conflict and the resulting drop in capacity of the airspace surrounding the metroplex.

The U.S metroplexes identified as candidates for flow de-confliction are Chicago and New York (FAA, 2012h). The FAA has implemented an RNP approach with RF leg at Midway International Airport (MDW) to de-conflict the Chicago metroplex and is currently testing the curved path RNP approach at John F Kennedy International Airport (JFK) to de-conflict flows at the New York metroplex (FAA, 2012h).

1.1.1 Chicago Metroplex De-confliction

Chicago metroplex is the second largest metroplex in the U.S in terms of traffic volume, with 3055 operations per year (ASPM 2012). It has two airports, the Chicago O'Hare International Airport (ORD) and the Chicago Midway International Airport (MDW) within thirteen nautical miles (NM) of each other. During low ceiling and visibility, and winds from the south east direction, aircraft arriving at MDW are required to use the Instrument Landing System (ILS) on runway 13C. The ILS approach to 13C starts 10.1 NM from the runway threshold and interferes with departures from runway 22L at ORD (see Figure 4 (a)). This is overcome by tactical time sharing of the common airspace, resulting in ground delay for ORD departures and airborne holding for MDW arrivals. Using the new RNP approach with radius to fix leg, aircraft can approach runway 13C without interfering with aircraft departing from runway 22L at ORD (see Figure 4 (b)). This approach procedure results in de-conflicting the metroplex airspace.

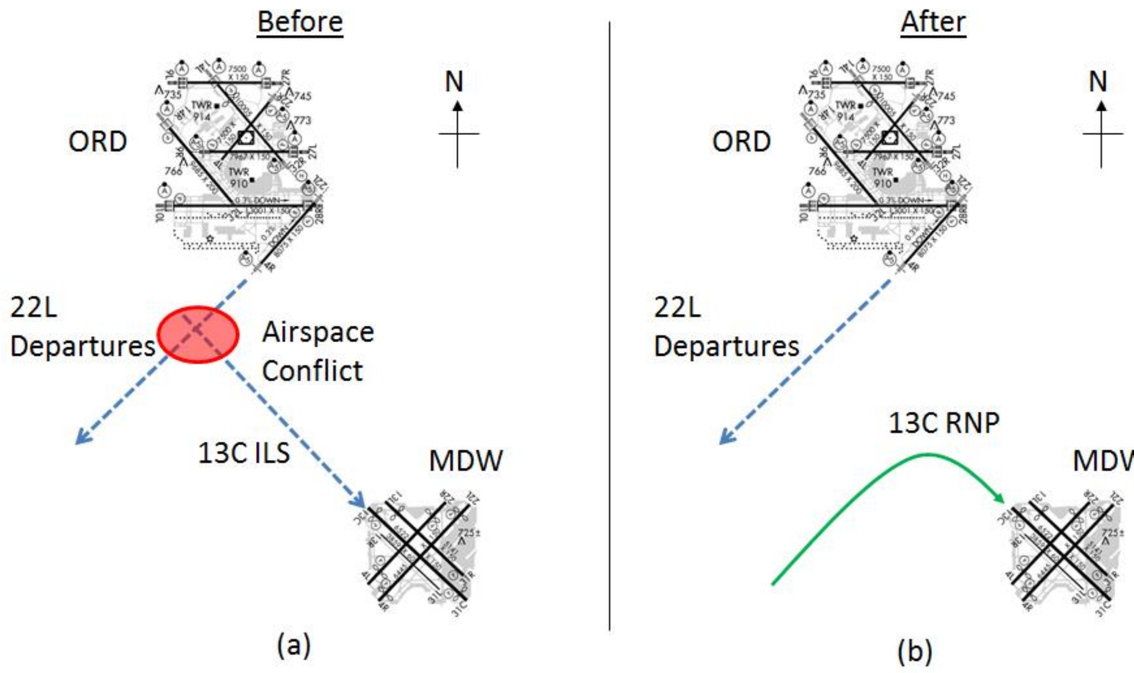


Figure 4: Using the new RNP approach with radius to fix leg to runway 13C at MDW, aircraft can approach runway 13C without interfering with aircraft departing from runway 22L at ORD

1.2 Challenges with Implementing RNP

To enable the RNP approach, the air navigation service providers (ANSPs) must design and approve RNP approaches, and train air traffic controllers. In addition, airlines must equip with RNP equipment, train the crew and achieve certification to fly the procedure. The adoption of RNP approach by airlines has been slow, primarily due to: (a) issues with estimating the Return-on-Investment (ROI) and (b) the “free rider” issue, i.e., the allocation of benefits to parties that choose not to equip but gain benefits when their competition equips.

1.2.1 Estimating the Return on Investment

Equipping with RNP approach capability is expensive. The FAA estimates the cost of adding the equipment to a new aircraft at the time of purchase is $260,000 and the cost of retrofitting an existing aircraft is $525,000 (FAA, 2012i). In addition, the airlines have to account for cost of training and certification of the crew, and the cost of down time associated with retrofitting the aircraft with the new equipment.

The primary benefit of this technology to individual airlines is fuel savings during instrument meteorological conditions (IMC) i.e., low ceiling and visibility, from:

(a) Shorter track distance in the terminal airspace compared to conventional instrument approach procedures

(b) Elimination of airborne holdings that would otherwise occur due to the metroplex flow conflict.

Ideally an airline would want to perform the precise curve path RNP approaches as often as possible and save on fuel burn. However, the candidate airports where RNP approach procedures are being implemented have IMC conditions on average less the 15% of the time in a year. Further, the percentage of time there are flow conflicts at metroplexes is further less. For instance, an analysis of ASPM data for years 2007 to 2012 shows MDW experiences IMC on average 13% of the time per year and a potential flow conflict at Chicago metroplex can occur on average 1.6% of the time per year. This lowers the potential use of RNP approach and the associated fuel burn savings.

The uncertainly in the costs associated with equipping and the actual usefulness of the RNP approach make it difficult to estimate the ROI and thus can prevent the airlines from equipping.

1.2.2 Free rider Issue

For metroplex operations, the additional cost of flow conflict is not evenly distributed among airlines operating at the airports and therefore the benefits from de-confliction by equipage can favor airlines at one airport more than the other. Also, only the airlines at the airport whose arrival flow is causing the airspace conflict are required to equip with the new capability.

For instance, the additional delay cost of flow conflict at Chicago metroplex was estimated at sixteen times more for airlines at ORD than MDW (Devlin, Mills, Porter, & Sprong, 2012). The addition of an RNP approach for arrivals at MDW de-conflicts the airspace and will require the airlines at MDW to equip. This results in airlines at ORD benefitting from investments made by airlines at MDW. This is referred to as the free rider issue.

The competitive nature of the business and the asymmetry in the distribution of benefits may keep airlines from making the investment in the new technology.

1.2.3 Research Questions

The fundamental research questions related to the RNP approach equipage and the associated challenges with it are:

1. Does airline investment in RNP approach capability yield an acceptable Return on Investment (ROI)?
2. Does an airline equipping with RNP approach capability offer a competitive advantage?
3. Are there opportunities to improve ROI?
4. What portfolio of incentives/strategies exists to achieve airline equipage?

1.3 Gaps in the Literature

A review of the existing literature (see section 2) shows the research questions above have not been addressed so far. Further, there are gaps in the type of analysis and the underlying methodology that need to be addressed.

1.3.1 Benefits analysis gaps.

The costs of metroplex flow conflicts and the potential benefits associated with the de-confliction have been analyzed from a system-wide perspective (Clarke et al., 2011; Devlin et al., 2012). The cost and benefits have been expressed in terms of system-wide delays, cancellation and fuel burn. The benefits and ROI from investing in the new PBN approach capabilities to individual airlines have not been analyzed. This is an over sight as airlines make investment based on their benefits, not system-wide benefits.

The fuel burn benefits are computed using time-in-mode method, which assume constant fuel burn rate for a given mode (e.g., descent, climb, and cruise) (Clarke et al., 2011). The main benefits of RNP approach to individual airlines are in terms of fuel burn savings from shorter and more efficient trajectories in the terminal airspace. Hence, it is important to compute fuel burn savings of RNP approach by taking into consideration the actual trajectories of aircraft in the terminal airspace

The cost of airborne holding as a result of the metroplex flow conflict have not been analyzed and quantified. This can be done by analysis of recently available track data.

1.3.2 Methodological gaps

The analyses of metroplex de-confliction are based on simulated de-coupled route structure (Clarke et al., 2011) or delay analysis of operational data (ASPM) (Devlin et al., 2012). These do not capture the actual real-world complexities of traffic flows and aircraft trajectories in the terminal airspace.

The analyses using track data are limited to prescribing methodologies to cluster track data, identifying variation in flows and detecting anomalies (Dorfman, Daily, Gonzalez, & Kondo, 2012; Enriquez, 2013; Levy, 2003; Vempati & Ramadani, 2012). There is lack of a systematic methodology to characterize flows in the terminal airspace for the purpose of differentiating and comparing performance of terminal flows in terms of track distance/time and fuel burn.

1.3.3 Summary of Gaps in the Literature

The existing metroplex de-confliction analyses have been performed from a system-wide perspective using simulated de-coupled routes or low fidelity operational data.

There is a need for a systematic methodology that uses ***high fidelity surveillance track data*** coupled with ***aerodynamic fuel burn model*** and ***weather data*** to estimate the efficiencies and costs of metroplex terminal air traffic flows for assessing benefits of associated concept-of-operations and technologies to individual airlines. This dissertation will address these gaps.

1.4 Research Objectives

The recent availability of high fidelity surveillance track data coupled with aerodynamic fuel burn models, and airport wind and weather data have created an opportunity to provide detailed analysis of metroplex traffic flows to include the real-world complexities of traffic flows and aircraft trajectories.

The objective of this dissertation is to develop a holistic methodology that leverages the accuracy of the high fidelity surveillance track data to:

1. Estimate the Return on Investment of the new PBN approach procedures to individual airlines. This has the following sub objectives:
    a. Estimate the track distance/time and fuel burn performance of the new PBN approach procedures to compare it to conventional approach procedures (i.e., ILS approaches).
    b. Use existing RNP approach flows to model additional potential RNP approaches to other runways at an airport and estimate their associated benefits.
    c. Estimate the fuel burn benefits of using the Optimal Runway Configuration model (see sections 1.9.2, 3.6.2 and 4.4.2).
2. Estimate the benefits of metroplex de-confliction to capture magnitude of the asymmetry and the potential for simultaneous adoption of the technology by the competing stakeholders.

1.5 Summary of the Methodology

This dissertation describes a holistic methodology to use *high fidelity surveillance track data* coupled with *aerodynamic models* and *weather data* to quantify the efficiencies and costs of metroplex terminal area air traffic flows. This methodology assesses the benefits of proposed terminal airspace concepts-of-operations and associated technologies that require simultaneous equipage and development of collaborative procedures by multiple stakeholders (airlines and ANSPs). The methodology includes the following six functions:

1. Perform terminal area flow analysis.
2. Analyze effects of metroplex flow conflict.
3. Define performance metrics and estimate the performance of terminal flows and holding patterns.
4. Estimate the benefits of metroplex airspace de-confliction.
5. Estimate the benefits of PBN approach procedures to an airline at an airport.
6. Estimate the return on investment for the equipped operator.

The first three functions are the building blocks of the overall methodology, which are used to develop models (in functions four, five and six) that annualize benefits of PBN approaches to the metroplex and the individual airlines.

1.6 Unique Contributions

The unique contributions of this dissertation are:

1. A systematic methodology that characterizes terminal flow and estimates the performance of terminal air traffic flows by integrating high fidelity surveillance track data, aerodynamic fuel burn model and airport wind and weather data.
2. A methodology that uses track data of existing RNP approach flows at an airport to model additional potential RNP approach flows to other runways at the airport

3. A methodology that determines the optimal runway configuration by ranking a set of feasible (for the given wind and meteorological conditions) runway configurations based on the weight average fuel burn for runways and selecting the runway configuration with the lowest (best) terminal area fuel burn performance.
4. A methodology for estimating the cost of holding pattern using surveillance track data.
5. Synthesis of micro and macro benefits analysis model that uses high fidelity surveillance track data and low fidelity operational data to estimate:
    a. The benefits of metroplex de-confliction and the associated asymmetry to competing stakeholders, to understand the potential for simultaneous adoption of the technology by the competing stakeholders.
    b. The benefits of PBN approach procedures to airlines at an airport based on the airport's arrival flow performance statistics, while taking into consideration the use of additional PBN approaches and runway configurations.
    c. The ROI of PBN approach procedure to individual airlines.
6. Application of the methodology for an analysis of Chicago Metroplex TRACON (C90) to estimate:

a. The performance of RNP approach to runway 13C by performing at TRACON flow analysis at MDW.

b. The cost of holding for 13C arrivals MDW due to conflicts with departures from 22L at ORD.

c. The annualized benefits of RNP approach to ORD and MDW from de-confliction of flows.

d. The annualized benefits of using Optimal Runway Configuration and RNP approach procedures to all major runways (13C, 31C, 22L, 4R) at MDW.

e. The Return on Investment (ROI) of RNP approach for the majority air carrier at MDW (Southwest Airlines).

1.7 Summary of Results –Chicago Metroplex TRACON (C90) Case Study

The methodology for metroplex air traffic flow analysis is demonstrated in a case-study of the benefits of the introduction of a Required Navigation Performance (RNP) approach procedure for air traffic arrival flows in the Chicago Terminal Radar Approach Control (TRACON), known as C90.

The analysis shows that this airspace, used to service both ORD and MDW, experiences a flow conflict (between 13C ILS arrivals at MDW and 22L departures at ORD) on an average 1.6% of the time per year. This results in holding patterns for 13C arrivals and departure delays for ORD departures. The additional airline direct operating cost per year on an average due this flow conflict is \$.04M for MDW arrivals and \$1.33M for ORD departures. The metroplex airspace is de-conflicted by the introduction of RNP approach to runway 13C at MDW. The ratio of the potential benefits (reduction in additional costs) between airlines at MDW and ORD is 1:33 in favor of ORD departures (non-equipped operators) as a result of the de-confliction. Therefore, there is no competitive advantage for airline at MDW to equip with RNP approach capability. However, the successful de-confliction of Chicago metroplex relies on achieving complete equipage for airlines operating at MDW.

The methodology is applied to perform arrival flow analysis at MDW to compare the performance of the new RNP approach to the conventional approach procedure in order to assess the benefits of the new procedure to the metroplex and to individual airlines. The RNP approach to 13C burns 14% less fuel than the corresponding ILS approach and 25% more fuel than the corresponding visual approach on an average. This limits the benefits of the current RNP approach to runway 13C to the IMC days (1.6% of the time).

Also, without efficient merging and spacing, the benefits of precise curved path RNP approach are not completely achieved as the "vectors" between the final waypoint on the STAR and the start of the RNP approach introduce as much variation in flight tracks as the ILS flows.

The methodology also enables the evaluation of the introduction of additional RNP approach procedures to other runways at MDW. This has the potential of saving on average 660K gallon per year of fuel for arrivals at MDW. At $3/gallon this amounts to a savings of $1.97M per year.

The analysis also identifies an opportunity to select optimal runway configuration at MDW based on wind magnitude/direction *and* flow fuel burn efficiency. The use of optimal runway configuration along with the additional RNP approach procedure has the potential of saving on average 890K gallons per year of fuel for arrivals at MDW. At $3/gallon this amounts to a savings of $2.67M per year.

The results from the analysis are used to estimate the ROI of investing in RNP approach for the major carrier (Southwest Airlines) at MDW. The results show the RNP approach does not yield a positive ROI for Southwest Airlines at MDW. The carrier will have to perform at least half a million RNP approaches per year throughout its network, saving at least 33 kg of fuel per approach on average to break-even in 10 years at a discount rate of 5%.

In conclusion, from an airline perspective (Southwest), the benefits of equipping with RNP approach capability for a single airport (MDW) does not yield a positive ROI. Also, for metroplex markets in which airlines compete there is no competitive advantage in equipping due to the free rider issue. In the case of Chicago metroplex, the competing airlines at ORD get up to 33 times more benefits compared to airlines at MDW from the de-confliction of flows in the metroplex airspace. This amount to 1:9 ratio per flight.

1.8 Strategies to Equipage of RNP

The market based approach relies on the inherent benefits of a technology to sell itself and achieve the desired equipage. In the case of metroplex flow de-confliction and RNP approach capability there are three major issues that negate the benefits of the RNP approach: (a) the terminal area vectoring for merging and spacing required to ensure safe separations, (b) limited potential use of the approach capability i.e. limited to IMC days as fuel burn performance of visual approaches are better than RNP approaches in most cases and (c) the free rider issue i.e. the asymmetric benefits to the competition serving the market. These issue and the high costs of equipage result in low ROI for the airlines.

Operational incentives can be provided to early adopters of the technology to overcome the free-rider issue, however these have limited scope. For instance, as a part of the FAA's Best Equipped Best Served (BEBS) program, a proposal to provide operational incentives in the form of priority arrival slots to equipped operators during traffic flow management initiatives (TMIs) like ground delay program (GDP) was investigated (AhmadBeygi, Bromberg, Elliott, Lewis, & Sud, 2013). Implementation of such TMIs will need new decision support tools and the associated training for the controllers to manage the duration of the program and allocation of slots based on the level of equipage. Also, the priority system will create equity issues for non-equipped operators resulting from excess delay allocation and will increase overall NAS delays due to network wide delay propagation (as a result of large delays for some flights). For example, 15 minutes for four flights can be more easily absorbed by the network than 1 hour delay for a single flight.

In cases of market failure, a theoretical case can be made to provide financial incentives to airspace system users to equip with costly avionics (Post, Wells, Bonn, & Ramsey, 2011). In the case of RNP, the benefits of the operational changes, while disproportionate, not only benefit the equipped operator but also other operations and the system as a whole. This asymmetry in distribution of benefits causes market failure. In such cases financial incentives can be provided to defray the cost of avionics. The financial incentives can be use of public funds, or creation of a tax pool that would tax every stakeholder proportional to the benefits accrued from the operational change. This will require accurate estimates of benefits to individual stakeholders, which can be done using this methodology.

Finally, the last option in the interest of modernization is to mandate the equipage. A federal mandate will ensure modernization of NAS required to meet the future demand (necessary for the growth of the nation). However, for metroplex flow de-confliction a mandate is not economically feasible. The overall cost to equip is higher than the additional airline operating cost due to metroplex flow conflict by orders of magnitude. The additional airline operating cost due to flow conflict at Chicago and New York are $4.5M and $3M (Devlin et al., 2012); whereas the cost to airlines to equip with RNP approach capability is in the hundreds of million ($175M for Southwest Airlines). Also the lack of RNP approaches will restrict the use and the potential benefits of the approach capability. For instance, at Chicago metroplex the airlines at ORD will not have any direct benefits from equipping unless new procedures are put in place to make use of the capability. Therefore, before a mandate to equip for RNP approaches is made the following key issues need to be addressed:

a. The air navigation service providers (ANSPs) must design and approve RNP approaches to all possible runways at all major airports for airlines to use.
b. The ANSPs must train air traffic controllers to smoothly merge and space aircraft at the start of the RNP approach.

1.9 Potential Applications of the Methodology

1.9.1 Analysis tool

The development of the capability to conduct benefits assessment of new concepts-of-operations and technologies using surveillance track data coupled with aerodynamic fuel burn models significantly improves the accuracy and reliability of benefits assessments. The methodology presented in this dissertation can be used to develop an analysis tool that can be used by policy-makers (Air Navigation Service Providers) and investors (Airlines) to better understand where the costs and benefits are accrued.

1.9.2 Optimal Runway Configuration

The Optimal Runway Configuration model built as a part of the methodology can be used in determining the optimal runway configuration for the tower control manager at the airports (see Figure 5). In current practice the runway configuration is determined based only on the wind direction. An alternate approach is to select optimal runway configuration by ranking a set of feasible (for the given wind and meteorological conditions) runway configurations based on the weight average fuel burn for runways and selecting the runway configuration with the lowest (best) terminal area fuel burn performance. The results of the dissertation show that there is potential for further fuel saving using this approach.

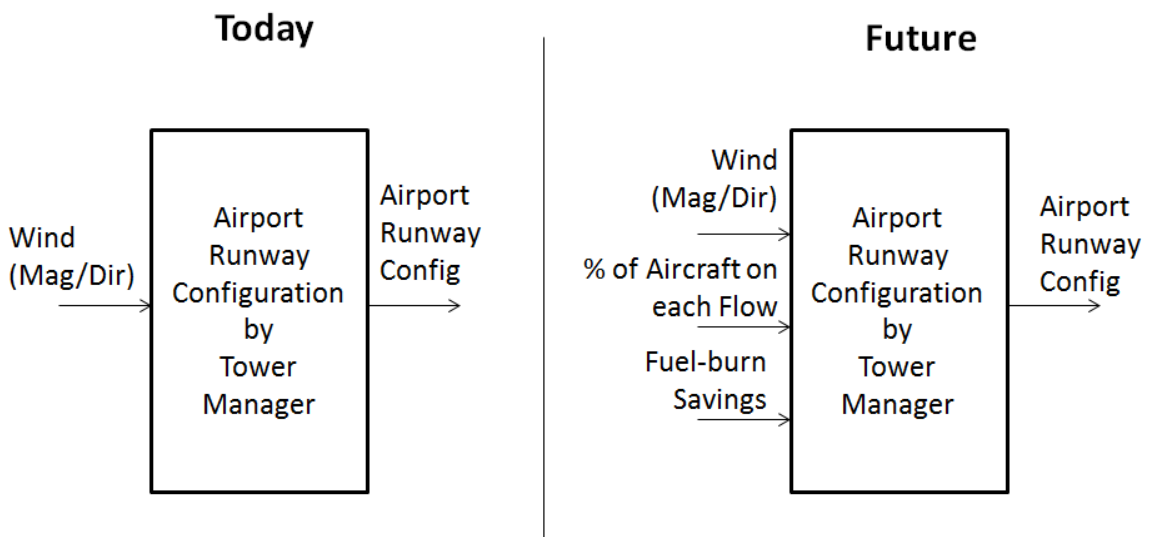

Today
Wind (Mag/Dir)
Airport Runway Configuration by Tower Manager
Airport Runway Config
Future
Wind (Mag/Dir)
% of Aircraft on each Flow
Fuel-burn Savings
Airport Runway Configuration by Tower Manager
Airport Runway Config

Figure 5: Current method for Airport runway configuration use only wind information. The proposed new method uses wind, traffic volume on each flow and estimated fuel burn.

2 CHAPTER2: LITERATURE REVIEW

This chapter is organized as follows: section 2.1 describes the metroplex, the flow conflict at metroplexes and the metroplex de-confliction Con-Ops; section 2.2 reviews the previous research on metroplex air traffic flow analysis; section 2.3 reviews previous research on terminal area flow analyses using track data; section 2.4 reviews previous research on aircraft fuel burn analyses using track data; section 2.5 reviews RNP deployment worldwide; section 2.6 describes the challenges with achieving RNP equipage and the potential for use of a mandate for achieving airline equipage; and section 2.7 provides a summary of literature review and the key gaps in the literature.

2.1 Metroplex Definition

The term “metroplex” was first coined and copyrighted by North Texas Commission (NTC) in 1972, to refer to the larger metropolitan area around Dallas and Fort Worth in Texas (NTC, 2013).

The Joint Planning and Development Office[3] (JPDO), defines metroplex as a group of two or more adjacent airports whose arrival and departure operations are highly interdependent (JPDO, 2007; pg B-6).

The Federal Aviation Administration (FAA) defines metroplex as a geographic area covering several airports serving major metropolitan areas and a diversity of aviation

[3]JPDO was created to manage the implementation of Next Generation Air Transportation System (NextGen) in the United States (U.S)

stakeholders such as National Airspace System (NAS) users, FAA, and other lines of business and airport operators (FAA, 2012e). As a part of the NextGen improvement program called the Optimization of Airspace and Procedures in the Metroplex (OAPM), the FAA has identified 21 metroplexes (Table 2) in the U.S.

Metroplexes are a critical component of the air transportation system and the economy. The 33 ASPM[4] airports at these 21 metroplexes account for more than 48% of the total operations in the NAS's hub[5] airports (FAA, 2012f) . The metropolitan regions these airports serve account for 35% (United States Census Bureau, 2012) of the nation's population (314 million as of 2012) and 44% (U.S. Department of Commerce, 2012) of the gross domestic product (U.S GDP in 2012 was $15.68 trillion).

Table 2: Metroplexes in the U.S, number of major airports in the metroplex, Operations per day, Population and Gross Domestic product.

Sl.no	Metroplex	# of ASPM Airports	Ops per day - ASPM 2012	Population-Year 2012	Gross Domestic Product - Year 2012 in $
1	New York	3	3257	1.98E+07	1.36E+12
2	Chicago	2	3055	9.52E+06	5.71E+11
3	Los Angeles	4	2797	1.31E+07	7.66E+11
4	Atlanta	1	2542	5.46E+06	2.95E+11
5	District of Columbia	3	2434	5.86E+06	4.49E+11
6	Dallas	2	2236	6.70E+06	4.20E+11
7	San Francisco	3	1903	4.46E+06	3.60E+11
8	Miami	2	1734	5.76E+06	2.74E+11

[4] The Aviation System Performance Metrics (ASPM) database system currently provides detailed data on flights to and from the ASPM airports (currently 77) (ASPM System Overview, 2012).
[5] U.S airports that have .05% or more of the total passenger boarding per year (FAA, 2012a)

9	Denver	1	1697	2.65E+06	1.68E+11
10	Charlotte	1	1498	2.30E+06	1.37E+11
11	Phoenix	1	1216	4.33E+06	2.02E+11
12	Detroit	1	1172	4.29E+06	2.08E+11
13	Minneapolis	1	1162	3.42E+06	2.20E+11
14	Las Vegas	1	1138	2.00E+06	9.56E+10
15	Boston	3	947	4.64E+06	3.36E+11
16	Seattle	1	842	3.55E+06	2.59E+11
17	Orlando	1	837	2.22E+06	1.06E+11
18	Memphis	1	735	1.34E+06	6.68E+10
19	Houston	2	497	6.18E+06	4.49E+11
20	Cleveland	1	495	2.06E+06	1.12E+11
21	Tampa	1	495	2.84E+06	1.20E+11

2.1.1 Metroplex Flow Conflict

A key determinant of the airspace capacity serving a metropolitan area with multiple airports is the extent of interaction between arrival and departure flows between the airports. The airports for some “metroplexes” are geographically located such that under certain wind and weather conditions, there exist conflicts between the flows (Atkins, 2008; Clarke et al., 2011; Devlin et al., 2012). This results in excess costs from ground holding for departures and airborne holding for arrivals to resolve the conflict temporarily.

The U.S metroplexes that have interdependent or coupled arrival and departure flows are New York, Chicago, Los Angeles, Dallas , San Francisco and Miami (Clarke et al., 2011).

The New York Metroplex contains three major commercial airports - Newark Liberty International Airport (EWR), John F. Kennedy International Airport (JFK) and LaGuardia Airport (LGA), as well as, another major general aviation airport— Teterboro

Airport (TEB)—within a circle of radius 10 nm. New York airspace is the most complex metroplex in the U.S. The configuration and operations of the airspace depend on the runway configurations at the various airports within the metroplex (Clarke et al., 2011). For instance, landing on runway 13L is a frequent and favored operation at JFK using the Parkway visual approach (FAA, 2012h). When the visual approach must be discontinued, air traffic controllers can ILS approach to runway 13L. This approach, however, has many impacts on the other New York City airports. The final approach segment for the ILS approach is much longer than the visual final approach and it conflicts with LGA's airspace. This forces LGA to use runway 13 for arrivals and creates a conflict between LGA and TEB arrivals allowing only one of the two airports to receive arrivals at a time. Due to these conflicts, JFK ILS 13L approach is rarely used, only being implemented when strong southeast winds eliminate the possibility of using any other runways for arrivals (AhmadBeygi et al., 2013).

The Chicago metropolitan area includes two OEP airports - Chicago O'Hare International Airport (ORD) and Chicago Midway International Airport (MDW)—less than 15 nm from each other. During low ceiling and visibility, and winds from the south east direction, aircraft arriving at MDW are required to use the Instrument Landing System (ILS) on runway 13C. The ILS approach to 13C starts 10.1 NM from the runway threshold and interferes with departures from runway 22L at ORD. This is overcome by tactical time sharing of the common airspace resulting in ground delay for ORD departures and airborne holding for MDW arrivals.

The Los Angeles metroplex has Los Angeles International Airport (LAX) and three airports - Van Nuys Airport (VNY), Long Beach Airport (LGB) and John Wayne-Orange County Airport (SNA) – within 20 NM of each other. The close proximity of these airports causes their arrival and departure paths to cross over and under each other and some of the airports also compete for arrival and departure fixes (Clarke et al., 2011).

The Dallas metroplex has two airports – Dallas Fort Worth International Airport (DFW) and Dallas Love Field airport (DAL) – less than 15 miles of each other. The runway configurations at DFW and DAL are typically aligned, therefore simultaneous visual departures from DAL are not allowed in north flow because their departure paths head toward the DFW departure paths (Clarke et al., 2011). When using instrument-landing-system (ILS) approaches in south flow, only a single stream of arrivals to DAL is allowed to avoid dependency with DFW arrivals because the extended final approach courses of the two airports converge (Clarke et al., 2011).

The Miami Metroplex has two OEP airports - Miami International Airport (MIA) and Fort Lauderdale–Hollywood International Airport (FLL) - within 20 NM of each other. The two airports have interdependent procedures due to their close proximity. However, traffic volume at airports in this metroplex is relatively moderate as compared with many other metroplexes and therefore the dependencies are less severe (Clarke et al., 2011).

2.1.2 Metroplex De-confliction Con-Ops

Metroplex flow conflicts can be resolved temporally or spatially. Further, the temporal or spatially de-confliction can be tactical (short-term) or strategic (long-term) (Clarke et al., 2011).

The tactical-temporal approach involves air or ground holding and speed adjustments by air traffic control (Clarke et al., 2011). The tactical-spatial approach involves vectors (horizontal and/or vertical) by air traffic control (Clarke et al., 2011). The tactical approaches are short term fixes and result in reduced safety margin and an increase in controller workload and costs of operations for the airlines.

The strategic-temporal approach involves NAS wide four dimensional trajectory (4D-T) schedule optimization that de-conflicts aircraft trajectory by assigning each aircraft a Required Time of Arrival (RTA) at the conflicting fixes (Clarke et al., 2011). The flows of aircraft are de-conflicted as long as each aircraft is able to meets it's RTA.

The strategic-spatial approach involves redesign of airspace around the metroplex to de-couple conflicting procedures (Clarke et al., 2011). This can be achieved through design of new precise curved path PBN procedure for terminal airspace called RNP 0.3 approach with Radius to Fix (RF) leg. The "RNP 0.3" is the level of performance required for the approach i.e., the aircraft are required to maintain centerline within 0.3 nautical miles (NM) 95 percent of the time and twice the RNP value, or 0.6 NM, 99.999 percent of the time, and the RF leg refers to the curved path between two fixes (see Figure 2). Using the RNP 0.3 approach with RF leg, aircraft are contained along a precise curved path allowing safe navigation near high terrain, obstacles and airspace occupied by other flows of air traffic (Ray, 2013).

To fly the RNP0.3 w/RF leg approach procedure an aircraft should be equipped with Global Positioning System (GPS) with Approach Capability, or RNP capable Flight Management Computer (FMC). The FMC should be capable of using both ground based navigation aids such as Distance Measuring Equipment (DME) and space-based GPS. It should also be capable to displaying the RF legs (Jeppesen Briefing Bulletin, 2005).

The FAA has implemented an RNP approach with RF leg at Midway International Airport (MDW) to de-conflict the Chicago metroplex (for details see section 1.1) and is currently testing the curved path RNP approach at John F Kennedy International Airport (JFK) to de-conflict flows at the New York metroplex (FAA, 2012h)

2.2 Review of Metroplex Analyses

Research related to the metroplex flow analysis is categorized as follows:

1. Metroplex Flow Conflict Analysis Methodology
2. Metroplex De-Confliction Benefits Metrics

2.2.1 Metroplex Flow Conflict Analysis Methodology

Metroplex flow conflict analysis and de-confliction benefits analysis have been conducted using simulated de-coupled routes (Clarke et al., 2011) and low fidelity operational data (Devlin et al., 2012; Donaldson & Hansman, 2011). The high fidelity surveillance track data is used to estimate excess path length flown by aircraft as a result of the flow conflict (Atkins, 2008) and to visualize the interaction between the various flows at the metroplex (Atkins, 2008; Donaldson & Hansman, 2011).

Metroplex flow conflicts have been analyzed at New York (Clarke et al., 2011; Devlin et al., 2012; Donaldson & Hansman, 2011), San Francisco (Atkins, 2008) and Chicago(Devlin et al., 2012) metroplexes

Clarke et al (2011) identify two strategies, temporal and spatial, to de-conflict New York metroplex operations. The benefits of de-conflicting the airspace using the temporal strategy is done using a scheduling algorithm that determines nominal fix-crossing and departure time to de-conflict flows temporarily. The benefits of de-conflicting the airspace using the spatial strategy is estimated using simulated de-couple routes. The simulation is performed using the New York Airport and Airspace Delay Simulation Model (SIMMOD).

Donaldson & Hansman (2011) analyze New York metroplex using airport operational data (ASPM) to quantify the inefficiencies found in different configurations, and the track data is used to identify the procedures that are likely constraining the airspace. The research prescribes a methodology to identify bottlenecks and their effect on capacity at metroplex airports. The analysis shows that the capacity of the metroplex is lower than the sum of the runway capacities of individual airports in the metroplex and that this capacity gap is due to conflict of flows in the metroplex airspace.

Devlin et al (2012) estimates the Airline Direct Operating Cost (ADOC) due to flow conflicts at New York and Chicago metroplexes. The analysis is conducted by using the airport configuration and weather information in the ASPM data. The data is used to identify time periods when airspace conflicts. The total minutes of arrival and departure delay and the number of cancelled flights are computed for the conflict periods. These are compared to the delays and cancelled flights during similar calendar and schedule time periods when the airspace conflict did not occur. Finally, the Airline Direct Operating

Cost (ADOC) due to the excess delay and cancellation as a result of airspace conflict are estimated.

Atkins (2008) analyzes San Francisco metroplex to generalize metroplex phenomenon i.e. interdependencies and sharing of resources (airspace, fixes, and routes) between proximate airports that result in reduced capacity or efficiency (Atkins, 2008). A detailed description of the operational issue at the San Francisco metroplex for two commonly used flows patterns, the West Plan and the South-East Plan is described. The analysis is conducted using the Enhanced Traffic Management System (ETMS) track data, using the Surface Operations Data Analysis and Adaptation (SODAA) tool. The ETMS data is used to visualize various flows in the San Francisco metroplex and to compute excess path length flown by aircraft as a result of flow conflict.

2.2.2 Metroplex De-confliction Benefits Metrics

The costs of metroplex flow conflicts and the potential benefits associated with the de-confliction have been analyzed from a system-wide perspective (Clarke et al., 2011; Devlin et al., 2012).

The costs of metroplex flow conflict have been expressed in terms of total Airline Direct Operating Cost (ADOC) per year (Devlin et al., 2012). The additional ADOC at New York metroplex (due to flow conflict between JFK and LGA) is $751,100 at JFK and $2,268,100 at LGA (Devlin et al., 2012). The additional ADOC at Chicago metroplex (due to flow conflict between MDW and ORD) is $275,000 at MDW and $4,365,000 at ORD (Devlin et al., 2012)

The benefits of de-coupling metroplex flows have also been expressed in terms of reduction in system-wide delays and fuel burn (Clarke et al., 2011). At New York metroplex the benefits of de-coupling the airspace spatially and temporarily is estimated using simulated de-couple routes using the New York Airport and Airspace Delay Simulation Model (SIMMOD). The results show that when applied separately the spatial and temporal de-confliction result in delay reduction of 28% and 60% respectively. Combined together the hybrid de-confliction resulted in delay reduction of 79% (Clarke et al., 2011). The de-coupled routes resulted in a system-wide fuel burn savings of 11%. The fuel burn benefits are computed using time in mode method, which assumes constant fuel burn rate for a given mode (descent, climb, and cruise) (Clarke et al., 2011)

2.2.3 Need for Analysis using High Fidelity Surveillance Track Data

The successful implementation of the metroplex de-confliction Con-Op (i.e., using new precise curved path PBN approach procedures to spatially de-couple conflicting metroplex flow) relies on achieving the airline equipage at the metroplex airports. Airlines invest in equipage for two reasons: (a) if the benefits of the equipage yield an acceptable ROI and (b) if equipage is required to meet regulatory requirements.

The metroplex flow conflict analyses have been performed from a system wide perspective and not from an individual airline's benefit perspective (Atkins, 2008; Clarke et al., 2011; Devlin et al., 2012; Donaldson & Hansman, 2011). The benefits and ROI from investing in the new PBN approach capabilities to individual airlines have not been analyzed. This is an oversight as airlines make investment based on their benefits and not system-wide benefits.

The metroplex flow de-confliction benefits analysis have been performed using simulated de-coupled routes (Clarke et al., 2011) or low fidelity operational data (Devlin et al., 2012). They do not capture the interaction between conflicting flows and their associated costs. The cost of airborne holding as a result of the metroplex flow conflict has not been analyzed and quantified.

The primary benefits of the precise curved path PBN approach procedures to individual airlines is in fuel burn savings from more shorter and more efficient trajectories in the terminal airspace. To compute the fuel burn savings of these approach procedures, the actual trajectories of aircraft in the terminal airspace must be taken into consideration.

There is a lack of a systematic approach to quantify the potential savings in fuel burn to airlines in using new PBN (RNP approach) approach procedures instead of conventional approaches. An assessment of benefits of fuel burn savings from RNP approach requires detailed track flow analysis that compares performance of RNP approach flows to convectional flows.

There is a need for a detailed analysis using high fidelity surveillance track data that captures real-world complexity of traffic flows and aircraft trajectories, characterizes terminal area flows and compares flows using statistics of flow performance metrics (i.e. track distance, time, fuel burn in the terminal airspace).

2.3 Review of Track Flow Analyses

The National Offload Program (NOP) data has flight track data for Terminal Radar Approach Control Facilities (TRACONs). The flight track data contains an identifying flight number and flight status (arrival, departure, or overflight), as well as, position reports of latitude, longitude, altitude, and time-of-report (DeArmon et al. 2011). A sample plot of Chicago TRACON (C90) NOP track data is shown in Figure 6. The metroplex terminal operations are a complex interaction of flows. To understand the effects of these interactions the individual flows at metroplex airports need to be analyzed. This section reviews existing research in the area of terminal track flow analysis to identify gaps in the existing methodology for characterizing and computing performance metrics for terminal area air traffic flows.

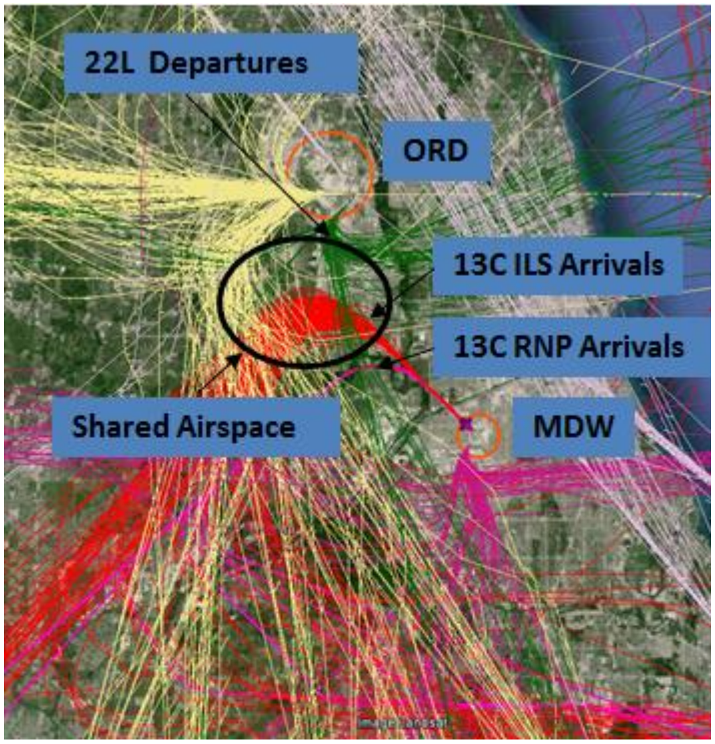


Figure 6: Traffic flow interaction at Chicago Metroplex

Levy (2003) presents a methodology for the mathematical characterization of three-dimensional airspace traffic flows from flight position data (Levy, 2003). The methodology uses the mean and the standardized skew of Normalized Cross-Track Distance (NCTD) to rank tracks and pick a typical track (or back bone) characterizing a flow. The NCTD is defined as the ratio of the cumulative cross-track distance and track-line distance. The NCTD value compares the tracks with the typical track to identify the traffic pattern. Three traffic patterns are defined based on the efficiency of the tracks, 'expedite', 'nominal' and 'delay'. Statistics are reported for each traffic pattern in terms of the NCTD rank and, the length, duration and ground speed of the tracks. The analysis identifies and compares traffic patterns within a flow; it does not extend the methodology to compare traffic patterns between different flows.

Dorfman et al (2012) analyze the vertical profile flight tracks using track data (Dorfman et al., 2012). The flows are defined by a start point and an end point (referred to as way-triangles in the analysis) and all tracks that pass through the start and the end point are assigned to the flow. The analysis highlights inefficiencies and potential for improvements in the vertical profiles of aircraft. The analysis does not compute cost of level off in terms of fuel burn and does not compare various flows in the terminal area.

Vempati & Ramadani (2012) present a methodology to measure the utilization of procedures implemented across the National Airspace System (NAS) (Vempati & Ramadani, 2012). The flight tracks are assigned to a procedure by checking the vertical and lateral proximity of the track to the published procedures along the track length. The paper focusses on the accuracy with which a flight track is assigned to a procedure. The assignment count is validated against the pilot-controller voice communications, the airline reported RNP usage and the scratch pad tally maintained as the TRACON. The analysis does not compute performance metric for the flow, nor does it analyze the relative performance of flow with respect to each other.

Enriquez (2013) presents a methodology identifying temporally persistent flows in the terminal area via spectral clustering (Enriquez, 2013). Spectral clustering uses graph partitioning approach to accomplish the grouping of flights. The paper explains the application and challenges of applying spectral clustering to group flight tracks into flows. The clustering algorithm is sensitive to the values of the clustering tolerance and requires calibration. The application of the methodology is limited to identifying irregular terminal operations, detecting flows that do not adhere to any of the published procedures and recommending a need to publish more RNAV procedures.

Gariel, Clarke, & Feron (2007) describe a methodology to analyze impact of TRACON capacity on terminal area delay and airport efficiency (Gariel, Clarke, & Feron, 2007). The analysis uses track data to estimate the arrival rate of aircraft in the TRACON, the number aircraft vectored in the terminal area and the delays associated with vectoring. These estimates are then used to build and calibrate a TRACON queuing and landing simulation model, which evaluates the impact of TRACON capacity on terminal area delays and airport efficiency. The analysis is limited to estimating delays and runway utilization as a function of TRACON capacity.

In summary the research on terminal flow analysis so far is limited to clustering of track data into flow. The methodologies have not been extended to computing performance metrics (in particular fuel burn) for terminal flow. Also, there is lack of a systematic approach in characterizing terminal area flows for comparing the performance of new PBN approach procedures with conventional approach procedures. The key benefit of new PBN approach to airlines is in terms of fuel burn savings. The next sub section review existing research on fuel burn analyses.

2.4 Review of Fuel burn analyses

Fuel costs currently constitute the largest fraction (29%) of an airline's operating cost (A4A Cost Index, 2012); therefore, it is important to evaluate and compare performance of terminal area flow using fuel burn estimates. This section presents a review of research on aircraft fuel burn analysis.

The fuel burn benefits of de-coupled metroplex airspace are computed using the standard fuel burn rate in the Landing and Takeoff (LTO) cycle of the International Civil Aviation Organization (ICAO) (Clarke et al., 2011). The ICAO fuel burn model uses a linear time-in-mode method, which assumes constant fuel burn rate and a standard duration (time) for a given mode (descent, climb, and cruise). The fuel burn for each aircraft type is estimated as the product of standard fuel burn rate and time for a given mode.

A comparison of the actual fuel burn information from aircraft's flight data recorder (FDR) and the ICAO model shows that total fuel burn for both departures and arrivals is overestimated by the ICAO method (i.e., actual fuel burn is between 70-85% of the ICAO maximum for each engine) (Patterson, Noel, Senzig, Roof, & Fleming, 2009). The comparative analysis is based on data collected for 2824 flight records, from 5 different airlines, with 14 unique engine combinations. The result suggests that while using ICAO method may be appropriate in comparative policy analyses, it is not suitable for comparing performance of flow in the terminal area, which has a tremendous variety in track profile of flow patterns.

Fuel burn for terminal area flows can be estimated within $\pm$5% actual fuel consumption using a regression model to estimate the thrust specific fuel consumption (Senzig, Fleming, & Iovinelli, 2009). The co-efficient of the regression expression are estimated for each airframe/engine combination based on aircraft performance data for an expected range of terminal-area operations. However, the use of this approach is limited by the availability of accurate fuel burn data required to accurately estimate the regression coefficients.

An aerodynamic model that uses actual flight trajectory, standard fuel flow and drag models can estimate fuel burn for terminal area flows within $\pm$5.4% of the actual value (Chatterji, 2011). This is provided accurate information is available for the wind, flight's position report and initial mass (Chatterji, 2011).

In summary a review of research on fuel burn model suggest that using standard time in mode and fuel burn rate will fail to capture the variation the vertical and lateral profile of terminal flows. The primary benefits of the precise curved path PBN approach procedures to individual airlines is in fuel burn savings from more shorter and more efficient vertical trajectories that do not level off in the terminal airspace. Using a hybrid fuel burn model that uses *high fidelity surveillance track data* coupled with *aerodynamic models* and *weather data* will capture the benefits of more shorter and efficient PBN approach procedures in the terminal airspace. The level accuracy of the fuel burn model based on actual trajectory provides enough motivation to use them in estimating and comparing performance of terminal flows

2.5 Review of RNP deployment world wide

This section describes worldwide deployment of RNP approach. The deployment of RNP approach procedures in the U.S, Canada, Australasia, Asia, Europe and South America are described in the following subsections. The benefits gained from implementation of RNP are summarized in the last subsection

2.5.1 RNP deployment in the United States

A summary of RNP approach deployment in the U.S is shown in Table 1. The RNP approach was first used in 1996 at Juneau Airport in Alaska by Alaska Airlines to improve access and schedule reliability (FAA, 2009). The approach at Juneau during bad weather using conventional ground-based instrument landing system (ILS) approach (requiring long unobstructed approach path) was not possible due to the tightly encircled mountains. The RNP approach allowed aircraft to navigate with increased precision around the high terrain and to the final approach of the runway.

In 2002, Horizon Airlines, a subsidiary of Alaska Airlines, initiated implementation of RNP approach procedures for airports in its network (Aviation Today, 2002). Horizon is a regional carrier in the northwestern United States. It operates from airports in mountainous terrain that are situated around its hubs, Seattle-Tacoma International Airport (SEA) and Portland International Airport (PDX). The use of RNP for approaches resulted in increased access, lower approach minima and schedule reliability at all its airports.

Table 3: RNP Deployment in the U.S

Year	Airline	Fleet Type	Location	Issue	Benefit type	Benefit Estimate
1996	Alaska Airlines	737-700,-900,-400,-200QC, MD-80	Started at Juneau, Alaska	Terrain, Bad weather	Increased Access - Lower Approach Minima, DA, Schedule Reliability	-
2002	Horizon Airlines	Dash 8, CRJ 700	Horizon's network	Terrain, Bad weather	Increased Access - Lower Approach Minima, DA, Schedule Reliability	-
2007	Southwest	737NG, 737Classics	Southwest hubs (BWI, MDW,DAL,LAS,HOU, PHX)	Operational inefficiency	Operating Cost, Schedule Reliability	-
2009	ConocoPhillips	737-700	Deadhorse, Alaska	Terrain, Weather	Increased Access	12650 gallons of fuel, 250 tons of CO2 reduction per year
2012	JetBlue	A320	KJFK, New York	Operational inefficiency	Operating Cost, Schedule Reliability	18 gallons fuel savings per flight

In 2007, Southwest Airlines contracted with GE Aviation – formerly Naverus – to develop tailored RNP approach procedures for all its operations. The cost of this transformation is estimated at $175 million (Hughes, 2008). The RNP approach so far had been adopted by airlines operating at terrain challenging high altitude airports to increase access and improve schedule reliability. Southwest Airline is the first airline to implement RNP approach with goal of reducing fuel burn and emissions, by having more efficient approaches compared to the conventional approaches.

In 2009, ConocoPhillips Airlines implemented RNP approach for its operations into Deadhorse Airport (PASC), Alaska. Depending on the runway in use, the new procedures were reported to reduce CO_2 emissions by at least 250 tons and jet fuel consumption by at least 12,650 gallons annually (GE Aviation, 2011).

In 2012, as a part of the joint venture with FAA, JetBlue conducted test flight into JFK using an RNP approach on to runway 13L. The precise curved path approach procedure cuts corner on the final approach and is expected to reduce fuel burn by 18 gallons per flight (Aviation Today, 2012b).

2.5.2 RNP deployment in Canada

RNP approach was first implemented in Canada, at Kelowna International Airport (CYLW) in 2003, by WestJet. Like Alaska, airports in Canada are affected by terrain and weather. The implementation of RNP approach at CYLW resulted in 41 Nautical Miles (NM) track-miles savings per flight (GE Aviation, 2011). This corresponds to 0.5 tons of fuel savings and 1.6 tons of CO_2 reduction per flight. WestJet currently has about 50 RNP approaches into 18 airports in Canada. These procedures save on average 10 track-miles per flight (GE Aviation, 2011).

Table 4: RNP deployment in Canada

Year	Airline	Fleet Type	Location	Issue	Benefit type	Benefit Estimate
2003	WestJet	737-600,-700,-800	Started at Kelowna, Canada	Terrain restricted ILS DA, Weather	Lowered DA by 310 feet, fewer diversion in bad weather.	41 NM track mile saving, 516kg of fuel, 1.6 ton CO2 reduction.

2.5.3 RNP deployment in Australia

RNP approach was first implemented in Australasia, at Queenstown International Airport (NZQN) in 2004, by Air New Zealand and Qantas (GE Aviation, 2011). The goal of implementing the RNP approach was to improve schedule reliability, which was affected by the terrain and weather at NZQN. The RNP approach resulted in 11NM track-miles savings per flights. This corresponds to 0.2 ton of fuel savings and 0.6 tons of CO_2 reduction per flight (GE Aviation, 2011).

Table 5: RNP deployment in Australia

Year	Airline	Fleet Type	Location	Issue	Benefit type	Benefit Estimate
2004	Air New Zealand	A320s	Queenstown, New Zealand	Terrain, Bad weather	New DH 250ft, Schedule Reliability	11NM track mile saving per procedure, 192kg of fuel saving, 603kg of CO2

						reduction
2004	Qantas Airways	737s	Queenstown, New Zealand	Terrain, Bad weather	New DH 250ft, Schedule Reliability	-
2006	Qantas , and Others	737s	Brisbane, Australia	Operational inefficiency	Successful sequencing of RNP and Non-RNP flights	2.6 minutes saved per flight, 126kg of fuel, 390kg of CO2 reduction

In 2006 Brisbane Green project was initiated by Airservices Australia in collaboration with GE Aviation, Qantas Airways and Civil Aviation Safety Authority of Autralia (CASA). The goal of this project was successful sequencing of RNP and Non-RNP flights into Brisbane Airport (YBBN). The new procedure saved on an average 2.6 minutes per flight (Airservices Australia, 2008). This resulted in 126 kg fuel savings and 390 kg CO_2 reduction per flight

2.5.4 RNP deployment in Asia

In 2007, Chinese Airlines in Asia started to use RNP approaches to improve access and schedule reliability at in high altitude airports in Tibet and China. In 2009, China Southern became the first airline to use tailored RNP approach for a wide-body (A330) aircraft into Lhasa airport (LXA), a mountainous high altitude airport (GE Aviation, 2011)

Table 6: RNP deployment in Asia

Year	Airline	Fleet Type	Location	Issue	Benefit type	Benefit Estimate
2007	Air China	A319	Linzhi, Tibet	Terrain, Bad weather	Increased Access to Linzhi, Schedule Reliability	-
2009	China Southern	A330	Lhasa, Tibet	Terrain, Bad weather	Increased Access to Lhasa, First for a wide body aircraft	-
2009	China Eastern	A319, B737	Lhasa, Tibet Yushu, China	Terrain, Bad weather	Increased Access to Lhasa	-
2010	Sichuan Airlines	A319	Lhasa, Tibet Lijiang, China	Terrain, Bad weather	Increased Access to Lhasa	-
2012	Eithad	A330-200	Abu Dhabi, UAE	Operational inefficiency	Operating Cost, Schedule Reliability	9% fuel burn savings
2012	IndiGo	A320	Kochi, India	Operational inefficiency	Operating Cost, Schedule Reliability	400kg fuel saving per flight

In 2012, Eithad Airline became the first airline in the middle-east to use RNP approach. By redesigning the horizontal and vertical flight paths of flights coming from the west, this new technology will reduce noise overflying the city of Abu Dhabi and optimize fuel consumption. Etihad estimates fuel consumption will be reduced between 100 kg and 200 kg per approach, which will result in a reduction of CO2 emissions by at least 20,000 tons per year (Aviation Today, 2012a)

In 2012, Indigo Airline flew its first RNP approach into Kochi International Airport (COK). Indigo is a low cost carrier operating in India. The new procedure is expected to save 400kg of fuel per flight (Air Transport World, 2012).

2.5.5 RNP deployment in Europe and South America

RNP in Europe is driven by the Minimum CO2 in Terminal Maneuvering Area (MINT) project. The first MINT demonstration flight took place on the 16th of June 2009, using the newly developed RNP procedure into Stockholm-Arlanda Airport (ESSA). The new procedure resulted in 20 NM track-mile savings (Euro Control, 2012a).

Table 7: RNP deployment in Europe and South America

Year	Airline	Fleet Type	Location	Issue	Benefit type	Benefit Estimate
2009	NovAir	A321	Stockholm Arlanda, Sweden	Operational inefficiency	Operating Cost, Environment	20NM track-mile savings per flight.
2009	LAN Airlines	A319	Cuzco,Peru	Terrain, Bad weather, Diversion	Increased Access, Schedule reliability	-

LAN airlines deployed RNP approach for its A319 fleet at Cuzco Airport (SPZO) in May 2009. Prior to the deployment of RNP, about 10% of LAN's arrivals into SPZO would be diverted, due to a combination of poor weather and low visibility coupled with the surrounding rugged terrain (GE Aviation, 2011).

2.5.6 Summary of RNP deployment

The application of RNP for approach provides benefits in terms of improved access, schedule reliability and savings in track miles and fuel burn. For about 70% of the airlines, RNP approach procedures improved access to airports located in mountainous terrain and affected by bad weather most of the time. Redesign of the horizontal and vertical flight paths using RNP 0.3 approach with RF leg can optimize fuel consumption. Depending upon the relative location of the start of the RNP approach with respect to the approach direction, the new procedure can save 10 to 40 nautical miles in the terminal airspace. This corresponds to 20 to 100 gallons of fuel savings and 0.5 to 1.6 tons of CO_2 reduction per flight.

2.6 Achieving Equipage for Required Navigation Performance Approach

The RNP 0.3 approach with RF leg capability is a key enabler for the metroplex de-confliction Concept-of-Operations (ConOps). However, for the ConOps to be fully functional, aircraft must be equipped with the associated avionics (FAA, 2012g).

The challenge for the Federal Aviation Administration (FAA) is that the adoption of RNP approach technology by airlines has been slow and haphazard, as equipping with RNP approach capability is expensive. The FAA estimates the cost of adding the equipment to a new aircraft at the time of purchase is $260,000 and the cost of retrofitting an existing aircraft is $525,000 (FAA, 2012i). The airlines also have to account for cost of training and certification of the crew and the down time associated with retrofitting the aircraft with the new equipment. In addition, the need for collaboration and simultaneous equipage across competing stakeholders and the allocation of benefits to parties that choose not to equip but gain benefits when their competition equips have created a "modernization stalemate."

Programs like Best Equipped Best Serve (BEBS) are being developed by the FAA to provide airlines operational incentives to equip with the technology (FAA, 2012h). The proposed operational incentives are in the form of priority arrival slots to equipped operator during traffic flow management initiatives (TMIs) (AhmadBeygi et al., 2013). The proposed TMI identifies periods (on flow conflict days) when equipage based priority can be applied. During these time periods, referred to as exclusionary periods, the metroplex flows are de-conflicted by only allowing equipped aircraft at the airports. This allows metroplex to resume normal operations, but penalizes flights that are not equipped to fly the procedure required to de-conflict the metroplex.

The implementation of such TMIs will need new decision support tools and the associated training for the controller to manage the duration of the program and allocation of slots based on the level of equipage (AhmadBeygi et al., 2013)

2.6.1 Mandate for Achieving Equipage

Airline equipage for improving capacity and/or safety of the NAS has been achieved through mandates. A summary of past modernization mandates are as follows:

The use of Very High Frequency (VHF) radio instead of High Frequency (HF) radio was mandated in 1961 to improve operational efficiency. The mandate requires two-way VHF radio communications for conducting flight operations on and around all controlled airports throughout the country (FAA, 2012b). The VHF radio provides higher bandwidth and voice clarity. Higher bandwidth means availability of more number of channels, which boosts' airspace capacity.

Transponders were mandated in 1978. The mandate requires all aircraft operating in Terminal Radar Service Areas (TRSAs) and Terminal Control Areas (TCAs) to have transponders to report identity (Mode A) and altitude (Mode C) installed by July 1981 (FAA, 2012b). An aircraft equipped with a beacon transponder can provide the terminal controller automatically with information on its identity, altitude, range, and bearing. This improves air traffic control service in terms of being able to safely handle higher level of traffic. Under the old system, the controller obtained an aircraft's altitude and identity only through voice contact with the aircraft's pilot.

The Traffic Collision Avoidance System (TCAS) was mandated in 1981 to prevent mid-air collision and improve NAS level of safety (FAA, 2012b). TCAS is a safety monitoring system independent of air traffic control, which monitors the airspace around an aircraft for other aircraft equipped with a corresponding active transponder and warns pilots of the presence of other transponder-equipped aircraft which may present a threat of mid-air collision. In the event of a potential mid-air collision the system also maneuvers the aircraft involved to avoid collision.

The Wind Shear equipage was mandated in 1988 (FAA, 2012b). Wind shear is a difference in wind speed and direction over a relatively short distance in the atmosphere. Presence of wind shear in the final approach to landing results in a decrease in aircraft's airspeed and an increase in the sink rate. This results in ground contact before the runway threshold (crash landing). The pilot must adjust the airspeed to deal with the effect of wind shear. The mandate requires all turbine-powered airliners seating 30 passengers or more to carry equipment to warn pilots when they encounter low-altitude wind shear and provide them with information needed to escape safely (FAA, 2012b).

The Reduced Vertical Separation Minima (RVSM) equipage was mandated in the U.S following a mandate in Europe in 2002, to increase capacity of airspace for flight level (FL) 290 to 410. The mandate requires all aircraft and flight crews operating in the NAS between flight level (FL) 290 to 410, to be RVSM compliant as of January 20, 2005 (FAA, 2012g). RVSM certified aircraft uses a certified altimeter which has an Altimetry System Error[6] (ASE) of less than 245 feet (Euro Control, 2012c). The reduced ASE enables the reduction of vertical separation requirement to 1000 feet from a previous requirement of 2000 feet for flight level (FL) 290 to 410. RVSM enhances ATC flexibility, mitigates conflict points, enhances sector throughput, reduces controller workload and enables crossing traffic. Operators gain fuel savings and operating efficiency benefits by flying at more fuel efficient flight levels and on more user preferred routings (FAA, 2012g).

2.7 Summary of Literature Review

The term Metroplex refers to a system of airports serving a large Metropolitan area (FAA, 2012e). The airports in a metroplex are often in close proximity to each other and can have interdependent arrival and departure procedures (JPDO, 2007).

Metroplexes are a critical component of the nation's economy and the air transportation system. A key determinant of the airspace capacity serving a metropolitan

[6] ASE is the difference between the altitude that the pilot, ground controller and aircraft systems believe the aircraft to be at and the actual altitude

area with multiple airports is the extent of interaction between arrival and departure flows between the airports. The airports for some “metroplexes” are geographically located such that under certain wind and weather conditions conflicts exist between the flows. This results in excess costs from ground holding for departures and airborne holding for arrivals.

Air traffic flows at a metroplex can be decoupled by re-design of airspace (metroplex de-confliction Con-Op) through implementation of new precise curved path PBN procedure for terminal airspace called RNP 0.3 approach with Radius to Fix (RF) leg. Using the RNP 0.3 approach with RF leg, aircraft are contained along a precise curved path allowing safe navigation near high terrain, obstacles and airspace occupied by other flows of air traffic (Ray, 2013).

The successful implementation of the metroplex de-confliction Con-Op (i.e. using new precise curved path PBN approach procedures to spatially de-couple conflicting metroplex flow) relies on achieving the airline equipage at the metroplex airports. Airlines invest in equipage for two reasons: (a) if the benefits of the equipage yield an acceptable ROI and (b) if equipage is required to meet regulatory requirements

The metroplex flow conflict analyses have been performed from a system wide perspective and not from an individual airline’s benefit perspective (Atkins, 2008; Clarke et al., 2011; Devlin et al., 2012; Donaldson & Hansman, 2011). The benefits and ROI from investing in the new PBN approach capabilities to individual airlines have not been analyzed. This is an oversight as airlines make investment based on their benefits and not system-wide benefits.

There is a need for a systematic methodology that uses ***high fidelity surveillance track data*** coupled with ***aerodynamic fuel burn model*** and ***weather data*** to estimate the efficiencies and costs of metroplex terminal air traffic flows for assessing benefits of associated concept-of-operations and technologies to individual airlines. This dissertation will address these gaps.

3 CHAPTER 3: METHODOLOGY

The recent availability of high fidelity surveillance track data coupled with aerodynamic fuel burn models, and airport wind and weather data have created an opportunity to provide detailed analysis of metroplex traffic flows to include the real-world complexities of traffic flows and aircraft trajectories.

This section describes a holistic methodology that uses *high fidelity surveillance track data* coupled with *aerodynamic models* and *weather data* to quantify the efficiencies and costs of metroplex terminal area air traffic flows.

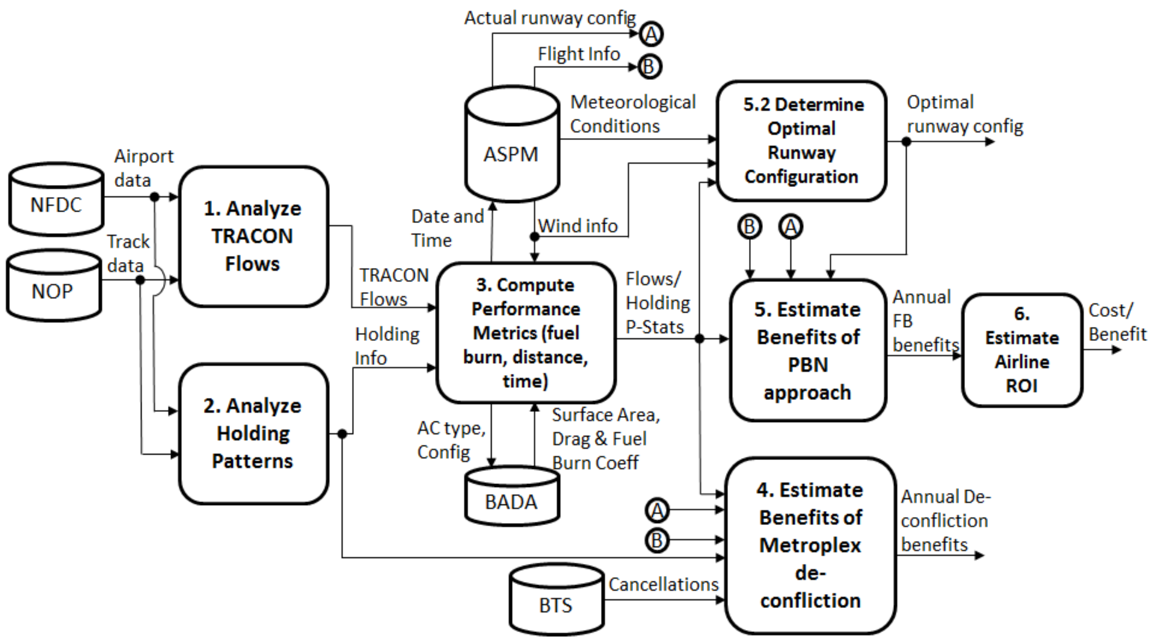
This methodology is intended for assessing benefits of proposed terminal airspace concepts-of-operations (e.g. metroplex de-confliction using RNP approach) and associated technologies that require simultaneous equipage and development of collaborative procedures by multiple stakeholders (airlines and ANSPs).

An overview of the methodology is shown in Figure 7. The methodology includes the following six functions:

1. Perform terminal area flow analysis: this involves characterizing terminal flows and assigning track data to each flow.
2. Analyze effects of metroplex flow conflict: this includes holding pattern analysis using track data

3. Define performance metrics and estimate the performance of terminal flows and holding patterns: this involves estimating the performance metrics for terminal area flows and holding patterns, in particular the fuel burn using track data and aerodynamic fuel burn model.
4. Estimate the benefits of metroplex airspace de-confliction: this involves identifying the effects of the metroplex flow conflict, annualizing the severity of the effects and estimating the benefits of de-conflicting the metroplex.
5. Estimate the benefits of PBN approach procedures to an airline at an airport: this involves annualizing the benefits of using the new PBN approach procedures to all possible runways at an airport, in addition to using the optimal runway configuration.
6. Estimate the return on investment for the equipped operator: this involves estimating the Net Present Value (NPV) based on the investment made in equipping and the annual benefits from using the new technology.

The first three functions are the building blocks of the overall methodology. These involve analysis of surveillance track data to estimate the track distance/time and fuel burn performance for terminal air traffic flows and holding patterns in the metroplex airspace. These building blocks are used to develop models (in functions four, five and six), that annualize benefits of PBN approaches to the metroplex and the individual airlines



Actual runway config
A
Flight Info
B
ASPM
Meteorological Conditions
5.2 Determine Optimal Runway Configuration
Optimal runway config
Airport data
NFDC
Track data
NOP
1. Analyze TRACON Flows
Date and Time
Wind info
B
A
TRACON Flows
3. Compute Performance Metrics (fuel burn, distance, time)
Flows/ Holding P-Stats
5. Estimate Benefits of PBN approach
Annual FB benefits
6. Estimate Airline ROI
Cost/ Benefit
Holding Info
2. Analyze Holding Patterns
AC type, Config
Surface Area, Drag & Fuel Burn Coeff
BADA
A
B
4. Estimate Benefits of Metroplex de-confliction
Annual De-confliction benefits
BTS
Cancellations

Figure 7: Overview of the methodology

This chapter is organized as follows: section 2.1 contains a summary of the data sources used; section 3.2 describes a methodology for TRACON arrival flow analysis using track data; section 3.3 describes a methodology for holding pattern analysis; section 3.4 describes a methodology for computing track distance, time and fuel burn using track data; section 3.5 describes a methodology for metroplex de-confliction benefits analysis; section 3.6 describes a methodology for benefits of future PBN approach procedure; section 3.7 describes a methodology for estimating airline's return on investment for the new PBN approach capability; and section 3.8 contains a summary of the method of implementation.

3.1 Data Sources

This section provides a description of the data used in the model.

3.1.1 ASPM Data

The Aviation System Performance Metrics (ASPM) online access system provides detailed data on flights to and from the ASPM airports (currently 77) and all flights by the ASPM carriers (currently 22), including flights by those carriers to international and domestic non-ASPM airports. ASPM also includes airport weather, runway configuration, and arrival and departure rates. (ASPM System Overview, 2012).

In the methodology, the ASPM data is used to identify days with potential metroplex flow conflict, get airport runway configuration, estimate the count of arrivals and departures get airport weather and wind conditions.

3.1.2 NOP Data

The National Offload Program (NOP) service is operated by the FAA (FAA, 2012c). It collects NAS operational data daily. One of the data items collected is flight tracks for Terminal Radar Approach Control Facilities (TRACONs). Flight tracks contain identifying flight number and flight status (arrival, departure, or overflight), as well as position reports including (latitude, longitude, altitude, and time-of-report) (DeArmon et al. 2011).

In the methodology, the NOP track data is used to estimate the performance of terminal arrival flows at an airport, estimate the cost of holding, and visualize and identify metroplex flow conflicts.

3.1.3 BADA Data

Base of Aircraft Data (BADA) is an Aircraft Performance Model (APM) developed and maintained by EUROCONTROL through active cooperation with aircraft manufacturers and operating airlines. For more information about BADA refer (Euro Control, 2012b).

In the methodology, the BADA is used to determine the aircraft performance related information i.e., equations and coefficients for aircraft drag, lift, thrust specific fuel consumption and stall speeds required for fuel burn computation.

3.1.4 BTS Airline On-Time Data

The Bureau of Transportation Statistics (BTS) was established as a statistical agency in 1992 (BTS, 2012). The BTS mission is to create, manage, and share transportation statistical knowledge with public and private transportation communities and the Nation (BTS, 2012).

In the methodology, BTS airline On-time data is used to compute the number of excess cancelled flights as a result of metroplex flow conflict.

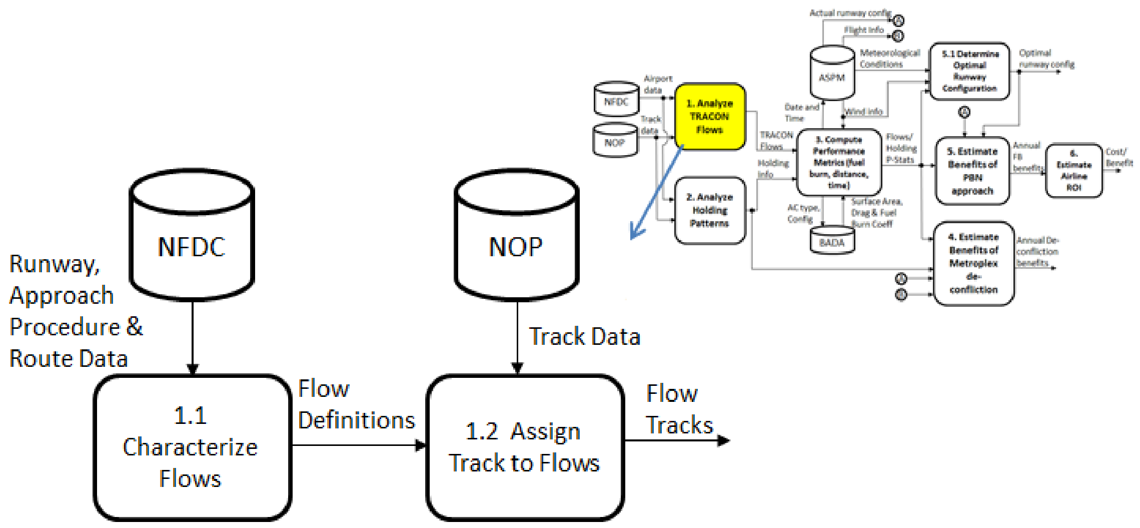
3.1.5 NFDC Data

The National Flight Data Center (NFDC), within the Aeronautical Information Management (AIM) directorate of Mission Support Services, is the central authority and official repository within the FAA responsible for the collection, validation and quality control of aeronautical information disseminated to support National Airspace System (NAS) operations. It contains details of the physical description, geographical position, and operational characteristics and status of all components of the NAS(FAA, 2012d).

In the methodology the NFDC data is used to determine coordinates of airports, runways, fixes and waypoints for procedures in the metroplex airspace.

3.2 Methodology for TRACON flow analysis

The methodology for airport arrival flow analysis has two functions: (1) Characterize TRACON flows and (2) Assign flight tracks to flows (see Figure 8). The methodology is described in detail in the following subsections.



Actual runway config
Flight info
ASPM
Meteorological Conditions
5.1 Determine Optimal Runway Configuration
Optimal runway config
Airport data
NFDC
1. Analyze TRACON Flows
Track data
NOP
Date and Time
Wind info
TRACON Flows
Holding Info
3. Compute Performance Metrics (fuel burn, distance, time)
Flows/ Holding P-Stats
5. Estimate Benefits of PBN approach
Annual FB benefits
6. Estimate Airline ROI
Cost/ Benefit
2. Analyze Holding Patterns
AC type, Config
Surface Area, Drag & Fuel Burn Coeff
BADA
4. Estimate Benefits of Metroplex de-confliction
Annual De-confliction benefits
Runway, Approach Procedure & Route Data
NFDC
NOP
Track Data
1.1 Characterize Flows
Flow Definitions
1.2 Assign Track to Flows
Flow Tracks

Figure 8: Methodology for Airport Arrival Flow Analysis

3.2.1 Flow Characterization

The TRACON arrival flows are defined as the flow of aircraft from the final waypoint on the Standard Terminal Arrival Route (STAR) to the runway threshold via an approach type. The flows are characterized as a combination of direction, runway and approach type. The characterization process is done using the following steps:

Step1: Identify the direction of TRACON flow

The location of the final waypoint on the STAR with respect to the runways determines the direction of the traffic flow. Figure 9 shows a sample STAR with its final waypoint (WP2) located south east of the airport. Therefore the cardinal direction associated with aircraft passing via WP2 is South East (SE). Similarly flows using WP1 to approach the runways would be characterized as North East (NE) flows and so on.

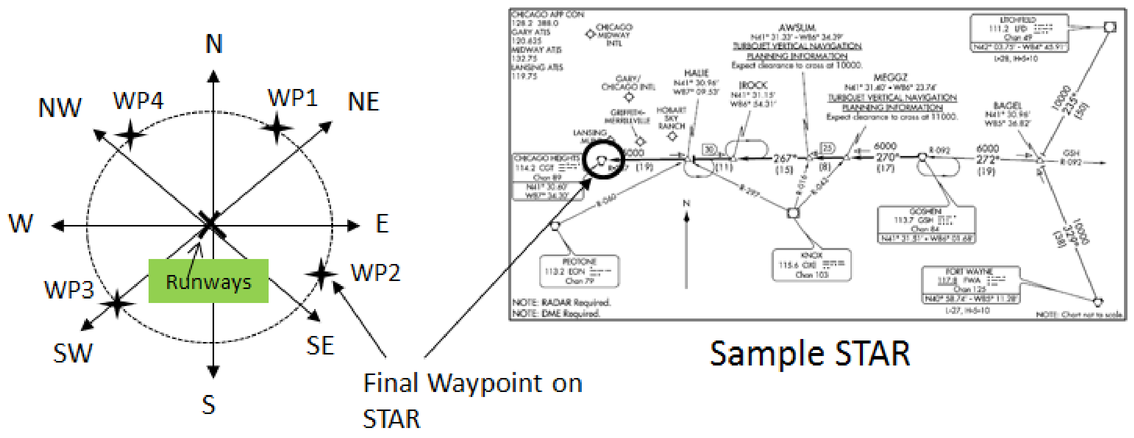


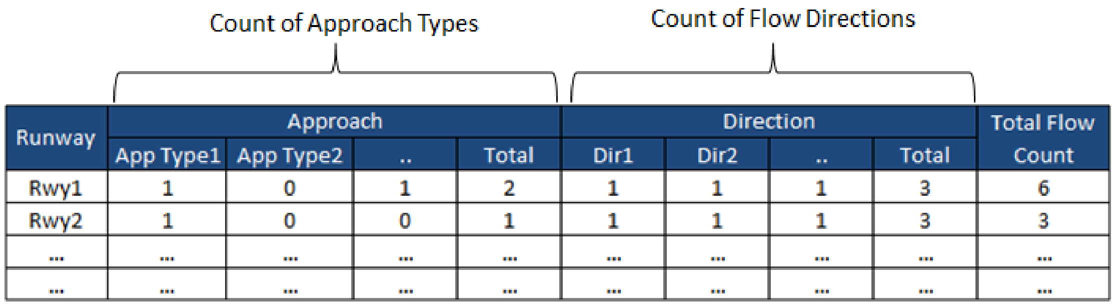
Figure 9: Depiction of how flows direction is determined from final waypoint on STAR

Step2: Identify the Approach procedure for each runway

The next step in the characterization process is to list the approach procedures published for each runway. Based on meteorological conditions (ceiling and visibility) at the airport, aircraft use either a visual approach or an instrument approach. The instrument approaches vary based on the level of lateral and vertical guidance required. The conventional instrument approach procedure is the Instrument Landing System (ILS) approach. The more recent Performance Based Navigation (PBN) approaches are Area

Navigation (RNAV) GPS approach and RNAV Required Navigation Performance (RNP) approach.

The output of the characterization process is a table that lists all possible flows for runways at an airport. A sample of this is shown in Figure 10, The table is populated with Boolean values, 1 indicating “applicable” and 0 indicating “not applicable”. The total number of flows for each runway is determined by multiplying the total number of approaches available for the runway with the total number of directions from where the flows can originate.


Count of Approach Types
Count of Flow Directions

Runway	Approach				Direction				Total Flow Count
	App Type1	App Type2	..	Total	Dir1	Dir2	..	Total	
Rwy1	1	0	1	2	1	1	1	3	6
Rwy2	1	0	0	1	1	1	1	3	3
...	...	...	...	...	...	...	...	...	...
...	...	...	...	...	...	...	...	...	...

Figure 10: Output of the Flow Characterization Process

3.2.2 Flow Assignment

The flow assignment process assigns each flight track to a flow. The flows are defined in terms of direction, runway and approach type. The arrival flight tracks to an airport (filtered from the NOP data) are high fidelity surveillance track data that originate 30-90 NM miles from the airport and terminate at the runway threshold.

The tracks are first assigned to a runway based on the final two track hits (each point of the 4D trajectory is referred to as a radar track hit). The tracks are then assigned a cardinal direction based on direction of approach. Finally, each track corresponding to a runway and an approach direction is assigned an approach type based on its proximity to published approach procedures for that runway. The detail of the algorithm for flow assignment is as follows:

Step1: Filter out arrival tracks from rest of the data.

The NOP data has track information for the whole TRACON. Before tracks are assigned to flows, the arrivals tracks are filtered out. This is done using the following logic:

a. Sort each track by time (descending order).
b. Get the first and last hit for each track.
c. Determine the top left and bottom right corner of the airport, by adding and subtracting 0.05 degrees from the coordinates of the airport. This will be the boundary of the airport.
d. If the first hit is within the airport boundary then assign the flight track as a departure, else assign flight track as an arrival

Step2: Assign runway to each arrival track.

The algorithm for assigning a track to a runway is as follows,

a. Sort the arrival track by time (descending order).
b. Get the last two hits of the arrival track, call it the sub-track.
c. Calculate the distance from each runway’s centerline for each sub-track

d. Calculate the heading, and compare it to each runway's alignment for sub-track.

e. Assign the sub-track to the runway with the minimum distance and difference in heading.

Step3: Assign Direction

Flight tracks are assigned direction based on the STAR they use to approach the runway. The algorithm for assigning track to a direction is as follows:

a. Assign each STAR a cardinal direction based on its location with respect to the airport

b. Define suitable rectangle boundaries for each STAR to capture all aircraft tracks using the STAR.

c. Determine if the track passes through the rectangle defined for the STAR.

d. Assign each track the cardinal direction associated with the STAR whose rectangle boundary it passes through.

Step4: Assign Approach procedure type

The tracks are assigned to an approach type based on its lateral and vertical proximity to the fixes/waypoints along the approach. The algorithm for assigning track to an approach type is as follows

a. Get all the arrival tracks for a given runway

b. Get the coordinates and the minimum altitude threshold for the waypoints of the approach procedure for the runway

c. Check the lateral and vertical proximity of the track to the approach procedure at the waypoints along the approach procedure.

d. Assign the track the approach procedure if it is within the vertical and lateral proximity thresholds.

3.3 Holding Pattern Analysis

Holding patterns are racetrack patterns based on a holding fix along the STAR (see Figure 11). They are used for delaying arriving aircraft that cannot land due to poor weather, runway unavailability or due to metroplex flow conflicts.

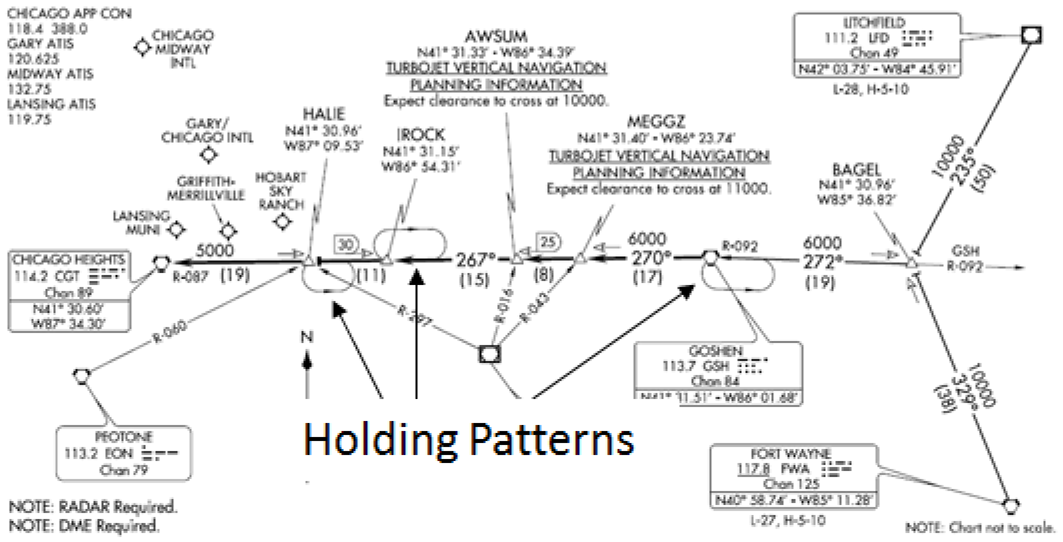
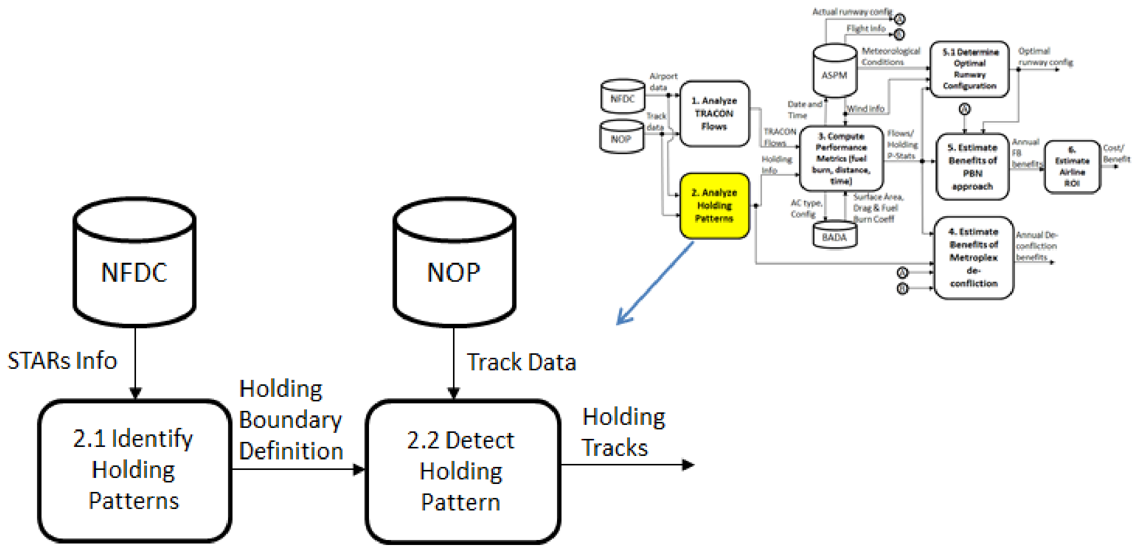

CHICAGO APP CON 118.4 388.0
GARY ATIS 120.625
MIDWAY ATIS 132.75
LANSING ATIS 119.75
CHICAGO MIDWAY INTL
GARY/ CHICAGO INTL
GRIFFITH-MERRILLVILLE
HOBART SKY RANCH
LANSING MUNI
HALIE N41° 30.96' W87° 09.53'
IROCK N41° 31.15' W86° 54.31'
AWSUM N41° 31.33' - W86° 34.39'
TURBOJET VERTICAL NAVIGATION PLANNING INFORMATION
Expect clearance to cross at 10000.
MEGGZ N41° 31.40' - W86° 23.74'
TURBOJET VERTICAL NAVIGATION PLANNING INFORMATION
Expect clearance to cross at 11000.
BAGEL N41° 30.96' W85° 36.82'
LITCHFIELD 111.2 LFD Chan 49 N42° 03.75' - W84° 45.91'
L-28, H-5-10
CHICAGO HEIGHTS 114.2 CGT Chan 89 N41° 30.60' W87° 34.30'
5000 R-087 (19)
267° (15) (11) (8)
6000 270° (17)
6000 272° (19)
R-092
GSH R-092
10000 235° (50)
10000 329° (38)
R-060
R-277
R-016
R-043
GOSHEN 113.7 GSH Chan 84 N41° 11.51' - W86° 01.68'
PEOTONE 113.2 EON Chan 79
FORT WAYNE 117.8 FWA Chan 125 N40° 58.74' - W85° 11.28'
L-27, H-5-10
N
Holding Patterns
NOTE: RADAR Required.
NOTE: DME Required.
NOTE: Chart not to scale.

Figure 11: Holding Patterns on STAR

A key difference in the holding pattern analysis and the arrival flow analysis is the scope of the analysis. Instead of performing the track data analysis in the TRACON, the track data analysis is done for each holding boundary i.e. the rectangular boundary defined around the published holding procedure to capture tracks in holding pattern. The methodology for holding pattern analysis is shown in Figure 12



Actual runway config
Flight info
ASPM
Meteorological Conditions
5.1 Determine Optimal Runway Configuration
Optimal runway config
NFDC
Airport data
Track data
NOP
1. Analyze TRACON Flows
Date and Time
Wind info
TRACON Flows
Holding Info
3. Compute Performance Metrics (fuel burn, distance, time)
Flows/ Holding P-Stats
5. Estimate Benefits of PBN approach
Annual FB benefits
6. Estimate Airline ROI
Cost/ Benefit
2. Analyze Holding Patterns
AC type, Config
Surface Area, Drag & Fuel Burn Coeff
BADA
4. Estimate Benefits of Metroplex de-confliction
Annual De-confliction benefits
A
B
NFDC
NOP
STARs Info
Track Data
2.1 Identify Holding Patterns
Holding Boundary Definition
2.2 Detect Holding Pattern
Holding Tracks

Figure 12: Holding Pattern Analysis

The algorithm for the holding pattern analysis is as follows:

a. Analyze each STAR in the terminal area and define suitable rectangular boundaries around the published holding patterns

b. Filter out tracks that fall within the rectangular boundaries.

c. Compute the cumulative turn angle for each filtered sub-track in the rectangular boundary.

d. Determine the holding tracks by filtering out sub-tracks with cumulative turn angle greater than 360 degrees..

e. Compute the time in holding and fuel burn for each holding track as described in section 3.4.1.

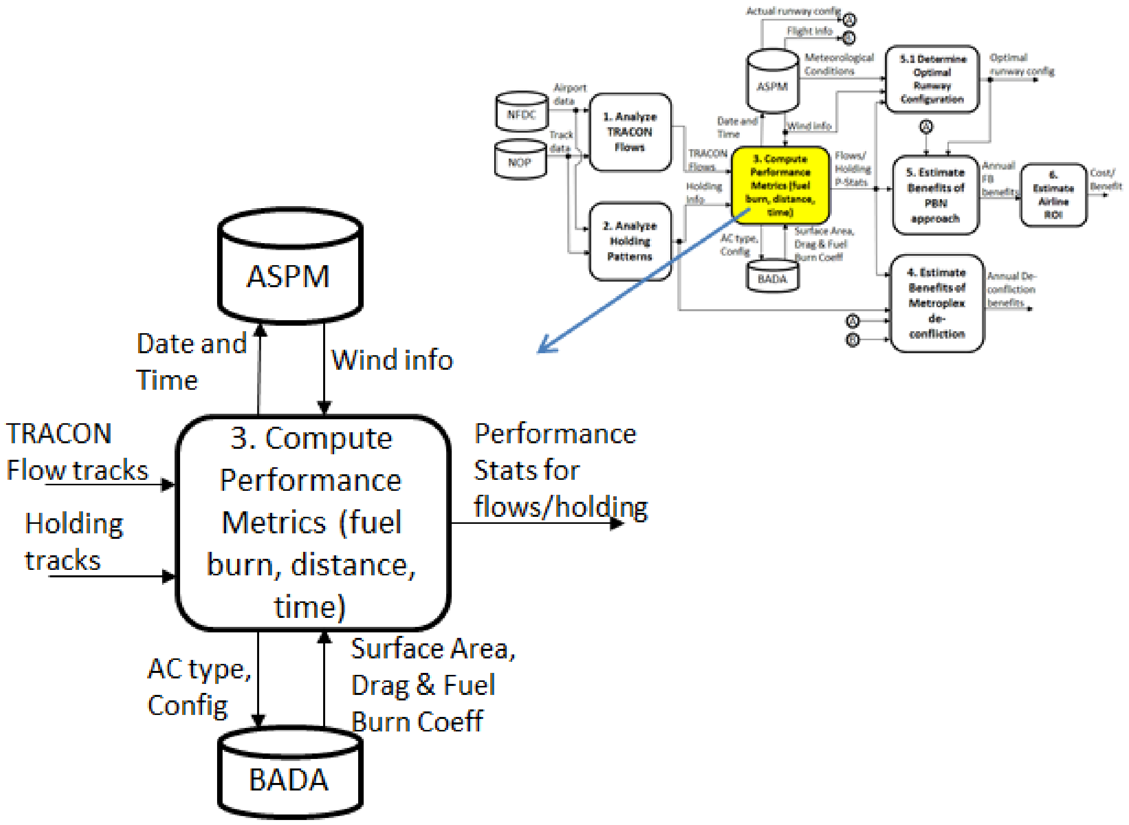
3.4 Performance Metrics

The performance metrics that can be derived from the flight track data are track distance (nautical mile), track time (minute) and track fuel burn (kilogram).

For TRACON flows the metrics are measured from the final STAR waypoint to the runway threshold. The metrics are computed for each flight track in a given flow. The performances of the flows are reported in terms of mean and standard deviation of the performance metric

For the holding patterns the performances metrics are reported in terms of frequency of holding pattern occurrence (number of holding patterns per 100 arrivals) and the mean and standard deviation of the holding pattern performance metric.

The overview of the methodology for computing performance metrics for TRACON flows and holding patterns is shown in Figure 13. The details of computing track distance (NM) and track time (minute) are discussed in section 3.4.1 and the details for computing the fuel burn is discussed in section 3.4.2



ASPM
Date and Time
Wind info
TRACON Flow tracks
Holding tracks
3. Compute Performance Metrics (fuel burn, distance, time)
Performance Stats for flows/holding
AC type, Config
Surface Area, Drag & Fuel Burn Coeff
BADA

Figure 13: Performance Analysis of flows and holding pattern

3.4.1 Track distance and time

The algorithm for estimating the track distance and track time for TRACON arrival flows is as follows:

a. Sort arrival track for each flow by time (descending order).

b. Designate a *Start Point* by defining abeam (line perpendicular to flow direction of each flow) across each of the final STAR waypoint to mark the start of the TRACON arrival flow (see Figure 27 and Figure 28).

c. The track time is the difference in the 4D track's time at the *Start point* and the time at the runway threshold

d. Compute the track distance at each time step from the *Start Point* to the runway threshold. (NOP arrival track terminate at the runway threshold).

e. Track distance is the cumulative distance from the Start point to the point at the runway threshold.

In case of holding patterns the track distance and time are the total distance/time in holding and are computed using the first and last 4D point of the holding track.

3.4.2 Track Fuel Burn

Aircraft fuel burn rate is higher in the terminal area due to level offs and change of aircraft configuration from clean to dirty. This fuel burn model captures these two aspects of terminal arrival flows by taking into consideration the energy state (kinetic – true airspeed and potential – altitude) of the aircraft at each position report of the flight trajectory. The inputs required to compute fuel burn are, the 4D trajectory, the wind magnitude and direction, and the aircraft mass estimate, thrust and drag coefficient. The details of computing the fuel burn, true airspeed, thrust and drag are described in the following sub sections.

3.4.2.1 Thrust Specific Fuel Burn

The model computes total fuel burn for a 4D trajectory by summing up the fuel burn at each time step i (see Equation 1). The fuel burn at each time step is computed as a product of the thrust and thrust specific fuel consumption (see Equation 2). The expression for thrust specific fuel consumption for jets and turboprops is shown in Equation 3 and Equation 4 respectively (Euro Control, 2012b). The thrust is estimated using the Total-Energy model, which equates the rate of work done by forces acting on the aircraft, to the rate of change of potential and kinetic energy (see Equation 5) (Euro Control, 2012b). By rearranging the total energy model, the equation of thrust is obtained (see Equation 6).

Equation 1: Total fuel burn for a flight track

$$F = \sum_i f_i \times (t_i - t_{i-1})$$

Equation 2: Fuel burn rate at each time step

$$f_i = \eta_{i,engine} \times T_i$$

Equation 3: Thrust specific fuel consumption for jets

$$\eta_{i,jet} = C_{f1} \times \left(1 + \frac{TAS_i}{C_{f2}}\right)$$

Equation 4: Thrust specific fuel consumption for turboprops

$$\eta_{i,turboprop} = C_{f1} \times (1 - \frac{TAS_i}{C_{f2}}) \times \left(\frac{TAS_i \times 1.94}{1000}\right)$$

Equation 5: Total-energy model

$$(T_i - D_i) \times TAS_i = m_i \times g \times \frac{dh_i}{dt} + m_i \times TAS_i \times \frac{dTAS_i}{dt}$$

Equation 6: Expression for thrusts

$$T_i = D_i + \frac{m_i \times g \times \frac{dh_i}{dt}}{TAS_i} + m_i \times \frac{dTAS_i}{dt}$$

Where,

F is the total fuel burn for a 4D trajectory in kg.

f_i is the fuel burn rate in kg/min.

t_i, t_{i-1} are the timestamps for positions $i \ and \ i-1$.

$\eta_{i,engine}$ is the thrust specific fuel consumption in kg/(min*kN).

T_i is the thrust in kN.

D_i is the drag in kN.

$C_{f1}$is the first thrust specific fuel consumption coefficients for an aircraft type in kg/(min*kN) for jets and kg/(min*knots*kN) for turbo jets

C_{f2} is the second thrust specific fuel consumption coefficient for an aircraft type in m/s.

TAS_i is the true airspeed of the aircraft in m/s.

dh_i is the change in altitude = $h_i - h_{i-1}$ in m

$dTAS_i$ is the change in true airspeed = $TAS_i - TAS_{i-1}$ in m/s

dt is the time step increment = $t_i - t_{i-1}$ in s

m_i is the mass of the aircraft in kg.

g is acceleration due to gravity = 9.81 m/s^2.

3.4.2.2 Initial Mass of the Aircraft and the Total Fuel Burn Sensitivity

The terminal area fuel burn is estimated for each aircraft track from the final waypoint on STAR to the runway threshold. The mass of the aircraft at start of the flight track (the final waypoint on STAR) is estimated using the following equation:

Equation 7: Initial mass of the aircraft

$$m_{initial} = ((OEW + MPW) + MLW)/2$$

Where,

OEW is the operating empty weight for a given aircraft type in kg

MPW is the maximum payload weight for a given aircraft type in kg

MLW is the maximum landing weight for a given aircraft type.

The mass of the aircraft at the time of landing can range from the operating empty weight (OEW) and the maximum landing weight (MLW). In addition the weight of the cargo and passengers has to be taken into account as it is not included in the OEW as specified by the manufacturer. The sum of the operating empty weight and the maximum payload weight is the lower bound for the mass of the aircraft in the terminal airspace. The maximum landing weight is the upper bound estimate for the mass of the aircraft in the terminal airspace. The sensitivity of the total fuel burn in the terminal airspace to the initial mass of the aircraft is analyzed by estimating the fuel burn for the lower and upper bounds.

In this analysis the initial mass (the time at which the aircraft track is at the final waypoint on STAR) of the aircraft is estimated as the mean of the lower and the upper bound (see Equation 7) for each aircraft type. For subsequent time steps the mass of aircraft is reduced by the amount of fuel burnt in the previous time step as shown Equation 8.

Equation 8: Mass of the aircraft varies at each time step along the flight track as function of the initial mass the fuel burn at each time step.

$$m_i = m_{i-1} - f_{i-1}$$

3.4.2.3 True Airspeed

The time step between position reports in the NOP data for the most part vary from four seconds to a minute. In this analysis the 4D trajectories are consolidated such that the time step is at least thirty seconds. This is done to reduce noise in the velocity profile. In addition a differential equation forward filter is used to further smoothen the velocity profile(Jamet, 2011). The true airspeed is computed at each time step based on the ground speed and the wind speed (Oaks & Ryan, 2010). The true airspeed is given by,

Equation 9: True Airspeed

$$TAS_i = \left| \begin{pmatrix} GS_i \times \sin\theta_i \\ GS_i \times \cos\theta_i \end{pmatrix} - \begin{pmatrix} WS_i \times \sin\phi_i \\ WS_i \times \cos\phi_i \end{pmatrix} \right|$$

Where,

GS_i is the ground speed in m/s.

θ_i is the aircraft bearing with respect to the north in radian.

WS_i is the wind speed in m/s.

ϕ_i is the wind bearing with respect to the north in radian.

The ground speed and the aircraft bearing are computed using the following equations:

Equation 10: Ground Speed

$$GS_i = \frac{R \times c_i}{t_i - t_{i-1}}$$

Equation 11: Central Angle between two coordinates

$$c_i = 2 \times atan2(\sqrt{b}, \sqrt{(1-b)})$$

Equation 12: Coordinates intercept

$$b = sin^2\left(\frac{\varphi_i - \varphi_{i-1}}{2}\right) + cos(\varphi_{i-1})cos(\varphi_i)sin^2\left(\frac{\lambda_i - \lambda_{i-1}}{2}\right)$$

Equation 13: Aircraft bearing with respect to the North

$$\theta_i = atan2\big(sin(\lambda_i - \lambda_{i-1})cos(\varphi_i), cos(\varphi_{i-1})sin(\varphi_i) - sin(\varphi_{i-1})cos(\varphi_i)cos(\lambda_i - \lambda_{i-1})\big)$$

Where,

R is the radius of the earth = 6378100m.

c_i is the haversine central angle between two coordinates $(\varphi_{i-1}, \lambda_{i-1})$ and (φ_i, λ_i).

φ_i, φ_{i-1} are the latitude for positions $i\ and\ i-1$.

λ_i, λ_{i-1} are the longitude for positions $i\ and\ i-1$.

t, t_{i-1} are the timestamps for position $i\ and\ i-1$s.

The wind speed and wind bearing reported in the ASPM data are measured at ten meters from the surface. The wind speeds increase with altitude and are estimated using the power law wind profile equation(Panofsky & Dutton, 1984).

Equation 14: Power law wind profile

$$WS_i = z_g \left(\frac{h_i}{h_g}\right)^{\alpha}$$

Where,

z_g is the velocity of the wind at height h_g=10m, in m/s.

h_i is the height of the aircraft above ground in m.

α is the Hellman exponent = 0.3 for human inhabited areas.

3.4.2.4 Drag Computation

The drag force on the aircraft is computed using the following equations (Euro Control, 2012b):

Equation 15: Drag

$$D_i = \frac{C_D \times \rho_i \times TAS_i^2 \times S}{2}$$

Equation 16: Density of air

$$\rho_i = \frac{p_i \times M}{R \times T}$$

Equation 17: Pressure as a function to altitude

$$p_i = p_0 \left(1 - \frac{L \times h_i'}{T_0}\right)$$

Equation 18: Temperature as a function of altitude

$$T = T_0 - L \times h_i'$$

Where,

D_i is drag in kN.

C_D is the coefficient of drag.

ρ_i is the density of the aircraft in kg/m^3.

h_i' is the altitude of the aircraft above sea level in m.

S is the wing reference area in m^2.

p_0 is the sea level atmospheric pressure = 101.325kPa.

p_i is the pressure at altitude h_i' in kPa.

L is the temperature lapse rate = 0.0065K/m.

T_0 is the sea level standard temperature = 288.15K.

M is the molar mass of dry air = 0.0289644 kg/mol.

R is the universal gas constant = 8.31477 J/(mol.K).

T is the absolute temperate in K.

The coefficient of drag in Equation 15 is a function of the coefficient of lift and the configuration of the aircraft. For terminal arrival flows, an aircraft is assumed to be in clean, approach or landing configuration at each time step based on the true airspeed and the altitude of the aircraft. The criteria assumed for selecting the configuration of the aircraft are shown in Table 8. The flight trajectories for arrival flows in the TRACON are all below 8000 feet. The minimum velocity (Vmin) for each configuration is 1.3 times the stall speed provided in the BADA's Operations Performance File (OPF) for each aircraft type.

Table 8: Criteria for Aircraft Configuration

Altitude(feet above ground level)	Velocity Threshold	Configuration
>1000ft & <=8000	TAS>=VminCruise	Clean
>1000ft & <=8000	VminApproach<=TAS<VminCruise	Approach
<=1000ft	None	Landing

The coefficient of drag for clean, approach and landing configuration is given by Equation 19, Equation 20 and Equation 21 (Euro Control, 2012b). The coefficient of lift is given by Equation 22 (Euro Control, 2012b).

Equation 19: Coefficient of drag in cruise configuration

$$C_D = C_{D0,CR} + C_{D2,CR} \times C_L^2$$

Equation 20: Coefficient of drag in approach configuration

$$C_D = C_{D0,AP} + C_{D2,AP} \times C_L^2$$

Equation 21: Coefficient of drag in landing configuration

$$C_D = C_{D0,LD} + C_{D0,LDG} + C_{D2,LD} \times C_L^2$$

Equation 22: Coefficient of lift

$$C_L = \frac{2 \times m \times g}{\rho_i \times TAS_i^2 \times S \times \cos\psi}$$

Where,

C_L is the coefficient of lift.

$C_{D0,CR}$ $C_{D0,AP}$ $C_{D0,LD}$ are the parasitic drag coefficient for cruise, approach and landing configuration.

$C_{D2,CR}$ $C_{D2,AP}$ $C_{D2,LD}$ are the induced drag coefficient for cruise, approach and landing configuration.

$C_{D0,LDG}$ is the parasitic drag coefficient of the landing gear.

ψ is bank angle, assumed to be zero in this analysis.

3.5 Methodology for benefits analysis of metroplex flow de-confliction

Metroplex flow conflicts occur when low visibility and certain wind (magnitude and direction) constrain metroplex airports runway configurations such that arrivals or departure flows of one airport interfere with arrival or departure flows of the neighboring airport. The metroplex flow conflicts can be identified by performing a metroplex capacity gap analysis (Donaldson & Hansman, 2011). The capacity gap analysis involves analyzing capacity for various operating configuration in the metroplex i.e. runway configuration at each airport in the metroplex. The configurations that have a dip in capacity are further analyzed to identify flows likely constraining the airspace capacity. An alternate approach to identify conflicting flows is to research the literature or to interview subject matter experts (tower manager and controllers).

Metroplex flow conflicts result in sharing of common airspace in the metroplex by flows from two adjacent airports. This is overcome by ground holding for departures and airborne holdings for arrivals at the airports in the metroplex. The ground holdings for departures result in departure delays and may result in cancelled flights. The airborne holding for arrivals results in en-route delays. The benefits of de-confliction are the reduction in: (1) additional departure delays (2) cancelled flights and (3) airborne holdings that occur due to flow conflict. The de-confliction using RNP approach has further benefits in fuel burn savings for arrivals from using RNP approaches that have a shorter and more efficient approach compared to conventional ILS approach. The fuel burn savings for arrival flows from the new PBN approach procedure (e.g. RNP approach) is described in section 3.6.

This section describes a methodology for estimating the cost of delays and cancelled flights for departures and airborne holdings for arrivals. The costs are computed in terms of the Airlines Direct Operating Costs (ADOC), which takes into account the cost of flight deck crew, fuel, maintenance, equipment charge and other miscellaneous charges. This dissertation focuses on airline's cost/potential savings rather than system-wide cost/potential savings, as airline make equipage decision based on their own individual savings.

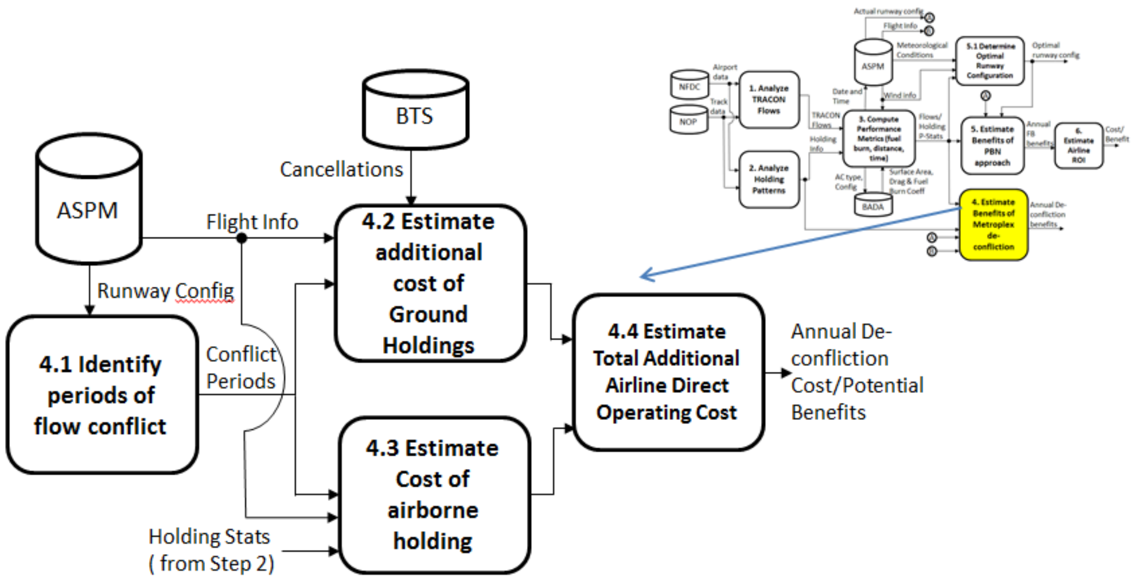

BTS
Cancellations
ASPM
Flight Info
Runway Config
4.1 Identify periods of flow conflict
Conflict Periods
4.2 Estimate additional cost of Ground Holdings
4.3 Estimate Cost of airborne holding
Holding Stats (from Step 2)
4.4 Estimate Total Additional Airline Direct Operating Cost
Annual De-confliction Cost/Potential Benefits

Figure 14: Methodology for estimating the benefits of Metroplex Flow De-confliction

3.5.1 Estimation of Cost of Ground Holding

Metroplex flow conflict results in ground holding for the departure flows. This results in additional departure delays and can result in additional cancelled flights. The additional departure delays and cancelled flights due to metroplex flow conflict are computed by comparing the delays and cancellation statistics using historic ASPM airport and flight data for flights during periods of flow conflict with a suitable baseline. The baseline period selected represents similar operational condition at the airport (ceiling and visibility). For instance, a suitable baseline for metroplex flow conflicts occurring during IMC can be other IMC periods with no flow conflicts.

The algorithm for estimating the cost of excess departure delays and cancelled flights due to metroplex conflict is as follows:

Step1: Identify the flow conflict and the no-flow conflict (baseline) periods

a. Identify set of time periods (i.e., start and end times from fields 2 to 5 in ASPM airport table) of <u>flow conflict</u> for a given year using ASPM airport data. This is done by analyzing the runway configuration (field 76 in ASPM airport table) at the airports and identifying time bins when the runways associated with the flow conflict are in use. These time bins are called the conflict periods.

Equation 23: Set of time periods of flow conflict for a given year

$$C_y : \{tc_1, tc_2, tc_3 \ldots .\}$$

Where,

C_y is the set of all conflict periods in a given year y

tc_1, tc_2, tc_3 .... are the flow conflict periods in a given year with each period defined by a start and end time.

b. Identify set of time periods (i.e., start and end times from fields 2 to 5 in ASPM airport table) of <u>*no* flow conflict</u> (baseline) for a given year using

ASPM airport data to. This is done by analyzing the runway configuration (field 76 in ASPM airport table) at the airports and identifying time bins which have similar meteorological conditions (field 70 in ASPM airport table) and use the runways *not* associated with the flow conflict. Call these the baseline periods

Equation 24: Set of time periods of no flow conflict (baseline)` for a given year

$$NC_y : \{tnc_1, tnc_2, tnc_3 \ldots .\}$$

Where,

NC_y is the set of all no-flow conflict periods in a given year $y = \{y1, y2..\}$

tnc_1, tnc_2, tnc_3 .... are the no-flow conflict periods in a given year with each period defined by a start and end time (fields 2 to 5 in ASPM airport table).

Step2: Compute excess delays (D_{dep}) and cancelled flights (ΔCancelled) due to flow conflict

a. Estimate the total schedule flights for the flow conflict (C_y) and no-flow conflict (NC_y) time periods, using the ASPM flight data and BTS Airline on-time performance data. ASPM does not have information on cancelled flights and BTS has cancellation information for only domestic flights. The total number of scheduled flights for a given year (for C_y and NC_y) is estimated by summing the operational flight count and cancellation count for each time period in the ASPM flight data table and the BTS on-time performance table respectively.

Equation 25: Total scheduled flights in a year for flow conflict or no-flow conflict time periods

$$F\_sch_T = (\#\ ASPM\ flights)_T + (\#\ Cancelled\ BTS)_T$$

Where,

F_sch_T is the total scheduled flights in a year for the flow conflict ($T = C_y$) or the no-flow conflict ($T = NC_y$) time periods

b. Compute the percentage of scheduled flights delayed, cancelled and the average delay per scheduled flight for the flow conflict (C_y) and no-flow conflict (NC_y) time periods.

Equation 26: Percentage of scheduled flights cancelled

$$P\_can_T = \frac{(\#\ Cancelled\ BTS)_T}{F\_sch_T}$$

Equation 27: Percentage of scheduled flights delayed

$$P\_del_T = \frac{(\#\ ASPM\ flight\ delayed > 15min)_T}{F\_sch_T}$$

Equation 28: Average delay per scheduled flight

$$Avg\_del_T = \frac{Total\_del_T}{Fsch_T}$$

$$Total\_del_T = \sum_T ASPM\ \ flight\ delayed > 15min$$

Where,

P_can_T is the percentage of scheduled flights cancelled for the flow conflict ($T = C_y$) or the no-flow conflict ($T = NC_y$) time periods.

P_del_T is the percentage of scheduled flights delayed for the flow conflict ($T = C_y$) or the no-flow conflict ($T = NC_y$) time periods.

Avg_del_T is the average delay per scheduled flight for the flow conflict ($T = C_y$) or the no-flow conflict ($T = NC_y$) time periods.

$Total\_del_T$ is the total delay for the flow conflict ($T = C_y$) or the no-flow conflict ($T = NC_y$) time periods computed using the ASPM flight data (field 54 in ASPM flight table).

c. If the average delay (Avg_del_T) and percentage cancelled flights (P_can_T) for flow conflict ($T = C_y$) time period is greater than no-flow conflict ($T = NC_y$) time periods, compute excess delays (Ddep) and cancelled flights (ΔCancelled) due to flow conflict using the following equations:

Equation 29: Excess delays (Ddep) due to flow conflict

$$D_{dep} = Total\_del_{T=C_y} - \left(F_{sch_{T=C_y}} * P_{del_{T=NC_y}}\right) * Avg\_del_{T=NC_y}$$

Equation 30: Excess cancelled flights (ΔCancelled) due to flow conflict

$$\Delta Cancelled = F_{sch_{T=C_y}} * (P_{can_{T=C_y}} - P_{can_{T=NC_y}})$$

Where,

$\left(F_{sch_{T=C_y}} * P_{del_{T=NC_y}}\right) * Avg\_del_{T=NC_y}$ is the total flight delay for flow conflict time period ($T = C_y$) if percentage flight delayed and the average delay per flight is same as no-flow conflict time periods (as a result of flow de-confliction).

$(P_{can_{T=C_y}} - P_{can_{T=NC_y}})$ is the difference in scheduled flights cancelled during the flow conflict ($T = C_y$) and the no-flow conflict ($T = NC_y$) time periods.

Step3: Compute total cost of excess departures delays (D_{dep}) and cancelled flights (ΔCancelled) due to flow conflict

The metroplex flow conflict between departure flows at one airport and arrival flows at the neighboring airport can prevent departures as aircraft taxi out to the runway threshold, resulting in departure queues at the runway threshold. The Airline's Direct

Operating Cost (ADOC) for additional departure delays due to metroplex flow conflict are accounted as taxi-out delay costs. They are computed using the following equations:

Equation 31: Total ADOC due to addition departure delays

$$T_{ADOC_{TO}} = ADOC_{TO} * D_{dep}.$$

Equation 32: ADOC for taxi out delay

$$ADOC_{TO} = WADOC * A_{TO}$$

Equation 33: ADOC weighted for fleet mix

$$WADOC = \sum AC_{type} * ADOC_{AC_{type}}$$

Where,

$ADOC_{AC_{type}}$ is the ADOC per block hour for a given class of aircraft in $/hour (FAA, 2005b; pg D-8).

AC_{type} is the fraction/percentage of aircraft of a given class in the fleet mix.

$WADOC$ is the weighted ADOC based on the fleet mix in $/hour.

A_{TO} is the adjustment factor for ADOC for the taxi out phase = 0.78 (FAA, 2005b; pg D-9).

$ADOC_{TO}$ is the ADOC per block hour for taxi out phase of the flight in $/hour.

$T_{ADOC_{TO}}$ is the Total ADOC due to taxi out delays in $.

D_{dep} is the Total additional departure delays due to metroplex flow conflict in hours (computed step2 of the algorithm).

The cost of a cancelled flight to an airline is estimated to be $4977 per cancellation (FAA, 2012g; pg 14). The total cost of cancellation due to metroplex flow conflict is:

Equation 34: Cost of Flight Cancellation

$$Cost_{Cancel} = \$4977 * \Delta Cancelled$$

3.5.2 Holding Patterns Delays due to Metroplex Flow Conflict

The metroplex flow conflict can prevents arrivals from approaching the airport for landing, resulting in airborne holding at pre-defined fixes along the STAR. The Airlines Direct Operating Cost (ADOC) for airborne holding due to metroplex flow conflict is accounted as en-route delay costs.

The total en route delay due to holding pattern as a result of the metroplex flow conflict can be estimated using NOP data, provided NOP track data is available for the all conflict periods. NOP track data is a larger data set and access to it is limited compared to ASPM data. In case NOP track data is not available for all the conflict periods, the sample NOP track data for the conflict days is used along with the ASPM flight data to estimate the total en-route delay due to holding pattern as a result of the metroplex flow conflict.

Not all holding patterns are due to the metroplex flow conflict. It is necessary to accurately account for holding patterns due to metroplex flow conflict. The algorithm for estimating cost of en route holding due to metroplex flow conflict is as follows:

Step1: Compute total delay due to airborne holding as a result of metroplex flow conflict ($D_{holding}$)

a. Collect NOP data to perform holding pattern analysis for the conflict periods identified in Step1 of section 3.5.1
b. Perform holding pattern analysis (see section 3.3) using NOP track data.
c. Identify holding patterns as a result of flow conflict with a neighboring arrival or departure flow.
    i. Lookup the start time of each holding pattern using the track data

ii. Lookup using the track data the start time of the other flow involved in the flow conflict. In case of departure flow get the start of departure time. In case of arrival flow get the start time track in the TRACON.

iii. Create time bins (15 minute) with start time of the conflicting flow events e.g. arrival and departure flows, causing the flow conflict.

iv. Identify time bins that have occurrence of both the conflicting flow events. Call these the conflict bins

d. If NOP data is available for all the conflict periods in a given year, compute total delay due to airborne holding as results of metroplex flow conflict by summing up the holding times for holding patterns in the conflict bins and GOTO step2. Else,

e. Compute the frequency of holding patterns using the number of aircraft in holding and the total number of arrivals in each conflict bin i.e. holding patterns per 100 arrivals.

f. Compute the total count of holding patterns due to the metroplex flow conflict as product of the frequency of the holding pattern and the total arrivals during the conflict period (computed from the ASPM flight data).

g. Compute the total delays due to holding patterns as product of the total count of holding patterns and the mean holding time.

Step2: Compute cost of holding patterns due to metroplex flow conflict ($T_{ADOC_{ER}}$)

The Airlines Direct Operating Cost (ADOC) for airborne holding due to metroplex flow conflict is accounted as en-route delay costs. They are computed using the following equations:

Equation 35: Total ADOC due to en route holding delays

$$T_{ADOC_{ER}} = ADOC_{ER} * D_{holding}.$$

Equation 36: ADOC for En route delay

$$ADOC_{ER} = WADOC * A_{ER}$$

Equation 37: ADOC weighted for fleet mix

$$WADOC = \sum AC_{type} * ADOC_{AC_{type}}$$

Where,

$ADOC_{AC_{type}}$ is the ADOC per block hour for a given class of aircraft in $/hour(FAA, 2005b; pg D-8).

AC_{type} is the fraction/percentage of aircraft of a given class in the fleet mix.

$WADOC$ is the weighted ADOC based on the fleet mix in $/hour.

A_{ER} is the adjustment factor for ADOC for the en route phase = 1.25 (FAA, 2005b; pg D-9).

$ADOC_{ER}$ is the ADOC per block hour in the en route phase of the flight in $/hour.

$T_{ADOC_{ER}}$ is the Total ADOC due to en route holding delays in $.

$D_{holding}$ is the Total delays due to holding patterns as a result of metroplex flow conflict in hours (computed in the step1 of the algorithm).

3.6 Methodology for Benefits Analysis of PBN approach procedures at an airport

The new PBN approach capability enable precise curved path approaches (e.g., RNP 0.3 w/ RF leg) in the terminal airspace. These new approach procedures are designed to eliminate trombone vectors in the base leg of the approach and make the approach from the final waypoint on STAR to runway threshold shorter on average compared to conventional approach procedures. This capability has two potential benefits at an airport in the metroplex:

a. The shorter approach results in fuel burn savings in the terminal airspace compared to the conventional approaches. Therefore, there is potential to design and enable new efficient procedures to all runways at the airport, not the just the ones required to de-conflict the metroplex airspace.
b. The more precise PBN approach path makes the operations at an airport in a metroplex independent of the neighboring airport. This allows the use of runway configurations that are more optimal in terms of performance of terminal area flows, compared to the historic configurations that are constrained on operations at the neighboring airport.

This section describes a methodology that captures the above two benefits to estimate the total potential benefits of the new PBN approaches to airlines at an airport. An overview of the methodology is shown Figure 15.

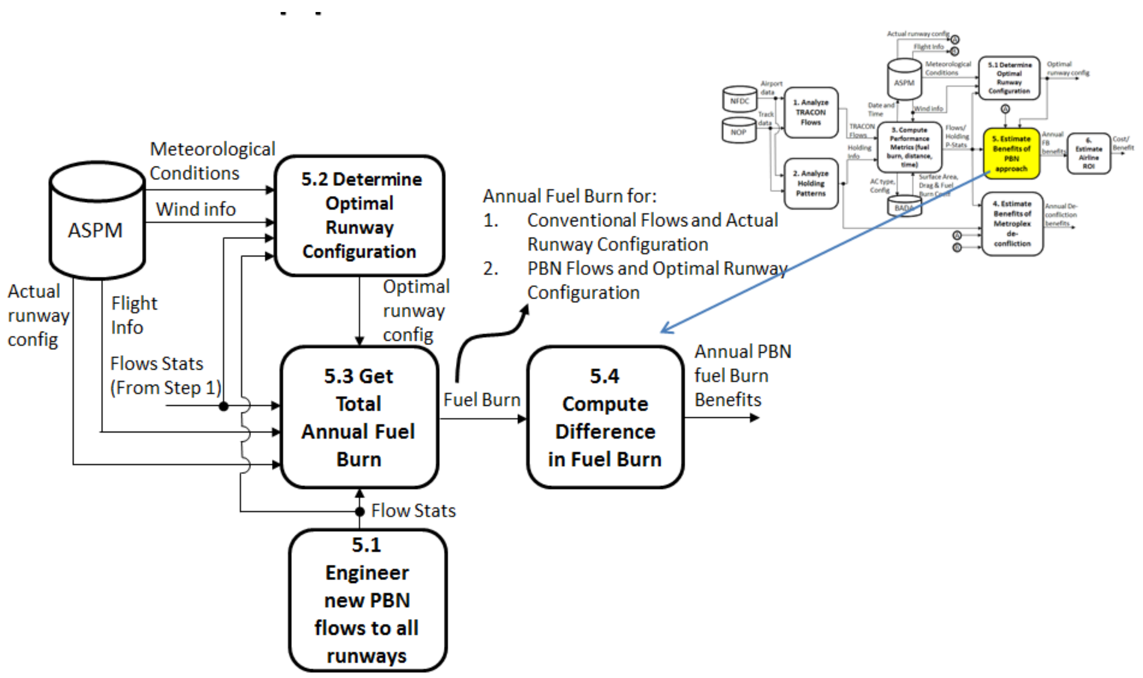

Meteorological Conditions
Wind info
ASPM
5.2 Determine Optimal Runway Configuration
Actual runway config
Flight Info
Flows Stats (From Step 1)
Optimal runway config
Annual Fuel Burn for:
1. Conventional Flows and Actual Runway Configuration
2. PBN Flows and Optimal Runway Configuration
5.3 Get Total Annual Fuel Burn
Fuel Burn
5.4 Compute Difference in Fuel Burn
Annual PBN fuel Burn Benefits
Flow Stats
5.1 Engineer new PBN flows to all runways

Figure 15: Methodology to estimate the total potential benefits of new PBN approach to airlines at an airport

3.6.1 Engineering new PBN flows

The approach described in this section to engineer new PBN flows (e.g. RNP approach flows) to runways at an airport relies on the availability of existing PBN flows. The flight tracks of existing PBN flows are reflected and rotated (see Figure 16) to get new PBN flows for other runways at the airport that do not have published PBN approach procedures, but could potentially benefit from them. The new flows are generated either by rotation or reflection and rotation, depending on the location of the arrival fix (final waypoint on STAR) with respect to the runway.

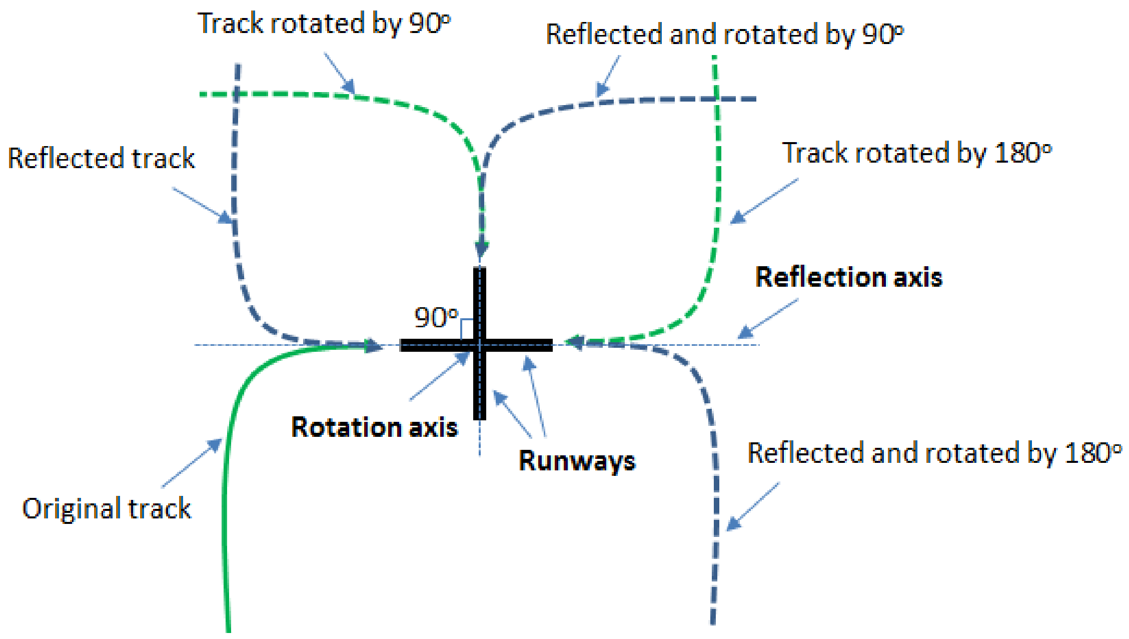


Figure 16: Rotation and Reflection of existing flight tracks to engineer new flows

The rotation and reflection of existing flight tracks is done using the following equations:

Equation 38: Formula for reflection of flight track

$$Ref_l(v) = 2 * \frac{v.l}{l.l} * l - v$$

Where,

$Ref_l(v)$ is the vector representing the reflected flight track coordinate

v is the flight track co-ordinate vector that needs to be reflected.

l is the vector representing the center line of the runway about which the flight track co-ordinate is reflected.

$v.l$ and $l.l$ are the dot product of the respective vectors

Equation 39: Formula for rotation of flight track

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} cos\theta & -sin\theta \\ sin\theta & cos\theta \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$

Where,

x, y are the flight track co-ordinate vectors with respect the point about which the vectors need to be rotated

x', y' are the rotated flight track co-ordinate vectors

θ is the angle of rotation (i.e., angular difference between the runways).

3.6.2 Optimal Runway Configuration

In current practice the runway configuration is determined based only on the wind direction and other tower considerations. An alternate approach is to select the most optimal runway configuration from a set of feasible configurations, by taking into account the direction of traffic intensity and the fuel burn performance of individual flows in the terminal. The algorithm for determining the optimal runway configuration is as follows:

a. For a given wind magnitude and direction compute the cross wind and head wind component for each runway using the following equation:

Equation 40: Formula for crosswind

$$CW = \sin(A) * WS$$

Equation 41: Formula for headwind

$$HW = \cos(A) . WS$$

Where,

WS is the magnitude of the wind in knots

CW is the magnitude of the crosswind in knots

HW is the magnitude of the headwind in knots

A is the difference in wind and runway bearing in radian.

b. Select a set of feasible runways or runway configurations based on the cross wind and headwind thresholds and the meteorological conditions. For a runway configuration to be feasible the cross wind component should not be greater than 20 knots and the headwind component should not be less than zero. Negative headwind component means the runway has a tailwind. Also, for IMC only runway configurations that support precision or precision-like approaches are selected.

c. Compute the weighted average fuel burn per flight for a given runway configuration and approach type while taking into account traffic intensity from each direction.

Equation 42: Average fuel burn per flight for a given runway and approach type

$$f_{y',t} = \sum_{t=1}^{M} W_{y,t} * f_y$$

Where,

$f_{y',t}$ is the average fuel burn per flight for a given runway and approach type.

$W_{y,t}$ is the weight of the arrival flows (percentage of aircrafts arriving in each flow) y at time period t and $\sum_{y,t=n} W_{y,t} = 1$ (i.e., the weight of arrival flows for a given time period t sum up to 1). These weights are used to allocate the total arrival at the airport to individual flows. These are estimated by performing TRACON flow analysis using NOP track data (see section 3.2) for all possible runway configurations and approach types which are a function of the meteorological conditions (visibility, wind magnitude and direction) at the airport.

f_y is the average fuel per aircraft for TRACON flow y in kg/min. The fuel burn statistics for TRACON flows are computed using the methodology in section 3.4.2

d. Rank the runway configurations based on the fuel burn performance and select the runway configuration with lowest weighted average fuel burn as the optimal runway configuration.

e. Repeat the process for each time bin in the ASPM airport table.

3.6.3 Total Annual Terminal Flow Fuel burn

The model presented in this section estimates the annual fuel-burn in the terminal airspace for arrivals at an airport using historic ASPM runway and flight data, and the flow fuel-burn statistics computed using NOP track data. The total annual fuel burn for aircraft in the terminal airspace is calculated using the following equations:

Equation 43: Total annual fuel burn in the terminal airspace for arrivals at an airport

$$F = \sum_{d=1}^{N} \sum_{t=1}^{M} A_{t,d} * W_{y,t} * f_y$$

Where,

F is the total annual fuel burn for aircraft in the terminal airspace in kg

d is the number of days considered in the year

t is the number of time periods/bins in a day (i.e., 15 min time bins to 1 hour time bins).

$A_{t,d}$ is the total number of arrivals for given day d and time period/bin t. This is computed from the ASPM flight table by counting the number of arrivals in each time bin for a given day. The time bins can be hourly or 15 minutes long.

3.6.3.1 The total potential benefits of the new PBN approach to airlines at an airport

The algorithm for estimating the total potential benefits of the new PBN approach to airlines at an airport is as follows:

a. Engineer new RNP flows for runway that currently do not have the new PBN procedures published by using existing RNP flows to (see section 3.6.1)
b. Compute fuel burn performance metrics for the new flows and existing flows (see sections 3.2 and 3.4)
c. Estimate the optimal runway configuration for existing flows and potential future flows for a given year for each 15 minute bin in the ASPM airport table (see section 3.6.2).
d. Estimate the total annual fuel burn for actual historic runway configuration and existing flows (see section 3.6.3) and call it the baseline scenario.
e. Repeat the process for alternate scenarios. The alternate scenarios are various combinations of runway configurations (actual and optimal) and flows (existing and future).
f. Compute the total potential benefits of the new PBN approach to airlines at an airport as the difference in the total annual fuel burn F for the baseline scenario and various alternatives.

3.7 Methodology for Estimating Airline Return on Investment

The airline ROI on equipage is estimated using the Net Present Value method shown below:

Equation 44: Net Present Value

$$NPV(i, N) = \sum_{t=0}^{N} \frac{R_t}{(1+i)^t}$$

Where,

i is the discount rate.

N is the total number of periods in years.

R_t is the net cash flow (i.e., cash inflow-cash outflow) at time t

The cash inflow per year is the savings from new PBN approach estimated using the methodology described in sections 3.5.2 and 3.6. The cash outflow is the initial airline investment in equipage which includes cost of equipment, training and certification. The assumptions made by the model are:

a. Benefits are accrued only after 100% of fleet is equipped
b. Cash outflow is the Initial Cost of Equipage when t=0
c. Cash outflow is zero for t>0
d. Cash Inflow is the benefits from new equipage per year
e. N equals 20 years.

The NPV model is also used to estimate the time to a positive ROI or Break Even Time (i.e. the smallest value of t for which the NPV is greater than zero).

3.8 Summary of the method of implementation

This methodology is demonstrated in a case-study of the benefits of the introduction of a Required Navigation Performance (RNP) approach procedure for air traffic arrival flows in the Chicago Terminal Radar Approach Control (TRACON). The methodology is implemented using GAWK (Daniel, 2010) which is an interpreted programming language designed for text processing and typically used as a data extraction and reporting tool. It is a standard feature of most Unix-like operating systems. A total of 3031 lines of code were written to implement the various modules and algorithms of the methodology. The results described in the next chapter are based off of processing a total of 43 days of NOP track data from year 2010, 2011 and 2012, and 6 years of ASPM airport and flight data from years 2007 to 2012. A total of 1 gigabyte (GB) of NOP track data and 2GB of ASPM data were processed.

4 CHAPTER 4: CHICAGO METROPLEX CASE STUDY

This chapter describes a case-study of the benefits of Chicago metroplex de-confliction by flow modification using new navigational capabilities. At Chicago metroplex Required Navigation Performance 0.3 (RNP) approach with Radius to Fix (RF) leg has been introduced to runway 13C at MDW to spatially de-conflict the arrival flows to 13C at MDW and departure flows from 22L at ORD (see section 4.1.1 for details). The successful de-confliction of the Chicago metroplex relies on collaboration and simultaneous equipage (RNP approach capability) by airlines at MDW and must be financially viable.

The adoption of RNP approach by airlines has been slow, primarily due to: (a) issues with estimating the Return-on-Investment (ROI) and (b) the "free rider" issue, i.e., the allocation of benefits to parties that choose not to equip but gain benefits when their competition equips. The fundamental research questions related to these issues are:

1. Does airline equipping with RNP approach capability offer a competitive advantage? (section 4.2)
2. Would airline investment in RNP approach capability yield an acceptable Return on Investment (ROI)? (section 4.5)
3. Are there opportunities to improve the ROI? (section 4.4)

4. What portfolio of incentives/strategies exists for achieving airline equipage? (section 5.2)

This case study uses *high fidelity surveillance track data* coupled with *aerodynamic models* and *weather data* to quantify the performance of MDW terminal area air traffic flows to estimate the potential benefits and financial feasibility of the new technology to airlines at MDW. The case study also estimates the allocation of benefits to ORD and MDW from the de-confliction of the metroplex to address the free rider issue.

The overview of the methodology (described in chapter 3) for benefits of RNP approach at Chicago metroplex is shown in Figure 17. The methodology has six functions.

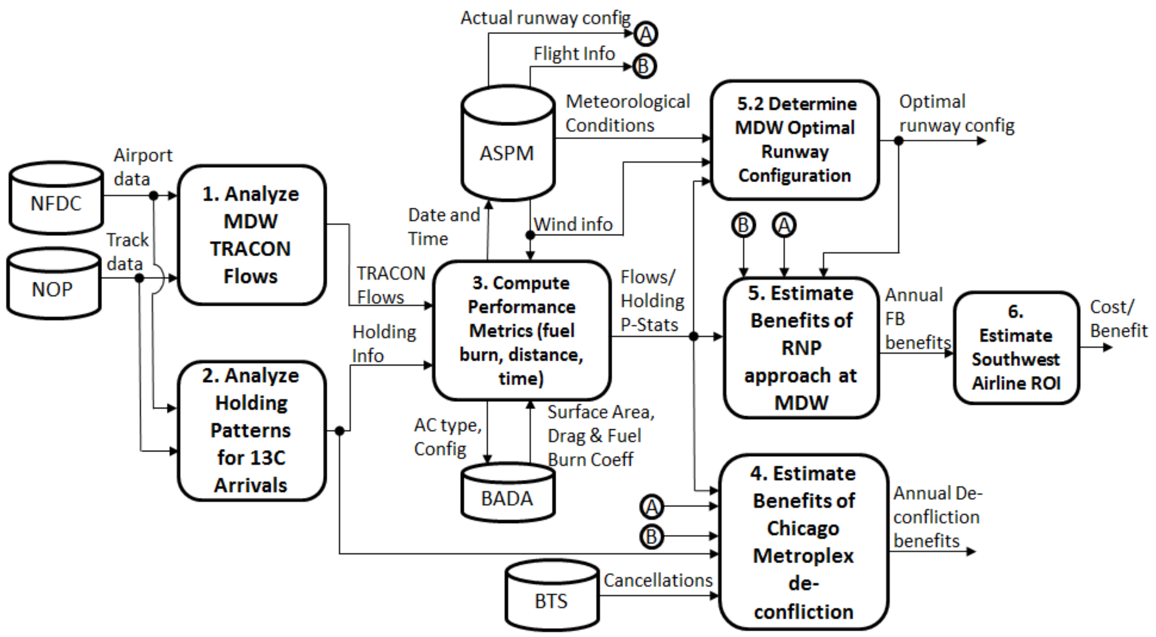

Actual runway config
A
Flight Info
B
ASPM
Meteorological Conditions
5.2 Determine MDW Optimal Runway Configuration
Optimal runway config
NFDC
Airport data
1. Analyze MDW TRACON Flows
NOP
Track data
Date and Time
Wind info
B
A
TRACON Flows
3. Compute Performance Metrics (fuel burn, distance, time)
Flows/ Holding P-Stats
5. Estimate Benefits of RNP approach at MDW
Annual FB benefits
6. Estimate Southwest Airline ROI
Cost/ Benefit
Holding Info
2. Analyze Holding Patterns for 13C Arrivals
AC type, Config
Surface Area, Drag & Fuel Burn Coeff
BADA
4. Estimate Benefits of Chicago Metroplex de-confliction
A
B
Annual De-confliction benefits
BTS
Cancellations

Figure 17: Overview of the methodology for benefits analysis of de-confliction of Chicago metroplex using RNP approach to 13C.

The summary of the overall methodology is as follows:

a. Chicago metroplex de-confliction analysis (functions 2, 3, 4) estimates the total Airline Direct Operating Cost (ADOC) of airborne holding at MDW and departure delays at ORD from the flow conflict between ILS arrivals to runway 13C at MDW and departures from 22L at ORD. The ADOC at MDW and ORD are the annual potential cost savings from elimination of holding patterns for 13C arrivals at MDW and queues for 22L departures at ORD as a result of metroplex de-confliction from using RNP approach to 13C at MDW.
b. Arrival flow analysis at MDW using NOP track data (functions 1 and 3) estimates the performance of the new RNP approach flows to runway 13C in comparison to the conventional approach flows.
c. MDW RNP benefits analysis (function 5) estimates the total potential benefits of using RNP approach procedures to all possible runways and using the optimal runway configuration at MDW.
d. Net Present Value (NPV) analysis (function 6) estimates the ROI of RNP approach for Southwest Airlines.

The Chicago metroplex is chosen for the following reasons:

a. Chicago metroplex is one of the two metroplexes (the other being New York) identified as a candidate for flow de-confliction using RNP approach with RF leg (FAA, 2012h). The new RNP approach to runway 13C at Midway International Airport (MDW) is published and ready for use (FAA, 2012h). At

New York, the new RNP approach procedure is in its testing phase and not yet published for commercial use.

b. The major carrier at MDW (Southwest Airline) is investing $175M in equipping with RNP approach capability (Martin, 2009).

c. Southwest Airlines uses RNP approach procedure at MDW. Analysis of track data for year 2011 shows, on days when 13C is used at MDW during IMC, 9% of the total arrivals use RNP approach.

d. The availability of actual RNP tracks provides an opportunity to estimate, while taking into consideration the variance in terminal flight tracks, its true performance and benefits in comparison to the conventional approaches (ILS and Visual).

This chapter is organized as follows: section 4.1 describes the flow conflict at Chicago metroplex and an overview of the RNP benefits analysis; section 4.2 describes the potential cost saving from elimination of holding patterns for 13C arrivals at MDW and queues for 22L departures at ORD as a result of metroplex de-confliction from using RNP approach to 13C at MDW; section 4.3 describes the arrival flow analysis at MDW and performance of RNP approach in comparison to conventional approach; section 4.4 describes the total potential benefits of RNP approach at MDW from using RNP approaches to other runways, in addition to existing approach to 13C and the optimal runway configuration; and section 4.5 computes the ROI of RNP approach for Southwest Airlines.

4.1 Chicago Metroplex Flow Conflict Analysis

With 3055 operations per day on an average in year 2012, Chicago metroplex is the second largest metroplex in the U.S in terms of traffic volume (ASPM 2012). It has two airports, the Chicago O'Hare International Airport (ORD) and the Chicago Midway International Airport (MDW) within thirteen nautical miles (NM) of each other (see Figure 18).

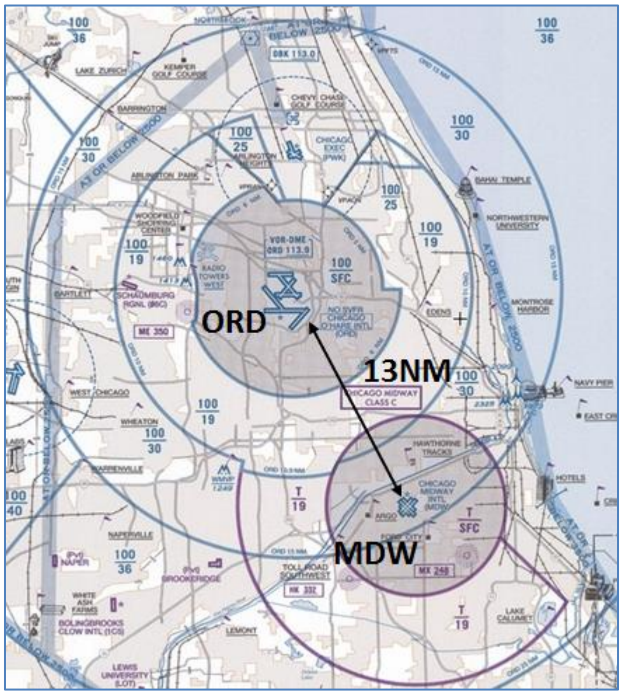


Figure 18: Chicago TRACON, ORD and MDW are 13NM and share common terminal airspace.

4.1.1 Flow Conflict at Chicago Metroplex

To avoid cross wind landings, when winds are greater than 20 knots and from southeast, the arrivals to MDW must use 13C. When Instrument Meteorological Condition (IMC) exist (i.e. cloud ceiling less than 1,000 feet above ground level or visibility less than 3 statute miles) these arrivals must use Instrument Landing System (ILS) to runway 13C. The turn on the base leg to the final leg of the ILS approach to 13C can occur 10.1 NM from the runway threshold and is 3NM from end of runway 22L at ORD (see Figure 19).

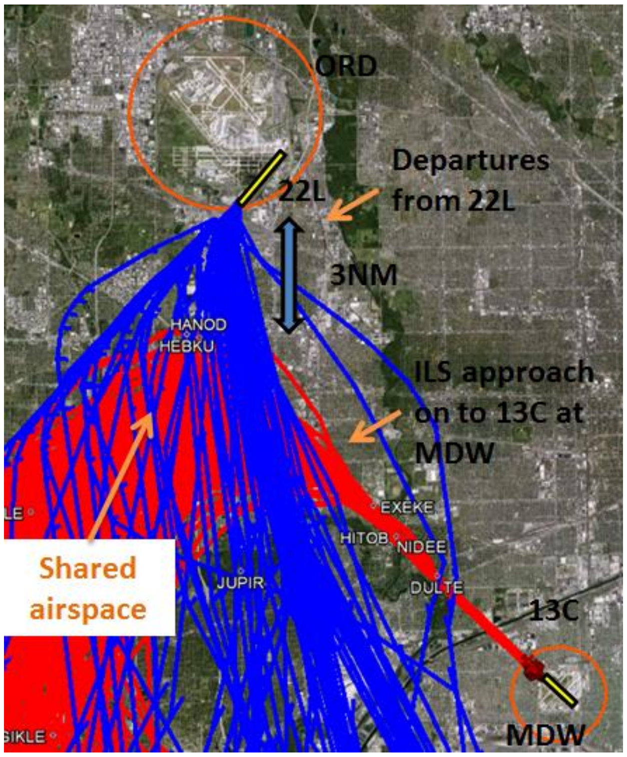


Figure 19: Flow Conflict at Chicago Metroplex between ORD 22L departures and MDW 13C arrivals, when winds are greater than 20 knots from southeast in IMC

This results in a flow conflict in the shared airspace due to the lack of vertical separation between departures from 22L at ORD and arrivals to 13C at MDW. An analysis of track data shows that arrivals to 13C at MDW are on an average at an altitude of 2800 feet when in the shared airspace. The departure from 22L at ORD using the shared airspace cannot always climb to an altitude of 3800 feet or more (1000 feet more), required to guarantee safe separation.

The tactical time-based sharing employed by the TRACON enables the use of the common airspace by alternating between departures at ORD and arrivals at MDW. This results in ground holding for departures from 22L at ORD and airborne holding for MDW arrivals to 13C. The ground holdings for departures result in departure delays and in some circumstances may contribute to cancelled flights. The airborne holding for arrivals results in en-route delays.

4.1.2 Chicago Flow Conflict Frequency

An analysis of ASPM data for years 2007 to 2012 shows MDW experiences IMC on an average 13% of the time per year (see Figure 20). Further, runway 13C at MDW is used on an average 12% of the time during IMC (see Figure 21).

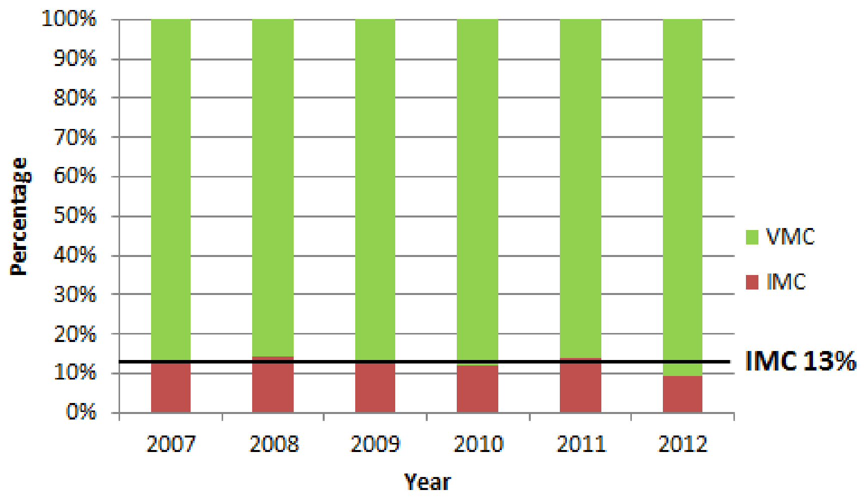


Figure 20: Meteorological Conditions at MDW shows MDW is IMC on an average 13%

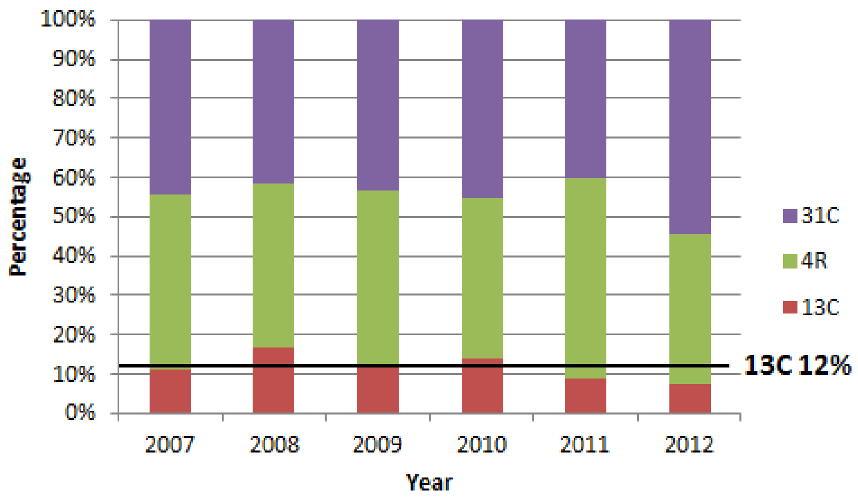


Figure 21: Runway Usage at MDW during IMC, 13C is used on an average 12% of the time in IMC

A potential flow conflict can occur on an average 1.6% of the time per year when runway 13C is in use during IMC. This amounts to on an average 17 IMC days per year, with the average duration of IMC lasting on an average 5.54 hours per day, as shown in Figure 22.

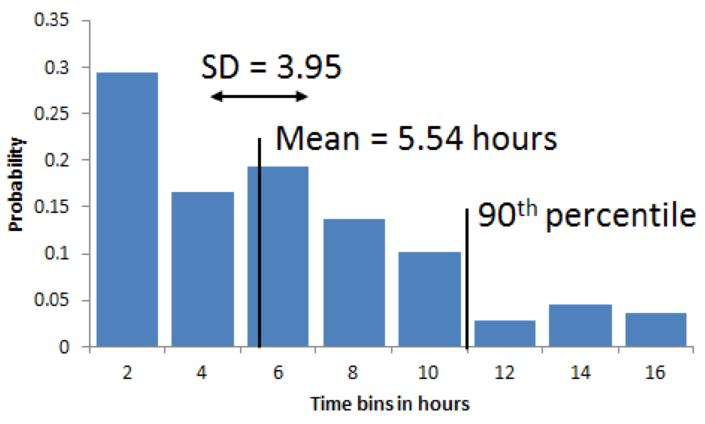


Figure 22: Distribution of the duration of usage of runway 13C at MDW per day during IMC

4.1.3 Chicago Metroplex flow de-confliction using RNP approach

The new RNP approach with radius to fix leg to runway 13C cuts the corner on the final leg of the approach adding sufficient distance (7NM) to maintain safe vertical separation between the departure flows from 22L at ORD and arrival flows to 13C at MDW. This allows arrivals to MDW without interfering with departures from runway 22L at ORD. Figure 23 shows sample RNP approach tracks to runway 13C at MDW and their relative position to departure from runway 22L at ORD.

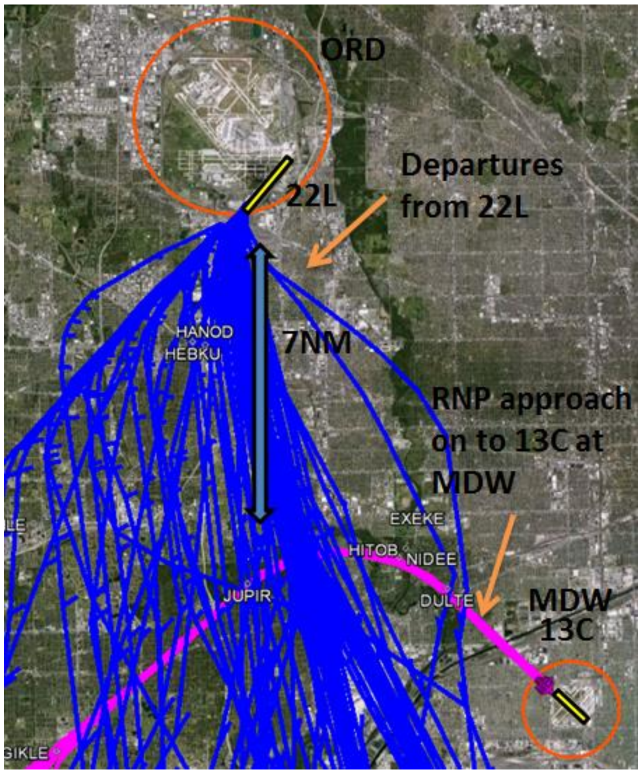


Figure 23: De-confliction of flows using RNP approach creates 7NM lateral separation to allow arrivals to MDW without interfering with departures from runway 22L at ORD.

4.1.4 Overall Approach to Chicago De-confliction Analysis

The successful de-confliction of the Chicago metroplex flows relies on achieving complete equipage for airlines operating at MDW. While the RNP approach procedure has been published and is ready for use, analysis of track data for year 2011 shows, on days when 13C is used at MDW during IMC, 9% of the total arrivals use RNP approach. Therefore, it is important to quantify the benefits and the ROI of RNP approach at MDW from an airlines perspective. The primary benefits of this technology to individual airlines at MDW are cost savings from:

a. Shorter track distance in the terminal airspace compared to conventional instrument approach procedures
b. Elimination of airborne holdings that would otherwise occur due to the metroplex flow conflict.

This chapter uses the methodology described in chapter 3 to quantify the above two benefits at MDW using surveillance NOP track data. In addition, the benefits of elimination of departures queue at runway 22L and the ROI of RNP approach to the majority carrier at MDW (Southwest Airline) have also been estimated. The summary of the overall approach is as follows:

a. Chicago metroplex de-confliction analysis (section 4.2) estimates the total Airline Direct Operating Cost (ADOC) of airborne holding at MDW and departure delays at ORD from the flow conflict between ILS arrivals to runway 13C at MDW and departures from 22L at ORD. The ADOC at MDW and ORD are the annual potential cost savings from elimination of holding patterns for 13C arrivals at MDW and queues for 22L departures at ORD as a result of metroplex de-confliction from using RNP approach to 13C at MDW.

b. Arrival flow analysis at MDW using NOP track data (section 4.3).estimates the performance of the new RNP approach flows to runway 13C in comparison to the conventional approach flows

c. MDW RNP benefits analysis (section 4.4) estimates the total potential benefits of using RNP approach procedures to all possible runways and using the optimal runway configuration at MDW

d. Net Present Value (NPV) analysis (section 4.5) estimates the ROI of RNP approach for Southwest Airlines.

4.2 Benefits of RNP Approach to 13C at MDW to the Chicago Metroplex

The RNP approach to 13C at MDW de-conflict flows between the ILS arrivals to runway 13C at MDW and departures from runway 22L at ORD. The flow conflict at Chicago metroplex is overcome through tactical time based sharing of the common airspace by alternating between departures at ORD and arrival at MDW. This results in ground holding for departures from runway 22L at ORD when arrivals using ILS to runway 13C at MDW are in progress and airborne holding for MDW arrivals to 13C when departures from 22L at ORD are in progress. The ground holdings for departures result in departure delays and cancelled flights. The airborne holdings for arrivals result in en-route delays.

This section computes total Airline Direct Operating Cost (ADOC) of airborne holding at MDW and departure delays at ORD from the flow conflict between ILS arrivals to runway 13C at MDW and departures from 22L at ORD. The ADOC at MDW and ORD are the annual potential cost savings from elimination of holding patterns for 13C arrivals at MDW and queues for 22L departures at ORD as a result of metroplex de-confliction from using RNP approach to 13C at MDW.

4.2.1 Holding pattern analysis

The holding pattern analysis is conducted using NOP track data and the methodology is described in section 3.3. Boundaries are defined around waypoints with published holding pattern procedures. The total cumulative turn angle of flight track is computed for tracks within the defined boundaries. Tracks with cumulative turn angle of 360 degrees or more are filtered out at holding patterns. The holding pattern tracks and the boundaries defined to capture the holding patterns are shown in Figure 24.

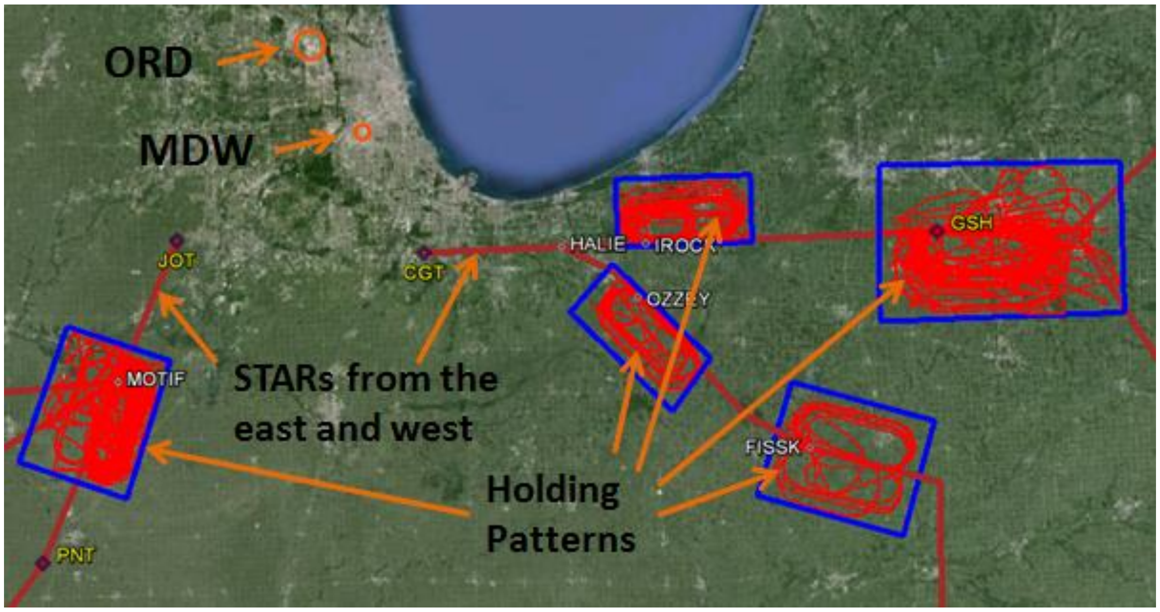


Figure 24: Holding patterns for 13C arrivals detected along the STARs at MDW

Using the algorithm in the step1 of section 3.5.2, the holding patterns for ILS arrival flow to runway 13C at MDW due to conflict with departures from 22L at ORD are detected. The duration of the conflict period along with the count of arrivals at MDW, departures from ORD, holding patterns and the frequency in terms of holdings per 100 arrivals are shown in Table 9. For the NOP data analyzed, there were a total of 811 ILS arrivals to 13C at MDW and 83 holding patterns during periods of flow conflict. There were on average 10.23 holdings per 100 arrivals.

Table 9: Airborne holding count estimated from NOP data analysis, for 13C arrival during periods of flow conflict with 22L departures at ORD

Date (yyyymmdd)	Duration of flow conflict (Hrs)	13C Arrivals	22L Departures	Holding Count	Holding per 100 arrivals
20111214	13.7	292	169	28	10
20110426	3.7	95	32	25	26
20110615	2.3	48	26	8	17
20110223	2.9	46	47	7	15
20101122	1.0	13	21	7	54
20110620	1.1	31	78	5	16
20111126	5.3	110	40	2	2
20100123	13.7	176	43	1	1
20120122	13.1	251	49	0	0
20110117	7.7	151	34	0	0
20120213	1.3	24	22	0	0
20101211	7.3	100	22	0	0
20100124	2.2	23	19	0	0
20110124	3.2	57	7	0	0

The mean and standard deviation of holding pattern's duration, track distance and fuel burn is shown in Table 10. These are estimated for all aircraft types and for B73's.

Table 10: Holding patterns statistics for holding duration, track distance and fuel burn for the fleet mix at MDW and for B73's

Aircraft Type	Track Count	Holding Duration (min)		Holding Track Distance (NM)		Holding Fuel burn (kg)	
		Mean	SD	Mean	SD	Mean	SD
all	83	18.37	9.08	79.75	38.67	346.27	255.24
B73	42	16.59	7.90	72.38	35.81	458.33	238.32

The total annual cost of holding pattern for 13C ILS arrivals at MDW is estimated based on the holding pattern statistics (Table 10) and frequency of holding patterns (Table 9). The analysis is done using the methodology in step 2 of section 3.5.2.

The number of days and duration when the Chicago metroplex had conflicting runway configuration (runway 13C in the arrival configuration at MDW and runway 22L in the departure configuration at ORD) during IMC, the corresponding totals for the number of arrival at 13C at MDW, holding count estimates, fuel burn, en route delay and airline direct operating cost due to the en route delay are shown in Table 11 for years 2007 to 2012.

Table 11: Cost of Holding due to flow conflict per year for 13C arrivals

Year	Conflict Days per Year	Conflict Duration per Year	Total 13C Arrivals per Year	Holding Count per Year	Total Fuel burn (gallons)	Total Time in Holding (min)	Airline Direct Operating Cost ($)
2007	8	2.81	391	40	4517.2	680.0	26618.7
2008	15	3.63	913	93	10547.8	1587.8	62155.7
2009	7	2.25	267	27	3084.6	464.3	18177.0

2010	10	3.05	424	43	4898.4	737.4	28865.3
2011	16	2.48	633	65	7313.0	1100.9	43093.7
2012	12	4.52	921	94	10640.3	1601.7	62700.3
Average	**11**	**3.19**	**592**	**61**	**6833.6**	**1028.7**	**40268.4**
Std Dev	**4**	**0.8**	**278**	**28**	**3215.0**	**484.0**	**18945.2**

The results indicate that on average there are 61 aircraft that are kept in holding per year as a result of flow conflict between 13C ILS arrivals at MDW and 22L departure at ORD. This results in fuel burn of 6,833 gallon per year on average with a standard deviation of 3,215 and delay of 1,028 minute per year on average with a standard deviation of 484. The average airline direct operating cost (ADOC) due en route delays is $40,268 per year with a standard deviation of 18,945. The ADOC is estimated using Equation 35, Equation 36 and Equation 37.

4.2.2 Departure Delay Analysis

The flow conflict at Chicago metroplex (between departures from 22L at ORD and ILS arrivals to 13C at MDW) results in the reduction of the arrival and departure capacity at ORD. This results in arrival and departure delays and can contribute to cancelled flights at ORD.

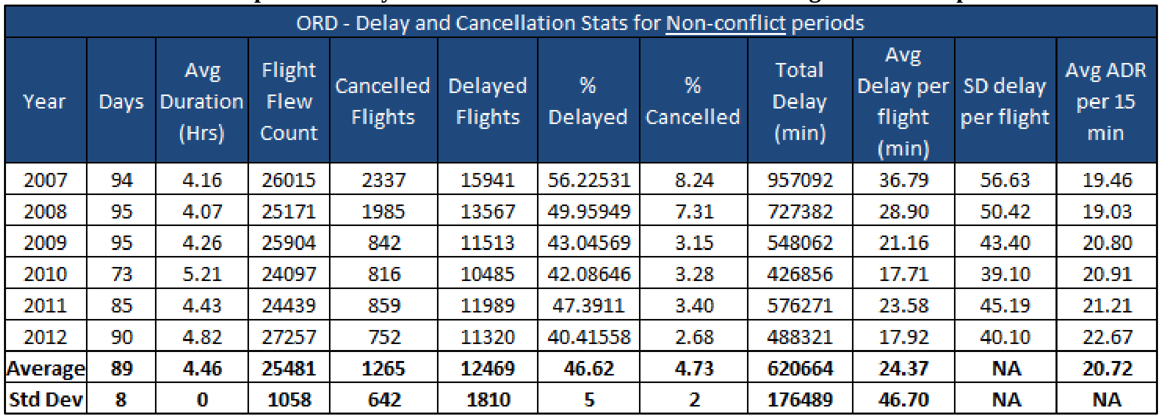
This section estimates the excess departure delays and cancellations and their associated costs as a lower bound for the additional cost of metroplex flow conflict to airlines operating at ORD. The excess departure delays and cancelled flights are estimated as the difference in the departures delays and cancelled flights at ORD for flow conflict and no-flow conflict periods. The flow conflict periods are during IMC when MDW is configured to use runway 13C as one of its arrival runways and ORD is configured to use runway 22L as one of its departure runways. The no-flow conflict periods are during IMC when MDW is not configured to use 13C as one of its arrival runways. The excess departure delays and cancelled flights thus estimated will show the impact of 13C arrivals on ORD departures during IMC.

The number of days and duration when the Chicago metroplex has conflicting runway configuration (runway 13C in the arrival configuration at MDW and runway 22L in the departure configuration at ORD) during IMC and the corresponding statistics for the number of departures, number of cancelled flights, number of delayed flights, percentage delayed and cancelled flights, total delays, average and standard deviation of delay per flight and average airport departure rate (ADR) are shown in Table 12 for year 2007 to 2012. The same set of statistics for periods with no metroplex flow conflict is shown in Table 13.

Table 12: Departure delay and cancellation statistics at ORD during flow conflict periods

ORD - Delay and Cancellation Stats for Conflict periods											
Year	Days	Avg Duration (Hrs)	Flight Flew Count	Cancelled Flights	Delayed Flights	% Delayed	% Cancelled	Total Delay (min)	Avg Delay per flight (min)	SD delay per flight	Avg ADR per 15 min
2007	8	2.81	1441	218	1235	74.44244	13.14	82405	57.19	65.69	18.84
2008	15	3.63	3561	179	2685	71.79144	4.79	139349	39.13	52.13	16.94
2009	7	2.25	892	76	515	53.20248	7.85	30579	34.28	57.41	17.90
2010	10	3.05	1843	130	1186	60.11151	6.59	56044	30.41	44.38	18.48
2011	16	2.48	2384	116	1333	53.32	4.64	69305	29.07	47.23	19.04
2012	12	4.52	3215	119	2014	60.40792	3.57	92745	28.85	49.58	19.17
Average	**11**	**3.19**	**2223**	**140**	**1495**	**63.27**	**5.91**	**78405**	**35.27**	**NA**	**18.36**
Std Dev	**3**	**1**	**942**	**46**	**687**	**8**	**3**	**33680**	**52.44**	**NA**	**NA**

Table 13: Departure delay and cancellation statistics at ORD during non-conflict periods



ORD - Delay and Cancellation Stats for Non-conflict periods

Year	Days	Avg Duration (Hrs)	Flight Flew Count	Cancelled Flights	Delayed Flights	% Delayed	% Cancelled	Total Delay (min)	Avg Delay per flight (min)	SD delay per flight	Avg ADR per 15 min
2007	94	4.16	26015	2337	15941	56.22531	8.24	957092	36.79	56.63	19.46
2008	95	4.07	25171	1985	13567	49.95949	7.31	727382	28.90	50.42	19.03
2009	95	4.26	25904	842	11513	43.04569	3.15	548062	21.16	43.40	20.80
2010	73	5.21	24097	816	10485	42.08646	3.28	426856	17.71	39.10	20.91
2011	85	4.43	24439	859	11989	47.3911	3.40	576271	23.58	45.19	21.21
2012	90	4.82	27257	752	11320	40.41558	2.68	488321	17.92	40.10	22.67
Average	**89**	**4.46**	**25481**	**1265**	**12469**	**46.62**	**4.73**	**620664**	**24.37**	**NA**	**20.72**
Std Dev	**8**	**0**	**1058**	**642**	**1810**	**5**	**2**	**176489**	**46.70**	**NA**	**NA**

The results indicate that during the flow conflict periods there are on average 2,362 scheduled departures per year at ORD, with 63.25% delayed and 5.91% cancelled flights. The average delay per flight on average is 35.27 minutes with standard deviation 52.44.

During periods of no flow conflict there are on average 26,746 scheduled departures per year at ORD, with 46.62% delayed and 4.73% cancelled flights. The average departure delay per flight on average is 24.37 minutes with standard deviation 46.70.

The flow conflict at Chicago metroplex results in 16.63% additional delayed flights and 1.18% additional cancelled flights. The average additional departure delay per flights is 11.35 minutes.

The de-confliction of the airspace by introduction of RNP approach to runway 13C at MDW could potentially result in savings of 51,563 minutes of excess total delays and 28 cancelled flights per year for departures at ORD. This amounts to $1.33M in additional airline direct operating cost (ADOC) at the rate of $1386 per hour for ground delays (taxi-out) at ORD (computed using Equation 31, Equation 32, Equation 33) and $4977 per cancellation (FAA, 2012i).

4.2.3 Summary of Benefit of PBN Approach Procedure at Chicago Metroplex

This section presents the benefits to Chicago metroplex from de-confliction of flows by airspace redesign using RNP approach to runway 13C at MDW.

The flow conflict between 13C ILS arrivals at MDW and 22L departures at ORD in the Chicago metroplex occur on an average 1.6% of the time per year. The introduction of the RNP approach to 13C at MDW to de-conflict the traffic flow for these periods is estimated to reduce the direct airline operating cost per year on an average by $.04M at MDW and $1.33M at ORD.

The magnitude of the benefits of de-confliction for the operator expected (airlines at MDW) to equip is small and the asymmetry in benefits between competing operators (airlines at MDW and ORD) in neighboring airports is large (1:33 in favor of ORD departures). As a result, there is no competitive advantage for MDW airlines in equipping due to the free rider issue. However, the successful de-confliction of Chicago metroplex relies on achieving complete equipage (RNP approach capability) for airlines operating at MDW.

4.3 Arrival Flow Analysis at MDW

4.3.1 Flow Characterization

The flows are characterized by specifying a direction, runway and arrival approach procedure. This is done by studying the runway configuration and published arrival procedure for the airport.

At MDW there are ten runways, as shown in Figure 25. Of these, runway 13C, 31C and 4R have ILS. Runway 13C has an RNP approach as well. In IMC, the arrivals into MDW are restricted (depending on the wind conditions) to one of the three ILS runways.

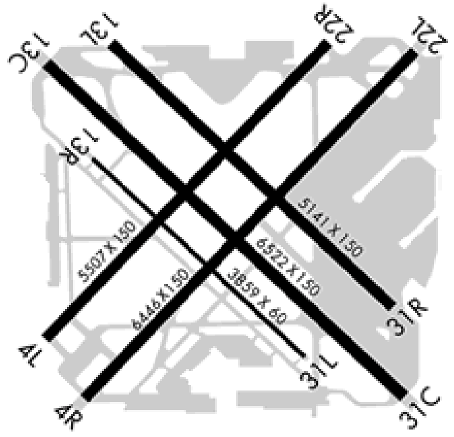


Figure 25: MDW runway configuration (Source: airnav.com), runways 13C, 31C and 22L have ILS, and only runway 13C has RNP 0.3 w/ RF leg approach (2011)

For arrivals into MDW there are three STARs one from the west and two from the east. The two STARs (one RNAV and one conventional) from the east terminate at Chicago Heights VORTAC (CGT) and the STAR from the west terminates at Joliet VORTAC (JOT) (Figure 2). These two waypoints feed traffic into the terminal area at MDW.

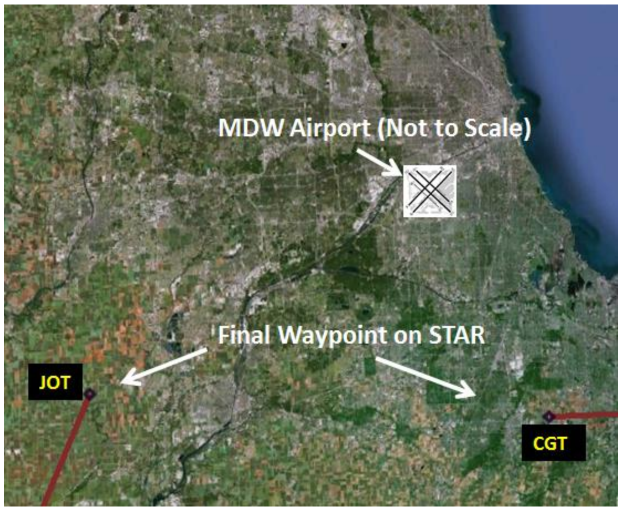


Figure 26: Location of Final waypoint on STAR w.r.t the airport (MDW).

The location of the final waypoint on the STAR with respect to the runways determines the direction from which the traffic flows. These combined with the runway and arrival approach procedure characterize various TRACON flows. For instance, at MDW flights arriving from JOT for an ILS approach to runway 13C are assigned to “W 13C ILS” flow. Similarly “E 13C RNP” flow will consist of flights arriving from the CGT (east waypoint) to runway 13C for an RNP approach

A total of 28 flows are possible at MDW based on the characterization process described above. A summary of the counting process is shown in Table 14

Table 14: Total Possible flows at MDW by matching the final STAR waypoint with the approach type and runway

	Approach				Direction			
Rwy	Visual	ILS	RNP	Total	East	West	Total	Flows
13C	1	1	1	3	1	1	2	6
13L	1	0	0	1	1	1	2	2
13R	1	0	0	1	1	1	2	2
4R	1	1	0	2	1	1	2	4
4L	1	0	0	1	1	1	2	2
31C	1	1	0	2	1	1	2	4
31L	1	0	0	1	1	1	2	2
31R	1	0	0	1	1	1	2	2
22L	1	0	0	1	1	1	2	2
22R	1	0	0	1	1	1	2	2
							Total	28

4.3.2 Flow Assignment Results

Forty three days of NOP track data for Chicago TRACON (C90) are analyzed. The days are selected from the years 2010, 2011 and 2012 to cover various meteorological conditions and runway configurations at MDW. After filtering out general aviation flights a total of 11,275 tracks are processed. A total of 438 tracks are filtered out as incomplete tracks or go-arounds.

Runways 13R/31L are excluded from the assignment process because they too short for commercial flight landings (FAA, 2005a) and too close to runways 13C/31C making it difficult to accurately assign them flight tracks.

The flight count obtained from the track data for the remaining 24 flows are shown in Table 15. The flow track count indicates that runways 13C, 31C, 4R and 22L are the four major runways at MDW. The 16 flows associated with these runways are highlighted in Table 15. This analysis will focus on these 16 flows.

Table 15: Flow Assignment results for MDW from track data. Runways 13C, 31C, 4R and 22L are the major runways used.

Sl.no	Direction	Runway	Approach	Count
1	E	**13C**	**ILS**	**798**
2			**RNP**	**87**
3			**Visual**	**1026**
4		13L	Visual	8
5		**22L**	**Visual**	**840**
6		22R	Visual	70
7		**31C**	**ILS**	**1467**
8			**Visual**	**345**
9		31R	Visual	5
10		4L	Visual	48
11		**4R**	**ILS**	**390**
12			**Visual**	**1181**
13	W	**13C**	**ILS**	**568**
14			**RNP**	**151**
15			**Visual**	**857**
16		13L	Visual	9
17		**22L**	**Visual**	**650**
18		22R	Visual	56
19		**31C**	**ILS**	**387**
20			**Visual**	**987**
21		31R	Visual	2
22		4L	Visual	50
23		**4R**	**ILS**	**729**
24			**Visual**	**564**

The major flows (in terms of the flow track count) from the east and the west are the ILS, RNP and Visual approaches on to runway 13C, the ILS and Visual approaches on to runway 4R and 31C and the Visual approaches on to runway 22L. A sample of these flows along with the legend is shown in Figure 7 and Figure 8.

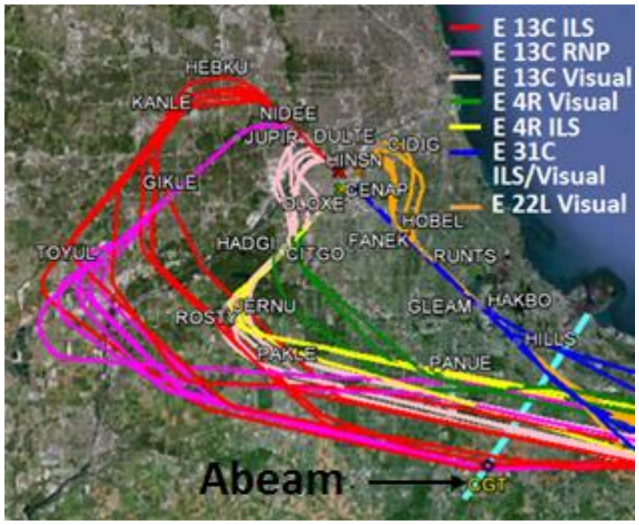


Figure 27: Sample of the eight major flows from the east at MDW

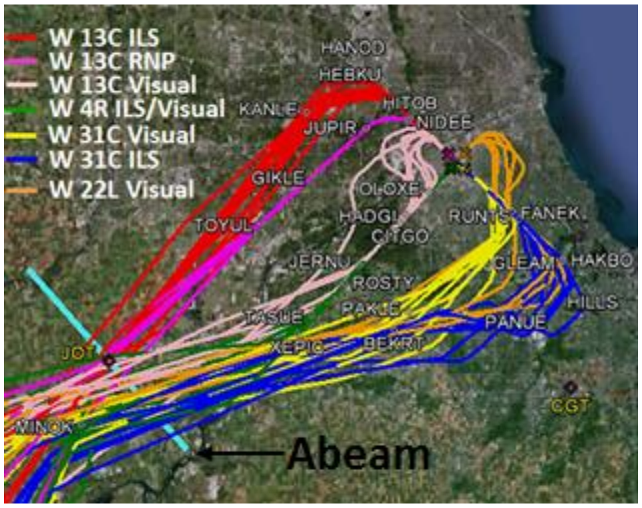


Figure 28: Sample of the eight major flows from the west at MDW

4.3.3 Flow Performance Statistics

This section describes the performance of TRACON flows in terms of track distance (NM), time (min) and fuel burn (kg) in the terminal airspace. The performance metrics are measured from the abeam (see Figure 27 and Figure 28) on the final waypoint to STAR to the runway threshold.

The analysis is conducted using National Offload Program (NOP) data for 43 days selected to provide data for the 16 flows at MDW. After filtering out general aviation flight a total 11,275 tracks are processed. A total of 438 tracks are filtered out as incomplete tracks or go-arounds.

The performance metrics are computed for runways 13C, 31C, 22L and 4R for:

1 Individual flows from the east and the west

2 Approach type (ILS, Visual or RNP); by combining flows from east and west for an approach type

3 All flows (approach type and direction) combined for a given runway.

The runways are ranked (from lowest to highest) based on the track distance, track time and fuel burn statistics.

4.3.3.1 Track Time and Distance Statistics

The ranking of runways, from best to worse, based on individual flows from the east is shown in Table 16. The flows are ranked based on mean track distance and mean track time of each flow.

From the east the ILS and Visual approach on to runway 31C are the shortest as this is a straight-in approach. This is followed by visual approach on to runways 22L, 4R, and 13C, the ILS approach on to 4R, RNP approach on to 13C and finally ILS approach on to 13C. For flows from the east, the ILS to 13C has the longest track distance and time.

Table 16: Ranking of runways by mean Track Time for flows from the East (lowest to highest)

Dir/Runway/Approach			Count	Track Time (min)		Distance (NM)		# of Turns in the Terminal Airspace
				Mean	SD	Mean	SD	
E	31C	ILS	1467	5.90	0.69	16.56	1.44	Straight-In
E	31C	Visual	345	6.12	0.80	16.77	2.48	Straight-In
E	22L	Visual	840	6.71	0.83	20.39	1.25	One turn
E	4R	Visual	1181	8.84	1.77	28.74	4.17	One turn
E	13C	Visual	1026	9.62	1.59	32.58	4.08	Two turns
E	4R	ILS	390	10.92	2.58	33.59	5.52	One turn
E	13C	RNP	87	13.40	1.88	45.31	4.81	Two turns
E	13C	ILS	798	14.37	2.21	48.20	5.37	Two Turns

It should be noted that despite the accuracy of the RNP approach procedure, the variance of the RNP flow from the east to runway 13C is almost as high as the corresponding ILS approach. The variance in RNP approach flow occurs on the downwind and turn-to-base, while the variance on the ILS approach occurs by "tromboning" on the downwind, base leg and the turn to final. This phenomenon is illustrated in Figure 29.

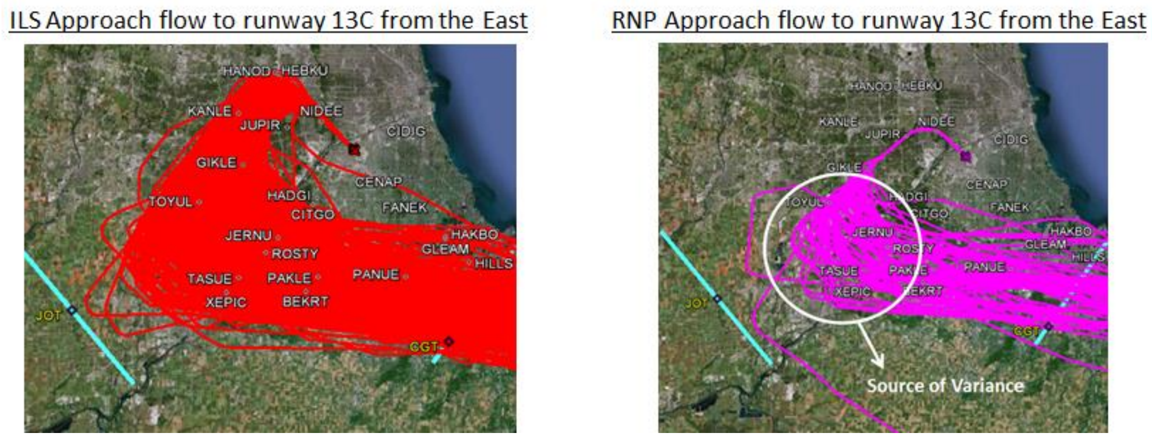


Figure 29: Comparison of ILS and RNP approach flow from the East to runway 13C. The variance in the RNP approach flows is induced in the base leg for safe merging and spacing.

The ranking of runways, from best to worse, based on individual flows from the west is shown in Table 17. For flows from the west, the ILS and visual approach on to runway 4R are the shortest followed by, RNP, visual and ILS on to 13C, visual approach on to 31C, visual approach on to 22L and ILS on to 31C.

Table 17: Ranking of runways by mean Track Time for flows from the West (lowest to highest)

Dir/Runway/Approach			Count	Track Time (min)		Track Distance (NM)		# of Turns in the Terminal Airspace
				Mean	SD	Mean	SD	
W	4R	ILS	729	8.58	0.89	28.94	0.64	Straight-In
W	4R	Visual	564	9.07	1.27	29.34	2.05	Straight-In
W	13C	RNP	151	9.47	0.67	32.67	0.64	One turn
W	13C	Visual	857	9.65	1.11	33.59	1.98	One turn
W	13C	ILS	568	10.63	1.01	36.22	1.78	One turn
W	31C	Visual	987	11.91	1.94	42.90	4.58	One turn
W	22L	Visual	650	12.39	2.07	46.16	5.20	Two turns
W	31C	ILS	387	13.77	2.39	47.44	5.60	One turn

The variance in flow increases with the total distance and time in the terminal area and the number of turns made. For instance, consider flows on to runway 13C from east and west. The 13C flows from the east are longer and make two turns before the final approach; whereas, the 13C flows from the west are direct and make one turn before the final approach. As a result, flows to runway 13C from the west are efficient and have lower variance compared to the flows from the East (Figure 30).

The x-axis is the track distance and the y-axis is the normalized frequency. The vertical line marks the mean of the distribution. For flows to runway 13C from the east, the track distance distribution about the mean is wide spread and has a fat tail. This is caused by the vectoring in the terminal. For flows to runway 13C from the west, the track distance distribution about the mean is tight, especially for the RNP flow

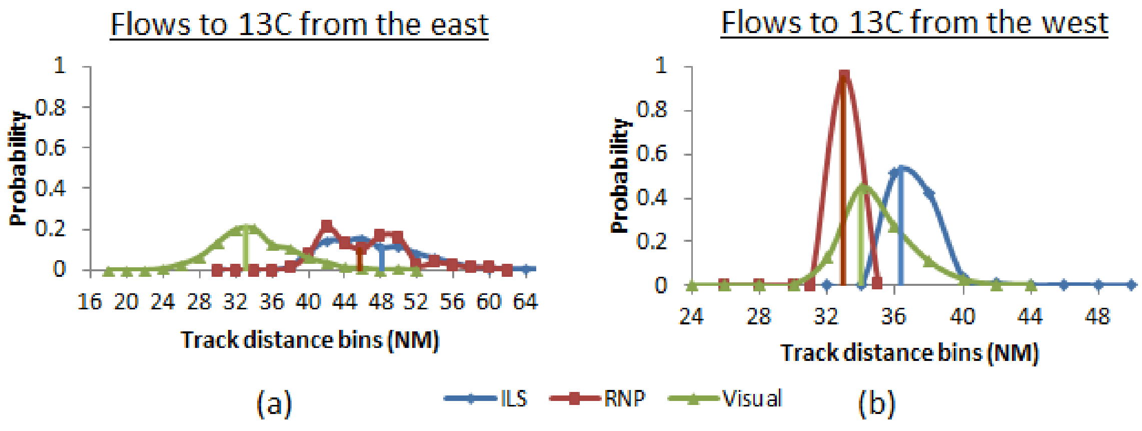


Figure 30: Comparison between east and west track distance distribution for flows to runways 13C

The flows from east and west consolidated and ranked by approach type are shown in Table 18. The first three rows show track time and track distance statistics for the ILS approach, followed by RNP approach to runway 13C, and visual approaches to the major four runways 4R, 31C, 22L and 13C at MDW. Runway 22L does on have an ILS approach and only runway 13C has a RNP approach.

Table 18: Runways/Approach type ranked by Mean Track Time (lowest to highest) for arrivals from the east and the west combined.

Runway/ Approach		Count	Track Time (min)		Track Distance (NM)	
			Mean	SD	Mean	SD
4R	ILS	1119	9.75	2.26	31.27	4.57
31C	ILS	1854	9.83	4.31	32.00	15.97
13C	ILS	1366	12.50	2.54	42.21	7.20
13C	RNP	238	11.44	2.42	38.99	7.19
4R	Visual	1745	8.95	1.54	29.04	3.30
31C	Visual	1332	9.02	3.25	29.84	13.58
22L	Visual	1490	9.55	3.25	33.27	13.43
13C	Visual	1883	9.64	1.37	33.08	3.24

For ILS approaches the flow statistics for 4R and 31C are nearly the same, followed by 13C. For visual approaches 4R is the shortest, followed by 31C, 22L and 13C. The RNP approach on to 13C is on an average shorter than the corresponding ILS approach by 1.06 minute in terms of time and 3.22NM in terms of distance. The RNP approach to 13C is on average 8% shorter track time and track distance than the ILS approach to 13C.

The visual approaches are shorter than the corresponding ILS approaches by 11% for 4R, 15% for 31C, and 23% for 13C.

The visual approach on to 13C is shorter than the RNP approach by 1.8 minute in terms of time and 5.91 NM in terms of distance

The overall ranking for runways is shown irrespective of direction of flow or type of approach is shown in Table 19. On the whole, flows to runway 4R have the lowest mean track distance (30.15 NM) and track time (9.35 minute) followed by 31C (30.92NM and 9.42 minute), 22L (33.27 NM and 9.55 minute, and 13C (38.10 NM and 11.19 minute).

Table 19: Runways ranked by mean Track Time (lowest to highest)

		Track Time (min)		Track Distance (NM)	
Rwy	Count	Mean	SD	Mean	SD
4R	2864	9.35	1.97	30.15	4.14
31C	3186	9.42	3.84	30.92	14.86
22L	1490	9.55	3.25	33.27	13.43
13C	3487	11.19	2.48	38.10	7.23

4.3.3.2 Fuel Burn Performance Statistics

At MDW there are 98 aircraft types (jets and turbo props). The list of aircraft types in each category is shown in Figure 31. The narrow body aircrafts account of 76% of the flights into MDW, followed by business jets (15%), regional jets (5%) and turbo props (4%). The most dominant aircraft type, the Boeing 73's (B73's), accounts for 72% of the flights at MDW.

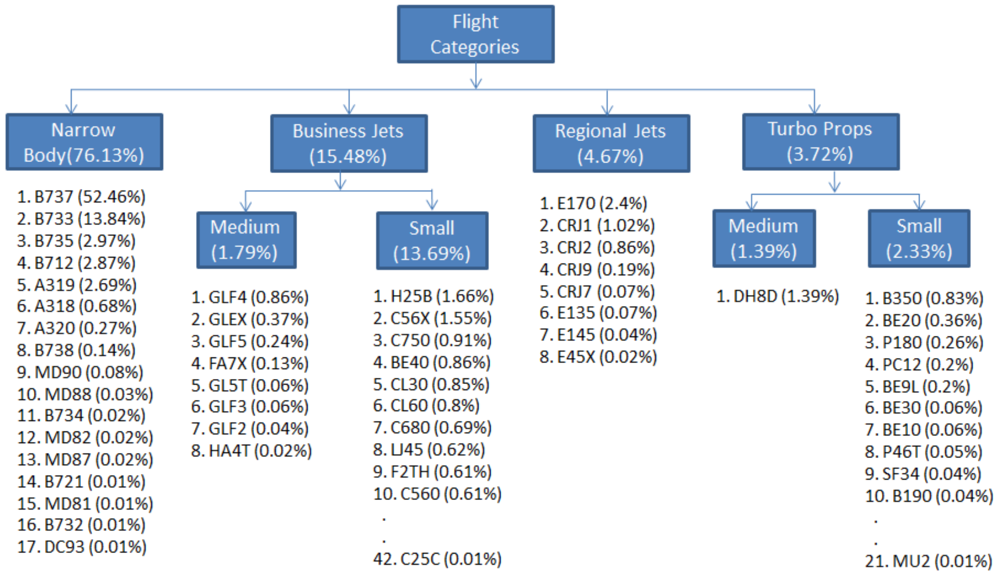


Figure 31: Aircraft types under each category at MDW

4.3.3.2.1 Fuel Burn Performance Statistics by Flow

At MDW all arrival flows except for the RNP approach on to 13C have a fleet mix shown in Figure 31. The RNP approach on to 13C, first published in 2011 is used on a limited basis by Southwest airlines, whose fleet consists of Boeing 737's. Analysis of track data for year 2011 shows on days when 13C is used at MDW during IMC nine percent of the Southwest arrivals use RNP approach.

This section describes fuel burn performance statistics for flows from the east and west for all aircraft types as shown in Table 20 and Table 21. Also, to show the benefits of the RNP approach, the fuel burn performance of B73's are also computed and shown in Table 22 and Table 23.

The tables also show track-time, level-time statistics and the mean level-time to track-time ratio. The track time is the total time taken by the flight from the final waypoint on the STAR to the runway threshold. The level time is the total level off time from the final waypoint on the STAR to the runway threshold, (i.e. when the change is the vertical profile is less than 100 feet per minute). The flows are ranked (from least to most) based on the average fuel burn per flight

From the east, the ILS and Visual approaches on to runway 31C (100.48 kg and 100.72kg) have the lowest average fuel burn per flight, followed by the visual approaches on to runways 22L, 4R and 13C (140.20kg, 189.14kg and 206.28 kg) and the ILS approach on to runway 4R (279 kg) and runway 13C (336.5kg) (Table 20).

Table 20: Fuel burn, Track time and Level time Statistics for flows from the east, for all aircrafts, ranked by fuel burn

Flow			Track Count	Fuel Burn (kg/flight)		Track time (min)		Level Time (min)		% Level	# of Turns in the Terminal Airspace
Dir/Rwy/Procedure				Mean	SD	Mean	SD	Mean	SD		
E	31C	Visual	345	100.48	47.00	6.12	0.80	0.42	0.86	6.90	Straight-In
E	31C	ILS	1467	100.72	45.77	5.90	0.69	0.43	0.74	7.26	Straight-In
E	22L	Visual	840	140.20	59.21	6.71	0.83	0.42	0.65	6.20	One turn
E	4R	Visual	1181	189.14	101.82	8.84	1.77	2.25	1.74	25.51	One turn
E	13C	Visual	1026	206.28	100.74	9.62	1.59	1.89	1.29	19.66	Two turns
E	4R	ILS	390	279.00	161.18	10.92	2.58	3.50	2.42	32.02	One turn
E	13C	ILS	798	336.50	164.49	14.37	2.21	6.40	2.50	44.53	Two turns

From the west, ILS and visual approaches on to runway 4R (160.66 kg and 162.12 kg) have the lowest fuel burn per flight, followed by the visual and ILS approaches on to runway 13C (181.19 kg and 215.68 kg), visual approaches on to runway 31C and 22L (249.9 kg and 275.32 kg) and the ILS approach on to runway 31C (313.4 kg) (Table 21).

Table 21: Fuel burn, Track time and Level time Statistics for flows from the west, for all aircrafts, ranked by fuel burn

Flow			Track Count	Fuel Burn (kg)		Track time (min)		Level Time (min)		% Level	# of Turns in the Terminal Airspace
Dir/Rwy/Procedure				Mean	SD	Mean	SD	Mean	SD		
W	4R	Visual	564	160.66	62.52	9.07	1.35	1.41	1.58	15.55	Straight-In
W	4R	ILS	729	162.12	54.50	8.57	0.92	1.26	1.12	14.72	Straight-In
W	13C	Visual	861	181.19	75.03	9.63	1.11	1.10	1.14	11.42	One turn
W	13C	ILS	568	215.68	86.33	10.61	1.03	2.51	1.64	23.71	One turn
W	31C	Visual	987	249.90	113.66	11.89	1.97	3.45	2.02	29.00	One turn
W	22L	Visual	650	275.32	109.19	12.37	2.11	3.24	1.95	26.16	Two turns
W	31C	ILS	387	313.40	142.85	13.75	2.42	4.68	2.47	34.03	One turn

When tower managers select a runway configuration in no wind conditions, 13C is the most equitable runway with the 14% difference in fuel burn for VFR flights from the east and the west, followed by 4R (18%), 22L (96%) and 31C (148%). The same is true during IMC, 13C (56%) is the most equitable runway followed by 4R (72%) and 31C (211%).

Operational efficiency is maximized when the arrival flow crosses the final waypoint on the STAR and flies a straight course to the runway. The ILS and visual approaches from the east on to runway 31C and from the west on to 4R have a straight approach from the final way point on the STAR. The fuel burn performances of these flows are the best when compared to other flows from the same direction

The fuel burn performance of RNP approach flows for B73's aircraft type is compared to the performance of other flows in Table 22 and Table 23. The RNP approach on to runway 13C ranks second last among flows from the east and third from the top among flows from the west. It also burns on average 41-52 kg less fuel per flight per approach compared to the corresponding ILS approach and 0-114 kg more fuel per flight per approach compared to the corresponding visual approach. For RNP approach to runway 13C from the east, the vectors (for safe merging and spacing) in the base leg (before the start of the approach procedure) result in level off and higher fuel burn rate.

Table 22: Fuel burn, Track time and Level time Statistics for flows from the east, for B73's, ranked by fuel burn

Flow			Track Count	Fuel Burn (kg)		Track time (min)		Level Time (min)		% Level	# of Turns in the Terminal Airspace
Dir/Rwy/Procedure				Mean	SD	Mean	SD	Mean	SD		
E	31C	ILS	961	119.49	39.59	5.92	0.72	0.41	0.77	6.91	Straight-In
E	31C	Visual	215	121.03	39.68	6.14	0.91	0.43	0.93	7.03	Straight-In
E	22L	Visual	606	158.09	48.35	6.69	0.80	0.39	0.64	5.76	One turn
E	4R	Visual	842	216.36	95.36	8.72	1.81	2.11	1.72	24.16	One turn
E	13C	Visual	673	248.06	78.27	9.59	1.60	1.84	1.31	19.22	Two turns
E	4R	ILS	285	328.15	149.86	10.94	2.61	3.52	2.40	32.20	One turn
E	13C	RNP	87	362.26	91.46	13.32	2.08	4.69	1.96	35.21	Two turns
E	13C	ILS	520	414.67	121.84	14.30	2.14	6.38	2.49	44.62	Two Turns

Table 23: Fuel burn, Track time and Level time Statistics for flows from the west, for B73's, ranked by fuel burn

Flow			Track Count	Fuel Burn (kg)		Track time (min)		Level Time (min)		% Level	# of Turns in the Terminal Airspace
Dir/Rwy/Procedure				Mean	SD	Mean	SD	Mean	SD		
W	4R	ILS	548	178.17	42.04	8.51	0.88	1.23	1.06	14.49	Straight-In
W	4R	Visual	423	178.40	54.16	8.95	1.17	1.22	1.40	13.68	Straight-In
W	13C	RNP	144	206.31	46.70	9.44	0.83	1.25	1.06	13.28	One turn
W	13C	Visual	615	206.47	56.89	9.59	1.11	1.02	1.10	10.68	One turn
W	13C	ILS	416	247.56	64.54	10.57	0.96	2.44	1.61	23.11	Two turns
W	31C	Visual	734	286.98	95.86	11.82	1.91	3.35	1.93	28.36	One turn
W	22L	Visual	495	306.80	90.87	12.37	2.04	3.19	1.90	25.79	Two turns
W	31C	ILS	295	355.2	122.9	13.63	2.4	4.535	2.5	33.27	Two Turns

4.3.3.2.2 Fuel Burn Performance Statistics by Approach Type

The fuel burn performance for flows from the east and west are consolidated (assuming equal weights for traffic from the east and the west) to get fuel burn per flight for each runway and approach. These are ranked based on the average fuel burn per flight, as shown in Table 24 (fleet mix) and Table 25 (only B737s). Runway 22L does not have an ILS approach and only runway 13C has a RNP approach (see Table 25).

For the fleet mix at MDW the ranking of runways for ILS approach is 31C (207.06 kg) followed by 4R (+6.5%) and 13C (+33%). For visual approach the ranking is 4R (174.9 kg) followed by 31C (+.1%), 13C (+11%), and 22L (+19%).

The fuel burn for visual approaches on to runways 31C, 4R and 13C is less than the corresponding ILS approach by 15%, 20% and 30% respectively.

Table 24: Fuel burn, Track time, Level Time Statistics for all aircrafts at MDW, by approach type, ranked by fuel burn

Runway/ Approach		Track Count	Fuel Burn (kg/flight)		Track time (min)		Level Time (min)		% Level
			Mean	SD	Mean	SD	Mean	SD	
31C	ILS	1854	207.06	150.20	9.83	4.31	2.55	2.80	25.99
4R	ILS	1119	220.56	133.75	9.75	2.27	2.38	2.19	24.41
13C	ILS	1366	276.09	144.58	12.49	2.55	4.46	2.87	35.69
4R	Visual	1745	174.90	85.68	8.95	1.58	1.83	1.71	20.46
31C	Visual	1332	175.19	114.65	9.01	3.25	1.94	2.17	21.49
13C	Visual	1887	193.74	89.70	9.62	1.37	1.50	1.28	15.54
22L	Visual	1490	207.76	110.81	9.54	3.25	1.83	2.03	19.14

For B737s, the RNP approach on to 13C requires an average of 14% less fuel than the ILS approach on to 13C. At $3/gallon, this is equivalent to on average savings of $47 per flight per approach. The visual approach to 13C burns on average 20% less fuel than the corresponding RNP approach (see Table 25).

Table 25: Fuel burn, Track time, Level Time Statistics for B73's at MDW, by approach type, ranked by fuel burn

Runway/ Approach		Track Count	Fuel Burn (kg/flight)		Track time (min)		Level Time (min)		% Level
			Mean	SD	Mean	SD	Mean	SD	
31C	ILS	1256	237.36	149.08	9.78	4.24	2.47	2.76	25.29
4R	ILS	833	253.16	133.18	9.72	2.30	2.38	2.18	24.45
13C	ILS	936	331.12	128.40	12.44	2.50	4.41	2.88	35.48
13C	RNP	231	284.28	106.55	11.38	2.50	2.97	2.33	26.12
4R	Visual	1265	197.38	79.84	8.83	1.53	1.67	1.63	18.85
31C	Visual	949	204.00	110.76	8.98	3.21	1.89	2.10	21.07
13C	Visual	1288	227.27	71.51	9.59	1.38	1.43	1.28	14.95
22L	Visual	1101	232.4	104.1	9.529	3.2	1.787	2	18.76

4.3.3.2.3 Fuel Burn Performance Statistics by runway

The overall ranking for runways based on the fuel burn, irrespective of direction of flow or type of approach is shown Table 26. In general, the overall ranking of the runways in terms of fuel burn performance is 31C, 4R (+3%), 22L (+8.4%) and 13C (+29%).

Table 26: Fuel burn, Track time, Level Time Statistics for all aircrafts at MDW, by runway, ranked by fuel burn

Runway	Track Count	Fuel Burn (kg/flight)		Track time (min)		Level Time (min)		% Level
		Mean	SD	Mean	SD	Mean	SD	
31C	3186	191.12	134.56	9.42	3.84	2.24	2.52	23.84
4R	2864	197.73	114.62	9.35	1.99	2.11	1.99	22.52
22L	1490	207.76	110.81	9.54	3.25	1.83	2.03	19.14
13C	3487	250.94	123.27	11.16	2.51	2.98	2.56	26.69

4.3.3.2.4 Fuel burn Level vs. Non Level

For each trajectory the fuel burn model takes into account the energy state (i.e., kinetic – true air speed, and potential – altitude) of the aircraft at each position report. The model estimates the fuel burn rate in level segments and non-level segments while taking into consideration the configuration of the aircraft (clean, or dirty). This is illustrated in Figure 32 , where two trajectories for a B737 to runway 13C from the east are shown. The lateral and the vertical profile are shown in plots (a) and (b), and the true airspeed and fuel burn rate are shown in plots (c) and (d). The vertical profile, the true airspeed and the fuel burn rate are shown as a function of distance to the runway threshold.

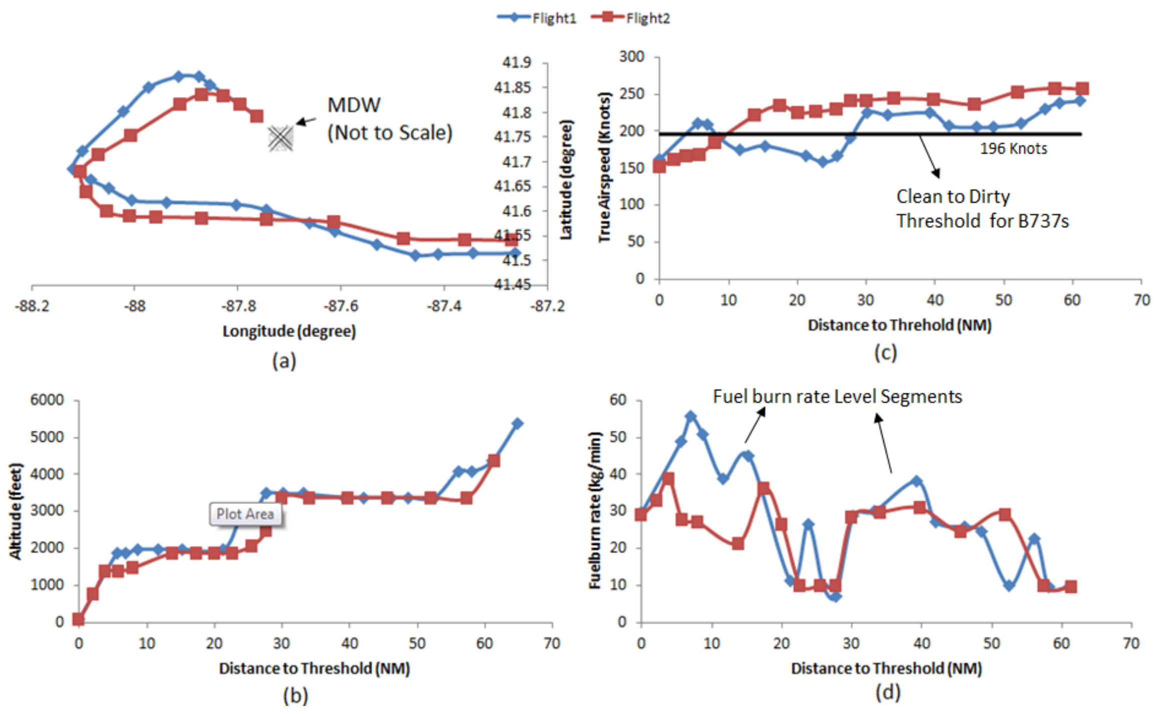


Figure 32: Flight Trajectories to runway 13C from the East, showing change in the true airspeed, altitude and fuel burn rate as function of distance from the runway threshold

The model captures the higher fuel burn rate in the level segments compared to the non-level segments (see Figure 32, plot b and d). The model also captures increases in fuel burn as a result of switching from clean to dirty configuration. For instance, flight1 (blue) burns more fuel as it switches from clean to dirty sooner than flight2 (red) as shown in Figure 32 (c).

The mean and standard deviation of fuel burnt on level, descent and final approach for narrow body aircrafts at MDW is shown in Figure 33. The fuel burn on the level segments is on an average 108% more than on the descent segments and 25% more than on the final approaches

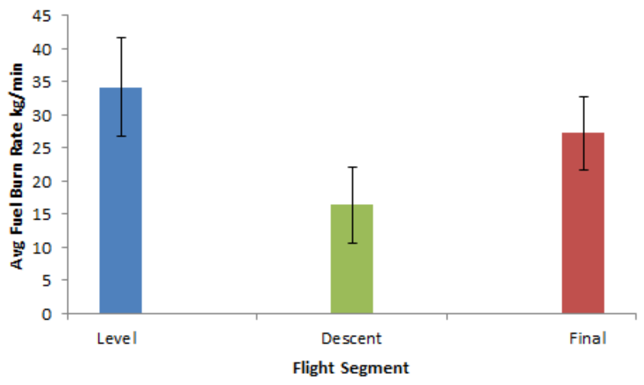


Figure 33: Average Fuel burn rate and Standard Deviation for Level, Descent and Final Approach Segments, for narrow body aircrafts

The fuel burn for individual TRACON flows is not proportional to the track time (see Figure 34 (b)). The main factors that contribute to higher fuel burn are the percentage level segments. The influence of level segments on total fuel burn is shown in Figure 34. In the terminal airspace (i.e. from final waypoint in STAR to the runway threshold) the time in descent and final approach and the corresponding fuel burn is more or less constant, as shown by the green and red line. Any increase in the total transit time in the terminal area results in an increase in the level segments, shown by the blue line. This is because aircraft have to maintain certain altitude at the start of the base leg and there is not much altitude to lose from the final waypoint of STAR to the start of the base leg. Therefore relative position of the final waypoint in STAR to the runway threshold, and the controller induced vectors for safe merging and spacing determine the duration of the terminal flow and the associated fuel burn. Therefore, longer duration flows have a higher percentage of level segments and a non-linear increase in the total fuel burn.

In case of MDW, flows to runway 13C from the east and flows to runway 22L and 31C from the west have longer level segments and a high level time to total time ratio. The straight-in flows to runway 31C from the east and runway 4R from the west have very short level segments and a low level segment to total time ratio.

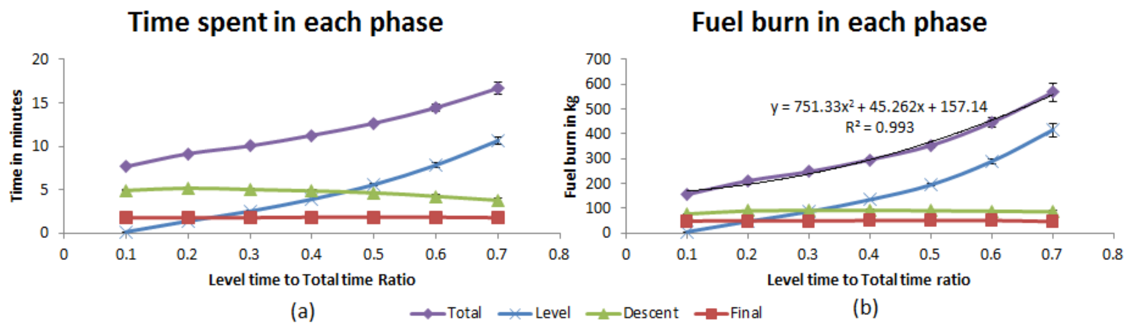


Figure 34: Influence of level segments on total fuel burn. Any increase in the total transit time in the terminal area results in an increase in the level segments as aircraft have to maintain certain altitude at the start of the base leg and there is not much altitude to lose from the final waypoint of STAR to the start of the base leg.

4.3.3.3 Summary of TRACON Arrival Flow Analysis

This section presents the estimates of track time, distance and fuel burn using the actual trajectories of the aircraft. The fuel burn model takes into consideration the energy state (kinetic – true air speed and potential - altitude) of the aircraft at each position report of the flight trajectory. The model captures the higher fuel burn rate in the level segments compared to the non-level segments. The model also captures increase in fuel burn as a result of a switch from clean to dirty configuration during the approach in the terminal airspace (i.e. from final waypoint on STAR to the runway threshold).

The mean track time, distance and fuel burn per flight are estimated for 16 major arrival flows at MDW using 43 days of NOP track data. The performance metrics are used to rank runways based on, the individual flows from east and west, the approach procedures irrespective of flow direction and the overall efficiency irrespective of approach type and flow direction.

The implications of the TRACON arrival flow analysis for MDW arrivals are as follows:

4.3.3.3.1 Performance of RNP approach

The RNP approach flows to 13C burns 14% less fuel than the corresponding ILS approach and 25% more fuel than the corresponding visual approach flows on an average. Therefore the benefits of RNP approach to 13C are limited to IMC days.

The variance in flight tracks in the terminal airspace, caused by controller vectoring for merging and spacing, is a function of the approach type along with the relative position and the distance of the final waypoint on the STAR to the runway threshold. Operational efficiency is maximized when the arrival flow crosses the final waypoint on the STAR and flies a straight course to the runway. TRACON flows that require turns to line-up for the final approach segment add vector complexity and result in increased track distance and track time, and more importantly increased variance in track distance and track time resulting in lost runway productivity. Without efficient merging and spacing the benefits of precise curved path RNP approach are nullified. The "vectors" between the final waypoint on the STAR and the start of the RNP approach introduce as much variation in flight tracks as the ILS flows.

4.3.3.3.2 Role of Level Segments in Approach

For narrow body aircrafts at MDW, the fuel burn rate in level flights (35 kg/min) is 108% greater than fuel burn for near-idle descent segments (15 kg/min) and 25% greater than fuel burn on the final approach segment (25 kg/min).

4.3.3.3.3 Potential for using Optimum Runway Configurations

During Visual Meteorological Conditions (VMC), assuming equal weights for arrivals from the east and west, the least average fuel burn arrival flow for VFR approach procedures is 4R (174.9 kg) followed by 31C (+.1%), 13C (+11%) and 22L (+19%). During Instrument Meteorological Conditions (IMC), assuming equal weights for arrivals from the east and west, the least average fuel burn arrival flow for ILS approach procedures is 31C (207.06 kg) followed by 4R (+6.5%) and 13C (+33%). This information combined with the wind information (magnitude and direction) can be used to select runway configurations that are optimal both in terms of wind and terminal area fuel burn.

The Optimal Runway Configuration model can be used to determine the optimal runway configuration for the tower control manager at the airports. In current practice the runway configuration is determined based only on the wind direction.

4.4 Benefits of RNP Approaches to all Runways at MDW

The analysis of benefits of RNP approaches to all runways at MDW is motivated by two factors: (1) the potential use of RNP approach to all runways and (2) the potential use of optimal runway configuration.

4.4.1 Engineering new RNP Approaches at MDW

As described in section 3.6, the new PBN approach with RF leg capability enable precise curved path approaches (e.g. RNP 0.3 w/ RF leg) in the terminal airspace. These new approach procedures are designed to eliminate trombone vectors in the base leg of the approach and make the approach from the final waypoint on STAR to runway threshold shorter on average compared to conventional approach procedures. The shorter approach result in fuel burn savings in the terminal airspace compared to the conventional approaches.

The use and the benefits of RNP approach with RF leg for an airline operating at MDW is just not limited to using RNP approach to 13C. This analysis uses the methodology describe in section 3.6.1 to reflect and rotate existing RNP tracks to runway 13C to construct RNP flows to other major runway (22L, 31C and 4R) at MDW.

Runway 22L is 90^o toward the north of runway 13C. Therefore, the RNP flow to runway 22L from the east is obtained by reflecting the west RNP flow to runway 13C about the runway's (13C) axis and rotating by 90^o. Similarly the RNP flow to runway 22L from the west is obtained by reflecting the RNP flow to runway 13C from the east about the runway's (13C) axis and rotating by 90^o. The new flows to 22L are shown in Figure 35 and Figure 36.

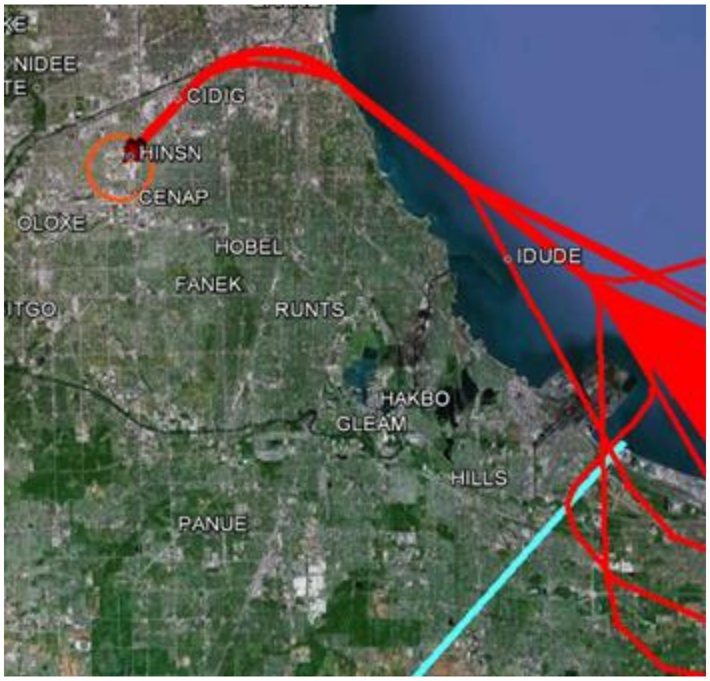


Figure 35: Hypothesized RNP approach flows to 22L from the east, obtained from reflecting and rotating RNP flows to 13C from the west by 90 degree

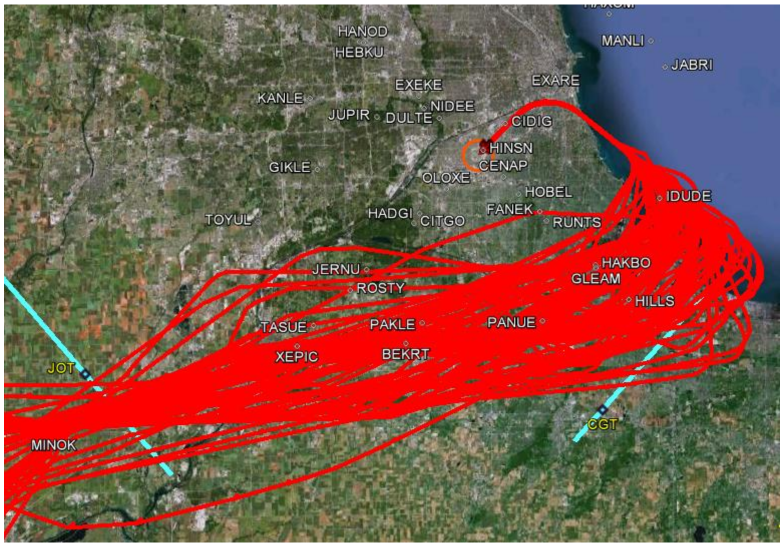


Figure 36: Hypothesized RNP approach flows to 22L from the west, obtained from reflecting and rotating RNP flows to 13C from the east by 90 degree

Runway 4R is 90^o toward the south of runway 13C. Therefore, the RNP flow to runway 4R from the east is obtained by rotating the west RNP flow to runway 13C by 90^o (see Figure 37). The west flows to runway 4R are aligned with the runway and do not need a RF leg. It is assumed that RNP flows from the west to runway 4R have the same performance as the ILS approach in terms of fuel burn.

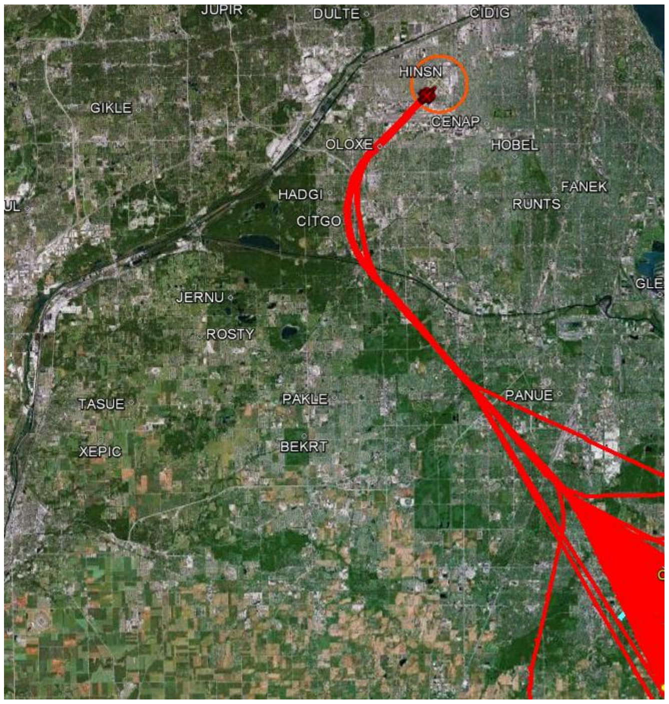


Figure 37: Hypothesized RNP approach flow to runway 4R from the east, obtained from reflecting and rotating RNP flows to 13C from the west by -90 degree

Runway 31C is the other side of runway 13C. Therefore, the RNP flow to runway 31C from the west is obtained by reflecting the west RNP flow to runway 13C about the runway’s (13C) axis and rotating by 180^o (see Figure 38).The east flows to runway 31C are aligned with the runway and do not need a RF leg. Therefore as in case of runway 31C, it is assumed that RNP flows from the east to runway 31C have the same performance as the ILS approach in terms of fuel burn.

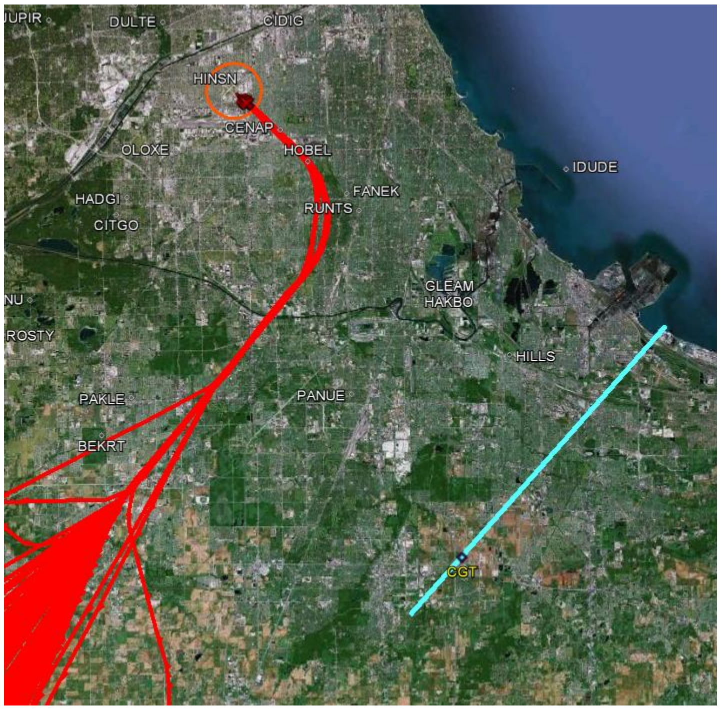


Figure 38: Hypothesized RNP flow to runway 31C from the west, obtained from reflecting and rotating RNP flows to 13C from the west by 180

The fuel burn performance of these new hypothesized flows are computed and included with performance statistics of the existing flows. The 4-D tracks of the existing RNP approach flows are for B73 aircraft type, as Southwest airline is the only commercial carrier that uses RNP approach at MDW. The average fuel burn per flight for each flow and its standard deviation for Boeing 73’s (B73’s) (which constitutes 72% of the fleet mix at MDW for all operations except general aviation) is shown in Figure 39.

The RNP flows at MDW only have B73's aircraft type. The estimates for the RNP flow's mean fuel burn per flight for the fleet mix at MDW (see Figure 31) are obtained by multiplying a factor of 0.84 to the fuel burn estimates of B73's. This factor is determined by analyzing ILS and visual approach flow at MDW that have the complete fleet mix at MDW (see section 4.3.3.2.1).

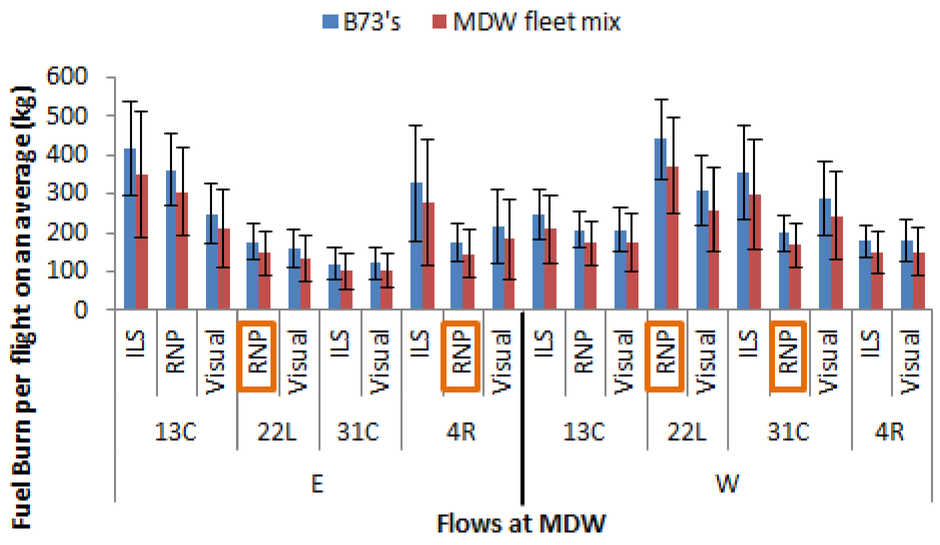


Figure 39: Fuel burn performance (mean and standard deviation) of existing and hypothesized flows at MDW, for B73's and the fleet mix at MDW

The hypothesized flows are marked in Figure 39. The RNP approaches to 22L from the east and the west have a higher fuel burn per flight than corresponding visual approaches on average by +10.5% for the east and + 43.5% for the west. Therefore, the hypothesized RNP approaches to runway 22L are not desirable over the conventional visual approaches.

The fuel burn performances of the hypothesized RNP approach to runway 4R from the east and the hypothesized RNP approach to runway 31C from the west are lower than the corresponding ILS and visual approach on average by -47% and -20% for 4R and -44% and -30% for 31C. These RNP approaches eliminate the trombone vectors on the base leg that are present in the corresponding ILS and visual approaches to runways 4R and 31C. Therefore, these new approaches would be preferred both in VMC and IMC.

4.4.2 Optimal Runway Configuration for MDW Arrivals

One of the implications of TRACON arrival flow analysis (see section 4.3.3.3.3) is that fuel burn performance for all terminal area arrival flows are not the same and depend on the relative position of the runway with respect of the approach direction and the type of approach. The current process of selecting runway configuration is based on wind direction and magnitude and other operational constrains like the metroplex flow conflict. For instance, at Chicago metroplex MDW avoids using ILS approach to 13C due the flow conflict with 22L departures from ORD. The de-confliction of flows at Chicago metroplex using RNP approach and the added fuel burn savings associated with the new approach provides an opportunity to assess the benefits of using optimal runway configurations that take in consideration the fuel burn performance of flows in addition to the wind magnitude and direction.

This section determines the optimal runway configuration for arrivals at MDW for given set of flows using the methodology described in section 3.6.2. For each fifteen minute periods, the set of feasible runways for the given wind (magnitude and direction) and meteorological conditions (IMC or VMC) is ranked based on the intensity of traffic volume from the east and west direction and the fuel burn performance of individual flows at MDW. The runway with lowest average fuel burn per flight is select as the optimal runway for the arrivals at MDW.

The terminal air traffic flows at MDW originate from the east or the west. The traffic volume from the east and the west vary throughout the day. The average fraction of traffic volume from the east for each hour of the day is shown in Figure 40. The fraction of traffic volume from the west is one minus the fraction of traffic volume from the east. The red line is the 50% mark when there are equal volumes of traffic from the east and the west. At MDW the traffic is predominantly from the east except for 1PM, 4PM, 7PM and 10PM when traffic volumes are higher from the west. From 6AM to 10 PM the fraction of traffic volume from the east at MDW ranges from 0.46 to 0.69.

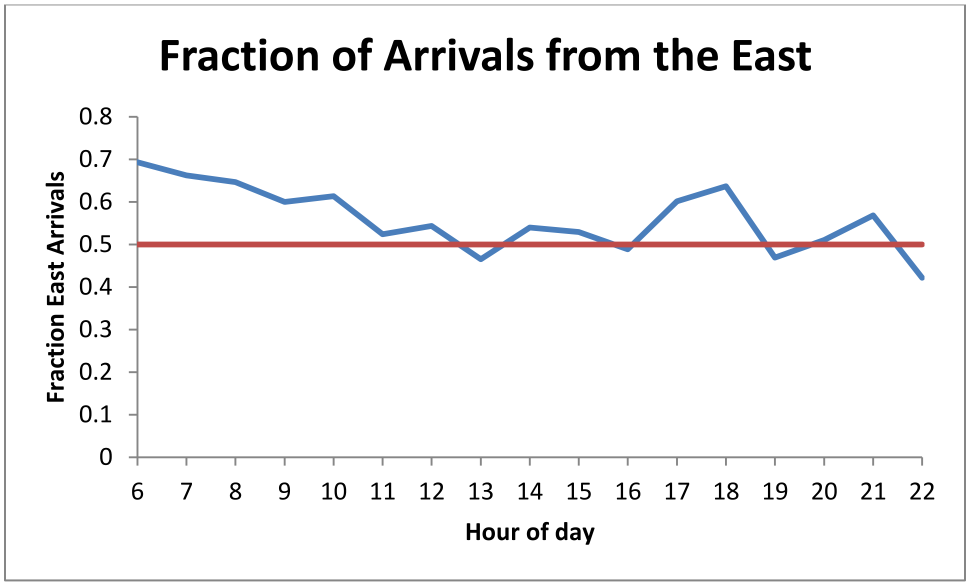


Figure 40: East – West Traffic flow volume ratio at MDW by hour of the day

In this analysis the fraction of traffic volume from the east and the west are used as weights to combine the mean fuel burn performance of flows from the east and west for a given runway and approach type. The weighted fuel burn averages are then used to rank and select the runway with the best (lowest) weighted average fuel burn for a given approach type. The runway thus selected will minimize the total fuel burn in the terminal airspace.

The weighted average fuel burn per flight for various ratios for traffic volume from the east and the west, for visual, ILS and RNP flows at MDW are shown in Figure 41, Figure 42, and Figure 43. For a given runway and approach type, the “zero” on the x-axis corresponds to the mean fuel burn of the flow from the west and the “one” corresponds to mean fuel burn of the flow from the east. Each line is the mean fuel burn per flight for various levels of traffic volume from the east and the west. The range (0.46 to 0.69) of traffic volume ratio applicable at MDW is highlighted as well.

For all approach types (visual, ILS and RNP) runways 31C or 4R have the best fuel burn performance. For visual approaches, performance of runway 4R is better than 31C as long as traffic from the east is 54% or less of the total traffic volume. When the traffic from the east is greater than 54% of the total traffic runway 31C has better fuel burn performance. A similar tradeoff occurs between runways 13C and 22L, when the traffic from the east is 55% or less of the total traffic, runway 13C is more favorable than runway 22L in terms of fuel burn per flight in the terminal airspace.

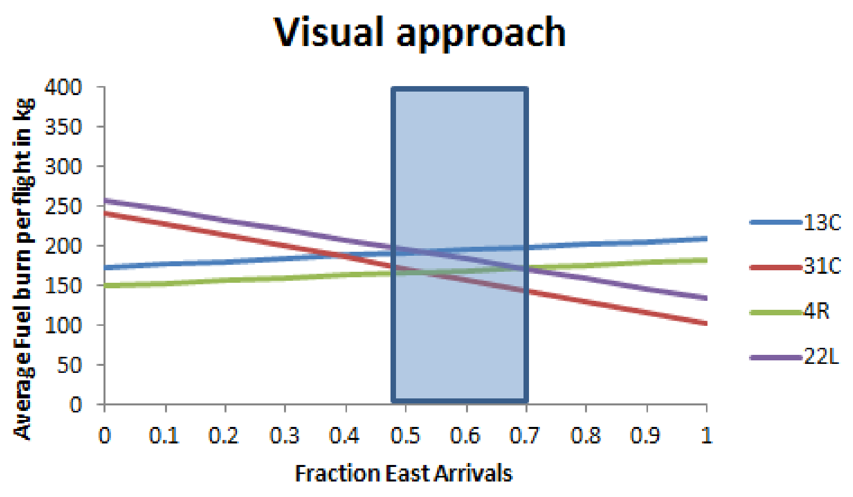


Figure 41: Runway ranking based on traffic volume ratio for visual approaches

At MDW only runways 13C, 31C and 4R have published ILS approach procedures. The ILS approach to runway 31C has the lowest weighted average fuel burn per flights followed by ILS approach to runways 4R and 13C (see Figure 42).

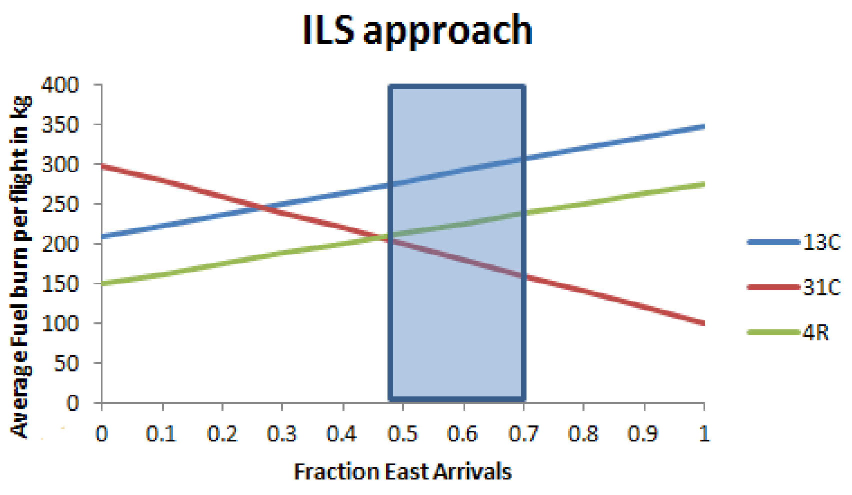


Figure 42: Runway ranking based on traffic volume ratio for ILS approaches

For RNP approaches runway 31C is the best with the lowest weighted average fuel burn per flight followed by runways 4R, and 13C or 22L. The performance of runway 13C is better than 22L until the traffic volume from the east is 57% or less of the total traffic volume, above which runway 22L has better fuel burn performance (see Figure 43).

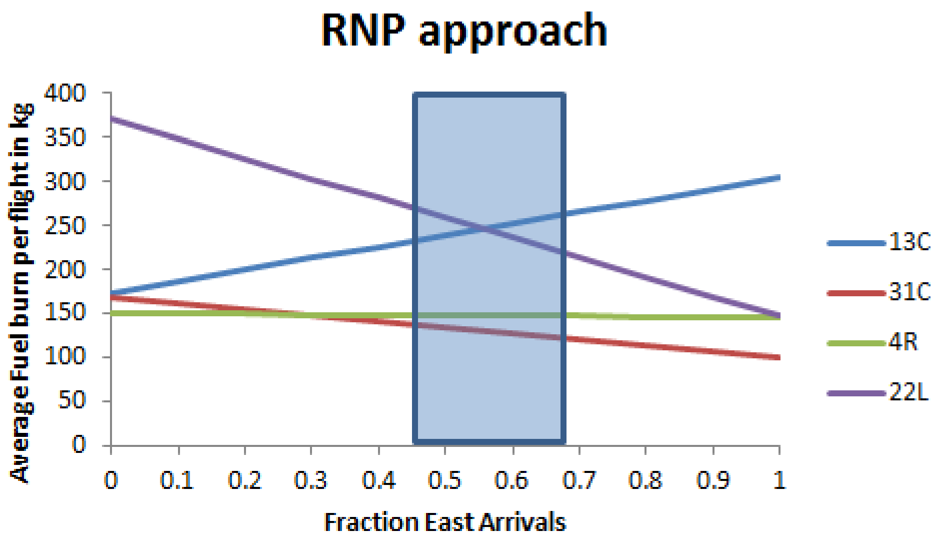


Figure 43: Runway ranking based on traffic volume ratio for RNP approaches

The fuel burn performances for all the runways (by approach type) at MDW as a function of traffic level from the east and the west are shown in Figure 44. The weighted average fuel burn for RNP approaches are shown in solid line, visual approaches are shown in dashes lines and ILS approaches are shown in dash-dotted line. The legend shows the ranking of the runway by available approach type for the runway. For example, runway 31C at MDW has the best fuel burn performance for RNP approaches followed by visual approaches and ILS approaches. At MDW, irrespective of the traffic volume from the east and the west, the RNP approach to runway 31C has the lowest (best) fuel burn performance followed by the RNP approach to runway 4R. The ILS approach to runway 13C has the highest fuel burn performance. All other approaches show trade-offs based on the level of traffic from the east and west.

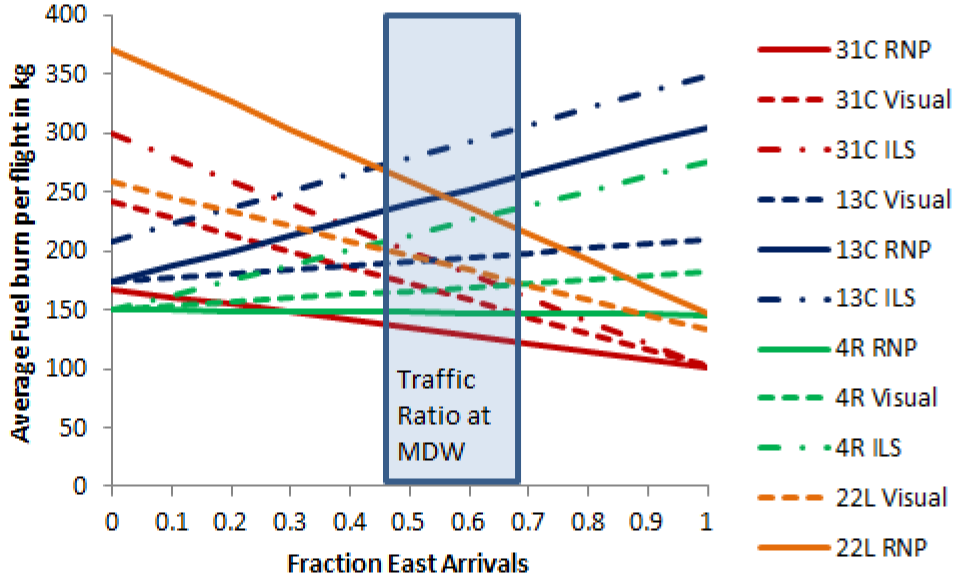


Figure 44: Runway ranking based on traffic volume ratio for all flows at MDW

Using the methodology described in section 3.6.2, the optimal runway for MDW arrivals is determined based on the existing flows and the hypothesized flows. The existing flows consists of the 16 major flows identified using the TRACON flow analysis. The additional hypothesized flows are RNP approaches to runway 22L, 31C and 4R.

The runways used at MDW for arrivals, obtained from analysis of ASPM airport data for year 2007 to 2012, are shown in Figure 45. This is compared to the optimal usage as estimated by the optimal runway configuration model. The optimal usage is estimated for two sets of flows: “Optimal 1” which is based on existing 16 major flows at MDW, namely the ILS, visual and RNP approaches to 13C, the ILS and visual approaches to 31C and 4R, and visual approaches to 22L from the east and the west and “Optimal 2” which includes hypothesized RNP flows to runways 22L, 31C and 4R in addition to the 16 major flows.

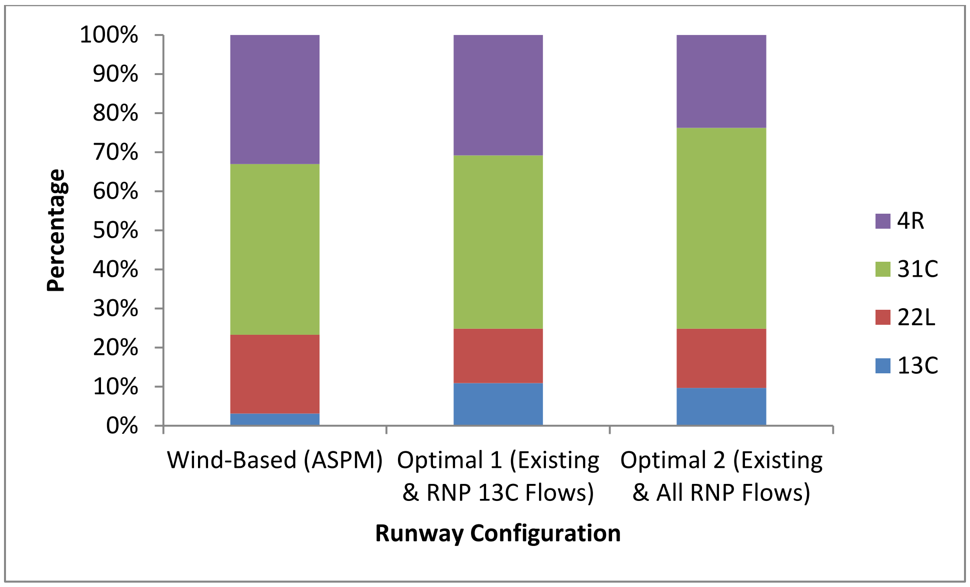


Figure 45: Runway usage at MDW, ASPM vs. optimal 1 and optimal 2.

The analysis of ASPM data for year 2007 to2012 shows that runways 31C, 4R, 22L and 13C are used for arrivals on average 44%, 33%, 20% and 3% of the time respectively.

The optimal runway usage (optimal 1) based on the 16 major flows and historic wind information indicate no major change in use of runways 31C and 4R for arrivals. However, the optimal runway configuration model shows that 13C is more favorable than 22L in presence of south east winds and heavier traffic from the west (more than 43% of the total traffic).

The optimal runway usage (optimal 2) based on the 16 major flows, 4 new RNP flows and historic wind information suggest an increase in usage of runway 22L(15%) over 13C (9%) and 31C (51%) over 4R(24%) compared to "optimal 1". This is because the RNP approach to 22L has better fuel burn performance than RNP approach to 13C when traffic volumes from the east are higher than 57% of the total traffic. Also, the RNP approach to 31C is always (irrespective of the traffic volumes from the east and the west) more efficient than any of the flows to 4R; therefore, for prevailing winds from the north 31C is always more favorable than 4R.

4.4.3 Annualized Benefits of RNP Approach at MDW

Using the methodology described in section 3.6.3, the annualized fuel burn benefits of RNP approaches at MDW are estimated as the difference between the total annual terminal area fuel burn for the baseline case and the four alternatives. For each case the total fuel burn for terminal area flows is computed for the total number of arrivals at MDW per year on average.

The total arrivals at MDW from 6AM to 10PM for years 2007 to 2012 are shown in Figure 46. On average there are 96,300 total arrivals and 69,430 (72%) Southwest arrivals (SWA) per year at MDW.

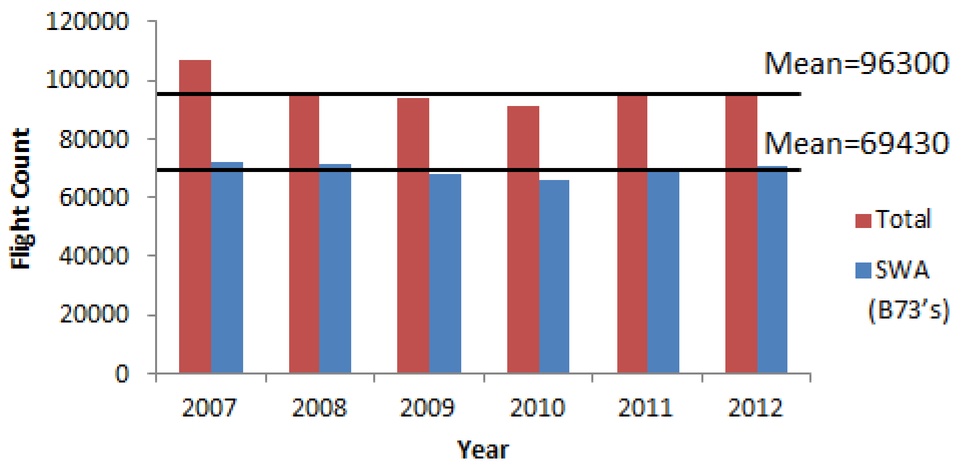


Figure 46: Total number of arrivals at MDW for years 2007 to 2012

The total fuel burn per year (for IMC, VMC and total) for the baseline case and the four alternatives are shown in Table 27. For each case the table also shows the flows and runway configuration considered for the fuel burn calculations in the first two columns. The potential savings in fuel burn per year and the associated costs from using the alternative flows and runway configurations over the baseline case are shown in the last column.

Table 27: Total fuel burn per year on an average for the baseline and the four alternatives and their associated benefits for all arrivals at MDW.

Flows	Runway Config	Fuelburn/year in gallons						Savings	
		VMC		IMC		Total			
		Mean	SD	Mean	SD	Mean	SD	Gallons	$3/gallon
13C (Visual, ILS) 31C(Visual, ILS) 4R(Visual, ILS) 22L(Visual)	ASPM- **Baseline**	4.8E+06	2.9E+05	7.9E+05	1.1E+05	5.0E+06	2.9E+05	NA	NA
13C (Visual, ILS, **RNP**) 31C(Visual, ILS) 4R(Visual, ILS) 22L(Visual)	ASPM - **A1**	4.8E+06	2.9E+05	7.7E+05	1.1E+05	5.0E+06	2.9E+05	5.8E+03	$17,520
	Optimal1 - **A2**	4.7E+06	2.9E+05	7.6E+05	1.1E+05	5.0E+06	2.9E+05	4.5E+04	$135,810
13C (Visual, ILS,**RNP**) 31C(Visual, ILS,**RNP**) 4R(Visual, ILS,**RNP**) 22L(Visual,**RNP**)	ASPM - **A3**	4.2E+06	2.6E+05	5.6E+05	7.7E+04	4.4E+06	2.6E+05	6.6E+05	$1,971,510
	Optimal2 - **A4**	4.1E+06	2.6E+05	5.7E+05	7.9E+04	4.1E+06	2.6E+05	8.9E+05	$2,665,410

The baseline case estimates the total terminal area fuel burn at MDW per year on average for the ASPM runway configuration and the associated conventional flows (i.e. visual approach flows during VMC and ILS approach flows during VMC).

The alternate case A1 estimates the benefits of using RNP approach to runway13C instead of the ILS approach. The total terminal area fuel burn at MDW per year on average is estimated for the ASPM runway configuration and the associated convectional and 13C RNP approach flows. The use of RNP approach to 13C instead of ILS approach during VMC yields a fuel burn saving of 5800 gallons of fuel per year. At $3/gallon this amounts to $17K savings per year on average for all MDW arrivals.

The alternate case A2 estimates the benefits of using the optimal runway configuration (Optimal 1 in Figure 45) instead of the runway usage as per the ASPM data and the associated flows. The alternate case A2 yields fuel burn savings of 45,000 gallons per year on average. The fuel burn savings are from the use of runway 13C which is more favorable than 22L in presence of south east winds and heavier traffic from the west (more than 43% of the total traffic). At \$3/gallon this amounts to \$135K savings per year on an average for all MDW arrivals

The alternate case A3 estimates the benefits of using ASPM runway configuration and the introduction of other RNP approaches to runways 31C, 4R and 22L. The alternate A3 yields a fuel burn savings of 660,000 gallons per year. At \$3/gallon this amounts to \$1.97M savings per year on an average for all MDW arrivals.

The alternate case A4 estimates the benefits of using the optimal runway configuration (Optimal 2 in Figure 45) instead of the runway usage as per the ASPM data and the associated flows. The alternate A4 yields a fuel burn savings of 890,000 gallons per year on an average. At \$3/gallon this amounts to \$2.67M savings per year on an average for all MDW arrivals.

The benefits of using the four alternatives over the baseline are also estimated for Southwest arrivals at MDW (see Table 28). The benefits to Southwest from using RNP approach to runway 13C under alternatives A1 and A2 are \$15K and \$115K per year on an average respectively. The benefits to Southwest from using RNP approach to all major runways (13C, 31C, 4R and 22L) under alternatives A3 and A4 are \$1.7 M and \$2.3M respectively.

Table 28: Total annual fuel burn for baseline and four alternative and their associated benefits for Southwest arrivals at MDW

Flows	Runway Config	Fuelburn/year in gallons						Savings	
		VMC		IMC		Total			
		Mean	SD	Mean	SD	Mean	SD	Gallons	$3/gallon
13C (Visual, ILS) 31C(Visual, ILS) 4R(Visual, ILS) 22L(Visual)	ASPM- **Baseline**	4.1E+06	1.7E+05	6.7E+05	9.2E+04	4.3E+06	1.6E+05	NA	NA
13C (Visual, ILS, **RNP**) 31C(Visual, ILS) 4R(Visual, ILS) 22L(Visual)	ASPM - **A1**	4.1E+06	1.7E+05	6.6E+05	9.0E+04	4.3E+06	1.6E+05	5.0E+03	$14,910
	Optimal1 - **A2**	4.1E+06	1.7E+05	6.5E+05	8.9E+04	4.3E+06	1.6E+05	3.8E+04	$114,690
13C (Visual, ILS,**RNP**) 31C(Visual, ILS,**RNP**) 4R(Visual, ILS,**RNP**) 22L(Visual,**RNP**)	ASPM - **A3**	3.6E+06	1.6E+05	4.8E+05	6.5E+04	3.7E+06	1.6E+05	5.6E+05	$1,692,510
	Optimal2 - **A4**	3.5E+06	1.6E+05	4.9E+05	6.6E+04	3.5E+06	1.6E+05	7.6E+05	$2,280,930

The noticeable increase in fuel burn savings for case A3 is from the use of RNP approaches to runways 31C and 4R which have better fuel burn performance than the corresponding ILS and visual approaches. For instance, assuming equal weights for traffic from the east and the west at MDW, the fuel burn savings in the terminal airspace from using RNP approach to 31C instead of the corresponding ILS and visual approach are 65.64 kg and 37.55 kg per flight on an average respectively. Similarly, the fuel burn saving in the terminal airspace from using RNP approach to 4R instead of the corresponding ILS and visual approach are 51.95 kg and 18 kg per flight on average respectively.

The use of optimal runway configuration “optimal 2” in case A4 further increases the fuel burn savings by 26%. This increase in fuel burn savings is from the increase use of runway 22L (15%) over 13C (9%) and 31C (51%) over 4R (24%), compared to optimal runway configuration “optimal 1”. During IMC the RNP approach to runway

22L has better fuel burn performance than RNP approach to runway 13C, when traffic volumes from the east are higher than 57% of the total traffic volume, As described before in section 4.4.2. Also, the RNP approach to 31C is always (irrespective of the traffic volumes from the east and the west) more efficient than any of the flows to 4R, therefore for prevailing winds from the north at MDW 31C is always more favorable than 4R.

The overall analysis shows that the new RNP approach procedures to all the major runways at MDW (13C, 31C, 4R and 22L) and their associated fuel burn savings over the convectional visual and ILS approaches enable the use of optimal runway configuration (see Figure 45, “optimal 2”). This results in a savings of 890K gallons per year on an average for all arrivals at MDW and a savings of 760K gallons per year on an average for Southwest arrivals at MDW. The fuel burn savings are from using the RNP approaches at MDW 77% of the time with fuel burn saving in the terminal airspace of 33kg per flight on an average. At $3/gallon the savings amount to $2.67M per year on an average for all arrivals and $2.3M per year on an average for Southwest arrivals at MDW.

4.4.4 Summary of Benefits of PBN Approach Procedures at MDW

The asymmetry in benefits between competing operators (airlines at MDW and ORD) in neighboring airports is large (1:33 in favor of ORD departures). In this way, there is no competitive advantage for MDW airlines in equipping due to the free rider issue. Using the methodology described in section 3.6.3, this section estimates the annualized benefits of using RNP approaches to all the major runways at MDW (13C, 4R, 22L, 31C) for the historic runway configuration (ASPM) and the optimal runway configurations (Optimal1 and Optimal 2).

In addition to reducing the additional cost due to metroplex flow conflict, RNP approach can be used to save fuel by having shorter and more efficient procedures compared to conventional approach procedures in the terminal airspace

The methodology enabled the evaluation of the introduction of additional RNP approach procedures to major runways (13C, 31C, 4R and 22L) at MDW. This has the potential of saving on an average 660K gallon per year of fuel for arrivals at MDW. At $3/gallon this amounts to a savings of $1.97M per year.

The de-confliction of metroplex airspace makes operations at neighboring airspace independent of each other allowing scope for further improvement in terminal fuel burn efficiency by choosing optimal runway configurations.

The methodology also identifies opportunities to select optimal runway configuration at MDW based on wind magnitude/direction and fuel burn efficiency of terminal area flows. The use of optimal runway configuration for wind and fuel burn, along with additional RNP approach procedure, has the potential of saving on average 890K gallons per year of fuel for arrivals at MDW. At $3/gallon this amounts to a savings of $2.67M per year.

The analysis also estimates the benefits of using RNP approach to the majority carrier at MDW (Southwest Airlines). The use of optimal runway configuration for wind and fuel burn, along with additional RNP approach procedures to runways 13C, 31C, 4R and 22L, has the potential of saving on an average 760K gallons per year of fuel for Southwest airlines at MDW. At $3/gallon this amounts to a savings of $2.3M per year.

4.5 Net Present Value Analysis of RNP Equipage for Southwest Airlines

This section estimates the return on investment (ROI) for Southwest Airlines using the NPV model described in section 3.7.

The Southwest Airline is investing $175M in equipping with RNP approach capability (Martin, 2009). The two financial parameters of interest to an airline are the Net Present Value (NPV) of the investment and the time it takes to obtain a positive ROI or Break Even Time (BET). The NPV analysis done to estimate these two parameters has the following assumptions:

a. Benefits are accrued only after 100% of fleet is equipped
b. Cash outflow = Initial Cost of Equipage, when time t=0
c. Cash outflow =0 for t>0
d. Cash Inflow = Benefits from new equipage per year
e. The NPV is computed for N=20 years

The RNP 0.3 approach with RF leg enabled by the precise curved path approach capability of the aircraft cuts the corner on the final approach making the final approach shorter. The shorter approach results in fuel burn savings in the terminal airspace compared to the conventional approaches. Analysis in section 4.4.3 shows that at MDW the use of RNP can result in a total potential savings of 760K gallons of fuel per year on average for Southwest Airlines. At $3/gallon this amounts to a savings of $2.3M per year.

The cash inflow of $2.3M per year does not yield a positive ROI at MDW. The NPV analysis of, the number of MDW like airports required for Southwest to break even at 5% and 10% discount rates is shown in Figure 47. The NPV curves are for savings at $3/ gallon. The analysis shows that Southwest will need at least 10 more airports like MDW to break even in less than 10 years at a discount rate of 5%. With 15 more airports like MDW the breakeven is achieved in 5 years with the 20 year NPV of $276M.

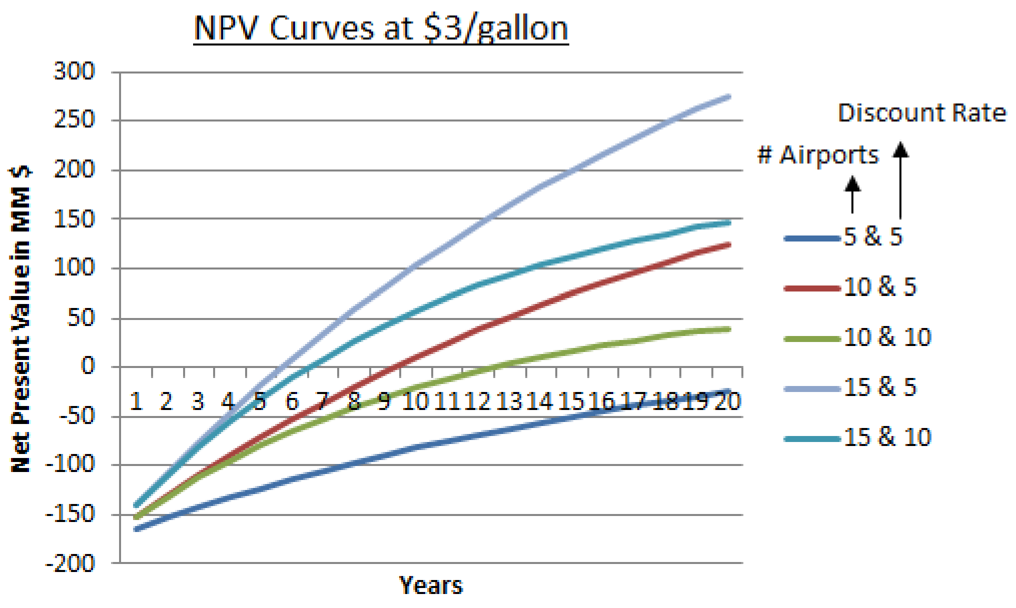


Figure 47: Analysis showing Break Even Time and Net Present Value for 20 years horizon for various combinations of number of MDW likes airports and discount rates.

The sensitivity of the Break Even Time and the 20 year NPV to fuel price for 5% and 10% discount rate is shown in Table 29 and Table 30. The analysis shows an increase in fuel prices will yield more acceptable ROIs and NPVs. This is assuming the increase in operations cost due to increase in fuel price is passed to the passengers and does not result in reduction in demand and daily operations.

Table 29: Break Even Time and 20 year NPV sensitivity to fuel price, at 5% discount rate

Fuel Price / # Airports	$3	$4	$5
5	>20 & $-25M	17 & $26M	12 & $75M
10	10 & $126M	7 & $226M	6 & $326M
15	6 & $276M	5 & $426M	4 & $577M

Table 30: Break Even Time and 20 year NPV sensitivity to fuel price, at 10% discount rate

Fuel Price / # Airports	$3	$4	$5
5	>20 & $-67M	>20 & $-31M	19 & $4M
10	13 & $40M	8 & $112M	6 & $183M
15	7 & $148M	5 & $255M	4 & $363M

Airports where Southwest Airlines have more than 10,000 arrivals per year on an average are shown in Figure 48.

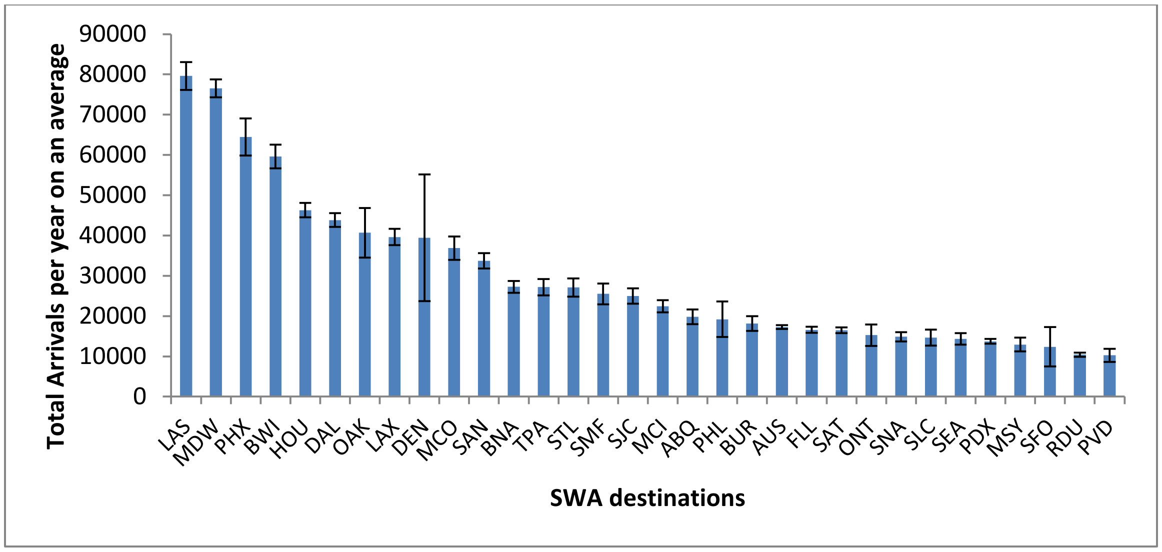


Figure 48: Southwest airports with 10,000 or more arrival per year on an average

MDW is the second largest operations center for Southwest Airline after Las Vegas International Airport (LAS). Based on the current level of operation Southwest Airlines does not have 10 more airports like MDW. However, Southwest can still break even based on the total number of RNP approach operation it carries out throughout its network. Analysis shows that MDW has a total of 1.1M arrival operations per year on average throughout its network. The carrier will break even in 10 years at a discount rate of 5% if it can perform half a million RNP approaches per year throughout its network, while saving at least 33 kg of fuel per approach.

4.5.1 Summary

The Southwest Airline is investing $175M in equipping with RNP approach capability (Martin, 2009).

Analysis in section 4.4.3 shows that at MDW the use of RNP can result in a total potential savings of 760K gallons of fuel per year on average for Southwest Airlines. At

$3/gallon this amounts to a savings of $2.3M per year. These savings are estimated using the optimal runway configuration model, which models the benefits of using of RNP approach to all major runways (13C, 31C, 22L and 4R) at MDW. Based on the fuel burn performance of existing and hypothesized RNP approaches, RNP can be used at MDW 77% of the time with fuel burn savings of 33kg per flight per approach over the alternate approach procedure (ILS or visual).

Investing in RNP approach for single airport use (for de-confliction of airspace and improving terminal approach efficiency) does not yield a positive ROI at MDW.

The results from the analysis were used to estimate the ROI for investing in RNP for the major carrier (Southwest Airlines) at MDW. The results show that the RNP approach does not yield a positive ROI at MDW, and that the carrier will have to perform at least half a million RNP approaches per year throughout its network, saving at least 33 kg of fuel per approach on an average to break-even in 10 years at a discount rate of 5%.

5 CHAPTER 5: CONCLUSIONS AND FUTURE WORK

A metroplex is a collection of airports serving a metropolitan area. A key determinant of the capacity of the metroplex airspace is the extent of interaction between arrival and departure flows at the airports. In some metroplexes the geometry of the airports is such that, under certain wind and weather conditions, there exist conflicts between the flows that require excessive ground holding for departures and airborne holding for arrivals. Advances in aircraft navigation technologies have created opportunities to improve arrival flow efficiencies and de-conflict metroplex flows. However, adoption of these technologies has been slow and haphazard due to issues with estimating the true Return-on-Investment (ROI), the need for collaboration and simultaneous equipage across competing stakeholders, and the allocation of benefits to parties that choose not to equip but gain benefits when their competition equips. Together these issues have created a "modernization stalemate."

The recent availability of high fidelity surveillance track data coupled with aerodynamic fuel burn models and airport wind and weather data create an opportunity to provide detailed analysis of metroplex traffic flows to include the real-world complexities of traffic flows and aircraft trajectories.

This dissertation describes a holistic methodology to use *high fidelity surveillance track data* coupled with *aerodynamic models* and *weather data* to quantify the efficiencies and costs of metroplex terminal area air traffic flows.

This methodology is intended for assessing benefits of proposed terminal airspace concepts-of-operations (e.g. metroplex de-confliction using RNP approach) and associated technologies that require simultaneous equipage and development of collaborative procedures by multiple stakeholders (airlines and ANSPs).

This chapter is organized as follows: section 5.1 describes the conclusions related to the methodology developed in this dissertation; section 5.2 describes the conclusions related to the results of the case study; section 5.3 describes the issue with achieving equipage and strategies for equipage; and section 5.4 describes the potential application of this methodology and future work.

5.1 Hybrid model for benefits analysis of PBN and metroplex de-confliction

This dissertation is motivated by the issues with the implementation of PBN approach procedures for de-confliction of metroplex airspace. The NAS modernization stalemate is due to: (a) lack of airlines confidence in the benefits of new technology and (b) free rider issue and asymmetry in distribution of benefits of the new technology. Both factors affect the implementation of PBN approach procedures for de-conflicting metroplex airspace. A review of the literature related to metroplex analysis identified the following methodological gaps:

1. The costs of metroplex flow conflict and the potential benefits associated with the de-confliction have been analyzed from a system-wide

perspective and not from an airlines perspective (Atkins, 2008; Clarke et al., 2011; Devlin et al., 2012). Any benefits analysis of operational changes that require airlines to equip should be done from an airlines perspective.

2. The benefits analysis of new PBN approach procedures has been estimated using simulated de-coupled routes (Clarke et al., 2011) or analysis of operational data (Devlin et al., 2012). The key benefits of these new approach procedures to individual airlines are fuel burn savings from shorter and more efficient flight paths in the terminal airspace. In the case of metroplex flow conflict, the airlines also benefit from elimination of holding patterns for de-confliction of terminal area flows. Therefore, fuel burn analysis using actual track data is required to account for the energy state of the aircraft and the vertical profile of the trajectory in the terminal airspace. Also a methodology to estimate for the cost of holding patterns using actual track data is required.
3. Existing analyses using surveillance track data (Dorfman et al., 2012; Enriquez, 2013; Levy, 2003; Vempati & Ramadani, 2012) have not been extended to estimating the benefits and the ROI of PBN approach procedures to individual airlines based on savings from shorter terminal area flight path and elimination of holding patterns.

This dissertation addresses these gaps by developing a holistic methodology to use *high fidelity surveillance track data* coupled with *aerodynamic models* and *weather data* to estimate the benefits of PBN approaches and in doing so:

1. Estimate the track distance/time, fuel burn performance of the new PBN approach procedures and compare it to conventional approach procedures.
2. Estimate the Return on Investment of the new PBN approach procedures to individual airlines.
3. Estimate the benefits of metroplex de-confliction to capture magnitude of the asymmetry and the potential for simultaneous adoption of the technology by the competing stakeholders.

5.2 Implications of the Chicago Metroplex Case Study

The methodology is demonstrated in a case-study of the benefits of the introduction of a Required Navigation Performance (RNP) approach procedure for air traffic arrival flow de-confliction at the Chicago Terminal Radar Approach Control (TRACON).

The high equipage, training and certification costs of RNP capability coupled with lack of confidence in the anticipated benefits are one reason airlines may be reluctant to invest in RNP approach capability. The results of this dissertation indicate that investing in RNP approach for single airport use (for de-confliction of airspace and improving terminal approach efficiency) does not yield a positive ROI in 20 years at MDW.

The free rider issue and the asymmetry in distribution of benefits associated with the de-confliction of airspace further prevent the airlines from making the investment.

Therefore the RNP approach capability on its own is not an attractive technology to adopt for operation in a network of airports with prevalent visual conditions. These are described in more details in the following subsections.

5.2.1 Performance of RNP approach

An arrival flow analysis at MDW was performed to compare the performance of the new RNP approach (13C) to the conventional approach procedure (ILS 13C) to assess the benefits of the new procedure to the metroplex and to individual airlines.

The RNP approach flows to 13C burns 14% less fuel than the corresponding ILS approach and 25% more fuel than the corresponding visual approach flows on average. Thus, the benefits of RNP approach to 13C are limited to IMC days.

The variance in flight tracks in the terminal airspace, due to controller vectoring for merging and spacing, is a function of the approach type along with the relative position and the distance of the final waypoint on the STAR to the runway threshold. Operational efficiency is maximized when the arrival flow crosses the final waypoint on the STAR and flies a straight course to the runway. TRACON flows that require turns to line-up for the final approach segment add vector complexity and result in increased track distance and track time, and more importantly increased variance in track distance and track time resulting in lost runway productivity.

Without efficient merging and spacing, the benefits of precise curved path RNP approach are nullified as the "vectors" between the final waypoint on the STAR and the start of the RNP approach introduce as much variation in flight tracks as the ILS flows.

5.2.2 Asymmetry in Benefits of De-confliction

The flow conflict between 13C ILS arrivals at MDW and 22L departures at ORD at Chicago metroplex occur on an average 1.6% of the time per year. The introduction of the RNP approach to 13C at MDW to de-conflict the traffic flow for these periods is estimated to reduce the direct airline operating cost per year on an average by $.04M at MDW and $1.33M at ORD.

The magnitude of the benefits of de-confliction for the operator expected (airlines at MDW) to equip is small and the asymmetry in benefits between competing operators (airlines at MDW and ORD) in neighboring airports is large (1:33 in favor of ORD departures). This results is no competitive advantage in equipping due to the free rider issue.

5.2.3 Potential for additional RNP approaches

In addition to reducing the additional cost due to metroplex flow conflict, RNP approach can be used to save fuel by having shorter and more efficient procedure compared to conventional approach procedures in the terminal airspace

The methodology also enables the evaluation of the introduction of additional RNP approach procedures on to other runways at MDW. This has the potential of saving on average 660K gallons per year of fuel for arrivals at MDW. At $3/gallon this amounts to a savings of $1.97M per year.

5.2.4 Potential for choosing better runway configuration

The de-confliction of metroplex airspace makes operations at neighboring airspace independent of each other, providing scope for choosing optimal runway configurations for improving terminal area fuel burn efficiency.

The analysis also identifies an opportunity to select optimal runway configuration at MDW based on wind magnitude/direction and fuel burn efficiency. The use of optimal runway configuration for wind and fuel burn, along with additional RNP approach procedure has the potential of saving on an average 890K gallons per year of fuel for arrivals at MDW. At $3/gallon this amounts to a savings of $2.67M per year.

5.2.5 ROI for individual airlines

Investing in RNP approach for single airport use (for de-confliction of airspace and improving terminal approach efficiency) does not yield a positive ROI at MDW for Southwest Airlines.

The results from the analysis are used to estimate the ROI for investing in RNP for the major carrier (Southwest Airlines) at MDW. The results show that the RNP approach does not yield a positive ROI at MDW for Southwest Airlines. The carrier will have to perform at least half a million RNP approaches per year throughout its network and save at least 33 kg of fuel per approach on average to break-even in 10 years at a discount rate of 5%.

5.3 Strategies for equipage

To enable the RNP approach the air navigation service providers (ANSPs) must design and approve RNP approaches, develop ATC procedures and train air traffic controllers. In addition, airlines must equip with RNP equipment, train the crew and receive certification to fly the procedure. The adoption of RNP approaches by airlines has been slow, primarily due to: (a) issues with estimating the Return-on-Investment (ROI) and (b) the “free rider” issue, (i.e., the allocation of benefits to parties that choose not to equip but gain benefits when their competition equips). This section describes various strategies for achieving RNP approach equipage.

5.3.1 Market-based Approach

The market based approach relies on the inherent benefits of a technology to sell itself and achieve the desired equipage.

In the case of metroplex flow de-confliction and RNP approach capability, there are three major issues that negate the benefits of the RNP approach: (a) the terminal area vectoring for merging and spacing required to ensure safe separations, (b) limited potential use of the approach capability (i.e. limited to IMC days as fuel burn performance of visual approaches are better than RNP approaches in most cases) and (c) the free rider issue (i.e. the asymmetric benefits to the competition serving the market). These issues and the high costs of equipage result in low ROI for the airlines making investment in RNP approach undesirable.

At Chicago metroplex south-east winds during IMC require the use of ILS on 13C for MDW arrivals. The ILS approach to 13C at MDW conflicts with departures from 22L at ORD. This flow conflict occurs on an average 1.6% of the time per year. The introduction of the RNP approach to 13C at MDW to de-conflict the traffic flow for these periods is estimated to reduce the direct airline operating cost per year on average by $.04M at MDW and $1.33M at ORD (an asymmetry of 1:33 in favor of airlines at ORD). Also the "vectors" (due to merging and spacing) between the final waypoint on the STAR and the start of the RNP approach introduce as much variation in flight tracks as the ILS flows. These factors negate the benefits of RNP approach. Therefore, the RNP approach capability on its own is not an attractive technology to adopt for operations at airports with prevalent Visual Meteorological Conditions (VMC).

5.3.2 Operational Incentives

Programs like Best Equipped Best Serve (BEBS) are being developed to provide operational incentives in the form of priority arrival slots to equipped operator during traffic flow management initiatives (TMIs) (AhmadBeygi et al., 2013). The proposed TMI identifies periods (on flow conflict days) when equipage-based priority can be applied. During these time periods, referred to as exclusionary periods, the metroplex flows are de-conflicted by only allowing equipped aircraft at the airports. This allows metroplex to resume normal operations, but penalizes flights that are not equipped to fly the procedure required to de-conflict the metroplex.

The implementation of such TMIs will need new decision support tools and the associated training for the controller to manage the duration of the program and allocation of slots based on the level of equipage (AhmadBeygi et al., 2013).

The use of such TMIs at Chicago metroplex will results in normal operations at ORD during the exclusionary periods at the expense of airlines operating at MDW. The investment in equipage by airlines at MDW (necessary for the de-confliction) will not yield significant savings in delays as arrival demand at MDW is well below the capacity. In addition, the limited access will create equity issues for non-equipped operators resulting from excess delay allocation. This will increase overall NAS delays due to network wide delay propagation as a result of large delays for some flights. For instance, 15 minute delays for four flights can be more easily absorbed by the network than 1 hour delay for a single flight

5.3.3 Financial Incentives

In cases of market failure, a theoretical case can be made to provide financial incentives to airspace system users to equip (Post et al., 2011). In case of RNP, the benefits of the operational changes while disproportionate, not only benefit the equipped operator but also other operations and the system as a whole. This asymmetry in distribution of benefits causes the market failure. In such cases, financial incentives can be provided to defray the cost of avionics. The financial incentives can use public funds or use a tax pool that would tax every stakeholder proportional to the benefits accrued from the operational change. This will require accurate estimates of benefits to individual stakeholders which can be done using this methodology.

5.3.4 Mandate

An alternative, in the interest of modernization, would be a government mandate to equip with RNP approach capability. Mandating equipage for RNP will solve the free rider issue and ensure modernization of the NAS that is required to meet the future demand necessary for the growth of the nation.

In the past, equipage for NAS modernization for improving capacity or safety has been achieved through FAA mandates. These include:

a. <u>VHF Radio Transceivers</u>: Mandated in 1961, requiring two-way radio communications using Very High Frequency (VHF) for conducting flight operations on and around all controlled airports throughout the country

(FAA, 2012b). The VHF radio provided higher bandwidth and voice clarity. Higher bandwidth meant availability of more number of channels, which boosted airspace capacity.

b. Transponders: Mandated in 1978, requiring all aircraft operating in Terminal Radar Service Areas (TRSAs) and Terminal Control Areas (TCAs) to have transponders to report identity (Mode A) and altitude (Mode C) installed by July 1981 (FAA, 2012b).

c. Traffic Alter and Collision Avoidance System (TCAS): Mandated in 1981 to prevent mid-air collision (FAA, 2012b).

d. Wind Shear: Mandated in 1988, requiring all turbine-powered airliners seating 30 passengers or more carry equipment to warn pilots when they encounter low-altitude wind shear and provide them with information needed to escape safely (FAA, 2012b).

e. Reduced Vertical Separation Minima (RVSM): The mandate required all aircraft and flight crews operating in domestic airspace between flight level (FL) 290 to 410 to be RVSM compliant as of January 20, 2005 (FAA, 2012g). RVSM increased capacity of airspace by reduction of vertical separation requirement to 1000 feet from a previous requirement of 2000 feet for flight level (FL) 290 to 410.

From the perspective of metroplex flow de-confliction, mandating equipage is inefficient as all aircraft do not need to equip. Also, the overall cost to equip is higher than the additional airline operating cost due to metroplex flow conflict by orders of magnitude. The additional airline operating cost due to flow conflict at Chicago and New York are $4.5M and $3M (Devlin et al., 2012); whereas, the cost to airlines to equip with RNP approach capability is in the hundreds of million ($175M for Southwest Airlines).

The air navigation service providers (ANSPs) must design and approve RNP approaches to all possible runways at all major airports and train air traffic controllers. The lack of RNP approaches will restrict the use and the potential benefits of the approach capability. For example, this analysis shows investing in RNP approach for single airport use (for de-confliction of airspace and improving terminal approach efficiency) does not yield a positive ROI at MDW for Southwest Airlines. The carrier will have to perform at least half a million RNP approaches per year throughout its network and save at least 33 kg of fuel per approach on average to break-even in 10 years at a discount rate of 5%. Also, at Chicago metroplex the airlines at ORD will not have any direct benefits from equipping unless new procedures are put in place to make use of the capability. Therefore, before a mandate to equip for RNP approaches is made, the following key issues need to be addressed:

a. The air navigation service providers (ANSPs) must design and approve RNP approaches to all possible runways at all major airports for airlines to use.
b. The ANSPs must upgrade the surveillance systems and train air traffic controllers to smoothly merge and space aircraft at the start of the RNP approach.

5.4 Future Work and Application

The methodology described in the dissertation can to be applied to other metroplexes to estimate the benefits of de-confliction of flows using new PBN approach procedures. In addition this dissertation presents several opportunities for continued work in the form of methodological improvements and potential applications.

5.4.1 Methodological Improvements

The key methodological improvements are:

a. Include airline's entire networks (all airports) to accurately estimate the potential benefits of the new PBN approach to individual airlines.
b. Minimize the setup time to run the model by automating the TRACON flow analysis. This will require automated extraction of coordinates of STAR and approach procedures from NFDC database and use of clustering algorithms (Enriquez, 2013; Levy, 2003) to assign tracks to flows.

5.4.2 Analysis tool

The development of the capability to conduct benefits assessment of new concepts-of-operations and technologies using surveillance track data coupled with aerodynamic fuel burn models significantly improves the accuracy and reliability of benefits assessments. The methodology presented in this dissertation can be used to develop an analysis tool that can be used by policy decision-makers and investors (airlines) to better understand where the costs and benefits are accrued.

5.4.3 Optimal Runway Configuration

The Optimal Runway Configuration model built as a part of the methodology can be used in determining the optimal runway configuration for the tower control manager at the airports. In current practice the runway configuration is determined based only on the wind direction. An alternate approach is to select the most optimal runway configuration from a set of feasible configurations, by taking into account the direction of air traffic and the fuel burn performance of individual flows in the terminal. The results of the dissertation show that there is potential for further fuel saving using this approach.

APPENDIX A: GAWK CODE FOR CHICAGO METROPLEX ANALYSIS

A1. Instructions to run the Code

The following code is run using unix/linux terminal or an emulator (e.g., Cygwin). The instructions to run the code are as follows:

1 Copy the input files, the scripts and the user-defined functions to a directory.

2 Change directory in the terminal to the one containing the files.

3 Run code by typing ./script_run_main.gawk ↲ in the unix/linux terminal.

A2. List of Input Files

The code requires the setup and input data files provided below:

File name: NOP_filelist.dat

```
# This contains names of all the NOP track data files that need to be processed. The NOP track data files should in the same directory as all the other files.
NOP_20110118_JobID316090
NOP_20110124_JobID316091
NOP_20110125_JobID316092
NOP_20110126_JobID316093
NOP_20110127_JobID316094
NOP_20110322_JobID316095
NOP_20110323_JobID316096
NOP_20110610_JobID316097
NOP_20110611_JobID316098
```

File name: airport_info.dat

```
# airport lat lon arr_color dep_color elev(feet)
MDW 41.7859722 -87.7524167 red blue 620
ORD 41.9816486 -87.9066714 green pink 672
```

File name: runway_info.dat

```
# airport lat lon arr_color dep_color rwy1 rwy2 lat1 lon1 lat2 lon2 color_arr_rwy1 color_dep_rwy1 color_arr_rwy2 color_dep_rwy2
MDW 41.7859722 -87.7524167 red blue 13C 31C 41.791573 -87.761067 41.779240 -87.743738 red crimson firebrick darkred
MDW 41.7859722 -87.7524167 red blue 4R 22L 41.779160 -87.759317 41.791996 -87.743056 pink hotpink deeppink mediumvioletred
MDW 41.7859722 -87.7524167 red blue 4L 22R 41.781368 -87.761192 41.792336 -87.747300 coral orangered tomato orange
MDW 41.7859722 -87.7524167 red blue 13L 31R 41.792315 -87.757938 41.782598 -87.744271 saddlebrown sienna brown maroon
# MDW 41.7859722 -87.7524167 red blue 13R 31L 41.788068 -87.758803 41.780772 -87.748552 mediumspringgreen green
yellowgreen darkolivegreen
ORD 41.9816486 -87.9066714 green pink 10 28 41.968995 -87.931532 41.969070 -87.883729 gold yellow peachpuff khaki
ORD 41.9816486 -87.9066714 green pink 14L 32R 42.002435 -87.915368 41.981405 -87.891713 thistle violet blueviolet indigo
ORD 41.9816486 -87.9066714 green pink 14R 32L 41.990435 -87.933140 41.970083 -87.910233 greenyellow lime limegreen
lightgreen
ORD 41.9816486 -87.9066714 green pink 4R 22L 41.953327 -87.899418 41.969922 -87.879743 mediumspringgreen seagreen green
darkgreen
ORD 41.9816486 -87.9066714 green pink 9R 27L 41.983897 -87.918352 41.983900 -87.889051 yellowgreen olivedrab olive
darkolivegreen
ORD 41.9816486 -87.9066714 green pink 4L 22R 41.981655 -87.913918 41.997537 -87.896371 mediumaquamarine darkseagreen
lightseagreen darkcyan
ORD 41.9816486 -87.9066714 green pink 9L 27R 42.002832 -87.926676 42.002831 -87.899084 darkturquoise cadetblue steelblue
lightsteelblue
```

File name: flow_direction.dat

```
# Airport Arrival/Dep Direction lat1(upperleft) lon1(upperleft) lat2(lowerright) lon2(lowerright)
MDW Atracks E 41.72663605364218 -87.06180839663364 41.22663605364218 -86.76180839663364
MDW Atracks W 41.65792541789417 -88.54367194784265 41.32349630243619 -88.28656624524224
```

File name: flow_info_all.dat

```
# Airport Arr(Atracks)_Dep(Dtracks) Direction Runway Procedure Fix1 Fix1_radius(NM) Vertical
MDW Atracks E 13C ILS HEBKU 4.5 0
MDW Atracks E 13C RNP GIKLE 5 0
# MDW Atracks E 13C RNP JUPIR 1.5 2000
# MDW Atracks E 13C Visual 0
MDW Atracks W 13C ILS HEBKU 4.5 0
MDW Atracks W 13C RNP GIKLE 5 0
# MDW Atracks W 13C RNP JUPIR 1.5 2000
# MDW Atracks W 13C Visual 0
MDW Atracks W 31C ILS GLEAM 6 4000
MDW Atracks E 31C ILS GLEAM 2.5 4000
# MDW Atracks W 31C ILS HILLS 5 0
# MDW Atracks W 31C Visual 0
# MDW Atracks E 4R ILS TASUE 6 0
MDW Atracks E 4R ILS CADON 6 4000
MDW Atracks W 4R ILS CADON 2.5 4000
# MDW Atracks E 4R Visual 0
```

File name: MDW_fix.dat

```
# waypoint lat lon
OKK      40.5278   -86.058
TROLY    40.6945   -86.0542
GOTNE    40.9613   -86.048
FISSK    41.0492   -86.4328
VEECK    41.1253   -86.6192
OZZEY    41.3992   -86.9335
AZUMO    41.4578   -87.0048
HALIE    41.5161   -87.1589
CGT      41.51     -87.5715
FWA      40.9782   -85.1912
BAGEL    41.516    -85.6137
GSH      41.5252   -86.028
MEGGZ    41.5234   -86.3957
AWSUM    41.5221   -86.5732
IROCK    41.5191   -86.9052
HALIE    41.5161   -87.1589
CGT      41.51     -87.5715
LFD      42.0625   -84.7651
BAGEL    41.516    -85.6137
```

SPI	39.8397	-89.6777
YEARY	40.6874	-88.8643
PNT	40.8212	-88.7335
MOTIF	41.2296	-88.501
MINOK	41.4689	-88.3633
JOT	41.5464	-88.3184
MAGOO	40.0249	-90.7618
PIA	40.6801	-89.793
KORTT	40.8858	-89.3157
MOTIF	41.2296	-88.501
IRK	40.135	-92.5917
PIA	40.6801	-89.793
LMN	40.5967	-93.9676
RENZO	41.1358	-89.8062
BDF	41.1597	-89.5879
MOTIF	41.2296	-88.501
CVA	41.7085	-90.4833
BDF	41.1597	-89.5879
TOYUL	41.7123	-88.0704
GIKLE	41.7693	-87.9846
JUPIR	41.8263	-87.8987
NIDEE	41.8356	-87.8303
DULTE	41.8252	-87.8083
TASUE	41.6006	-87.9891
JERNU	41.6613	-87.9128
HADGI	41.7219	-87.8364
PAKLE	41.6039	-87.8323
JERNU	41.6613	-87.9128
BANER	41.6106	-87.972
CADON	41.6457	-87.9278
CITGO	41.7117	-87.8447
OLOXE	41.7487	-87.7978
KANLE	41.846	-87.9957
HANOD	41.9065	-87.9188
EXEKE	41.8491	-87.8379
JABRI	41.8807	-87.4811
HAXOM	41.9381	-87.5621
EXARE	41.8776	-87.6391
CIDIG	41.8196	-87.7127
CGT	41.51	-87.5715
HILLS	41.6294	-87.534
GLEAM	41.6634	-87.5814
RUNTS	41.7144	-87.6529
HOBEL	41.7433	-87.6934
PANUE	41.6063	-87.6586
HAKBO	41.6667	-87.582
FANEK	41.7244	-87.6626
CENAP	41.7619	-87.7152
IDUDE	41.7387	-87.4892
HAKBO	41.6667	-87.582
HEBKU	41.9075	-87.9245
HITOB	41.8386	-87.8273
MANLI	41.9089	-87.5016
HINSN	41.7907	-87.7447

File name: goaround_holding.dat

```
# Airport Type(Arrival/Departures) Checkradius
MDW Atracks 20
# ORD Atracks 20
```

File name: flow_measuring_fix.dat

```
# Airport Arrival/Dep Direction (measuring line)lat1 lon1 lat2 lon1 check_radius
MDW Atracks E CGT 41.646154 -87.410911 41.476136 -87.612629 25
MDW Atracks W JOT 41.437972 -88.192559 41.658152 -88.44957 40
```

File name: BADA_coefficient_2.dat
This file contains the aircraft operation performance information obtained fromt the BADA OPF files. A sample of the information is provided below. Make sure all the information as consistent with the header is enter for every aircraft type at the airport. See BADA manual and “function.badacoefficient” in section A4 for information about each field in the file.

```
# actype Cf1 Cf2 Cf3 Cf4 Cfcr Cd0CR Cd2CR VstallCR Cd0AP Cd2AP VstallAP Cd0LD Cd2LD Cd0delLD VstallLD mass S
MTOW engine
A306 0.63936 516.862 21.196 67071 0.98852 0.020591 0.051977 151 0.038031 0.044932 109 0.078935 0.044822 0.0225 97 133000
260 171700 Jet
A30B 0.68406 538.109 38.59 46972 0.95523 0.022844 0.05373 146 0.039333 0.045401 101 0.076977 0.03684 0.0245 99 131950 260
165000 Jet
A310 0.63947 586.621 15.664 -332800 0.95439 0.024934 0.03963 142 0.05189 0.0401 105 0.10591 0.03604 0.02208 97 116925 219
150000 Jet
A318 0.64411 678.861 11.956 72889 0.98776 0.028747 0.034614 138 0.0344 0.0445 99 0.0746 0.0475 0.0179 93 57500 122.6 68000
Jet
A319 0.72891 889.886 11.114 133850 0.99224 0.025954 0.025882 139 0.046986 0.035779 100 0.097256 0.036689 0.02568 94 58500
122.6 70000 Jet
A320 0.6333 441.923 9.134 79668 0.95423 0.025149 0.036138 145 0.0456 0.0381 107 0.0838 0.0371 0.0312 101 62250 122.6 77000
Jet
A321 0.72987 636.316 14.159 68867 1 0.026984 0.035074 145 0.047354 0.040818 106 0.07959 0.037708 0.029751 103 70750 122.6
83000 Jet
A332 0.59426 424.849 25.897 79790 0.95422 0.018953 0.032965 141 0.0564 0.03221 111 0.08197 0.02995 0.0316 104 179000 361.6
230000 Jet
A333 0.61503 472.79 21.033 112230 0.93655 0.019805 0.031875 134 0.0555 0.0325 105 0.078 0.0345 0.0257 99 173500 361.6
212000 Jet
A343 0.62965 525.196 31.094 75361 0.92082 0.021718 0.035123 142 0.05398 0.03552 104 0.07664 0.03302 0.027 101 192000 361.6
275000 Jet
A345 0.63894 779.898 34.292 84087 0.98057 0.022634 0.036883 167 0.0344 0.0447 126 0.0497 0.0421 0.0179 123 273500 437
372000 Jet
A346 0.64295 755.667 33.958 90040 0.95843 0.02214 0.038022 158 0.0364 0.0465 121 0.0477 0.0441 0.0177 118 261470 437
365000 Jet
A388 0.54336 445.622 64.145 74435 0.97182 0.01813 0.043198 154 0.0328 0.0528 117 0.0452 0.051 0.0181 114 434000 845 560000
Jet
A3ST 0.61103 740.131 21.127 84803 0.93593 0.035226 0.046765 146 0.0378 0.0495 112 0.0828 0.0458 0.0174 100 131000 260
153000 Jet
AT43 3.6224 695.58 6.3855 77290 1.1396 0.021851 0.034779 98 0.039 0.032 85 0.058 0.029 0.021 75 15100 54.5 16700 Turboprop
AT45 3.9933 345.887 5.4649 68373 1.0492 0.035466 0.031737 98 0.0525 0.0298 81 0.0633 0.0293 0.0211 77 16600 54.51 18600
Turboprop
AT72 3.6731 1018.29 7.8788 72338 1.2227 0.022072 0.0293 102 0.0396 0.0297 86 0.07972 0.0296 0.02115 76 19650 61 21500
Turboprop
ATP 3.0341 514444444000000 5.9858 1000000000000000 1.0502 0.02633 0.027511 99 0.05266 0.027511 83 0.07899 0.027511 0.02
75 20684.5 78.82 22930 Turboprop
B462 0.75195 391.4 18.993 -642700 0.98117 0.032992 0.039381 135 0.065984 0.039381 93 0.098976 0.039381 0.02 90.5 35000 77.3
42200 Jet
B703 1.15 565.889 28 49990 1 0.0142 0.062707 125 0.0284 0.062707 96 0.0426 0.062707 0.02 96 96300 274.6 140000 Jet
B712 0.60281 267.11 10.444 133650 1.0429 0.020945 0.045998 145 0.0465 0.0423 107 0.0588 0.041 0.0224 104 44750 90.02 52600
Jet
B722 0.32166 60.3135 28.211 60154 0.92644 0.018415 0.061258 157 0.03683 0.061258 106 0.055245 0.061258 0.02 104 68750
157.9 86400 Jet
B732 1.1423 1108.73 20.144 53383 0.99363 0.021609 0.043078 143 0.0411 0.0416 102 0.079 0.038 0.0254 99 43950 91.09 52300
Jet
B733 0.78052 525.608 14.768 52584 0.99371 0.024958 0.040885 141 0.0605 0.0414 108 0.092 0.0373 0.02 106 50400 91.09 62800
Jet
B734 0.7595 508.95 14.769 52343 0.97905 0.025953 0.044644 152 0.0477 0.0433 115 0.0833 0.0373 0.0228 109 55310 91.09 68000
Jet
B735 0.77318 454.846 14.733 52667 0.99746 0.022776 0.045399 143 0.0542 0.041 110 0.089 0.0383 0.015 103 49360 91.09 60680
Jet
B736 0.63557 360.955 15.384 61221 0.96824 0.021696 0.036752 137 0.0497 0.0403 100 0.0721 0.0401 0.0208 99 54000 124.65
65000 Jet
B737 0.6864 490.188 10.592 59399 0.9342 0.023738 0.037669 143 0.0477 0.0423 105 0.0653 0.0412 0.0235 103 57510 124.65
70080 Jet
B738 0.70057 549.478 14.19 65932 0.92958 0.025452 0.035815 149 0.0492 0.0424 109 0.0689 0.0404 0.0249 107 63375 124.65
78300 Jet
B739 0.70675 583.997 13.133 61936 0.93516 0.025734 0.033615 156 0.046859 0.037823 118 0.080202 0.034566 0.022 111 69686
124.58 85139 Jet
```

File name: MDW_wind_info.dat
This file contains information (see header) extracted from the ASPM airport table for years 2006 to 2012. In addition the date time information is converted to unix timestamp. A sample of the file is shown below. Make sure all the field are present.

```
# LOCID YYYYMM DAYNUM HR_LOCAL QTR TIMESTAMP(UNIX) MC CEILING VISIBLE TEMP WND_ANGL
WND_SPED RUNWAY
MDW 200601 1 0 1 1136095200 V 120 8 - 210 5 31C31L31R|22L31C31L31R
MDW 200601 1 0 2 1136096100 V 120 8 - 210 5 31C31L31R|22L31C31L31R
MDW 200601 1 0 3 1136097000 V 120 8 - 210 5 31C31L31R|22L31C31L31R
MDW 200601 1 0 4 1136097900 V 120 8 32 210 4 31C31L31R|22L31C31L31R
MDW 200601 1 1 1 1136098800 V 120 8 32 210 4 31C31L31R|22L31C31L31R
MDW 200601 1 1 2 1136099700 V 120 8 32 210 4 31C31L31R|22L31C31L31R
MDW 200601 1 1 3 1136100600 V 120 8 32 210 4 31C31L31R|22L31C31L31R
MDW 200601 1 1 4 1136101500 V 120 8 32 200 7 31C31L31R|22L31C31L31R
MDW 200601 1 2 1 1136102400 V 120 8 32 200 7 31C31L31R|22L31C31L31R
MDW 200601 1 2 2 1136103300 V 120 8 32 200 7 31C31L31R|22L31C31L31R
```

File name: aircraft_cat.dat
This file contains aircraft categories at MDW. A sample of the file is shown below. This can derived from the NOP data being processed. The aircraft are classified as turboprops, business jet (small, medium, large), regional jets (medium, small), B73's and narrow body based on their MTOW.

```
# Actype Engine Count MTOW Category
B737 Jet 5785 70080 B737s
B738 Jet 15 78300 B737s
B733 Jet 1526 62800 B737s
B735 Jet 328 60680 B737s
B732 Jet 1 52300 B737s
B734 Jet 2 68000 B737s
B738 Jet 15 78300 narrow_body
A320 Jet 30 77000 narrow_body
B721 Jet 1 77000 narrow_body
MD90 Jet 9 77000 narrow_body
B737 Jet 5785 70080 narrow_body
A319 Jet 297 70000 narrow_body
```

File name: flow_colors_2.dat

```
# direction runway approach color
E 13C ILS red
E 13C RNP magenta
E 13C Visual pink
# E 13L SA
E 22L Visual orange
# E 22R Visual 64 15.111 22.47 18.6924 1.59441
E 31C ILS blue
E 31C Visual violet
# E 4L SA 35 23.6128 39.851 31.543 4.54613
E 4R ILS yellow
E 4R Visual green
W 13C ILS red
W 13C RNP magenta
W 13C Visual pink
# W 13L Visual 6 31.034 97.834 44.0956 26.4114
W 22L Visual orange
# W 22R Visual 62 36.295 78.4852 48.9974 7.92077
W 31C ILS blue
W 31C Visual yellow
# W 31R Visual 1 41.7532 41.7532 41.7532 0
# W 4L Visual 34 25.6115 39.2291 30.2807 2.73918
W 4R ILS green
W 4R Visual lime
```

File name: rnp_gen_info.dat
This file contains the criteria for reflecting and rotating 13C RNP tracks to get RNP tracks for 22L, 31C and 4R at MDW.

```
# airport direction1  runway1 direction2 runway2 rotate(0,1) reflect
MDW E 4R W 13C RNP 1 0
MDW E 22L W 13C RNP 1 1
MDW W 22L E 13C RNP 1 1
MDW W 31C W 13C RNP 1 1
```

File name: MDW_flight_data_2007_2012.TAB
This the ASPM flight table (tab separated) for MDW from year 2007 to 2012. The can be downloaded from the FAA ASPM website.

File name: MDW_star_holding_fix.dat
This file contains the coordinates for boundaries to capture holding patterns on the arrival STARs for MDW.

```
# direction holdingfixes coordinates
# W MINOK
W MOTIF -88.31906309616981 41.27655670829328 -88.45340850652383 40.9712032958472 -88.7719189001181
41.05381907437446 -88.62588191895068 41.35448425806798
# W PNT
# W PIA
# W BDF
E OZZEY -86.84127460957141 41.14297499358587 -87.13463422320871 41.36712572344572 -87.00603998870756
41.47145373742799 -86.72558787251218 41.25764817533518
E FISSK -86.48080818382758 41.19791552102979 -86.05732347451041 41.09936272533439 -86.16326659945729
40.84334443177345 -86.59425723283579 40.9450811340405
E HALIE -87.42489445004537 41.57038497020047 -87.1442507845394 41.56647298242147 -87.13370082771226
41.42736351560021 -87.4334148456261 41.4254552145423
E IROCK -86.99004519299315 41.50828170749523 -86.99059307454925 41.65809435393067 -86.59169834322361
41.65976256473662 -86.59209867470884 41.51051470725858
E GSH -85.49007757056091 41.65951969124372 -85.49697772117916 41.32506632044909 -86.22257425653612
41.32597546754238 -86.2090567269282 41.67198864319884
```

File name: MDW_2007_2012_arr_rwy
This file combine ASPM airport and flight table to get the count of flight from each direction to each runway.

```
# LOCID YYYYMM DAYNUM HR_LOCAL QTR TimeStamp MC CEILING VISIBLE TEMP WND_ANGL WND_SPED
RUNWAY Traffic_east Traffic_west Ratio Arrival_Rwy
MDW 200701 1 7 2 1167657300 I 013 9 38 260 11 31C31L31R|22L31C31L31R 1 0 0.662356 31C
MDW 200701 1 8 1 1167660000 I 015 10 39 270 10 31C31L31R|22L31C31L31R 2 1 0.646597 31C
MDW 200701 1 8 2 1167660900 I 015 10 39 270 10 31C31L31R|22L31C31L31R 2 1 0.646597 31C
MDW 200701 1 8 3 1167661800 I 015 10 39 270 10 31C31L31R|22L31C31L31R 4 3 0.646597 31C
MDW 200701 1 8 4 1167662700 I 015 10 39 290 13 31C31L31R|22L31C31L31R 2 2 0.646597 31C
MDW 200701 1 9 1 1167663600 I 015 10 39 290 13 31C31L31R|22L31C31L31R 3 2 0.6 31C
MDW 200701 1 9 2 1167664500 I 015 10 39 290 13 31C31L31R|22L31C31L31R 4 2 0.6 31C
MDW 200701 1 9 3 1167665400 I 015 10 39 290 13 31C31L31R|22L31C31L31R 3 1 0.6 31C
MDW 200701 1 9 4 1167666300 V 019 10 40 310 10 31C31L31R|22L31C31L31R 2 2 0.6 31C
```

File name: ORD_ASPM_airport.txt (tab separated)
ASPM airport table for ORD for years 2007 to 2012.

File name: ORD_flight_data_2007_2012.TAB (tab separated)
ASPM flight table for ORD for years 2007 to 2012

.

File name: ORD_2007_2012_dep_cancelled.tab (tab separated)
BTS on-time performance table for year 2007 to 2012.

A3. Gawk scripts

The scripts for the analysis are provided below. Each script needs to be saved (as the file name provided) in the same directory as the other files

```
File name: script_run_main.gawk
# This script runs all the other scripts
infile="NOP_filelist.dat"

# run script_process_tracks_2.gawk
gawk '{if($1!="#") print $0}' $infile > temp_input
gawk '{execute="./script_process_tracks_2.gawk "$1; print execute; system(execute)}'
temp_input

# run script_compute_NOP_stats.gawk
gawk 'BEGIN{execute="./script_compute_NOP_stats.gawk"; print execute; system(execute)}'

# run script_plot_tracks_by_flow0.gawk
gawk 'BEGIN{execute="./script_plot_tracks_by_flow0.gawk"; print execute;
system(execute)}'

# run script_get_new_rnp_approach0.gawk
gawk 'BEGIN{execute="./script_get_new_rnp_approach0.gawk"; print execute;
system(execute)}'

# run script_get_fb_all_flows.gawk
gawk 'BEGIN{execute="./script_get_fb_all_flows.gawk"; print execute; system(execute)}'

# run script_MDW_annual_fb0.gawk
gawk 'BEGIN{execute="./script_MDW_annual_fb0.gawk"; print execute; system(execute)}'

# run script_get_holding_patterns.gawk
gawk 'BEGIN{execute="./script_get_holding_patterns.gawk"; print execute;
system(execute)}'

# run script_metroplex_benefits.gawk
gawk 'BEGIN{execute="./script_metroplex_benefits.gawk"; print execute; system(execute)}'
```

```
File name: script_MDW_annual_fb0.gawk
# This script estimate the annual fuel burn for MDW arrivals
# Input files
# Input1 ASPM runway and wind info
rwy_wind_info="MDW_wind_info.dat"
# Input2 ASPM flight info
flight_info="MDW_flight_data_2007_2012.TAB"
# Input3 valid ac types
actypes="aircraft_cat.dat"
# NOP data
nopflight="NOP_data_all_good_2"
# Year range
year_start="2007"
```

```
year_end="2012"
# Time range
time_start=7
time_end=22
# Fuel burn flow
MDW_fb_flow="MDW_all_flows_mean_fb"

# runways
r1="13C"
r2="31C"
r3="22L"
r4="4R"

# basic process repeat flag
# Change this to zero if the basic data process does not have to be repeated
firstrun=1

if [ $firstrun -eq 1 ]
then
# Process flight information and get flight count every 15 min
# make actype tab separated
sed 's/ /\t/g' $actypes > temp_actypes
gawk -v year_start=$year_start -v year_end=$year_end -v airline="SWA" 'BEGIN{FS="\t"}
(NR==FNR){arr[$1]; next} ($20 in arr){if(substr($5,1,4)>=year_start &&
substr($5,1,4)<=year_end){bin[$5" "$6" "$7" "$8]++; if($17==airline){bin2[$5" "$6" "$7"
"$8]++}}} END{for(no in bin){print no,bin[no],bin2[no]}}' temp_actypes $flight_info |
sort -k1,1n -k2,2n -k3,3n -k4,4n > MDW_flightcount

##gawk -v year_start=$year_start -v year_end=$year_end 'BEGIN{FS="\t"}
{if(substr($5,1,4)>=year_start && substr($5,1,4)<=year_end){bin[$5" "$6" "$7" "$8]++}}
END{for(no in bin){print no,bin[no]}}' $flight_info | sort -k1,1n -k2,2n -k3,3n -k4,4n >
temp_flightcount
#
# Get the hourly east west traffic ratio
TZ=UTC gawk -v zone=-6 '{if($1!="#"){mnth=strftime("%m",$13); hr=strftime("%H",$13);
qtr=int(strftime("%M",$13)/15)+1; if(mnth>=3 && mnth<11){hr=hr+zone-1} else{hr=hr+zone};
if(hr<0){hr=hr+24}; arr[hr" "$23]++}} END{for(no in arr) print no, arr[no]}' $nopflight |
sort -k1,1n -k2,2 > temp_tratio_1

gawk 'BEGIN{x=99} {if($1!=x){if(FNR!=1){if(count==2) print hr,y1/(y1+y2)}; x=$1; count=1;
y1=$3; hr=$1} else{count++; y2=$3}} END{if(count==2) print hr,y1/(y1+y2)}' temp_tratio_1
> MDW_east_west_ratio

# Combine wind info with traffic and traffic ratio information for every qtr
gawk '(NR==FNR){arr[$1" "$2" "$3" "$4]=$5" "$6; next} ($2" "$3" "$4" "$5 in arr){print
$0, arr[$2" "$3" "$4" "$5]}' MDW_flightcount $rwy_wind_info | gawk
'{if($15~/^$|0/){$15=0}}1' > temp_qtr_info1

gawk '(NR==FNR){arr[$1]=$2; next} ($4 in arr){print $0, arr[$4]}' MDW_east_west_ratio
temp_qtr_info1 > temp_qtr_info2

# Filter out periods before 7AM and after 10PM
gawk -v time_start=$time_start -v time_end=$time_end '{if($4>=time_start && $4<=time_end)
print $0}' temp_qtr_info2 > temp_qtr_info3

# Create proxy approach for RNP approach on to 31C and 4R from the east and west
respectively
# E 31C ILS = E 31 RNP
# W 4R ILS = W 4R RNP
gawk '{if(($1=="E" && $2=="31C" && $3=="ILS") || ($1=="W" && $2=="4R" &&
$3=="ILS")){print $0}}' $MDW_fb_flow | gawk '{$3="RNP"}1' > temp_MDW_fb_flow_1
cat temp_MDW_fb_flow_1 $MDW_fb_flow | sort -k2,2 -k3,3 > temp_MDW_fb_flow_2
gawk '{if($2" "$3!=x){x=$2" "$3; fb1=$7; fb2=$9} else{print x,fb1,fb2,$7,$9}}'
temp_MDW_fb_flow_2 | sort -k2,2 -k1,1 > temp_MDW_fb_flow_3

# print average fuel burn per runway per approach for various values of east west ratio
(0 to 1)
```

```
gawk '{fb_east[FNR]=$4; fb_west[FNR]=$6}
END{for(i=0;i<=1;i=i+.1){for(j=1;j<=length(fb_east);j++){fb=i*fb_east[j]+(1-
i)*fb_west[j]; if(j==length(fb_east)){printf "%.2f\n", fb} else{printf "%.2f ", fb}}}}'
temp_MDW_fb_flow_3 > temp_MDW_fb_flow_4


# get the actual runway for each 15 min  bin
gawk -v r1=$r1 -v r2=$r2 -v r3=$r3 -v r4=$r4 '{if($13~/^'"$r1"'/){print $0,r1}
else{if($13~/^'"$r2"'/){print $0,r2} else{if($13~/^'"$r3"'/){print $0,r3}
else{if($13~/'"$r4"'/){print $0,r4} else{print $0,"NA"}}}}}' temp_qtr_info3 >
temp_qtr_info4

# Output MDW arrival runway config
cp temp_qtr_info4 MDW_$year_start"_"$year_end"_"arr_rwy


# Get average fuel burn per flight for actual and optimal runway configuration
# print out three cases of flows
# case1: only Visual and ILS, case2: case1 plus 13C RNP, case3: all flows
gawk 'BEGIN{print 1; print 2; print 3}' > temp_flowcases
# For all aircrafts get actual and optimal runway configuration
gawk -v infile="temp_qtr_info4" -v fb_flow="temp_MDW_fb_flow_3" '{execute="time
./script_get_fb_15min_actual_config.gawk "$1" "infile" "fb_flow; print execute;
system(execute)}' temp_flowcases

# The output of the above script is temp_qtr_info4 with additional information

# Compute total number of flight and fuel burn for each case
# flight count field for All flights is $14 and for SWA is $15
# Also fuel burn for B737 is more than all fleet mix by 1.19
# Compute total fuelburn per year for all year
gawk -v fcf=14 -v factor=1 '{if($fcf!=0){year=substr($2,1,4); if($18!=999){fltcnt[year"
"$7]+=$fcf; base[year" "$7]+=($18*$fcf)*factor; a1[year" "$7]+=($20*$fcf)*factor;
a2[year" "$7]+=($23*$fcf)*factor; a3[year" "$7]+=($25*$fcf)*factor; a4[year"
"$7]+=($28*$fcf)*factor}}} END{for(no in fltcnt){print no,
fltcnt[no],base[no],a1[no],a2[no],a3[no],a4[no]}}' temp_qtr_info4 | sort -k1,1n -k2,2n >
temp_flight_count_fb_by_year1
# Compute average fuelburn per year
gawk -f function.mean -f function.std -f function.avgfb_allcases
temp_flight_count_fb_by_year1 > temp_flight_count_fb_by_year2

# Compute total fuelburn per year for all year
gawk -v fcf=15 -v factor=1.19 '{if($fcf!=0){year=substr($2,1,4);
if($18!=999){fltcnt[year" "$7]+=$fcf; base[year" "$7]+=($18*$fcf)*factor; a1[year"
"$7]+=($20*$fcf)*factor; a2[year" "$7]+=($23*$fcf)*factor; a3[year"
"$7]+=($25*$fcf)*factor; a4[year" "$7]+=($28*$fcf)*factor}}} END{for(no in fltcnt){print
no, fltcnt[no],base[no],a1[no],a2[no],a3[no],a4[no]}}' temp_qtr_info4 | sort -k1,1n -
k2,2n > temp_flight_count_fb_by_year1_swa
# Compute average fuelburn per year
gawk -f function.mean -f function.std -f function.avgfb_allcases
temp_flight_count_fb_by_year1_swa > temp_flight_count_fb_by_year2_swa

# Get actual and optimal runway config
gawk '{actual[$17]++; opt1[$21]++; opt2[$26]++} END{for(no in actual){print no,
actual[no], opt1[no],opt2[no]}}' temp_qtr_info4 | sort -k1,1 > MDW_runway_config1

# Get IMC VMC for each year
gawk '{arr[substr($2,1,4)" "$7]++} END{for(no in arr) print no, arr[no]}' temp_qtr_info4
| sort -k1,1n > temp_MDW_MC


else

gawk 'BEGIN{y="do nothing"}'


# The output of the above script is temp_qtr_info4 with additional information
# Compute total number of flight and fuel burn for each case
# flight count field for All flights is $14 and for SWA is $15
# Also fuel burn for B737 is more than all fleet mix by 1.19
# Compute total fuelburn per year for all year
gawk -v fcf=14 -v factor=1 '{if($fcf!=0){year=substr($2,1,4); if($18!=999){fltcnt[year]
"$7]+=$fcf; base[year" "$7]+=($18*$fcf)*factor; a1[year" "$7]+=($20*$fcf)*factor;
a2[year" "$7]+=($23*$fcf)*factor; a3[year" "$7]+=($25*$fcf)*factor; a4[year"
"$7]+=($28*$fcf)*factor}}} END{for(no in fltcnt){print no,
fltcnt[no],base[no],a1[no],a2[no],a3[no],a4[no]}}' temp_qtr_info4 | sort -k1,1n -k2,2n >
temp_flight_count_fb_by_year1
# Compute average fuelburn per year
gawk -f function.mean -f function.std -f function.avgfb_allcases
temp_flight_count_fb_by_year1 > temp_flight_count_fb_by_year2

# Compute total fuelburn per year for all year
gawk -v fcf=15 -v factor=1.19 '{if($fcf!=0){year=substr($2,1,4);
if($18!=999){fltcnt[year" "$7]+=$fcf; base[year" "$7]+=($18*$fcf)*factor; a1[year"
"$7]+=($20*$fcf)*factor; a2[year" "$7]+=($23*$fcf)*factor; a3[year"
"$7]+=($25*$fcf)*factor; a4[year" "$7]+=($28*$fcf)*factor}}} END{for(no in fltcnt){print
no, fltcnt[no],base[no],a1[no],a2[no],a3[no],a4[no]}}' temp_qtr_info4 | sort -k1,1n -
k2,2n > temp_flight_count_fb_by_year1_swa
# Compute average fuelburn per year
gawk -f function.mean -f function.std -f function.avgfb_allcases
temp_flight_count_fb_by_year1_swa > temp_flight_count_fb_by_year2_swa

# Get actual and optimal runway config
gawk '{actual[$17]++; opt1[$21]++; opt2[$26]++} END{for(no in actual){print no,
actual[no], opt1[no],opt2[no]}}' temp_qtr_info4 | sort -k1,1 > MDW_runway_config1

# Get IMC VMC for each year
gawk '{arr[substr($2,1,4)" "$7]++} END{for(no in arr) print no, arr[no]}' temp_qtr_info4
| sort -k1,1n > temp_MDW_MC

fi
```

File name: script_assign_airport.gawk

```
# This script filters out noise and assign track to the airport of interest
# This also separates arrival from departures
infile1=$1
airport=$2
lat=$3
lon=$4
color1=$5
color2=$6
input1=$infile1"_first_last"
input2=$infile1"_all_filtered"

out1=$infile1"_"$airport"_AD"
out2=$infile1"_"$airport"_Atracks"
out3=$out2".kml"
out4=$infile1"_"$airport"_Dtracks"
out5=$out4".kml"


# Assign airport, arrival departure information to each flight
gawk -v airport=$airport -v lat=$lat -v lon=$lon -v alt=2000 -v span=.05 -f
function.assignairports $input1  > temp_AD

# Seperate arrival track from departure tracks
gawk '(NR==FNR){if($NF=="A"){arr[$1]=$NF; next}} ($1 in arr){print $0,arr[$1]}' temp_AD
$input2 > temp_A_tracks
gawk '(NR==FNR){if($NF=="D"){arr[$1]=$NF; next}} ($1 in arr){print $0,arr[$1]}' temp_AD
$input2 > temp_D_tracks

# Print output files
# airport flights
cp temp_AD $out1
# Arrival track information
cp temp_A_tracks $out2
# kml file for arrival tracks
if [ -s $out2 ]
then
    gawk -v filename=$out1 -v linecolor=$color1 -f function.getcolor -f
function.kmlfile_1 $out2 > $out3
fi
# Departure track information
cp temp_D_tracks $out4
# kml file for arrival tracks
if [ -s $out4 ]
then
    gawk -v filename=$out1 -v linecolor=$color2 -f function.getcolor -f
function.kmlfile_1 $out4 > $out5
fi
```

File name: script_assign_flow_direction.gawk

```
# This script assigns direction to each track
# Input
infile1=$1
recno=$2
airport=$3
type=$4
direction=$5
lat1=$6
lon1=$7
lat2=$8
lon2=$9
# Step1 Get input file
if [ $recno -eq 1 ]
then
input1=$infile1"_"$airport"_"$type
input2=$infile1"_"$airport"_"$type"rwy"
rm temp_assignedids
else
input1="temp_remainingtracks"
input2=$infile1"_"$airport"_"$type"rwy"
fi

if [ $recno -ne 0 ]
then
# Get id of track inside the box
gawk -v lat1=$lat1 -v lon1=$lon1 -v lat2=$lat2 -v lon2=$lon2 '{if($7<lat1 && $7>lat2 &&
$8>lon1 && $8<lon2) print $1}' $input1 | sort -u > temp_validid
# Print out remaining track
gawk '(NR==FNR){arr[$1];next} !($1 in arr){print $0}' temp_validid $input1 > temp_dummy
cp temp_dummy temp_remainingtracks
# Assign valid id the flow direction
gawk -v direction=$direction '(NR==FNR){arr[$1];next} ($1 in arr){print $0,direction}'
temp_validid $input2 >> temp_assignedids
else
out1=$infile1"_"$airport"_"$type"rwy_dir"
cp temp_assignedids $out1
fi
```

File name: script_assign_flow_direction0.gawk

```
# This script executes the script_assign_flow_direction.gawk
#
# Input
infile1=$1
```

```
airport=$2
type=$3
flowdirection=$4


### type airport is the airport of interest type is Arrival tracks (Atracks) or Departure
tracks (Dtracks)
gawk -v airport=$airport -v type=$type '{if($1==airport && $2==type){i++;print i,$0}}
END{print 0,airport,type}' $flowdirection > temp_flowdirection
## Assign flow direction
gawk -v infile1=$infile1 '{execute="time ./script_assign_flow_direction.gawk "infile1"
"$0; print execute; system(execute)}' temp_flowdirection
```

File name: script_assign_flow_procedure.gawk

```
# This script assigns flow procedure to each track
# e.g. ILS RNP or visual approach
## This script assigns direction to each track
# Input
infile1=$1
recno=$2
airport=$3
type=$4
direction=$5
runway=$6
procedure=$7
fix=$8
check_radius=$9
vertical=${10}
fix_lat=${11}
fix_lon=${12}

# Step1 Get input file
if [ $recno -eq 1 ]
then
input1a=$infile1"_"$airport"_"$type
input2=$infile1"_"$airport"_"$type"rwy_dir_pro"
# rm temp_assignedids
# filter out tracks for the current runway and direction
gawk -v runway=$runway -v direction=$direction '{if($21==runway && $22==direction) print
$1,$21,$22}' $input2 > temp_ids_runway_direction
gawk '(NR==FNR){arr[$1];next} ($1 in arr){print $0}' temp_ids_runway_direction $input1a >
temp_track_runway_direction
input1="temp_track_runway_direction"
else
input1="temp_remainingtracks"
input2=$infile1"_"$airport"_"$type"rwy_dir_pro"
fi

# body of the code where each track is assigned to a procedure
if [ $recno -ne 0 ]
then
# Get id of track around each fix
## Debug test print
#gawk -v fix_lat=$fix_lat -v fix_lon=$fix_lon -v check_radius=$check_radius -f
function.distfromfix1 -f function.gcd_haversine_2 $input1 | sort -u > temp_test
gawk -v vertical=$vertical -v fix_lat=$fix_lat -v fix_lon=$fix_lon -v
check_radius=$check_radius -v procedure=$procedure -v runway=$runway -v scale=0.2 -f
function.distfromfix1 -f function.bearing -f function.absvalue -f function.radian -f
function.degrees -f function.gcd_haversine_2 -f function.dist_point_line_2 -f
function.getcartesian_2 $input1 | sort -u > temp_validid
# Print out remaining track
if [ -s temp_validid ]
then
gawk '(NR==FNR){arr[$1];next} !($1 in arr){print $0}' temp_validid $input1 > temp_dummy
cp temp_dummy temp_remainingtracks
```

```
# Assign valid id the flow direction
gawk -v procedure=$procedure '(NR==FNR){arr[$1]; next} ($1 in arr){$23=procedure}1'
temp_validid $input2 > temp_assignedids
cp temp_assignedids $input2
else
# If there are no ILS approach then copy the input1 file to temp_remainingtracks
cp $input1 temp_dummy2
cp temp_dummy2 temp_remainingtracks
fi
# gawk -v direction=$direction '(NR==FNR){arr[$1];next} ($1 in arr){print $0,direction}'
temp_validid $input2 >> temp_assignedids
else
##out1=$infile1"_"$airport"_"$type"rwy_dir_pro"
##cp temp_assignedids $out1
# Assign Visual approach to track that have not been assigned ILS or RNP
gawk -v runway=$runway -v direction=$direction '{if($21==runway && $22==direction) print
$1,$23}' $input2 > temp_ids_procedure
gawk '{if($2==1) print $0}' temp_ids_procedure > temp_ids_notassigned
if [ -s temp_ids_notassigned ]
then
gawk -v procedure="Visual" '(NR==FNR){arr[$1]; next} ($1 in arr){$23=procedure}1'
temp_ids_notassigned $input2 > temp_assignedids
cp temp_assignedids $input2
fi
fi
```

File name: script_assign_flow_procedure0.gawk

```
# This script executes script_assign_flow_procedure.gawk
# Input
infile1=$1
recno=$2
airport=$3
type=$4
direction=$5
runway=$6
flowinfo=$7

if [ $recno -eq 1 ]
then
# Create a file to record the procedure for each track
gawk '{print $0,1}' $infile1"_"$airport"_"$type"rwy_dir" >
$infile1"_"$airport"_"$type"rwy_dir_pro"
## test
##cp $infile1"_"$airport"_"$type"rwy_dir_pro" temp_basefile
fi


# ## Filter out flow information along with the lat lon of the fix around which the flow
is to be filtered
gawk -v airport=$airport -v type=$type -v runway=$runway -v direction=$direction
'{if($1==airport && $2==type && $3==direction && $4==runway){i++;print i,$0}} END{print
0,airport,type,direction,runway}' $flowinfo > temp_flowinfo
## Assign flow direction
gawk -v infile1=$infile1 '{execute="time ./script_assign_flow_procedure.gawk "infile1"
"$0; print execute; system(execute)}' temp_flowinfo
```

File name: script_assign_runway.gawk

```
# This script assign each track to a runway
#
arr=$1
infile1=$2
runwayinfo=$3
airport=$4
lat=$5
```

```
lon=$6

# Step1 : Get the first two track points in case of departures and last two in case of
arrivals for each flight
if [ $arr -eq 1 ]
then
input1=$infile1"_"$airport"_Atracks"
out1=$infile1"_"$airport"_Atracksrwy"
gawk '{if($1!=x){x=$1; if(FNR!=1){print arr[i-1],arr[i]}; i=1; delete arr; arr[i]=$0}
else{i++; arr[i]=$0}} END{print arr[i-1],arr[i]}' $input1 > temp_1record
else
input1=$infile1"_"$airport"_Dtracks"
out1=$infile1"_"$airport"_Dtracksrwy"
gawk '{if($1!=x){x=$1; if(FNR!=1){print arr[1],arr[2]}; i=1; delete arr; arr[i]=$0}
else{i++; arr[i]=$0}} END{print arr[1],arr[2]}' $input1 > temp_1record
fi

# Filter out runway co-ordinates for airport of interest
gawk -v airport=$airport '{if($1==airport) print $6,$7,$8,$9,$10,$11}' $runwayinfo >
temp_runway_coordinates


# Assign runway to each track
gawk -v arr=$arr -v lat=$lat -v lon=$lon -f function.getrunway_2 -f function.getcartesian
-f function.bearing -f function.radian -f function.degrees -f function.dist_point_line -f
function.absvalue temp_runway_coordinates temp_1record > temp_runway
# Output runway assignments
cp temp_runway $out1
# print out kml file
# Get the color information for each runway
# get runways
gawk '{if($NF!="NA") print $NF}' temp_runway | sort -u > temp_runway2
# get runway info
if [ $arr -eq 1 ]
then
gawk -v airport=$airport '(NR==FNR){arr[$1];next} ($6 in arr){if($1==airport) print
$6,$12}' temp_runway2 $runwayinfo > temp_runwayinfo1
gawk -v airport=$airport '(NR==FNR){arr[$1];next} ($7 in arr){if($1==airport) print
$7,$14}' temp_runway2 $runwayinfo > temp_runwayinfo2
cat temp_runwayinfo1 temp_runwayinfo2 > temp_runwayinfo3
else
gawk -v airport=$airport '(NR==FNR){arr[$1];next} ($6 in arr){if($1==airport) print
$6,$13}' temp_runway2 $runwayinfo > temp_runwayinfo1
gawk -v airport=$airport '(NR==FNR){arr[$1];next} ($7 in arr){if($1==airport) print
$7,$15}' temp_runway2 $runwayinfo > temp_runwayinfo2
cat temp_runwayinfo1 temp_runwayinfo2 > temp_runwayinfo3
fi

# print kml file
gawk -v input1="temp_runway" -v input2=$input1 '{execute="./script_print_kml_byrwy.gawk
"input1" "input2" "$1" "$2; print execute; system(execute)}' temp_runwayinfo3

# remove temp file
rm temp*
```

File name: script_combine_NOP_files.gawk

```
# This files combines all NOP processed data
# Input
infile1=$1
recno=$2
all=$3
airport=$4
type=$5
custom=$6
```

```
if [ "MDW" == "$airport" ]
then
if [ $custom -eq 1 ]
then
if [ $recno -eq 1 ]
then
rm NOP_data_all_good_custom
fi
input=$infile1
out1="NOP_data_all_good_custom"

else
if [ $recno -eq 1 ]
then
if [ $all -eq 1 ]
then
rm NOP_data_all
else
rm NOP_data_all_good
fi
fi

if [ $all -eq 1 ]
then
out1="NOP_data_all"
input=$infile1"_"$airport"_"$type"rwy"
else
out1="NOP_data_all_good"
input=$infile1"_"$airport"_"$type"rwy_dir_pro"
fi
fi

###if [ "MDW" == "$airport" ]
###then
gawk -v infile1=$infile1 '{print infile1,$0}' $input | gawk '{if($24==1){if($22=="13L" ||
$22=="4L" || $22=="22L" || $22=="22R" || $22=="31R"){$24="Visual"} else{$24="SA"}}}1' >>
$out1
fi

if [ "ORD" == "$airport" ]
then
if [ $recno -eq 1 ]
then
rm NOP_data_all_good_ORD
fi

out1="NOP_data_all_good_ORD"
input=$infile1"_"$airport"_"$type"rwy"

gawk -v infile1=$infile1 '{print infile1,$0}' $input >> $out1

fi
```

File name: script_combine_flow_stats.gawk

```
# This script computes stats for combined flows
# input files
input=$1
MC=$2 # (values are 0-VMC, 1-IMC(ILS), 2-IMC(RNP))

infile1="NOP_"$input"_stats"
infile2="NOP_"$input"_stats_custom"

out1="NOP_"$input"_stats_combined_"$MC
```

```
# Get required records
if [ $MC -eq 0 ]
then
cat $infile1 $infile2 | gawk '{if($1!="NA") print $0}' | gawk '{if($3=="Visual" ||
$3=="SA") print $0}' | sort -k2,2 > temp_infile
else
if [ $MC -eq 1 ]
then
cat $infile1 $infile2 | gawk '{if($1!="NA") print $0}' | gawk '{if($3=="ILS" || $3=="SA")
print $0}' | sort -k2,2 > temp_infile
else
cat $infile1 $infile2 | gawk '{if($1!="NA") print $0}' | gawk '{if($3=="RNP" || $3=="SA")
print $0}' | sort -k2,2 > temp_infile
fi
fi

# Get stats for each runway by combining flows.
gawk -f function.combine_flow_stats temp_infile > $out1
```

File name: script_combine_fuelburn_actypes.gawk

```
# This script combines fuel burn for all aircraft types for various flows
# Input1 aircraft types
ac_cat=$1
percentage=$2

# runways
r1=$3
r2=$4
r3=$5
r4=$6

input="NOP_fuelburn_"$ac_cat"_stats"

gawk -v percentage=$percentage -v r1=$r1 -v r2=$r2 -v r3=$r3 -v r4=$r4 '{if($1!="NA" &&
$2~/('"$r1"'|'"$r2"'|'"$r3"'|'"$r4"')/ && $4>10) print $0,percentage}' $input >>
temp_actypes3
```

File name: script_combine_fuelburn_actypes0.gawk

```
# This script combines fuel burn for all aircraft types for various flows
# This script is only for MDW arrivals
# Input1 aircraft types
actype="aircraft_cat.dat"
infile="NOP_data_all_good"

# runways
r1=$1
r2=$2
r3=$3
r4=$4

# Get percentage of flights under each aircraft category
gawk '(NR==FNR){arr[$1]=$5; next} ($29 in arr){print $29,arr[$29]}' $actype $infile >
temp_actypes1

gawk '{{arr[$2]++; count++}} END{for(no in arr){print no,arr[no],arr[no]/count}}'
temp_actypes1 | sort -k3,3nr > temp_actypes2

# Combine files
# The output of the script is temp_actypes3
rm temp_actypes3
```

```
gawk -v r1=$r1 -v r2=$r2 -v r3=$r3 -v r4=$r4
'{execute="./script_combine_fuelburn_actypes.gawk "$1" "$3" "r1" "r2" "r3" "r4; print
execute; system(execute)}' temp_actypes2

# Caculate average fuel burn per flight for each flow
sort -k1,1 -k2,2 -k3,3 temp_actypes3 | gawk '{if($1" "$2"
"$3!=x){if(FNR!=1){if(count>=2){print x,flights,"NA","NA",fb,"NA"}}; x=$1" "$2" "$3;
count=1; flights=0; flights=flights+$4; fb=0; fb=fb+$7*$9} else{count++;
flights=flights+$4; fb=fb+$7*$9}} END{if(count>=2){print x,flights,"NA","NA",fb,"NA"}}' >
temp_actypes4
```

File name: script_compute_NOP_stats.gawk

```
# This script compute stats for processed NOP data
# For MDW Arrivals
# Input
filelist="NOP_filelist.dat"
out1="NOP_MDW_Atracks_flow_stats.dat"
bada="BADA_coefficient_2.dat"
aircraft_category="aircraft_cat.dat"

# Combine all the files

# Good tracks for MDW Atracks
# # The output file name is NOP_data_all_good
# Don't have the run this unless script_process_tracks_2.gawk has been modified and run
# MDW_arrivals
gawk -v all=0 -v airport="MDW" -v type="Atracks" -v custom=0
'{execute="./script_combine_NOP_files.gawk "$1" "FNR" "all" "airport" "type" "custom;
print execute; system(execute)}' $filelist
# ORD departures
# The output file name is NOP_data_all_good_ORD
gawk -v all=0 -v airport="ORD" -v type="Dtracks" -v custom=0
'{execute="./script_combine_NOP_files.gawk "$1" "FNR" "all" "airport" "type" "custom;
print execute; system(execute)}' $filelist


# MDW flow stats analysis
# mark track with noise that had not been filtered
gawk '{if($2=="20110426181228C902654SWA709"){print "#",$0} else{print $0}}'
NOP_data_all_good > temp
cp temp NOP_data_all_good

# Filter out general aviation flights
gawk '{if($30!="NA") print $0}' NOP_data_all_good > NOP_data_all_good_2

# Get count of all the flows
gawk -v level=3 '{if($1!="#"){arr[$23" "$22" "$24]++}} END{for(no in arr) print
level,no,arr[no]}' NOP_data_all_good_2 | sort -k3,3 -k4,4 > temp_flow_counts3
gawk '{print $0} END{print 0,"NA","NA","NA","NA"}' temp_flow_counts3 > temp_flow_counts

# Compute statistics for each flow
# Track mile stats (field no 26)
gawk -v fieldno=26 -v input="NOP_data_all_good_2" -v GA=0
'{execute="./script_get_min_max_mean_std.gawk "FNR" "input" "$1" "$2" "$3" "$4" "$5"
"fieldno" "GA; print execute; system(execute)}' temp_flow_counts
##sort -k1,1 -k2,2 -k3,3 temp_stats_out NOP_trackmile_stats
cp temp_stats_out NOP_trackmile_stats

# Track mile stats (field no 27)
gawk -v fieldno=27 -v input="NOP_data_all_good_2" -v GA=0
'{execute="./script_get_min_max_mean_std.gawk "FNR" "input" "$1" "$2" "$3" "$4" "$5"
"fieldno" "GA; print execute; system(execute)}' temp_flow_counts
##sort -k1,1 -k2,2 -k3,3 temp_stats_out > NOP_tracktime_stats
cp temp_stats_out NOP_tracktime_stats
```

```
# fuel burn stats (field no 30)
gawk -v fieldno=30 -v input="NOP_data_all_good_2" -v GA=0
'{execute="./script_get_min_max_mean_std.gawk "FNR" "input" "$1" "$2" "$3" "$4" "$5"
"fieldno" "GA; print execute; system(execute)}' temp_flow_counts
##sort -k1,1 -k2,2 -k3,3 temp_stats_out > NOP_tracktime_stats
cp temp_stats_out NOP_fuelburn_stats

# level time stats (field no 31)
gawk -v fieldno=31 -v input="NOP_data_all_good_2" -v GA=0
'{execute="./script_get_min_max_mean_std.gawk "FNR" "input" "$1" "$2" "$3" "$4" "$5"
"fieldno" "GA; print execute; system(execute)}' temp_flow_counts
##sort -k1,1 -k2,2 -k3,3 temp_stats_out > NOP_tracktime_stats
cp temp_stats_out NOP_leveltime_stats

# Get goaround stats
# Track mile stats (field no 26)
gawk -v fieldno=26  -v input="NOP_data_all_good_2" -v GA=1
'{execute="./script_get_min_max_mean_std.gawk "FNR" "input" "$1" "$2" "$3" "$4" "$5"
"fieldno" "GA; print execute; system(execute)}' temp_flow_counts
# sort -k1,1 -k2,2 -k3,3 temp_stats_out > NOP_trackmile_stats_GA
cp temp_stats_out NOP_trackmile_stats_GA

# Track mile stats (field no 27)
gawk -v fieldno=27  -v input="NOP_data_all_good_2" -v GA=1
'{execute="./script_get_min_max_mean_std.gawk "FNR" "input" "$1" "$2" "$3" "$4" "$5"
"fieldno" "GA; print execute; system(execute)}' temp_flow_counts
# sort -k1,1 -k2,2 -k3,3 temp_stats_out > NOP_tracktime_stats_GA
cp temp_stats_out NOP_tracktime_stats_GA

### Get distribution aircraft types

#
# Get the engine type and MTOW for each aircraft
gawk '{print $0,"NA","NA"}' NOP_data_all_good > temp_engine_MTOW_1
gawk '(NR==FNR){mtow[$1]=$19;engine[$1]=$20;next} ($29 in mtow){$33=mtow[$29];
$34=engine[$29]}1' $bada temp_engine_MTOW_1 > temp_engine_MTOW_2
# Get distribution by a/ctype
gawk '{if($25=="NA") print $0}' temp_engine_MTOW_2 | sort -k29,29 | gawk -f
function.fb_actype -f function.min_max_mean_sd_3 -f function.mean -f function.std | sort
-k4,4nr > distribution_actype

# Compute fuel burn for each aircraft category
gawk '{print $5}' $aircraft_category | sort -u > temp_ac_cat
gawk -v infile="NOP_data_all_good_2" -v actypes=$aircraft_category
'{execute="./script_compute_fuelburn_bycat.gawk "infile" "actypes" "$1; print execute;
system(execute)}' temp_ac_cat
```

File name: script_compute_fb_new_rnp.gawk

```
# This script computes track distance/time and fuel burn for the new rnp tracks generated
by reflection and rotation of 13C RNP tracks
# get track info for all the valid ids
# input variables
infile1=$1
badadata=$2
recno=$3
airport=$4
type=$5
direction=$6
fix=$7
lat1=$8
lon1=$9
lat2=${10}
lon2=${11}
check_radius=${12}
lat0=${13}
lon0=${14}
```

```
alat=${15}
alon=${16}
elev=${17}
windinfo=${18}

## Set input and output filenames
input1="tracks_"$infile1

out1="metrics_"$infile1


if [ $recno -eq 1 ]
then
# Create a file to record the procedure for each track
cp $out1 temp_out
# Two fields for track mile and track time
# Three fields for airline name, ac type and fuel burn
gawk '{print $0,"NA","NA","NA","NA","NA","NA","NA","NA","NA"}' temp_out > $out1
fi

# Get track info close to the airport
gawk -v fix_lat=$alat -v fix_lon=$alon -v check_radius=$check_radius -f
function.distairport -f function.gcd_haversine_2 $input1 | sort -k1,1 -k2,2n >
temp_validtracks1a

# Compute cumulative distance to threshold
gawk -f function.dist2thresh -f function.gcd_haversine_2 temp_validtracks1a >
temp_validtracks2

# Compute distance from fix
gawk -v fix_lat=$lat0 -v fix_lon=$lon0 -v lat1=$lat1 -v lon1=$lon1 -v lat2=$lat2 -v
lon2=$lon2 -v scale=3 -f function.dist_point_line_2 -f function.getcartesian_2 -f
function.absvalue -f function.distfromfix2 temp_validtracks2 > temp_validtracks3

# Get the distance from fix to threshold
gawk -f function.distimefix2thresh_2 -f function.absvalue temp_validtracks3 >
temp_validtracks4


# Update output file with distance and time to threshold from corner post
cp $out1 temp_out
gawk '(NR==FNR){d2t[$1]=$2; t2t[$1]=$3; next} ($1 in d2t){$25=d2t[$1]; $26=t2t[$1]}1'
temp_validtracks4 temp_out > $out1

#########################################################################################
##################################
# Fuel burn calculation
# Fuel burn is a function of track mile flow and hence had to be computed for each
direction

# Update airline name and ac type in the output file
if [ $recno -eq 1 ]
then
# Get airline code and ac type for each flight
###gawk -F"," '{if($4!="") print $1,substr($3,1,3),$4}' $input4 | sort -u >
temp_airline_ac_type
gawk '(NR==FNR){arr[$2]=$28" "$29; next} ($1 in arr){print $1,arr[$1]}' NOP_data_all_good
$input1 > temp_airline_ac_type
cp $out1 temp_out
gawk '(NR==FNR){arr1[$1]=$2; arr2[$1]=$3; next} ($1 in arr1){$27=arr1[$1];
$28=arr2[$1]}1' temp_airline_ac_type temp_out > $out1
fi

# Get information about the 4D profile
# Get combine wind information with track data
# Get the min and max timestamp
```

```
###sort -k2,2n temp_validtracks3 | gawk '{if(FNR==1){first=$2}; last=$2} END{print
first,last}' > temp_timestamp_first_last
# Get wind information for valid timestamp range
###gawk '{if(NR==FNR){first=$1-15*60; last=$2+15*60}
else{if(FNR!=1){if($6>=first){if($6<=last){print $0} else{exit}}}}}'
temp_timestamp_first_last $windinfo > temp_valid_windinfo
# Combine wind information with track data in temp_validtracks3
###gawk '{if(NR==FNR){arr1[FNR]=$6;arr2[FNR]=$11" "$12}
else{for(i=1;i<=length(arr1);i++){if($2>arr1[i] && $2<=arr1[i]+15*60){print
$0,arr2[i]}}}}' temp_valid_windinfo temp_validtracks3 > temp_validtracks3b

# Get wind information
# get the first time stamp for each flight
gawk '{if($1!=x){print $0; x=$1}}' temp_validtracks3 | sort -k2,2n > temp_validtracks3_1
# Get wind mag and direction
gawk 'BEGIN{x=1}{if(NR==FNR){arr1[FNR]=$1; arr2[FNR]=$2}
else{if(x<=length(arr1)){for(i=x;i<=length(arr1);i++){if(arr2[i]>=$6 &&
arr2[i]<$6+15*60){print arr1[i],$11,$12; x++} else{break}}} else{exit}}}'
temp_validtracks3_1 $windinfo > temp_validtracks3_2
gawk '(NR==FNR){arr[$1]=$2" "$3; next} ($1 in arr){print $0, arr[$1]}'
temp_validtracks3_2 temp_validtracks3 > temp_validtracks3b


# For  each time step, compute the time increment,change in altitude,distance,velocity,
acceleration
gawk -v elev=$elev -f function.bearing -f function.degrees -f function.radian -f
function.computeTAS -f function.get_track_profile_info -f function.gcd_haversine_2 -f
function.absvalue temp_validtracks3b > temp_validtracks5
# Assign ac type to each track
gawk '(NR==FNR){arr[$1]=$3; next} ($1 in arr){print $0, arr[$1]}' temp_airline_ac_type
temp_validtracks5 >  temp_validtracks6

# Compute fuel burn for each flight
gawk -v elev=$elev -f function.absvalue -f function.computefuelburn1 -f
function.airdensity $badadata temp_validtracks6 > temp_validtracks7
cp temp_validtracks7 temp_validtracks7_$direction

# Get fuel burn per flight for the flow and for the level and final segments
# specify start of the final as start of final approach in case of MDW it is 1700 feet
gawk -v final=1700 -f function.fuelburn_levelsegment -f function.absvalue
temp_validtracks7 > temp_validtracks8


# Update output file
cp $out1 temp_out
gawk '(NR==FNR){fb2t[$1]=$2; ls2t[$1]=$3; fbl2t[$1]=$4; fs_t[$1]=$7; fs_fb[$1]=$8; next}
($1 in fb2t){$29=fb2t[$1]; $30=ls2t[$1]; $31=fbl2t[$1]; $32=fs_t[$1]; $33=fs_fb[$1]}1'
temp_validtracks8 temp_out > $out1
```

File name: script_compute_fb_new_rnp0.gawk

```
# This script computes track distance/time and fuel burn for the new rnp tracks generated
by reflection and rotation of 13C RNP tracks

# get track distance, track time and fuelburn statistics for each RNP flow
# input
airport=$1
direction=$2
runway=$3
procedure=$4

# initialize input files
airportinfo="airport_info.dat"
runwayinfo="runway_info.dat"
flowdirection="flow_direction.dat"
flowinfo="flow_info_all.dat"
```

```
fixinfo="MDW_fix.dat"
goaround_holding="goaround_holding.dat"
flowmeasuringfix="flow_measuring_fix.dat"
badadata="BADA_coefficient_2.dat"
windinfo1="MDW_wind_info.dat"

# input file
infile1=$direction"_"$runway"_"$procedure

input1="tracks_"$infile1

out1="metrics_"$infile1

# Next few script reform track information to NOP output format (does this makes sense??)
# get the last two hits for each track
gawk '{if($1!=x){x=$1; if(FNR!=1){print arr[i-1],arr[i]}; i=1; delete arr; arr[i]=$0}
else{i++; arr[i]=$0}} END{print arr[i-1],arr[i]}' $input1 > temp_1record
cp temp_1record $out1

# print the runway, direction and procedure
gawk -v runway=$runway -v direction=$direction -v procedure=$procedure '{print
$0,runway,direction,procedure,"NA"}' $out1 > temp_out1
cp temp_out1 $out1


# Get lat lon the fix from where distance needs to be computed
gawk '(NR==FNR){lat[$1]=$2;lon[$1]=$3;next} ($4 in lat){if($1!="#"){print
$0,lat[$4],lon[$4]}}}' $fixinfo $flowmeasuringfix > temp_flowfix_0
# Get lat lon the airport to filter out track with some check radius
gawk '(NR==FNR){lat[$1]=$2;lon[$1]=$3;alt[$1]=$6;next} ($1 in lat){if($1!="#"){print
$0,lat[$1],lon[$1],alt[$1]}}}' $airportinfo temp_flowfix_0 > temp_flowfix

# get flow reference point for the direction
gawk -v direction=$direction -v airport=$airport '{if($1==airport && $3==direction){print
$0}}' temp_flowfix > temp_flowfix2

# Compute fuel burn for all tracks
gawk -v infile1=$infile1 -v badadata=$badadata -v windinfo=$windinfo1 '{execute="time
./script_compute_fb_new_rnp.gawk "infile1" "badadata" "FNR" "$0" "windinfo; print
execute; system(execute)}' temp_flowfix2

# This part is temp addition needs to be removed later on
cut -d" " -f 1-7,9-21 temp_validtracks7 > temp_mitre_$infile1.txt
```

File name: script_compute_fuelburn_bycat.gawk

```
# This file computes fuel burn statistics for each aircraft category
# inputs
# 1
infile=$1
# 2
actypes=$2
# 3
category=$3


# get valid actype for each category

gawk -v category=$category '{if($5==category) print $1}' $actypes > temp_valid_ac_types

# filter out valid records from the NOP filelist
gawk '(NR==FNR){arr[$1]; next} ($29 in arr){print $0}' temp_valid_ac_types $infile >
temp_valid_records_1

# Compute fuel burn statistics
```

```
gawk '{if($1!="#"){if($30!="NA"){print $0,$31/$27}}}' temp_valid_records_1 >
temp_all_good_valid_actype
# Get stats for specific fields
# flight fuel burn rate for level, non level and final approach
gawk '{if($25=="NA"){if($31>0 && $35>0){print $0,$32/$31,$34/$33,$36/$35} else{if($31<=0
&& $35<=0){print $0,0,$34/$33,0} else{if($31>0){print $0,$32/$31,$34/$33,0} else{print
$0,0,$34/$33,$36/$35}}}}}' temp_all_good_valid_actype | gawk -v field1=38 -v field2=39 -v
field3=40 -f function.min_max_mean_sd_3 -f function.std -f function.mean -f
function.stats_fld > fb_level_nonlevel_stats"_"$category

#  Get plots for track time vs fuel burn and level time vs fuel burn
### track time vs level time track time ratio curve
###gawk -v category=$category -v infile="temp_all_good_valid_actype" -v timefield=27 -v
fbfield=33 -v GA=0 '{execute="./script_get_time_vs_fuelburn.gawk "infile" "timefield"
"fbfield" "GA" "FNR" "$2" "$3" "$4" "category; print execute; system(execute)}'
temp_flow_counts3
if [ "$category" == "B737s" ]
then
# level time track time ratio vs fuel burn
gawk -v category=$category -v infile="temp_all_good_valid_actype" -v timefield=37 -v
fbfield=30 -v GA=0 '{execute="./script_get_time_vs_fuelburn.gawk "infile" "timefield"
"fbfield" "GA" "FNR" "$2" "$3" "$4" "category; print execute; system(execute)}
END{execute="./script_get_time_vs_fuelburn.gawk "infile" "timefield" "fbfield" "GA" 0 all
all all "category; print execute; system(execute)}' temp_flow_counts3

# level time track time ratio vs track time
gawk -v category=$category -v infile="temp_all_good_valid_actype" -v timefield=37 -v
fbfield=27 -v GA=0 '{execute="./script_get_time_vs_fuelburn.gawk "infile" "timefield"
"fbfield" "GA" "FNR" "$2" "$3" "$4" "category; print execute; system(execute)}
END{execute="./script_get_time_vs_fuelburn.gawk "infile" "timefield" "fbfield" "GA" 0 all
all all "category; print execute; system(execute)}' temp_flow_counts3

# level time track time ratio vs level time
gawk -v category=$category -v infile="temp_all_good_valid_actype" -v timefield=37 -v
fbfield=31 -v GA=0 '{execute="./script_get_time_vs_fuelburn.gawk "infile" "timefield"
"fbfield" "GA" "FNR" "$2" "$3" "$4" "category; print execute; system(execute)}
END{execute="./script_get_time_vs_fuelburn.gawk "infile" "timefield" "fbfield" "GA" 0 all
all all "category; print execute; system(execute)}' temp_flow_counts3

# level time track time ratio vs fuel burn
gawk -v category=$category -v infile="temp_all_good_valid_actype" -v timefield=37 -v
fbfield=32 -v GA=0 '{execute="./script_get_time_vs_fuelburn.gawk "infile" "timefield"
"fbfield" "GA" "FNR" "$2" "$3" "$4" "category; print execute; system(execute)}
END{execute="./script_get_time_vs_fuelburn.gawk "infile" "timefield" "fbfield" "GA" 0 all
all all "category; print execute; system(execute)}' temp_flow_counts3

# level time track time ratio vs final approach time
gawk -v category=$category -v infile="temp_all_good_valid_actype" -v timefield=37 -v
fbfield=33 -v GA=0 '{execute="./script_get_time_vs_fuelburn.gawk "infile" "timefield"
"fbfield" "GA" "FNR" "$2" "$3" "$4" "category; print execute; system(execute)}
END{execute="./script_get_time_vs_fuelburn.gawk "infile" "timefield" "fbfield" "GA" 0 all
all all "category; print execute; system(execute)}' temp_flow_counts3

# level time track time ratio vs final approach fuel burn
gawk -v category=$category -v infile="temp_all_good_valid_actype" -v timefield=37 -v
fbfield=34 -v GA=0 '{execute="./script_get_time_vs_fuelburn.gawk "infile" "timefield"
"fbfield" "GA" "FNR" "$2" "$3" "$4" "category; print execute; system(execute)}
END{execute="./script_get_time_vs_fuelburn.gawk "infile" "timefield" "fbfield" "GA" 0 all
all all "category; print execute; system(execute)}' temp_flow_counts3

# level time track time ratio vs non-level time
gawk -v category=$category -v infile="temp_all_good_valid_actype" -v timefield=37 -v
fbfield=35 -v GA=0 '{execute="./script_get_time_vs_fuelburn.gawk "infile" "timefield"
"fbfield" "GA" "FNR" "$2" "$3" "$4" "category; print execute; system(execute)}
END{execute="./script_get_time_vs_fuelburn.gawk "infile" "timefield" "fbfield" "GA" 0 all
all all "category; print execute; system(execute)}' temp_flow_counts3
```

```
# level time track time ratio vs non-level fuel burn
gawk -v category=$category -v infile="temp_all_good_valid_actype" -v timefield=37 -v
fbfield=36 -v GA=0 '{execute="./script_get_time_vs_fuelburn.gawk "infile" "timefield"
"fbfield" "GA" "FNR" "$2" "$3" "$4" "category; print execute; system(execute)}
END{execute="./script_get_time_vs_fuelburn.gawk "infile" "timefield" "fbfield" "GA" 0 all
all all "category; print execute; system(execute)}' temp_flow_counts3
fi


# Get fuel burn stats
gawk -v fieldno=30 -v input="temp_all_good_valid_actype" -v GA=0
'{execute="./script_get_min_max_mean_std.gawk "FNR" "input" "$1" "$2" "$3" "$4" "$5"
"fieldno" "GA; print execute; system(execute)}' temp_flow_counts
cp temp_stats_out NOP_fuelburn_$category"_stats"
# Get Level time stats
gawk -v fieldno=31 -v input="temp_all_good_valid_actype" -v GA=0
'{execute="./script_get_min_max_mean_std.gawk "FNR" "input" "$1" "$2" "$3" "$4" "$5"
"fieldno" "GA; print execute; system(execute)}' temp_flow_counts
cp temp_stats_out NOP_leveltime_$category"_stats"
# Get track time stats
gawk -v fieldno=27 -v input="temp_all_good_valid_actype" -v GA=0
'{execute="./script_get_min_max_mean_std.gawk "FNR" "input" "$1" "$2" "$3" "$4" "$5"
"fieldno" "GA; print execute; system(execute)}' temp_flow_counts
cp temp_stats_out NOP_tracktime2_$category"_stats"
```

File name: script_compute_stats_new_rnp.gawk

```
# This script computes stats for the user defined rnp approach tracks
#
# initialize input file
rnp_gen="rnp_gen_info.dat"
# create file list
gawk '{print "metrics_"$2"_"$3"_"$6}' $rnp_gen > temp_filelist

filelist="temp_filelist"

# combine all the files
gawk -v all=0 -v airport="MDW" -v type="Atracks" -v custom=1
'{execute="./script_combine_NOP_files.gawk "$1" "FNR" "all" "airport" "type" "custom;
print execute; system(execute)}' $filelist

# Get count of all the flows
gawk -v level=3 '{if($1!="#"){arr[$23" "$22" "$24]++}} END{for(no in arr) print
level,no,arr[no]}' NOP_data_all_good_custom | sort -k3,3 -k4,4 > temp_flow_counts

# Compute statistics for each flow
# Track mile stats (field no 26)
gawk -v fieldno=26 -v input="NOP_data_all_good_custom" -v GA=0
'{execute="./script_get_min_max_mean_std.gawk "FNR" "input" "$1" "$2" "$3" "$4" "$5"
"fieldno" "GA; print execute; system(execute)}' temp_flow_counts
##sort -k1,1 -k2,2 -k3,3 temp_stats_out NOP_trackmile_stats
cp temp_stats_out NOP_trackmile_stats_custom

# Track mile stats (field no 27)
gawk -v fieldno=27 -v input="NOP_data_all_good_custom" -v GA=0
'{execute="./script_get_min_max_mean_std.gawk "FNR" "input" "$1" "$2" "$3" "$4" "$5"
"fieldno" "GA; print execute; system(execute)}' temp_flow_counts
##sort -k1,1 -k2,2 -k3,3 temp_stats_out > NOP_tracktime_stats
cp temp_stats_out NOP_tracktime_stats_custom

# This is code is only for B73's
### gawk '{if($1!="#"){if($30!="NA" && $29~"B73"){if($31>0){print
$0,$31/$27,$30/$27,$32/$31,($30-$32)/($27-$31)} else{print $0,$31/$27,$30/$27,"NA",($30-
$32)/($27-$31)}}}}' NOP_data_all_good_custom > temp_all_good_valid_actype_2
```

```
gawk '{if($1!="#"){if($30!="NA"){print $0,$27-($31+$33),$30-($32+$34),$31/$27}}}'
NOP_data_all_good_custom > temp_all_good_valid_actype_2

# Get fuel burn stats
gawk -v fieldno=30 -v input="temp_all_good_valid_actype_2" -v GA=0
'{execute="./script_get_min_max_mean_std.gawk "FNR" "input" "$1" "$2" "$3" "$4" "$5"
"fieldno" "GA; print execute; system(execute)}' temp_flow_counts
cp temp_stats_out NOP_fuelburn_stats_custom
# Get Level time stats
gawk -v fieldno=31 -v input="temp_all_good_valid_actype_2" -v GA=0
'{execute="./script_get_min_max_mean_std.gawk "FNR" "input" "$1" "$2" "$3" "$4" "$5"
"fieldno" "GA; print execute; system(execute)}' temp_flow_counts
cp temp_stats_out NOP_leveltime_stats_custom
# Get track time stats
gawk -v fieldno=27 -v input="temp_all_good_valid_actype_2" -v GA=0
'{execute="./script_get_min_max_mean_std.gawk "FNR" "input" "$1" "$2" "$3" "$4" "$5"
"fieldno" "GA; print execute; system(execute)}' temp_flow_counts
cp temp_stats_out NOP_tracktime2_stats_custom
```

File name: script_compute_track_mile_time_fuel.gawk

```
# This script computes the track mile and track time for each track from the fix to the
runway threshold
# Input
infile1=$1
badadata=$2
recno=$3
airport=$4
type=$5
direction=$6
fix=$7
lat1=$8
lon1=$9
lat2=${10}
lon2=${11}
check_radius=${12}
lat0=${13}
lon0=${14}
alat=${15}
alon=${16}
elev=${17}
windinfo=${18}

# Set input file name
input1=$infile1"_"$airport"_"$type
input2=$infile1"_"$airport"_"$type"rwy_dir_pro"
input3=$infile1"_"$airport"_"$type"_GA_ids"
input4=$infile1".csv"

# Set output filename
out1=$infile1"_"$airport"_"$type"rwy_dir_pro"

if [ $recno -eq 1 ]
then
# Create a file to record the procedure for each track
cp $out1 temp_out
# Two fields for track mile and track time
# Three fields for airline name, ac type and fuel burn
gawk '{print $0,"NA","NA","NA","NA","NA","NA","NA","NA","NA","NA","NA"}' temp_out > $out1
## test
##cp $infile1"_"$airport"_"$type"rwy_dir_pro" temp_basefile
fi

# Filter out relevant flight ids
# Get all flight for the given direction that have been assigned to a flow
gawk -v direction=$direction '{if($22==direction) print $1}' $input2 > temp_valids1
```

```
## Remove ids that are goarounds
##if [ -s $input3 ]
##then
##gawk '(NR==FNR){arr[$1]; next} !($1 in arr){print $0}' $input3 temp_validids1 >
temp_validids2
##else
cp temp_validids1 temp_validids2
##fi

# get track info for all the valid ids
gawk '(NR==FNR){arr[$1];next} ($1 in arr){print $0}' temp_validids2 $input1 >
temp_validtracks1

# Get track info close to the airport
gawk -v fix_lat=$alat -v fix_lon=$alon -v check_radius=$check_radius -f
function.distairport -f function.gcd_haversine_2 temp_validtracks1 | sort -k1,1 -k2,2n >
temp_validtracks1a
# Filter out flight above 8000 ft for flows from the east and 12000ft for flows from the
west
gawk -v direction=$direction '{if(direction=="E"){if($9<=8000) print $0}
else{if($9<=12000) print $0}}' temp_validtracks1a > temp_validtracks1b

# Compute cumulative distance to threshold
gawk -f function.dist2thresh -f function.gcd_haversine_2 temp_validtracks1b >
temp_validtracks2

# Compute distance from fix
gawk -v fix_lat=$lat0 -v fix_lon=$lon0 -v lat1=$lat1 -v lon1=$lon1 -v lat2=$lat2 -v
lon2=$lon2 -v scale=3 -f function.dist_point_line_2 -f function.getcartesian_2 -f
function.absvalue -f function.distfromfix2 temp_validtracks2 > temp_validtracks3

# Get the distance from fix to threshold
gawk -f function.distimefix2thresh_2 -f function.absvalue temp_validtracks3 >
temp_validtracks4


# Update output file with distance and time to threshold from corner post
cp $out1 temp_out
gawk '(NR==FNR){d2t[$1]=$2; t2t[$1]=$3; next} ($1 in d2t){$25=d2t[$1]; $26=t2t[$1]}1'
temp_validtracks4 temp_out > $out1

#######################################################################################
##################################
# Fuel burn calculation
# Fuel burn is a function of track mile flow and hence had to be computed for each
direction

# Update airline name and ac type in the output file
if [ $recno -eq 1 ]
then
# Get airline code and ac type for each flight
gawk -F"," '{if($4!="") print $1,substr($3,1,3),$4}' $input4 | sort -u >
temp_airline_ac_type
cp $out1 temp_out
gawk '(NR==FNR){arr1[$1]=$2; arr2[$1]=$3; next} ($1 in arr1){$27=arr1[$1];
$28=arr2[$1]}1' temp_airline_ac_type temp_out > $out1
fi

# Get information about the 4D profile
# Get combine wind information with track data
# Get the min and max timestamp
sort -k2,2n temp_validtracks3 | gawk '{if(FNR==1){first=$2}; last=$2} END{print
first,last}' > temp_timestamp_first_last
# Get wind information for valid timestamp range
gawk '{if(NR==FNR){first=$1-15*60; last=$2+15*60}
else{if(FNR!=1){if($6>=first){if($6<=last){print $0} else{exit}}}}}'
temp_timestamp_first_last $windinfo > temp_valid_windinfo
```

```
# Combine wind information with track data in temp_validtracks3
gawk '{if(NR==FNR){arr1[FNR]=$6;arr2[FNR]=$11" "$12}
else{for(i=1;i<=length(arr1);i++){if($2>arr1[i] && $2<=arr1[i]+15*60){print
$0,arr2[i]}}}}' temp_valid_windinfo temp_validtracks3 > temp_validtracks3b
# For  each time step, compute the time increment,change in altitude,distance,velocity,
acceleration
gawk -v elev=$elev -f function.bearing -f function.degrees -f function.radian -f
function.computeTAS -f function.get_track_profile_info -f function.gcd_haversine_2 -f
function.absvalue temp_validtracks3b > temp_validtracks5
cp temp validtracks5 temp validtracks5_$direction
# Assign ac type to each track
gawk '(NR==FNR){arr[$1]=$3; next} ($1 in arr){print $0, arr[$1]}' temp_airline_ac_type
temp_validtracks5 >  temp_validtracks6

# Compute fuel burn for each flight
gawk -v elev=$elev -f function.absvalue -f function.computefuelburn1 -f
function.airdensity $badadata temp_validtracks6 > temp_validtracks7
cp temp_validtracks7 temp_validtracks7_$direction

# Get fuel burn per flight for the flow and for the level and final segments
# specify start of the final as start of final approach in case of MDW it is 1700 feet
gawk -v final=1700 -f function.fuelburn_levelsegment -f function.absvalue
temp_validtracks7 > temp_validtracks8


# Update output file
cp $out1 temp_out
gawk '(NR==FNR){fb2t[$1]=$2; ls2t[$1]=$3; fbl2t[$1]=$4; fs_t[$1]=$7; fs_fb[$1]=$8;
nl_t[$1]=$5; nl_fb[$1]=$6; next} ($1 in fb2t){$29=fb2t[$1]; $30=ls2t[$1]; $31=fbl2t[$1];
$32=fs_t[$1]; $33=fs_fb[$1]; $34=nl_t[$1]; $35=nl_fb[$1]}1' temp_validtracks8 temp_out >
$out1
```

File name: script_compute_track_mile_time_fuel0.gawk

```
# This script executes script_compute_track_mile_time.gawk
# Input
infile1=$1
flowfix=$2
badadata=$3
windinfo=$4
##if [ $recno -eq 1 ]
##then
# Create a file to record the procedure for each track
##gawk '{print $0,1,"NA","NA"}' $infile1"_"$airport"_"$type"rwy_dir" >
$infile1"_"$airport"_"$type"rwy_dir_pro"
## test
##cp $infile1"_"$airport"_"$type"rwy_dir_pro" temp_basefile
##fi


## Compute track mile and track time for each flow direction
gawk -v infile1=$infile1 -v badadata=$badadata -v windinfo=$windinfo '{execute="time
./script_compute_track_mile_time_fuel.gawk "infile1" "badadata" "FNR" "$0" "windinfo;
print execute; system(execute)}' $flowfix

###gawk -v infile1=$infile1 -v badadata=$badadata -v windinfo=$windinfo '{execute="time
./script_compute_track_mile_time_fuel_2.gawk "infile1" "badadata" "FNR" "$0" "windinfo;
print execute; system(execute)}' $flowfix




# File name: script_detect_goaround_holding.gawk
# This script detects go arounds and holding patterns
# Step1: Calculate bearing between two consecutive points
# Step2: Calcute change in bearing
# Step3: Add up change in bearing
# Step4: If above threshold then classify as go around or holding pattern
#
# Input
infile1=$1
airport=$2
type=$3
check_radius=$4
lat=$5
lon=$6

# infile
input1=$infile1"_"$airport"_"$type
input2=$infile1"_"$airport"_"$type"rwy_dir_pro"

# output file
out=$infile1"_"$airport"_"$type"rwy_dir_pro"
out1=$infile1"_"$airport"_"$type"_GA_all"
out2=$infile1"_"$airport"_"$type"_GA_ids"

out3=$infile1"_"$airport"_"$type"_HOL_all"
out4=$infile1"_"$airport"_"$type"_HOL_ids"

# filter out track with valid runway assignment
gawk '{if($21!="NA"){print$1}}' $input2 > temp_assignedids
# Get track information for all valid tracks
gawk '(NR==FNR){arr[$1];next} ($1 in arr){print $0}' temp_assignedids $input1 >
temp_validtracks


# Get track info close to the airport
gawk -v fix_lat=$lat -v fix_lon=$lon -v check_radius=$check_radius -f
function.distairport -f function.gcd_haversine_2 temp_validtracks | sort -k1,1 -k2,2n >
temp_tracks_close

# Get tracks away from the airport
gawk '(NR==FNR){arr[$1" "$2]; next} !($1" "$2 in arr){print $0}' temp_tracks_close
temp_validtracks | sort -k1,1 -k2,2n > temp_tracks_away

# Compute turn angle for each flight
gawk -f function.bearing -f function.absvalue -f function.goaround_holding -f
function.radian -f function.degrees temp_tracks_close > temp_turn_angle1
# Assign direction runway and procedure type to each track ID
gawk '(NR==FNR){arr[$1]=$21$22$23; next} ($1 in arr){print $0, arr[$1]}' $input2
temp_turn_angle1 > $out1
# Get goaround ids
###gawk -v threshold1=330 -v threshold2=450 '{if($1!=x){x=$1; alt=$9; gacheck=0;
if(FNR!=1){if(rwydirpro~"13CE"){threshold=threshold2} else{threshold=threshold1};
if((turnangle1*-1)>threshold || turnangle2>threshold || gacheck==1){print
id,turnangle1,turnangle2,diff,gacheck}}} else{alt_diff=$9-alt; alt=$9;
if(alt_diff>=300){gacheck=1};id=$1; turnangle1=$13; turnangle2=$14; diff=$15;
rwydirpro=$16}}' $out1 > $out2

gawk -v threshold=360 '{if($1!=x){if(FNR!=1){if((turnangle1*-1)>threshold ||
turnangle2>threshold || gacheck==1){print id,turnangle1,turnangle2,diff,gacheck}}; x=$1;
alt=$9; gacheck=0} else{alt_diff=$9-alt; alt=$9; if(alt_diff>=300){gacheck=1};id=$1;
turnangle1=$13; turnangle2=$14; diff=$15; rwydirpro=$16}} END{if((turnangle1*-
1)>threshold || turnangle2>threshold){if(gacheck==1){print
id,turnangle1,turnangle2,diff,gacheck}}}' $out1 > $out2

# Update output file
if [ -s $out2 ]
then
cp $out temp_out
gawk '(NR==FNR){arr[$1]="GA"; next} ($1 in arr){$24=arr[$1]}1' $out2 temp_out > $out
fi
```

```
# Get holding tracks
#gawk -f function.bearing -f function.absvalue -f function.goaround_holding -f
function.radian -f function.degrees temp_tracks_away > $out3
# Get holding ids
#gawk -v threshold=200 '{if($1!=x){x=$1; if(FNR!=1){if(turnangle<0){turnangle=turnangle*-
1}; if(turnangle>threshold){print id,turnangle}}} else{id=$1; turnangle=$13}}' $out3 >
$out4
```

File name: script_detect_goaround_holding0.gawk

```
# This script executes the script_detect_goaround_holding.gawk
#
# Input
infile1=$1
recno=$2
airport=$3
type=$4
goaround_holding=$5
airportinfo=$6


if [ $recno -eq 1 ]
then
# Create a file to record the procedure for each track
cp $infile1"_"$airport"_"$type"rwy_dir_pro" temp_out
gawk '{print $0,"NA"}' temp_out > $infile1"_"$airport"_"$type"rwy_dir_pro"
## test
##cp $infile1"_"$airport"_"$type"rwy_dir_pro" temp_basefile
fi


##airport is the airport of interest type is Arrival tracks (Atracks) or Departure tracks
(Dtracks)
gawk -v airport=$airport -v type=$type '{if($1==airport && $2==type){print $0}}'
$goaround_holding > temp_goaround_holding2
## get the lat lon information of the airport of interest
gawk '(NR==FNR){lat[$1]=$2;lon[$1]=$3;next} ($1 in lat){print $0,lat[$1],lon[$1]}'
$airportinfo temp_goaround_holding2 > temp_goaround_holding3
## Assign flow direction
gawk -v infile1=$infile1 '{execute="time ./script_detect_goaround_holding.gawk "infile1"
"$0; print execute; system(execute)}' temp_goaround_holding3



# File name: script_filter_holding_patterns.gawk
# This script filters holding patterns from the track data
# Input
recno=$1
runway=$2
direction=$3
fix=$4
lon1=$5
lat1=$6
lon2=$7
lat2=$8
lon3=$9
lat3=${10}
lon4=${11}
lat4=${12}

# get infile
infile="tracks_"$runway"_"$direction

if [ $recno -eq 1 ]
then
```

```
rm temp_tracks_rectangle

fi

# Filter valid tracks
gawk -v lat1=$lat1 -v lon1=$lon1 -v lat2=$lat2 -v lon2=$lon2 -v lat3=$lat3 -v lon3=$lon3
-v lat4=$lat4 -v lon4=$lon4 -v scale=5 -v fix=$fix -f function.pointinrectangle -f
function.getcartesian_2 $infile >> temp_tracks_rectangle
```

File name: script_get_BADA_coefficients.gawk

```
# This script tabulates BADA coefficients required for fuelburn calculation for all BADA
flights
# Input is the flight type and recno
flighttype=$1
recno=$2
# Output file
out1="BADA_coefficient.dat"

# get the name of the BADA file
if [ ${#flighttype} -eq 4 ]
then
infile=$flighttype"__.OPF"
else
infile=$flighttype"___.OPF"
fi

# Remove out file before the first file is run
if [ $recno -eq 1 ]
then
rm $out1
fi

# Get coefficients fot each flight type
gawk -v flighttype=$flighttype -f function.badacoefficient $infile >> $out1
```

File name: script_get_BADA_coefficients0.gawk

```
# This script runs script_get_BADA_coefficient.gawk
# input is BADA ac type list
infile="BADA_actypes.dat"

# Get co-efficients from BADA OPF files
gawk '{execute="./script_get_BADA_coefficients.gawk "$1" "FNR; print execute;
system(execute)}' $infile

# Get co-efficients for non-BADA aircrafts
gawk 'BEGIN{execute="./script_get_non_BADA_coefficients.gawk BADA_coefficient.dat
NOP_data_all_good actype_nonBADA_MTOW.dat"; print execute; system(execute)}'
```

File name: script_get_FIX_NAV_latlon.gawk

```
# This script processes FIX and NAV data to get lat lon info in google earth format
#
# STAR and APPROACH info
infile="MDW_STARS_approach"
# Combine FIX and NAV file
cat FIX.txt NAV.txt > temp_FIX_NAV

# Get required information
```

```
## INFO start and length
# FIX
# NAME 5,30
# lat 67,14
# lon 81,14
# NAV
# NAME 5,4
# TYPE 9,20
# lat 372,14
# lon 397,14
gawk 'BEGIN{OFS="\t"} {if(substr($0,1,4)=="FIX1"){print
substr($0,1,4),substr($0,5,5),"NONE",substr($0,67,12),substr($0,81,13)};
if(substr($0,1,4)=="NAV1"){print
substr($0,1,4),substr($0,5,3),substr($0,9,7),substr($0,372,12),substr($0,397,13)}}'
temp_FIX_NAV > temp_FIX_NAV_2

# Convert lat lon format
gawk 'function sumarr(arr){out=0; out=arr[1]+arr[2]/60+arr[3]/3600; return out}
BEGIN{FS=OFS="\t"} {split($4,a1,"-");sumarr(a1);$4=out;split($5,a1,"-");sumarr(a1); $5=-
1*out}1' temp_FIX_NAV_2 > temp_FIX_NAV_3

# Get lat lon info for each fix or navaid in "MDW_STARS_approach.dat"
gawk 'BEGIN{FS=OFS="\t"} (NR==FNR){arr[$2]=$4"\t"$5; next} ($1 in arr){$2=arr[$1]}1'
temp_FIX_NAV_3 $infile".dat" > $infile

# Generate kml file
color1="brown"
color2="yellow"

gawk -v filename=$infile -v linecolor1=$color1 -v linecolor2=$color2 -f function.getcolor
-f function.kmlfile_2 $infile > $infile".kml"

# Get all the fixes and navaids and put them in kml format
gawk 'BEGIN{FS=OFS="\t"} (NR==FNR){arr[$1]; next} ($2 in arr){print $0}' $infile".dat"
temp_FIX_NAV_3 > temp_FIX_NAV_4
color1="white"
color2="purple"
gawk -v filename=$infile -v linecolor1=$color1 -v linecolor2=$color2 -f function.getcolor
-f function.kmlfile_3 temp_FIX_NAV_4 > $infile"_fix_nav.kml"
```

File name: script_get_fb_15min_actual_config.gawk

```
# This script computes average fuel burn per flight given the east west flow ratio and
the runway configuration
#
# Input 1, flow scope
flow_scope=$1
# Input 2, ASPM data
infile=$2
# Input 3, flow fuel burn
fb_flow=$3

# Filter out flows
if [ $flow_scope -eq 1 ]
then
# Flow scope 1, existing visual and ILS flows
gawk '{if($2=="ILS" || $2=="Visual") print $1,$2,$4,$6}' $fb_flow > temp_fb_flow
else
# Flow scope 2, existing visual, ISL and RNP flows
if [ $flow_scope -eq 2 ]
then
gawk '{if($2=="ILS" || $2=="Visual" || $1$2=="13CRNP") print $1,$2,$4,$6}' $fb_flow >
temp_fb_flow
else
gawk '{print $1,$2,$4,$6}' $fb_flow > temp_fb_flow
```

```
fi
fi

# Print average fuel burn for each record
gawk '{if($2=="Visual"){print "V",$0} else{if($2=="ILS"){print "I",$0} else{print "I",$0;
print "V",$0}}}' temp_fb_flow > temp_fb_flow_2
gawk -v flow_scope=$flow_scope -f function.radian -f function.absvalue -f
function.getfbrwyconfig1 temp_fb_flow_2 $infile > temp_fb_allrecords
cp temp_fb_allrecords $infile
```

File name: script_get_fb_all_flows.gawk

```
# This script brings together fuel burn for all flows
# Actual data and simulated data
# For B737s and for all jets combined.
#
# Print fuel burn for new rnp flows with other flows for B737s
# Inputs
fb_stats_b737_existing="NOP_fuelburn_B737s_stats"
fb_stats_b737_new="NOP_fuelburn_stats_custom"

# This script is valid for only MDWs major runways
r1="13C"
r2="31C"
r3="4R"
r4="22L"

# print stats for all flows for B737s
gawk -v r1=$r1 -v r2=$r2 -v r3=$r3 -v r4=$r4 '{if($1!="NA" &&
$2~/('"$r1"'|'"$r2"'|'"$r3"'|'"$r4"')/) print $0 }' $fb_stats_b737_existing >
temp_stats_b737s

cat $fb_stats_b737_new temp_stats_b737s | sort -k1,1 -k3,3 -k7,7 > temp_stats_b737s_2

# Compute avearge fuel for each flow for all jet types combined (output file
temp_actypes4)
gawk -v r1=$r1 -v r2=$r2 -v r3=$r3 -v r4=$r4
'BEGIN{execute="./script_combine_fuelburn_actypes0.gawk "r1" "r2" "r3" "r4; print
execute; system(execute)}'

# Compute percentage reduction in fuel burn
gawk '(NR==FNR){arr[$1" "$2" "$3]=$7; next} ($1" "$2" "$3 in arr){print $0, arr[$1" "$2"
"$3], (arr[$1" "$2" "$3]-$7)/arr[$1" "$2" "$3]}' temp_stats_b737s_2 temp_actypes4 >
temp_actypes5

# Get the average fraction reduction in average fuel burn per flight per flow
gawk '{sum+=$10; count++} END{print sum/count}' temp_actypes5 > temp_b737_all_fraction

# Get fuel burn for all flows B737 and all aircrafts combined
gawk '{if(NR==FNR){factor=$1} else{print $0,$7*(1-factor)}}' temp_b737_all_fraction
temp_stats_b737s_2 > MDW_all_flows_mean_fb
```

File name: script_get_first_last_hit.gawk

```
# This script filters NOP data and get the firt and last hit for each track
#
infile1=$1

input1=$infile1".csv"
out1=$infile1"_all_filtered"
out2=$out1".kml"
out3=$infile1"_first_last"
```

```
# Print out required fields from the NOP data
gawk -F"," 'function round(x1){sec=substr(x1,1,2); dec=substr(x1,4,1);
if(dec>=.5){out=sec+1} else{out=sec}; if(length(out)==2){return out} else{return "0"out}}
{if(NF==11){if(substr($3,1,3)!="VFR" &&
substr($3,1,3)!="UNK"){if(substr($1,1,4)/2!=0){if($6==""){org="NA"} else{org=$6};
if($7==""){des="NA"} else{des=$7}; time1="NR"; print
$1,time1,org,des,substr($11,1,length($11)-7)":"round(substr($11,length($11)-
5,4)),$8,$9,$10*100}}}}' $input1 > temp_all1

# Remove record with less than 10 data points
gawk '{arr[$1]++} END{for(no in arr){if(arr[no]>=10){print no, arr[no]}}}' temp_all1 >
temp_valid_rec
gawk '(NR==FNR){arr[$1]; next} ($1 in arr){print $0}' temp_valid_rec temp_all1 | TZ=UTC
gawk '{$2=mktime(substr($5,1,4)" "substr($5,6,2)" "substr($5,9,2)" "substr($6,1,2)"
"substr($6,4,2)" "substr($6,7,2))}1' > temp_all2

# Get the first and last point of each track
gawk '{if($1!=x){x=$1; if(FNR!=1){print record,lat,lon,alt}; record=$0}
else{lat=$7;lon=$8;alt=$9}} END{print record,lat,lon,alt}' temp_all2 > temp_first_last

# output files
# output temp_all2 in kml format
cp temp_all2 $out1
# print kml file
if [ -s temp_all2 ]
then
    gawk -v filename=$out1 -v linecolor="darkgreen" -f function.getcolor -f
function.kmlfile_1 $out1 > $out2
fi
# output the first and last stamp for each track
cp temp_first_last $out3

# remove temp files
rm temp*
```

File name: script_get_holding_patterns.gawk

```
# This script detects holding patterns and get stats on fuel burn and time on holding
# Input1 List of flights
input1="NOP_data_all_good_2"
# Input2 list of
input2="NOP_filelist.dat"

# Input 3 holding pattern fix info
holding_fix="MDW_star_holding_fix.dat"

# Input4 runway, direction and aircraft type
runway="13C"
app="ILS"
direction1="E"
direction2="W"
actype="B73"
airport="MDW"

# Input 5 BADA and wind info
badadata="BADA_coefficient_2.dat"
windinfo1="MDW_wind_info.dat"

# output files
out1="tracks_"$runway"_"$direction1
out2="tracks_"$runway"_"$direction2
out3="tracks_holding_"$runway

# Time range to considered
# Time range
time_start=7
time_end=22
# Specify the time zone to compute local time
tcurrent=-5
tzone=-6


firstrun=0

if [ $firstrun -eq 1 ]
then

# Get tracks for runway 13C
# change type to NA for normal tracks and GA for tracks with excess turn angle
## gawk  -v runway=$runway -v type="NA" -v actype=$actype '{if($22==runway &&
$29~/^'"$actype"'/){print $2,$23,$29}}' $input1 > temp_test1
# filter out valid record
# Get day with ILS approaches on to 13C (indicative of IMC)
gawk  -v runway=$runway -v app=$app '{if($22==runway && $24==app){print $1}}' $input1 |
sort -u > temp_validfiles
gawk '(NR==FNR){arr[$1]; next} ($1 in arr){print $0}' temp_validfiles $input1 >
temp_valid_records1

gawk -v time_start=$time_start -v time_end=$time_end -v tcurrent=$tcurrent -v
tzone=$tzone 'BEGIN{tdiff=tcurrent-tzone} {hr_local=strftime("%H",$13); hr_zone=hr_local-
tdiff; if(hr_zone>=time_start && hr_zone<=time_end) print $2,$23,$29}'
temp_valid_records1 > temp_valid_records2
gawk '(NR==FNR){arr[$1];next} ($2 in arr){print $0}' temp_valid_records2
temp_valid_records1 > NOP_summary_valid_records

gawk  -v runway=$runway -v app=$app '{if($22==runway && $24==app){print $2,$23,$29}}'
NOP_summary_valid_records > temp_test1


# remove temp_test2
rm temp_test2

# get track info for ids in temp_test1
# The output of this script in temp_test2 containing the track info
gawk -v flightids="temp_test1" '{execute="./script_print_error_tracks.gawk "$1"
"flightids; print execute; system(execute)}' $input2

# Copy temp_test2 to another file along with the direction information
# Printing out track from the east
gawk -v direction=$direction1 '(NR==FNR){if($2==direction){arr[$1]; next}} ($1 in
arr){print $0}' temp_test1 temp_test2 > $out1
# print temp_test2 in kml format
gawk -v out=$out1 -v color="red" 'BEGIN{execute="./script_print_kml.gawk "out" "color;
print execute; system(execute)}'

# Printing out track from the east
gawk -v direction=$direction2 '(NR==FNR){if($2==direction){arr[$1]; next}} ($1 in
arr){print $0}' temp_test1 temp_test2 > $out2
# print temp_test2 in kml format
gawk -v out=$out2 -v color="red" 'BEGIN{execute="./script_print_kml.gawk "out" "color;
print execute; system(execute)}'

# Filter out track hits and detect holding patterns that fall within the holding pattern
boundary defined in MDW_star_holding_fix.dat
# out out file temp_tracks_rectangle
# gawk get list of valid holding patterns boundaries
gawk '{if($1!="#") print $0}' $holding_fix > temp_holding_fix
gawk -v runway=$runway '{execute="./script_filter_holding_patterns.gawk "FNR" "runway"
"$0; print execute; system(execute)}' temp_holding_fix

# Get the cumulative turn angle for each flight
```

```
gawk -f function.bearing -f function.absvalue -f function.goaround_holding -f
function.radian -f function.degrees temp_tracks_rectangle > temp_turn_angle1

gawk -v threshold=360 '{if($1!=x){if(FNR!=1){if((turnangle1*-1)>threshold ||
turnangle2>threshold){print x,fix}}; x=$1; fix=$11; turnangle1=0; turnangle2=0}
else{if($14!="Start" && $15!="Start"){turnangle1=$14; turnangle2=$15}}}
END{if((turnangle1*-1)>threshold || turnangle2>threshold){print x,fix}}' temp_turn_angle1
> temp_turn_angle2

# print out holding patterns
gawk '(NR==FNR){arr[$1" "$2]; next} ($1" "$11 in arr){print $0}' temp_turn_angle2
temp_turn_angle1 > $out3

# Print kml file
gawk -v out=$out3 -v color="red" 'BEGIN{execute="./script_print_kml.gawk "out" "color;
print execute; system(execute)}'

# Compute holding stats
# time and fuel burn
gawk -v infile1=$out3 -v runway=$runway -v airport=$airport -v
infile2="NOP_summary_valid_records" -v badadata=$badadata -v windinfo=$windinfo1
'BEGIN{execute="time ./script_get_holding_stats.gawk "infile1" "badadata" "windinfo"
"infile2" "runway" "airport; print execute; system(execute)}'


else
gawk 'BEGIN{y="do nothing"}'

# Compute holding stats
# time and fuel burn
gawk -v infile1=$out3 -v runway=$runway -v airport=$airport -v
infile2="NOP_summary_valid_records" -v badadata=$badadata -v windinfo=$windinfo1
'BEGIN{execute="time ./script_get_holding_stats.gawk "infile1" "badadata" "windinfo"
"infile2" "runway" "airport; print execute; system(execute)}'


fi
```

File name: script_get_holding_stats.gawk

```
# This script computes holding stats, time in holding, fuel burn
# Input
infile=$1
badadata=$2
windinfo=$3
nop_summary=$4
runway=$5
airport=$6

# Output file names
out1=$airport"_"$runway"_holdings_metrics"
out2=$airport"_"$runway"_holding_stats"

# specify rwy and approach type that cause the metroplex flow conflict
rwy="13C"
app="ILS"
app2="RNP"
out3=$airport"_arr_"$rwy"_"$app
out4=$airport"_holdingratio_"$rwy"_"$app
# Specify the time zone to compute local time
tcurrent=-5
tzone=-6


# Compute holding pattern track distance
# Filter required fields, and bring the input file in the standard format for functions
can be used
cut -d" " -f1-10 $infile > temp_holdingtracks1
gawk -f function.dist2thresh -f function.gcd_haversine_2 temp_holdingtracks1 >
temp_validtracks2

# Distance from fix does not apply, enter a dummy field
gawk '{print $0, "NA"}' temp_validtracks2 > temp_validtracks3

# Get wind information
# get the first time stamp for each flight
gawk '{if($1!=x){print $0; x=$1}}' temp_validtracks3 | sort -k2,2n > temp_validtracks3_1
# Get wind mag and direction
gawk 'BEGIN{x=1}{if(NR==FNR){arr1[FNR]=$1; arr2[FNR]=$2}
else{if(x<=length(arr1)){for(i=x;i<=length(arr1);i++){if(arr2[i]>=$6 &&
arr2[i]<$6+15*60){print arr1[i],$11,$12; x++} else{break}}} else{exit}}}'
temp_validtracks3_1 $windinfo > temp_validtracks3_2
gawk '(NR==FNR){arr[$1]=$2" "$3; next} ($1 in arr){print $0, arr[$1]}'
temp_validtracks3_2 temp_validtracks3 > temp_validtracks3b

# For  each time step, compute the time increment,change in altitude,distance,velocity,
acceleration
gawk -v elev=$elev -f function.bearing -f function.degrees -f function.radian -f
function.computeTAS -f function.get_track_profile_info -f function.gcd_haversine_2 -f
function.absvalue temp_validtracks3b > temp_validtracks5
# Assign ac type to each track
gawk '(NR==FNR){arr[$2]=$29; next} ($1 in arr){print $0, arr[$1]}' $nop_summary
temp_validtracks5 >  temp_validtracks6

# Compute fuel burn for each flight
gawk -v holding=1 -v elev=$elev -f function.absvalue -f function.computefuelburn1 -f
function.airdensity $badadata temp_validtracks6 > temp_validtracks7

# For each aircraft in holding get the start and end time, total time, distance and fuel
burnt in holding.
gawk '{if($1!=x){if(FNR!=1){print x,actype,tstart,tend,(tend-tstart)/60,tdistance,tfb};
x=$1; tstart=$2; actype=$18} else{tend=$2;tdistance=$9;tfb=$21}} END{print
x,actype,tstart,tend,(tend-tstart)/60,tdistance,tfb}' temp_validtracks7 >
temp_validtracks7b
# Check if 22L departure happen while holding
gawk '(NR==FNR){arr[$2]=$1; next} ($1 in arr){print arr[$1],$0,strftime("%H",$3)}'
$nop_summary temp_validtracks7b | sort -k1,1 -k4,4n > temp_holdingids1
# get the list of files (days) that need to be examined
gawk '{print $1}' temp_holdingids1 | sort -u > temp_holdingids2
# Check if holding and departure coincided
gawk -v deprwy="22L" -v depairport="ORD" -v infile="temp_holdingids1"
'{execute="./script_holding_check.gawk "FNR" "$1" "infile" "deprwy" "depairport; print
execute; system(execute)}' temp_holdingids2
# Print out valid holding tracks
gawk '{if($10=="NA" && $11=="NA" && $12=="NA"){y="do nothing"} else{print $0}}'
temp_holdingids1 > temp_holdingidsall
gawk '(NR==FNR){arr[$2];next} ($1 in arr){print $0}' temp_holdingidsall
temp_validtracks7b > $out1


# Compute the mean and standard deviation for holding metrics
gawk -v actype="all" -f function.holdingstats -f function.mean -f function.std $out1 >
temp_stats1
gawk -v actype="B73" -f function.holdingstats -f function.mean -f function.std $out1 >
temp_stats2
cat temp_stats1 temp_stats2 | gawk 'BEGIN{print "actype count mean_htime sd_htime
mean_hdis sd_hdis mean_hfb sd_hfb"}{print $0}' > $out2

# Get ratio of total arrivals on to runway 13C during IMC (ILS approach) and total number
of holding patterns
# The ratio is computed per event
# Event is defined by the start and stop of a flow
# In case of MDW its the start and stop of ILS approach on to 13C
sort -k1,1 -k3,3n $nop_summary | gawk -v rwy=$rwy -v app=$app -v app2=$app2
'{if($1$22$24!=x1){if(FNR!=1){if(app1==app || app1==app2){if(rwy==rwy1 && count>1){y++;
print y,x2,tstart,tend,(tend-tstart)/3600,count,x3}}}; x1=$1$22$24;
x2=substr($1,length($1)-19,8); x3=$1; tstart=$3; rwy1=$22; app1=$24; count=0; count++}
else{tend=$3; count++}} END{if(app1==app || app1==app2){if(rwy==rwy1 && count>1){y++;
print y,x2,tstart,tend,(tend-tstart)/3600,count,x3}}}' > temp_arr_count1
# Get 22L departure count for each bin that has arrival onto 13C during IMC
# get list of files to get the counts from
### gawk '{print $7}' temp_arr_count1 | sort -u > temp_filelist
# get the buffer between time windows
gawk '{if($7!=x){x=$7;tend=$4;print $0,0} else{print $0,$3-tend;tend=$4}}'
temp_arr_count1 > temp_arr_count2
gawk -v deprwy="22L" -v depairport="ORD" -v infile="temp_arr_count1"
'{execute="./script_mplex_arr_dep_count.gawk "FNR" "$3" "$4" "$7" "$8" "infile" "deprwy"
"depairport; print execute; system(execute)}' temp_arr_count2

# Sum up the arrival for each day
gawk '{if(FNR==1){x=$2}; if($2!=x){print x,hr,arr,dep; x=$2; hr=$5; arr=$6; dep=$8;
count=1} else{hr+=$5; arr+=$6; dep+=$8}} END{print x,hr,arr,dep}' temp_arr_count1 > $out3

# get count of holding pattern for each day
gawk '(NR==FNR){arr[$2]=$1;next} ($1 in arr){print arr[$1],$0}' $nop_summary $out1 |gawk
'{arr[substr($1,length($1)-19,8)]++} END{for(no in arr) print no, arr[no]}' >
temp_holding_day1
# print the holding count along with duration of each day
gawk '(NR==FNR){arr[$1]=$2; next} ($1 in arr){print $0, arr[$1]}' temp_holding_day1 $out3
> temp_event_duration_2
gawk '(NR==FNR){arr[$1]=$2; next} !($1 in arr){print $0, 0}' temp_holding_day1 $out3 >
temp_event_duration_3
cat temp_event_duration_2 temp_event_duration_3 | gawk 'BEGIN{print "Date Durtn_Hr
13C_Arr 22L_Dep Holding_count Holdingper100"}{if($2>1) print $0,$5*100/$3}' > $out4
```

File name: script_get_min_max_mean_std.gawk

```
# This script get stats on each flow
# Input
recno=$1
input=$2
level=$3
direction=$4
runway=$5
procedure=$6
count=$7
fieldno=$8
GA=$9

if [ $level -eq 0 ]
then
# Combine flows by approach and runway by giving equal weight to each flow
# Get stats by approach
gawk '{if($2$3!=x){if(FNR!=1){m1=0;s1=0;
for(i=1;i<=length(arr_mean);i++){s1=s1+count[i];m1=m1+arr_mean[i]*(1/length(arr_mean))};
v1=0; for(i=1;i<=length(arr_sd);i++){v1=v1+(1/length(arr_sd))*(arr_sd[i]^2+(arr_mean[i]-
m1)^2)};sd1=sqrt(v1); print "NA",runway,approach,s1,"NA","NA",m1,sd1}; x=$2$3; i=1;delete
arr_mean; delete arr_sd; delete count; count[i]=$4; arr_mean[i]=$7;
arr_sd[i]=$8;runway=$2; approach=$3} else{i++; count[i]=$4; arr_mean[i]=$7;
arr_sd[i]=$8}} END{m1=0;s1=0;
for(i=1;i<=length(arr_mean);i++){s1=s1+count[i];m1=m1+arr_mean[i]*(1/length(arr_mean))};
v1=0; for(i=1;i<=length(arr_sd);i++){v1=v1+(1/length(arr_sd))*(arr_sd[i]^2+(arr_mean[i]-
m1)^2)};sd1=sqrt(v1); print "NA",runway,approach,s1,"NA","NA",m1,sd1}' temp_stats_out >
temp_stats_out_approach
# Get stats by runway
gawk '{if($2!=x){if(FNR!=1){m1=0;s1=0;
for(i=1;i<=length(arr_mean);i++){s1=s1+count[i];m1=m1+arr_mean[i]*(1/length(arr_mean))};
v1=0; for(i=1;i<=length(arr_sd);i++){v1=v1+(1/length(arr_sd))*(arr_sd[i]^2+(arr_mean[i]-
m1)^2)};sd1=sqrt(v1); print "NA",runway,"NA",s1,"NA","NA",m1,sd1}; x=$2; i=1;delete
arr_mean; delete arr_sd; delete count; count[i]=$4; arr_mean[i]=$7;
arr_sd[i]=$8;runway=$2; approach=$3} else{i++; count[i]=$4; arr_mean[i]=$7;
arr_sd[i]=$8}} END{m1=0;s1=0;
for(i=1;i<=length(arr_mean);i++){s1=s1+count[i];m1=m1+arr_mean[i]*(1/length(arr_mean))};
v1=0; for(i=1;i<=length(arr_sd);i++){v1=v1+(1/length(arr_sd))*(arr_sd[i]^2+(arr_mean[i]-
m1)^2)};sd1=sqrt(v1); print "NA",runway,"NA",s1,"NA","NA",m1,sd1}' temp_stats_out >
temp_stats_out_runway
# Combine all
cp temp_stats_out temp_stats_out_b
cat temp_stats_out_runway temp_stats_out_approach temp_stats_out_b > temp_stats_out

else
if [ $recno -eq 1 ]
then
if [ $GA -eq 0 ]
then
# remove go arounds from the data
gawk '{if($25!="GA") print $0}' $input > temp_input1
else
# keep only go around data
gawk '{if($25=="GA") print $0}' $input > temp_input1
fi
rm temp_stats_out
fi

# Compute and print out the stats for each flow
gawk -v direction=$direction -v level=$level -v runway=$runway -v procedure=$procedure -v
fieldno=$fieldno -f function.getflowstats -f function.min_max_mean_sd_3 -f function.mean
-f function.std temp_input1 >> temp_stats_out

fi
```

File name: script_get_new_rnp_approach.gawk

```
# This script uses the existing RNP approach track data to generate RNP data for other
runways
# input parameters
airport=$1
new_rnp_rwy=$3
base_rnp_rwy=$5
rotate=$7
reflect=$8
alat=$9
alon=${10}
rwylat1=${11}
rwylon1=${12}
rwylat2=${13}
rwylon2=${14}

# input track file
infile="tracks_"$4"_"$5"_"$6

# outfile
out="tracks_"$2"_"$3"_RNP"

# get track points within 60NM of the airport
gawk -v fix_lat=$alat -v fix_lon=$alon -v check_radius=60 -f function.gcd_haversine_2 -f
function.distairport $infile > temp_rnp_track_1

# reflect and rotote tracks
gawk -v alat=$alat -v alon=$alon -v rwylat1=$rwylat1 -v rwylon1=$rwylon1 -v
rwylat2=$rwylat2 -v rwylon2=$rwylon2 -v rotate=$rotate -v reflect=$reflect -v
baserwy=$base_rnp_rwy -v newrwy=$new_rnp_rwy -v scale=0 -f function.reflectrotate -f
function.getcartesian_2 -f function.getlatlon -f function.radian temp_rnp_track_1 >
temp_rnp_track_2
```

```
# Copy track to output
cp temp_rnp_track_2 $out
# print kml file
#
```

File name: script_get_new_rnp_approach0.gawk

```
# This script runs script_get_new_rnp_approach.gawk for all the records in
rnp_gen_info.dat
#
#
runway_info="runway_info.dat"

airport_info="airport_info.dat"

rnp_gen="rnp_gen_info.dat"

airport="MDW" # Make sure to match this with the airport of the which the data need to be
processed

# Get information about the rnp procedure to be generated for the airport
gawk -v airport="MDW" '{if($1==airport){print $0}}' $rnp_gen > temp_rnp_gen_1

# get airport coordinates
gawk '(NR==FNR){arr[$1]=$2" "$3; next} ($1 in arr){print $0,arr[$1]}' $airport_info
temp_rnp_gen_1> temp_rnp_gen_2

# get the runway coordinates
gawk -v airport="MDW" '{if(NR==FNR){if($1==airport){rwy1[FNR]=$6;coord1[FNR]=$8" "$9"
"$10" "$11; rwy2[FNR]=$7;coord2[FNR]=$10" "$11" "$8" "$9}}
else{for(i=1;i<=length(rwy1);i++){if($5==rwy1[i]){print $0,coord1[i]; break}
else{if($5==rwy2[i]){print $0,coord2[i]; break}}}}}' $runway_info temp_rnp_gen_2 >
temp_rnp_gen_3

# run script_get_new_rnp_approach.gawk
gawk '{if($1!="#"){execute="./script_get_new_rnp_approach.gawk "$0; print execute;
system(execute)}}' temp_rnp_gen_3

# get flow metrics, track distance/time and fuel burn for new RNP flow
gawk '{if($1!="#"){execute="./script_compute_fb_new_rnp0.gawk "$1" "$2" "$3" "$6; print
execute; system(execute)}}' temp_rnp_gen_3

# get stats for the metrics
gawk 'BEGIN{execute="./script_compute_stats_new_rnp.gawk"; print execute;
system(execute)}'
```

File name: script_get_non_BADA_coefficients.gawk

```
# This script get co-efficient for non BADA aircraft type
# input files
# Input1 BADA coefficients
badadata=$1
# input2 actypes in NOP data
nopactype=$2
# input3 actype and MTOW
actypemtow=$3

# output file
out="BADA_coefficient_2.dat"

## remove bada actypes which do not have coefficient values namely cf1,cf2,cf3 and cf4
gawk '{if($2!=0 && $3!=0 && $4!=0 && $5!=0) print $0}' $badadata > temp_badagood

# Get aircraft type not in BADA
```

```
gawk '{if($29!="NA") print $29}' $nopactype | sort -u > temp_actypes_nop

# Get ac type that are not in bada
gawk '(NR==FNR){arr[$1];next} !($1 in arr){print $0}' temp_badagood temp_actypes_nop >
temp_actype_nonbada
## old code not to be used
###gawk '{if($30=="NA"){arr[$29]++}; sum1++} END{for(no in arr){print no,
arr[no],arr[no]/sum1,"NA"}}' NOP_data_all_good | sort -k1,1 > temp_actype_nonBADA
### Get MTOW for each aircraft type from actype_performance_faa.dat
###gawk '(NR==FNR){if($4=="LBS"){arr[$1]=$5*0.453592} else{arr[$1]=$5}; next} ($1 in
arr){$4=arr[$1]}1' aircraft_performance.dat temp_actype_nonBADA >
temp_actype_nonBADA_MTOW

# get the MTOW for each non bada aircraft
gawk '(NR==FNR){mtow[$1]=$2;engine[$1]=$3; next} ($1 in mtow){print
$1,mtow[$1],engine[$1]}' $actypemtow temp_actype_nonbada > temp_actype_nonbada_mtow

gawk -f function.nonbadacoefficient -f function.absvalue temp_badagood
temp_actype_nonbada_mtow > temp_subactype

gawk '{actual=$21; replace=$1; $1=actual; $21=replace}1' temp_subactype > temp_subactype2

cat temp_badagood temp_subactype2 > $out
```

File name: script_holding_check.gawk

```
# This script check if holding patterns happen at the same time as departures from a
neighboring airport
# In case of MDW (its 13C ILS holding and 22L departures)
# Inputs
recno=$1
fileid=$2
holding_list=$3
deprwy=$4
depairport=$5

if [ $recno -eq 1 ]
then
gawk '{print $0,"NA","NA","NA"}' $holding_list > temp
cp temp $holding_list
fi


# Get departure flight and bin them
infile1=$fileid"_"$depairport"_Dtracksrwy"

gawk -v deprwy=$deprwy '{if($21==deprwy){print $1,strftime("%H",$2)}}' $infile1 >
temp_dep1

gawk '{arr[$2]++} END{for(no in arr){print no, arr[no]}}' temp_dep1 > temp_dep2

# Get holding list
gawk -v fileid=$fileid '{if($1==fileid) print $0}' $holding_list > temp_arr1

# Check if any departures were there in the current bin
gawk '(NR==FNR){arr[$1]=$2; next} ($9 in arr){$10=arr[$9]}1' temp_dep2 temp_arr1 >
temp_arr2
# Check if any departures were there in the previous time bin bin
gawk '(NR==FNR){arr[$1]=$2; next} ($9-1 in arr){$11=arr[$9-1]}1' temp_dep2 temp_arr2 >
temp_arr3
# Check if any departures were there in the next time bin bin
gawk '(NR==FNR){arr[$1]=$2; next} ($9+1 in arr){$12=arr[$9+1]}1' temp_dep2 temp_arr3 >
temp_arr4

# update holdinglist
```

```
gawk '(NR==FNR){a1[$1]=$10; a2[$1]=$11; a3[$1]=$12; next} ($1 in
a1){$10=a1[$1];$11=a2[$1]; $12=a3[$1]}1' temp_arr4 $holding_list > temp_arr5
cp temp_arr5 $holding_list
```

File name: script_metroplex_benefits.gawk

```
# This script does the following
# This script is specific to chicago metroplex and will need modification to apply to
other metroplexes
# Determine excess fuel burn per year as a result of holding as a result of metroples
flow conflict
# Determine excess departure delays as a result of metroplex flow conflict
# The analysis is conduction for the years
# # Year range
year_start="2007"
year_end="2012"
# Time range
time_start=7
time_end=2
# Input files
# MDW
input1_arpt="MDW_2007_2012_arr_rwy"
# ORD
input2_arpt="ORD_ASPM_airport.txt"
input3_flts="ORD_flight_data_2007_2012.TAB"
input4="ORD_2007_2012_dep_cancelled.tab"

# Metroplex configuration conflict details
# runways effected and the meteorological conditions
rwy1="13C"
rwy2="22L"
MC="I"

# first run flag
firstrun=0

if [ $firstrun -eq 1 ]
then

# Get the departure runway config, total count of flight delayed, delay in minutes and
average delay per flight
# Get departure runway config
gawk 'BEGIN{OFS=FS="\t"} {print $1,$2,$3,$4,$5,$6,$25,$26,$27,$76,$79}' $input2_arpt |
sed 's/|/\t/g' | sed 's/ //g' | sed 's/\t/ /g' > temp_arpt_delay_config1
# Check if 22L is in the departure runway configuration
gawk -v rwy="22L" '{if($11~rwy){print $0,rwy} else{print $0,"NA"}}'
temp_arpt_delay_config1 > temp_arpt_delay_config2

# Combine MDW and ORD information
gawk '(NR==FNR){arr[$2$3$4$5]=$0; next} ($2$3$4$5 in arr){print
arr[$2$3$4$5],$1,$6,$7,$8,$9,$13,$12}' $input1_arpt temp_arpt_delay_config2 | sort -k2,2n
-k3,3n -k4,4n -k5,5n > temp_arpt_delay_config3

# Step1 ORD delay saving benefits
# Step1a Get delay stats for Metroplex Configuration Conflict (MCC) days
# Get list of days when the metroplex configuration conflict occurs
gawk -v rwy1=$rwy1 -v rwy2=$rwy2 -v MC=$MC '{if($7==MC && $17==rwy1 && $23==rwy2){arr[$2"
"$3]++; flts[$2" "$3]+=$20; delay[$2" "$3]+=$21}} END{for(no in arr) print no, arr[no]/4,
flts[no], delay[no]}' temp_arpt_delay_config3 | sort -k1,1n -k2,2n >
temp_MCC_days_duration
# Get number of flight delayed from flight data
# Get records with MCC
gawk -v rwy1=$rwy1 -v rwy2=$rwy2 -v MC=$MC 'BEGIN{OFS="\t"} {if($7==MC && $17==rwy1 &&
$23==rwy2){print $2,$3,$4,$5}}' temp_arpt_delay_config3 > temp_MCC_records
gawk 'BEGIN{FS="\t"} (NR==FNR){if(length($2)==1){a1=$2" "} else{a1=$2};
if(length($3)==1){a2=$3" "} else{a2=$3}; arr[$1" "a1" "a2" "$4]; next} ($1" "$2" "$3" "$4
in arr){delay=$54-15; if(delay<0){delay=0}; print $1,$2,$3,$4,delay}' temp_MCC_records
$input3_flts | sort -k1,1n -k2,2n -k3,3n -k4,4n > temp_MCC_flts_delays

gawk '{flts[$1" "$2]++; if($5>0){flts_d[$1" "$2]++}; delay[$1" "$2]+=$5} END{for(no in
flts) print no,flts[no],flts_d[no],delay[no]}' temp_MCC_flts_delays | sort -k1,1n -k2,2n
> temp MCC flts delays 2
# Combine delay and MCC duration information
gawk '(NR==FNR){arr[$1" "$2]=$3" "$4" "$5; next} ($1" "$2 in arr){print $1,$2,$3,arr[$1"
"$2]}' temp_MCC_flts_delays_2 temp_MCC_days_duration > temp_MCC_duration_delays
# Get total number of MCC days, average duration, total delays, percentage flight delayed
and average delays per flight for each year
gawk '{year1=substr($1,1,4); totaldays[year1]++; totalduration[year1]+=$3;
totalflts[year1]+=$4; totalfltsdel[year1]+=$5; totaldelay[year1]+=$6} END{for(no in
totaldays) {print
no,totaldays[no],totalduration[no]/totaldays[no],totalflts[no],totalfltsdel[no],totalflts
del[no]*100/totalflts[no], totaldelay[no], totaldelay[no]/totalflts[no]}}'
temp_MCC_duration_delays | sort -k1,1n > temp_ORD_MCC_stats

# Get mean and standard deviation of fuel burn per flight
gawk 'function mean(arr){zz1=0; yy1=0; for(no in arr){zz1+=arr[no]; yy1++}; out=zz1/yy1;
return out} function std(mean, arr){zz2=0; xx2=0; yy2=0; for(no in arr){xx2=(arr[no]-
mean)^2; zz2+=xx2; yy2++}; if(yy2==1){out=0} else{out=sqrt(zz2/(yy2-1))}; return out}
{if(substr($1,1,4)!=x){if(FNR!=1){mean1=mean(arr); std1=std(mean1,arr); print
x,mean1,std1}; x=substr($1,1,4); delete arr; arr[FNR]=$5} else{arr[FNR]=$5}}
END{{mean1=mean(arr); std1=std(mean1,arr); print x,mean1,std1}}'  temp_MCC_flts_delays |
sort -k1,1n > temp_MCC_avg_flt_delay

# Get the ADR stats
gawk -v rwy1=$rwy1 -v rwy2=$rwy2 -v MC=$MC '{if($7==MC && $17==rwy1 && $23==rwy2){print
$2,$3,$4,$5,$24}}' temp_arpt_delay_config3 > temp_MCC_ADR
gawk 'function mean(arr){zz1=0; yy1=0; for(no in arr){zz1+=arr[no]; yy1++}; out=zz1/yy1;
return out} function std(mean, arr){zz2=0; xx2=0; yy2=0; for(no in arr){xx2=(arr[no]-
mean)^2; zz2+=xx2; yy2++}; if(yy2==1){out=0} else{out=sqrt(zz2/(yy2-1))}; return out}
{if(substr($1,1,4)!=x){if(FNR!=1){mean1=mean(arr); std1=std(mean1,arr); print
x,mean1,std1}; x=substr($1,1,4); delete arr; arr[FNR]=$5} else{arr[FNR]=$5}}
END{{mean1=mean(arr); std1=std(mean1,arr); print x,mean1,std1}}'  temp_MCC_ADR | sort -
k1,1n > temp_MCC_avg_ADR

# Get cancellation stats
# Get the date time in the ASPM format
gawk 'BEGIN{FS=OFS="\t"} {if(length($19)==1){ym=$18"0"$19} else{ym=$18$19}; print
ym,$20,$21,$23}' $input4 > temp_cancelled_datetime
gawk 'BEGIN{FS="\t"} (NR==FNR){arr[$1$2$3$4]; next} ($1$2$3$4 in
arr){y[substr($1,1,4)]++} END{for(no in y) print no, y[no]}' temp_MCC_records
temp_cancelled_datetime | sort -k1,1n > temp_MCC_cancel_counts


# Step1b Get delay stats for Non-Metroplex Configuration Conflict (MCC) days
# Get list of days when the metroplex configuration conflict occurs
gawk -v rwy1=$rwy1 -v rwy2=$rwy2 -v MC=$MC '{if($7==MC && $17!=rwy1 && $23==rwy2){arr[$2"
"$3]++; flts[$2" "$3]+=$20; delay[$2" "$3]+=$21}} END{for(no in arr) print no, arr[no]/4,
flts[no], delay[no]}' temp_arpt_delay_config3 | sort -k1,1n -k2,2n >
temp_nonMCC_days_duration
# Get number of flight delayed from flight data
# Get records with MCC
gawk -v rwy1=$rwy1 -v rwy2=$rwy2 -v MC=$MC 'BEGIN{OFS="\t"} {if($7==MC && $17!=rwy1 &&
$23==rwy2){print $2,$3,$4,$5}}' temp_arpt_delay_config3 > temp_nonMCC_records
gawk 'BEGIN{FS="\t"} (NR==FNR){if(length($2)==1){a1=$2" "} else{a1=$2};
if(length($3)==1){a2=$3" "} else{a2=$3}; arr[$1" "a1" "a2" "$4]; next} ($1" "$2" "$3" "$4
in arr){delay=$54-15; if(delay<0){delay=0}; print $1,$2,$3,$4,delay}' temp_nonMCC_records
$input3_flts | sort -k1,1n -k2,2n -k3,3n -k4,4n > temp_nonMCC_flts_delays

gawk '{flts[$1" "$2]++; if($5>0){flts_d[$1" "$2]++}; delay[$1" "$2]+=$5} END{for(no in
flts) print no,flts[no],flts_d[no],delay[no]}' temp_nonMCC_flts_delays | sort -k1,1n -
k2,2n > temp_nonMCC_flts_delays_2
# Combine delay and MCC duration information
```

```
gawk '(NR==FNR){arr[$1" "$2]=$3" "$4" "$5; next} ($1" "$2 in arr){print $1,$2,$3,arr[$1"
"$2]}' temp_nonMCC_flts_delays_2 temp_nonMCC_days_duration > temp_nonMCC_duration_delays
# Take out MCC days
gawk '(NR==FNR){arr[$1$2$3]; next} !($1$2$3 in arr){print $0}' temp_MCC_duration_delays
temp_nonMCC_duration_delays > temp_nonMCC_duration_delays_2
# Get total number of MCC days, average duration, total delays, percentage flight delayed
and average delays per flight for each year
gawk '{year1=substr($1,1,4); totaldays[year1]++; totalduration[year1]+=$3;
totalflts[year1]+=$4; totalfltsdel[year1]+=$5; totaldelay[year1]+=$6} END{for(no in
totaldays) {print
no,totaldays[no],totalduration[no]/totaldays[no],totalflts[no],totalfltsdel[no],totalflts
del[no]*100/totalflts[no], totaldelay[no], totaldelay[no]/totalflts[no]}}'
temp_nonMCC_duration_delays_2 | sort -k1,1n > temp_ORD_nonMCC_stats

# Get mean and standard deviation of fuel burn per flight
gawk 'function mean(arr){zz1=0; yy1=0; for(no in arr){zz1+=arr[no]; yy1++}; out=zz1/yy1;
return out} function std(mean, arr){zz2=0; xx2=0; yy2=0; for(no in arr){xx2=(arr[no]-
mean)^2; zz2+=xx2; yy2++}; if(yy2==1){out=0} else{out=sqrt(zz2/(yy2-1))}; return out}
{if(substr($1,1,4)!=x){if(FNR!=1){mean1=mean(arr); std1=std(mean1,arr); print
x,mean1,std1}; x=substr($1,1,4); delete arr; arr[FNR]=$5} else{arr[FNR]=$5}}
END{{mean1=mean(arr); std1=std(mean1,arr); print x,mean1,std1}}'  temp_nonMCC_flts_delays
| sort -k1,1n > temp_nonMCC_avg_flt_delay

# Get the ADR stats
gawk -v rwy1=$rwy1 -v rwy2=$rwy2 -v MC=$MC '{if($7==MC && $17!=rwy1 && $23==rwy2){print
$2,$3,$4,$5,$24}}' temp_arpt_delay_config3 > temp_nonMCC_ADR
gawk 'function mean(arr){zz1=0; yy1=0; for(no in arr){zz1+=arr[no]; yy1++}; out=zz1/yy1;
return out} function std(mean, arr){zz2=0; xx2=0; yy2=0; for(no in arr){xx2=(arr[no]-
mean)^2; zz2+=xx2; yy2++}; if(yy2==1){out=0} else{out=sqrt(zz2/(yy2-1))}; return out}
{if(substr($1,1,4)!=x){if(FNR!=1){mean1=mean(arr); std1=std(mean1,arr); print
x,mean1,std1}; x=substr($1,1,4); delete arr; arr[FNR]=$5} else{arr[FNR]=$5}}
END{{mean1=mean(arr); std1=std(mean1,arr); print x,mean1,std1}}'  temp_nonMCC_ADR | sort
-k1,1n > temp_nonMCC_avg_ADR

# Get cancellation stats
# Get the date time in the ASPM format
gawk 'BEGIN{FS="\t"} (NR==FNR){arr[$1$2$3$4]; next} ($1$2$3$4 in
arr){y[substr($1,1,4)]++} END{for(no in y) print no, y[no]}' temp_nonMCC_records
temp_cancelled_datetime | sort -k1,1n > temp_nonMCC_cancel_counts

# Get a count of total number of MDW arrivals
gawk -v rwy1=$rwy1 -v rwy2=$rwy2 -v MC=$MC '{if($7==MC && $17==rwy1 &&
$23==rwy2){year=substr($2,1,4);configcon[year]++; configconops[year]+=$14;
configconopsswa[year]+=$15}} END{for(no in configcon) print no,
configcon[no]/4,configconops[no],configconopsswa[no]}' temp_arpt_delay_config3 | sort -
k1,1n > temp_MDW_MCC_arrival_counts

else
gawk 'BEGIN{y="do nothing"}'

# Get mean and standard deviation of fuel burn per flight
gawk 'function mean(arr){zz1=0; yy1=0; for(no in arr){zz1+=arr[no]; yy1++}; out=zz1/yy1;
return out} function std(mean, arr){zz2=0; xx2=0; yy2=0; for(no in arr){xx2=(arr[no]-
mean)^2; zz2+=xx2; yy2++}; if(yy2==1){out=0} else{out=sqrt(zz2/(yy2-1))}; return out}
{if(substr($1,1,4)!=x){if(FNR!=1){mean1=mean(arr); std1=std(mean1,arr); print
x,mean1,std1}; x=substr($1,1,4); delete arr; arr[FNR]=$5} else{arr[FNR]=$5}}
END{{mean1=mean(arr); std1=std(mean1,arr); print x,mean1,std1}}'  temp_MCC_flts_delays |
sort -k1,1n > temp_MCC_avg_flt_delay

# Get the ADR stats
gawk -v rwy1=$rwy1 -v rwy2=$rwy2 -v MC=$MC '{if($7==MC && $17==rwy1 && $23==rwy2){print
$2,$3,$4,$5,$24}}' temp_arpt_delay_config3 > temp_MCC_ADR
gawk 'function mean(arr){zz1=0; yy1=0; for(no in arr){zz1+=arr[no]; yy1++}; out=zz1/yy1;
return out} function std(mean, arr){zz2=0; xx2=0; yy2=0; for(no in arr){xx2=(arr[no]-
mean)^2; zz2+=xx2; yy2++}; if(yy2==1){out=0} else{out=sqrt(zz2/(yy2-1))}; return out}
{if(substr($1,1,4)!=x){if(FNR!=1){mean1=mean(arr); std1=std(mean1,arr); print
x,mean1,std1}; x=substr($1,1,4); delete arr; arr[FNR]=$5} else{arr[FNR]=$5}}
```

```
END{{mean1=mean(arr); std1=std(mean1,arr); print x,mean1,std1}}'  temp_MCC_ADR | sort -
k1,1n > temp_MCC_avg_ADR

# Get cancellation stats
# Get the date time in the ASPM format
gawk 'BEGIN{FS=OFS="\t"} {if(length($19)==1){ym=$18"0"$19} else{ym=$18$19}; print
ym,$20,$21,$23}' $input4 > temp_cancelled_datetime
gawk 'BEGIN{FS="\t"} (NR==FNR){arr[$1$2$3$4]; next} ($1$2$3$4 in
arr){y[substr($1,1,4)]++} END{for(no in y) print no, y[no]}' temp_MCC_records
temp_cancelled_datetime | sort -k1,1n > temp_MCC_cancel_counts

# Get mean and standard deviation of fuel burn per flight
gawk 'function mean(arr){zz1=0; yy1=0; for(no in arr){zz1+=arr[no]; yy1++}; out=zz1/yy1;
return out} function std(mean, arr){zz2=0; xx2=0; yy2=0; for(no in arr){xx2=(arr[no]-
mean)^2; zz2+=xx2; yy2++}; if(yy2==1){out=0} else{out=sqrt(zz2/(yy2-1))}; return out}
{if(substr($1,1,4)!=x){if(FNR!=1){mean1=mean(arr); std1=std(mean1,arr); print
x,mean1,std1}; x=substr($1,1,4); delete arr; arr[FNR]=$5} else{arr[FNR]=$5}}
END{{mean1=mean(arr); std1=std(mean1,arr); print x,mean1,std1}}'  temp_nonMCC_flts_delays
| sort -k1,1n > temp_nonMCC_avg_flt_delay

# Get the ADR stats
gawk -v rwy1=$rwy1 -v rwy2=$rwy2 -v MC=$MC '{if($7==MC && $17!=rwy1 && $23==rwy2){print
$2,$3,$4,$5,$24}}' temp_arpt_delay_config3 > temp_nonMCC_ADR
gawk 'function mean(arr){zz1=0; yy1=0; for(no in arr){zz1+=arr[no]; yy1++}; out=zz1/yy1;
return out} function std(mean, arr){zz2=0; xx2=0; yy2=0; for(no in arr){xx2=(arr[no]-
mean)^2; zz2+=xx2; yy2++}; if(yy2==1){out=0} else{out=sqrt(zz2/(yy2-1))}; return out}
{if(substr($1,1,4)!=x){if(FNR!=1){mean1=mean(arr); std1=std(mean1,arr); print
x,mean1,std1}; x=substr($1,1,4); delete arr; arr[FNR]=$5} else{arr[FNR]=$5}}
END{{mean1=mean(arr); std1=std(mean1,arr); print x,mean1,std1}}'  temp_nonMCC_ADR | sort
-k1,1n > temp_nonMCC_avg_ADR

# Get cancellation stats
# Get the date time in the ASPM format
gawk 'BEGIN{FS="\t"} (NR==FNR){arr[$1$2$3$4]; next} ($1$2$3$4 in
arr){y[substr($1,1,4)]++} END{for(no in y) print no, y[no]}' temp_nonMCC_records
temp_cancelled_datetime | sort -k1,1n > temp_nonMCC_cancel_counts



fi
```

File name: script_mplex_arr_dep_count.gawk

```
# This script count the number of arrivals and departures for neighboring airports
# In case of Chicago metroplex, count number of ILS arrivals onto 13C at MDW
anddepartures from 22L at ORD at the same time.

# Input
recno=$1 # record number
tstart=$2 # bin start time
tend=$3 # bin end time
fileid=$4 # file id
buffer=$5 # time buffer for each bin
arr_count=$6 # arrival counts
deprwy=$7 # dep runway
depairport=$8 # dep airport

if [ $recno -eq 1 ]
then
gawk '{print $0,"NA"}' $arr_count > temp
cp temp $arr_count
fi

# Get departure flight and bin them
infile1=$fileid"_"$depairport"_Dtracksrwy"
```

```
# Count the number of departures for record
gawk -v tstart=$tstart -v tend=$tend -v deprwy=$deprwy -v fileid=$fileid -v
buffer=$buffer -v recno=$recno 'BEGIN{count=0; if(buffer==0 ||
buffer>=1800){buffer2=1800} else{buffer2=buffer}} {if($21==deprwy){if($2>=tstart-buffer2
&& $2<=tend){count++}}} END{print recno,fileid, count}' $infile1 > temp_count1
# gawk -v tstart=$tstart -v tend=$tend -v deprwy=$deprwy -v fileid=$fileid
'BEGIN{count=0} {if($21==deprwy){count++}} END{print fileid, count}' $infile1 >
temp_count1

# update count
gawk '(NR==FNR){arr[$1" "$2]=$3; next} ($1" "$7 in arr){$8=arr[$1" "$7]}1' temp_count1
$arr_count > temp2
cp temp2 $arr_count
```

File name: script_plot_tracks_by_flow.gawk

```
# This script plot track for each flow
# The flows and their colors are defined in flow_color_2.dat

# Input
direction=$1
runway=$2
procedure=$3
color=$4

input1="NOP_data_all_good_2"
input2="NOP_filelist.dat"


# filter out flows and print kml files
# change type to NA for normal tracks and GA for tracks with excess turn angle
gawk -v direction=$direction -v runway=$runway -v procedure=$procedure -v type="NA" -v
actype="B73" '{if($22==runway && $23==direction && $24==procedure && $25==type &&
$29~/'"$actype"'/){print $2}}' $input1 > temp_test1

# remove temp_test2
rm temp_test2

# get track info for ids in temp_test1
# The output of this script in temp_test2 containing the track info
gawk -v flightids="temp_test1" '{execute="./script_print_error_tracks.gawk "$1"
"flightids; print execute; system(execute)}' $input2

# Copy temp_test2 to another file
# change remove "_GA" while processing normal tracks
# out1="tracks_"$direction"_"$runway"_"$procedure"_GA"
out1="tracks_"$direction"_"$runway"_"$procedure
cp temp_test2 $out1
# print temp_test2 in kml format
gawk -v out1=$out1 -v color=$color 'BEGIN{execute="./script_print_kml.gawk "out1" "color;
print execute; system(execute)}'
```

File name: script_plot_tracks_by_flow0.gawk

```
# This script executes script_plot_tracks_by_flow.gawk
# The script is executed for all valid entries in flow_color_2.dat
flowcolor="flow_colors_2.dat"

# get valid entries from flow_color_2.dat
gawk '{if($1!="#") print $0}' $flowcolor > temp_flowcolor

# execute script_plot_tracks_by_flow.gawk
gawk '{execute="./script_plot_tracks_by_flow.gawk "$0; print execute; system(execute)}'
temp_flowcolor
```

```
# File name: script_print_error_testing.gawk
# This script prints out error files

gawk '{execute="./script_print_error_tracks.gawk "$1; print execute; system(execute)}'
NOP_filelist.dat
```

File name: script_print_error_tracks.gawk

```
# This script print all track for error testing
infile=$1

input1=$infile"_MDW_Atracks"
input2=$2

gawk '(NR==FNR){arr[$1]; next} ($1 in arr){print $0}' $input2 $input1 >> temp_test2
```

File name: script_print_flow_label_kml.gawk

```
# Given the label information, this script print out a kml file for locating the label on
google earth
# Input is the flow legend file
# Should have the flow name, color, location coordinates to place the legend on the map
# The file should be tab separated

BEGIN{FS=OFS="\t"}
{flow[FNR]=$1;color[FNR]=$2;lat[FNR]=$3;lon[FNR]=$4}
END{
# print header
print "<?xml version=\42""1.0\42 encoding=\42UTF-8\42?>";
print "<kml xmlns=\42http://www.opengis.net/kml/2.2\42
xmlns:gx=\42http://www.google.com/kml/ext/2.2\42
xmlns:kml=\42http://www.opengis.net/kml/2.2\42
xmlns:atom=\42http://www.w3.org/2005/Atom\42>";

# print document name and style information
print "<Document id=\42doc1\42>";
print "\t<name>flowlabel</name>";
print "\t<visibility>0</visibility>";
print "\t<open>1</open>";

# print style
for(i=1;i<=length(flow);i++){
colorcode=getcolor(color[i]);
# Specific line style and color
print "\t<Style id=\42style"i"\42>";
# Icon style
print "\t\t<IconStyle>";
print "\t\t\t<color>"colorcode"</color>";
print "\t\t\t<scale>0.8</scale>";
print "\t\t\t<Icon>";
print "\t\t\t\t<href>airplane.png</href>";
print "\t\t\t</Icon>";
print "\t\t</IconStyle>";
# Lable style
print "\t\t<LabelStyle>";
print "\t\t\t<color>ffffffff</color>";
print "\t\t\t<scale>0.8</scale>";
print "\t\t</LabelStyle>";
# Line style
print "\t\t<LineStyle>";
print "\t\t\t<color>"colorcode"</color>";
print "\t\t</LineStyle>";
print "\t</Style>";
};
# print folder information
print "\t<Folder id=\42Flight Tracks\42>";
print "\t\t<name>"filename"</name>";
print "\t\t<visibility>0</visibility>";

# print label information
for(i=1;i<=length(flow);i++){
print "\t\t<Placemark id=\42"flow[i]"\42>";
print "\t\t\t<name>"flow[i]"</name>";
print "\t\t\t<visibility>0</visibility>";
print "\t\t\t<styleUrl>#style"i"</styleUrl>";
print "\t\t\t<Point>";
print "\t\t\t\t<altitudeMode>clampToGround</altitudeMode>";
print "\t\t\t\t<coordinates>";
print "\t\t\t\t\t\t"lon[i]","lat[i];
print "\t\t\t\t</coordinates>";
print "\t\t\t</Point>";
print "\t\t</Placemark>"
};

# Close open tags
print "\t</Folder>";
print "</Document>";
print "</kml>"

}
```

File name: script_print_kml.gawk

```
# This script coverts track data to kml format to be viewed on google earth
# Syntax to run the script
# ./script_print_kml.gawk inputfile trackcolor

# The inputfile should be space delimited
# It should have the following fields
# field 1 - flight id (unique)
# field 2 - timestamp (seconds)
# field 3 - Origin
# field 4 - Destination
# field 5 - Date (YYYY-MM-DD)
# field 6 - Time (HH:MM:SS)
# field 7 - flight latitude (-ve for eastern hemisphere)
# field 8 - flight longitude (-ve for southern hemisphere)
# field 9 - flight altitude (in feet)
# Should be sorted by field1 followed by field2

# Color has to be small cases
# Color options voilet, indigo, blue, green, yellow, orange, red

# Inputs
filename=$1
color=$2
out=$1".kml"

# Generate kml file
gawk -v filename=$filename -v linecolor=$color -f function.getcolor -f function.kmlfile_1
$filename > $out
```

File name: script_print_kml_byrwy.gawk

```
# This script output kml file for each runway
infile1=$1
```

```
infile2=$2
rwy=$3
color=$4
out1=$infile2"_"$rwy

# Filter out runway tracks
gawk -v rwy=$rwy '{if($NF==rwy) print $1}' $infile1 > temp_rwy_trackid
gawk '(NR==FNR){arr[$1];next} ($1 in arr){print $0}' temp_rwy_trackid $infile2 >
temp_rwy_tracks

# print out kml file
gawk -v filename=$out1 -v linecolor=$color -f function.getcolor -f function.kmlfile_1
temp_rwy_tracks > $out1".kml"
```

File name: script_process_tracks_2.gawk

```
# This script assign each track at runway a flow
# Flow is defined as direction runway and procedure
# e.g W 13C ILS, E 13C RNP, E 31C Visual
# Logic
# Step1: Assign track to Airport using point in square algorithm
# Step2: Separate Arrivals from Departures
# Step3: Assign track to runway
# run using ./script_process_tracks_2 NOP_data_name
infile1=$1 # NOP data
airportinfo="airport_info.dat"
runwayinfo="runway_info.dat"
flowdirection="flow_direction.dat"
flowinfo="flow_info_all.dat"
fixinfo="MDW_fix.dat"
goaround_holding="goaround_holding.dat"
flowmeasuringfix="flow_measuring_fix.dat"
badadata="BADA_coefficient_2.dat"
windinfo1="MDW_wind_info.dat"

# Filter out file and get the first and last hit for each record
gawk -v infile1=$infile1 'BEGIN{execute="time ./script_get_first_last_hit.gawk "infile1;
print execute; system(execute)}'

# Assign airport and separate arrivals from departures
gawk -v infile1=$infile1 '{execute="time ./script_assign_airport.gawk "infile1" "$1" "$2"
"$3" "$4" "$5; print execute; system(execute)}' $airportinfo

# Assign runways to arrivals and departures
# Arrivals
gawk -v arr=1 -v infile1=$infile1 -v runwayinfo=$runwayinfo '{execute="time
./script_assign_runway.gawk "arr" "infile1" "runwayinfo" "$1" "$2" "$3; print execute;
system(execute)}' $airportinfo
# Departures
gawk -v arr=0 -v infile1=$infile1 -v runwayinfo=$runwayinfo '{execute="time
./script_assign_runway.gawk "arr" "infile1" "runwayinfo" "$1" "$2" "$3; print execute;
system(execute)}' $airportinfo

# Assign arrival or departure direction to each flight
# For instance identify general direction of flow
# i.e. if the flight is coming in from the south, east, west etc
# The flight tracks are assigned to flows by using heuristics
# The heuristic uses predefined boxes to filter and assign flows
# If a flight track passes through a box it is filtered out and labeled
# See appendix for input file format

# Assign flow direction to all entries in flow_direction.dat
gawk '{if($1!="#") print $1,$2}' $flowdirection | sort -u > temp_airport_type
# For each entry of airport and type (arrival/departure) assign direction
```

```
gawk -v infile1=$infile1 -v flowdirection=$flowdirection
'{execute="./script_assign_flow_direction0.gawk "infile1" "$1" "$2" "flowdirection; print
execute; system(execute)}' temp_airport_type

## Next Procedure type , ILS RNP or Visual
## This has to be done by runway and direction
# Assign flow procedure to all entries in flow_info_all.dat
gawk '{if($1!="#") print $1,$2,$3,$4}' $flowinfo | sort -u >
temp_airport_type_direction_runway
# Get Co-ordinates of all the fixes
gawk '(NR==FNR){arr[$1]=$2" "$3; next} ($6 in arr){print $0, arr[$6]}' $fixinfo $flowinfo
> temp_flow_fix
# For each entry of airport, type, direction and runway assign procedure (ILS, RNP,
Visual)
gawk -v infile1=$infile1 -v flowinfo="temp_flow_fix"
'{execute="./script_assign_flow_procedure0.gawk "infile1" "FNR" "$1" "$2" "$3" "$4"
"flowinfo; print execute; system(execute)}' temp_airport_type_direction_runway


# Detect go around
# Filter out goaround and holding for all entries in goaround_holding.dat
gawk '{if($1!="#") print $1,$2}' $goaround_holding | sort -u > temp_goaround_holding
# For each entry of airport and arrival/departure type detect goarounds and holding
patterns
gawk -v infile1=$infile1 -v goaround_holding=$goaround_holding -v
airportinfo=$airportinfo '{execute="time ./script_detect_goaround_holding0.gawk "infile1"
"FNR" "$1" "$2" "goaround_holding" "airportinfo; print execute; system(execute)}'
temp_goaround_holding

# Compute track mile and track time for each flight in each flow
# Compute track mile and track time for all entries in flow_measuring_fix.dat
# Get lat lon the fix from where distance needs to be computed
gawk '(NR==FNR){lat[$1]=$2;lon[$1]=$3;next} ($4 in lat){if($1!="#"){print
$0,lat[$4],lon[$4]}}' $fixinfo $flowmeasuringfix > temp_flowfix_0
# Get lat lon the airport to filter out track with some check radius
gawk '(NR==FNR){lat[$1]=$2;lon[$1]=$3;alt[$1]=$6;next} ($1 in lat){if($1!="#"){print
$0,lat[$1],lon[$1],alt[$1]}}' $airportinfo temp_flowfix_0 > temp_flowfix
# Compute track mile and track time for each flow direction
gawk -v infile1=$infile1 -v flowfix="temp_flowfix" -v badadata=$badadata -v
windinfo=$windinfo1 'BEGIN{execute="time ./script_compute_track_mile_time_fuel0.gawk
"infile1" "flowfix" "badadata" "windinfo; print execute; system(execute)}'
```

A4. User defined functions

The list of function required to execute the script are provided below. Each function needs to be saved (as the file name provided) in the same directory at the GAWK scripts.

```
File name: function.absvalue
# This function return the absolute value of a number
function abs(num){if(num<0){newnum=-1*num} else{newnum=num}; return newnum}
```

```
File name: function.acos
# This function computes cosine inverse
function acos(x) {return atan2(sqrt(1-x*x),x)}
```

File name: function.airdensity

```
# This function computes density of air as a function of altitude
function rho_alt(h){
p0=101.325 # sea level standard atmospheric pressure kPa
T0=288.15 # sea level standard temprature in K
g0=9.80665 # acceleration due to gravity m/s^2
L=.0065 # temperature lapse rate K/m
R=8.31447 # ideal gas constant J/(mol-K)
M=.0289644 # molar mass of dry air kg/mol

T=T0-L*h # temperature at altitude h

p=p0*(1-(L*h/T0))^((g0*M)/(R*L))

rho_x=(p*M)/(R*T)*1000; # air density in kg/m^3
return rho_x;
}
```

File name: function.asin

```
# This function computes sine inverse
function asin(x) {return atan2(x, sqrt(1-x*x))}
```

File name: function.assignairports

```
# This function assign airport to each track
# The arrival or departure information is also dertermine
{
# Check if arrival or departure
# If first point is within the airport then its a departure
# If the last point is within the airport then its a arrival
if($7>=lat-span && $7<=lat+span && $8>=lon-span && $8<=lon+span && $9<=alt){print
$0,airport,"D"} else{if($10>=lat-span && $10<=lat+span && $11>=lon-span && $11<=lon+span
&& $12<=alt){print $0,airport,"A"}}}
```

File name: function.atan

```
# This function computes tan inverse
function atan(x) {return atan2(x,1)}
```

File name: function.avgfb_allcases

```
# This function computes average of the total fuel burn for MDW for all the 5 cases
BEGIN{
# kg to gallon convertion
factor=(1/.81)*0.264172
}
{
if($1!=x){
x=$1;
base_v=$4*factor;
a1_v=$5*factor;
a2_v=$6*factor;
a3_v=$7*factor;
a4_v=$8*factor;
} else{
base[FNR]=($4+base_v)*factor;
a1[FNR]=($5+a1_v)*factor;
a2[FNR]=($6+a2_v)*factor;
a3[FNR]=($7+a3_v)*factor;
a4[FNR]=($8_a4_v)*factor;
};
if($2=="I"){
base_I[FNR]=$4*factor;
a1_I[FNR]=$5*factor;
a2_I[FNR]=$6*factor;
a3_I[FNR]=$7*factor;
a4_I[FNR]=$8*factor};
if($2=="V"){
base_V[FNR]=$4*factor;
a1_V[FNR]=$5*factor;
a2_V[FNR]=$6*factor;
a3_V[FNR]=$7*factor;
a4_V[FNR]=$8*factor;}} END{
# total fuel burn during IMC on an average
mean_base_I=mean(base_I); sd_base_I=std(mean_base_I,base_I);
mean_a1_I=mean(a1_I); sd_a1_I=std(mean_a1_I,a1_I);
mean_a2_I=mean(a2_I); sd_a2_I=std(mean_a2_I,a2_I);
mean_a3_I=mean(a3_I); sd_a3_I=std(mean_a3_I,a3_I);
mean_a4_I=mean(a4_I); sd_a4_I=std(mean_a4_I,a4_I);

# total fuel burn during VMC on an average
mean_base_V=mean(base_V); sd_base_V=std(mean_base_V,base_V);
mean_a1_V=mean(a1_V); sd_a1_V=std(mean_a1_V,a1_V);
mean_a2_V=mean(a2_V); sd_a2_V=std(mean_a2_V,a2_V);
mean_a3_V=mean(a3_V); sd_a3_V=std(mean_a3_V,a3_V);
mean_a4_V=mean(a4_V); sd_a4_V=std(mean_a4_V,a4_V);

# total fuel burn during VMC and IMC on an average
mean_base=mean(base); sd_base=std(mean_base,base);
mean_a1=mean(a1); sd_a1=std(mean_a1,a1);
mean_a2=mean(a2); sd_a2=std(mean_a2,a2);
mean_a3=mean(a3); sd_a3=std(mean_a3,a3);
mean_a4=mean(a4); sd_a4=std(mean_a4,a4);

# Print all the numbers
print mean_base_V,sd_base_V,mean_base_I,sd_base_I,mean_base,sd_base;
print mean_a1_V,sd_a1_V,mean_a1_I,sd_a1_I,mean_a1,sd_a1;
print mean_a2_V,sd_a2_V,mean_a2_I,sd_a2_I,mean_a2,sd_a1;
print mean_a3_V,sd_a3_V,mean_a3_I,sd_a3_I,mean_a3,sd_a3;
print mean_a4_V,sd_a4_V,mean_a4_I,sd_a4_I,mean_a4,sd_a4;

}
```

File name: function.badacoefficient

```
# This file contains the fuel burn model and mapping information onto BADA OPF files
{
# Get engine type
if(FNR==14){engine=$(NF-2)};
# Fuelburn(kg/min) = eta(kg/(min*kN)) * thrust (kN)
# eta = Cf1 * (1+ (Vtas/Cf2))
## Cf1 = 1st Thrust specfic fuel consumption (kg/(min*kN))
## Cf2 = 2nd Thrust specifuc fuel consumption (knots)
## Vtas = True Airspeed (knots) (from NOP data) ## Conver to m/s
# Cf1 and Cf2 from BADA OPF file line 52, field 2 and 3
if(FNR==52){Cf1=$2*1; # conversion to kg/(sec*kN)
Cf2=$3*0.514444444} # Conversion factor to m/s
# Co-efficients for minimum fuel flow
if(FNR==54){
Cf3=$2*1; # kg/min
Cf4=$3*1; # in feet
}
# Get the cruise fuel flow factor
if(FNR==56){Cfcr=$2*1};

# thrust = drag (kN) + mass(kg)*d(Vtas)/dt + mass*g0*(h0/Vtas)
##  drag = (Cd * rho * (Vtas)^2 * S)/2
### Cd = coefficient of drag (two cases Approach and Landing) (no unit)
### Cd for Cruise
#### Cd,CR = Cd0,CR + Cd2,CR * (Cl)^2
### Cd for Approach
#### Cd,AP = Cd0,AP + Cd2,AP * (Cl)^2
### Cd for Landing
#### Cd,LD = Cd0,LD + Cd0,delLD + Cd2,LD * (Cl)^2
#### Cd0,AP and Cd2,AP are parasitic and induced drag coefficients during approach
# If co-efficient of drag is missing for AP use clean configuration
if(FNR==29){Cd0CR=$(NF-3)*1; Cd2CR=$(NF-2)*1; VstallCR=$(NF-4)*1};
# Cd0,AP and Cd2,AP from BADA OPF file line 32, field 6 and 7
if(FNR==32){if($(NF-3)==0){Cd0AP=Cd0CR*2} else{Cd0AP=$(NF-3)*1}; if($(NF-
2)==0){Cd2AP=Cd2CR} else{Cd2AP=$(NF-2)*1}; VstallAP=$(NF-4)*1};
#### Cd0,LD and Cd2,LD are parasitic and induced drag coefficients during Landing
# Cd0,LD and Cd2,LD from BADA OPF file line 33. field 6 and 7
# if(FNR==33){Cd0LD=$(NF-3)*1; Cd2LD=$(NF-2)*1};
if(FNR==33){if($(NF-3)==0){Cd0LD=Cd0CR*3} else{Cd0LD=$(NF-3)*1}; if($(NF-
2)==0){Cd2LD=Cd2CR} else{Cd2LD=$(NF-2)*1}; VstallLD=$(NF-4)*1};
#### Cd0,delLD parasitic drag landing gear
# Cd0,delLD from BADA OPF file line 39 filed 4
if(FNR==39){if($4==0){Cd0delLD=.02} else{Cd0delLD=$4*1}};
##### Cl = lift Coefficient (no unit)
##### Cl = (2 * mass * g0)/(rho * Vtas^2 * S * Cos(phi))
##### mass = reference mass of the aircraft type (tonnes) 3 Convert to kg
# mass from BADA OPF file line 19 field 2
if(FNR==19){mass=((($3+$5)+$2)/2)*1000; MTOW=$4*1000}; # Conversion factor to kg
#### g0 = acceleration due to gravity (m/s^2)
#### g0=9.80665;
#### rho = density of air (function of altitude) (kg/m^3) from NOP data and density
function
#### S = wing reference area (m^2)
# from BADA OPF file line 26 field 3
if(FNR==26){S=$3*1};
#### phi is bank angle (assumed to be zero)
} END{
print
flighttype,Cf1,Cf2,Cf3,Cf4,Cfcr,Cd0CR,Cd2CR,VstallCR,Cd0AP,Cd2AP,VstallAP,Cd0LD,Cd2LD,Cd0
delLD,VstallLD,mass,S,MTOW,engine;
}
```

File name: function.bearing

```
# This function gives the bearing given two lat lon
function bearing(lat1,lon1,lat2,lon2){
rlat1=radian(lat1);
rlon1=radian(lon1);
rlat2=radian(lat2);
rlon2=radian(lon2);
dellon=rlon2-rlon1;
var1=sin(dellon)*cos(rlat2);
var2=cos(rlat1)*sin(rlat2)-sin(rlat1)*cos(rlat2)*cos(dellon);
out=degrees(atan2(var1,var2));
return out}
```

File name: function.combine_flow_stats

```
# This function combines flow stats
{
if(x1!=$2){
if(FNR!=1){
t1=0;
s=0;
# get combined mean
for(i=1;i<=length(count);i++){
##t1=t1+mean[i]*count[i];
s=s+count[i]};
##m1=t1/s;
m1=m1+mean[i]*(1/length(count));}
# get combined standard deviation
v1=0;
for(i=1;i<=length(count);i++){
##v1=v1+count[i]*(sd[i]^2+(mean[i]-m1)^2)};
##sd1=sqrt(v1/s);
v1=v1+(1/length(count))*(sd[i]^2+(mean[i]-m1)^2)};
sd1=sqrt(v1);
# Print results
print x1,s,m1,sd1;
};
# initialize array
delete count;
delete mean;
delete sd;
count[FNR]=$4;
mean[FNR]=$7;
sd[FNR]=$8;

x1=$2;
} else{
count[FNR]=$4;
mean[FNR]=$7;
sd[FNR]=$8;
}} END{
t1=0;
s=0;
# get combined mean
for(i=1;i<=length(count);i++){
##t1=t1+mean[i]*count[i];
s=s+count[i]};
##m1=t1/s;
m1=m1+mean[i]*(1/length(count));}

# get combined standard deviation
v1=0;
for(i=1;i<=length(count);i++){
##v1=v1+count[i]*(sd[i]^2+(mean[i]-m1)^2)};
##sd1=sqrt(v1/s);
v1=v1+(1/length(count))*(sd[i]^2+(mean[i]-m1)^2)};
sd1=sqrt(v1);
# Print results
print x1,s,m1,sd1;
}
```

File name: function.combine_flow_stats_2

```
# This function combines flow stats
{
if(x1!=$2){
if(FNR!=1){
t1=0;
s=0;
# get combined mean
for(i=1;i<=length(count);i++){
##t1=t1+mean[i]*count[i];
s=s+count[i]};
##m1=t1/s;
m1=m1+mean[i]*(1/length(count));}

# get combined standard deviation
v1=0;
for(i=1;i<=length(count);i++){
##v1=v1+count[i]*(sd[i]^2+(mean[i]-m1)^2)};
##sd1=sqrt(v1/s);
v1=v1+(1/length(count))*(sd[i]^2+(mean[i]-m1)^2)};
sd1=sqrt(v1);
# Print results
print x1,s,m1,sd1;
};
# initialize array
delete count;
delete mean;
delete sd;
count[FNR]=$4;
mean[FNR]=$7;
sd[FNR]=$8;

x1=$2;
} else{
count[FNR]=$4;
mean[FNR]=$7;
sd[FNR]=$8;
}} END{
t1=0;
s=0;
# get combined mean
for(i=1;i<=length(count);i++){
##t1=t1+mean[i]*count[i];
s=s+count[i]};
##m1=t1/s;
m1=m1+mean[i]*(1/length(count));}

# get combined standard deviation
v1=0;
for(i=1;i<=length(count);i++){
##v1=v1+count[i]*(sd[i]^2+(mean[i]-m1)^2)};
##sd1=sqrt(v1/s);
v1=v1+(1/length(count))*(sd[i]^2+(mean[i]-m1)^2)};
sd1=sqrt(v1);
# Print results
print x1,s,m1,sd1;
}
```

File name: function.computeTAS

```
# This function computes True Air Speed (TAS)
function airspeed(lat1,lon1,lat2,lon2,deltime,alt,w_dir,w_mag){
# Compute aircraft bearing
f_bearing=bearing(lat1,lon1,lat2,lon2);
if(f_bearing<0){f_bearing=360+f_bearing};
f_bearing_rad=radian(f_bearing);
# Compute wind bearing
if(w_dir<180){w_bearing=w_dir+180} else{w_bearing=w_dir-180};
w_bearing_rad=radian(w_bearing);
# Compute ground speed
dist1=gcd(lat1,lon1,lat2,lon2)*1852; # Convert to meter
g_speed=dist1/deltime; # meter per second
# Covert wind speed to meter per second
w_speed1=w_mag*0.514444444;
# Ajust wind speed for altitude
w_speed2=w_speed1*(alt/33)^0.3 # using hellman wind gradient formula
# Compute airspeed
g_ver=g_speed*sin(f_bearing_rad);
g_hor=g_speed*cos(f_bearing_rad);
w_ver=w_speed2*sin(w_bearing_rad);
w_hor=w_speed2*cos(w_bearing_rad);
del_ver=g_ver-w_ver;
del_hor=g_hor-w_hor;
vtas=sqrt(del_ver^2 + del_hor^2);
return vtas;
}
```

File name: function.computefuelburn1

```
# This function computes fuel burn using BADA model
# The fuel burn is computed for each time step in the NOP data
# It is assumed that the flight is in its approach phase throughout
BEGIN{
g=9.80665; # acceleration due to gravity m/s^2
}
{
if(NR==FNR){
# record BADA coefficients
cf1[$1]=$2; # thrust specific fuel consumption coefficient one unit kg/(min*N)
cf2[$1]=$3; # thrust specific fuel consumption coefficient two unit m/s
cf3[$1]=$4; # idle descent fuel flow coefficient one kg/min
cf4[$1]=$5; # idle descent fuel flow coefficient two in feet
cfcr[$1]=$6; # Cruise fuel flow factor no unit
cd0cr[$1]=$7; # parasitic drag coefficients during cruise no unit
cd2cr[$1]=$8; # induced drag coefficients during cruise no unit
vstallcr[$1]=$9; # cruise stall speed in knots
cd0ap[$1]=$10; # parasitic drag coefficients during approach no unit
cd2ap[$1]=$11; # induced drag coefficients during approach no unit
vstallap[$1]=$12; # approach stall speed in knots
cd0ld[$1]=$13; # parasitic drag coefficients during landing no unit
cd2ld[$1]=$14; # induced drag coefficients during landing no unit
cd0ldg[$1]=$15; # parasitic drag coefficients during landing no unit
vstallld[$1]=$16; # landing stall speed in knots
mass[$1]=$17; # mass in kg
s[$1]=$18; # wing reference area m^2
engine[$1]=$20; # engine type piston, turboprop, or jet
} else{
if($1!=x){
x=$1;
cumfb=0;
check1=0; # check for clean to dirty
actype=$18;
mass1=mass[actype]; # initialize mass of aircraft
print $0,"NA","NA","NA"
} else{
if(cf1[actype]=="" || engine[actype]=="Piston"){
fb="NA";
cumfb="NA";
phase="NA";
print $0,phase,fb,cumfb;
} else{
deltime=$13; # time step
alt_feet=$7;
alt_agl=$7-elev; # altitude above ground level in feet
alt_meter=$7*0.3048; # altitude for calculating density in meter
delalt=$14*0.3048; # change in altitude in meter
dist=$15; # change in distance in meter
vtas=$16; # velocity in m/s
vtas2=$16*1.94384; # veloctity in knots
accel=$17; # acceleration in m/s^2
rho1=rho_alt(alt_meter); # density of air at given alt in kg/m^3

# Check phase of the flight
# i.e., cruise,approach or landing
# initialize standards from BADA data
Hmaxap=8000; # maximum altitude threshold for approach in feet
Hmaxld=1700; # maximum altitude thresholf for landing in feet
Vcr=1.3*vstallcr[actype]; # minimum velocity threshold for cruise in knots
Vap=1.3*vstallap[actype]; # minimum velocity threshold for approach in knots
# Check if flight in cruise phase, approach or landing, climb
###if(delalt<=0){
###if(alt_agl>Hmaxap){phase="CR"} else{
###if(alt_agl<=Hmaxap && alt_agl>Hmaxld){
###if(vtas2>=Vcr){phase="CR"} else{phase="AP"}} else{if(vtas2>=Vcr){phase="CR"}
else{if(vtas2>=Vap){phase="AP"} else{phase="LD"}}}}
###} else{phase="IC"};
### if(alt_feet>=2000){if(vtas2>=Vcr){phase="CR"} else{phase="AP"}}
else{if(vtas2>=Vap){phase="AP"} else{phase="LD"}};
# Altude threshold is airport specific, in case of MDW it is 1700
### if(alt_feet>=elev+1000){if(alt_feet>=3000){if(vtas2>=Vcr){phase="CR"}
else{phase="AP"}} else{phase="AP"}} else{phase="LD"};
#
if(holding==1){
if(delalt<=0){phase="CR"} else{phase="IC"}
} else{if(delalt<=0){
if(alt_feet>Hmaxap){phase="CR"} else{
if(alt_feet<=Hmaxap && alt_feet>Hmaxld){
if(vtas2>=Vcr && check1==0){phase="CR"} else{check1=1; phase="AP"}} else{phase="LD"}}}
else{check1=0; phase="IC"}};

# If descent in cruise phase then use minimum fuel flow model
if(phase=="CR" && delalt<0){
fb_t=cf3[actype]*(1-(alt_feet/cf4[actype])); # minimum fuel flow in kg/min
} else{
# Compute Drag
Cl=(2*mass1*g)/(rho1*vtas^2*s[actype]) # coefficient of lift

if(Cl>3){Cl=3}; # max lift coefficient is set to 3

# get coefficient for cruise, approach or landing
if(phase=="CR" || phase=="IC"){cd0=cd0cr[actype];cdldg=0; cd2=cd2cr[actype]} else{
if(phase=="AP"){cd0=cd0ap[actype];cdldg=0; cd2=cd2ap[actype]} else{
cd0=cd0ld[actype];cdldg=cd0ldg[actype]; cd2=cd2ld[actype]}};

Cd=cd0+cdldg+cd2*Cl^2; # Coefficient of drag

drag=(Cd*rho1*vtas^2*s[actype])/2 ; # Drag in N

# Compute forces due to mass and altitude rate
force1=mass1*accel; # Force1 (mass related) in N

force2=mass1*g*(delalt/deltime)/vtas # Force2 (altitude rate) in N

# Compute thrust
thrust=(drag+force1+force2)/1000; # total thrust in kN
# if thrust is negative then apply minimum fuel flow model
if(thrust>0){
if(abs(cf2[actype])>0){if(engine[actype]=="Jet"){
eta=cf1[actype]*(1+(vtas/cf2[actype]))} else{
eta=cf1[actype]*(1-(vtas/cf2[actype]))*(vtas2/1000)}} else{eta=cf1[actype]}; # thrust
specific fuel consumption kg/(min*N)
if(phase=="CR"){fb_t=eta*thrust*cfcr[actype]} else{fb_t=eta*thrust}; # kg/min
} else{
fb_t=cf3[actype]*(1-(alt_feet/cf4[actype])); # minimum fuel flow in kg/min
}
}

if(accel==0){yyy="do nothing"} else{fb=fb_t*(deltime/60);
cumfb=cumfb+fb;
mass1=mass1-fb;}

#print $0,rho1,Cl,Cd,drag,force1,force2,thrust,eta,fb_t,phase,fb,cumfb;
if(accel==0){print $0,"NA","NA","NA"} else{
print $0,phase,fb,cumfb,rho1,Cl,Cd,drag,force1,force2,thrust,eta,fb_t;
}}}}
```

File name: function.configbyqtr

```
# This function print out config info by every quater from 6:00 to 23:00
{
x1[FNR]=$5;
x2[FNR]=$6;
rec[FNR]=$0;
}
END{
for(i=6.25;i<=23;i=i+.25){
for(j=1;j<=length(x1);j++){
if(i>=contime(x1[j]) && i<=contime(x2[j])){
print i,rec[j];
break
}}}
}
```

File name: function.converttime

```
# This function convert time from 6|1 format to 6.25
function contime(intime){
split(intime,arr,"|");
outtime=arr[1]+(arr[2]*.25);
return outtime}
```

File name: function.degrees

```
# This function converts radian to degree
function degrees(value){return value*57.2957795}
```

File name: function.dist2thresh

```
# This function computes the cumulative distance of track from start of data point to the
end (runway threshold)
{
if($1!=id){
print $0, 0;
id=$1; lat1=$7; lon1=$8; cum_dis=0} else{
cum_dis=cum_dis+gcd(lat1,lon1,$7,$8);
print $0, cum_dis;
lat1=$7;
lon1=$8;
}
}
```

File name: function.dist_point_line

```
# This function calculates the distance from a point to a line
function getdist(lat0,lon0,lat1,lon1,lat2,lon2,alat,alon){
# lat0,lon0 are the co-ordinates of the point
# lat1,lon1,lat2,lon2 are end of the airport
# alat,alon is the center of the airport
# step convert lat lon to cartesian coordinates using flat earth approximation
getcar(alat,alon,lat0,lon0) # For the point
x0=x1;
y0=y1;
getcar(alat,alon,lat1,lon1) # For one end of the runway
xa=x1;
ya=y1;
getcar(alat,alon,lat2,lon2) # For other end of the runway
xb=x1;
yb=y1;
# Apply formula
numerator=abs((xb-xa)*(ya-y0)-(xa-x0)*(yb-ya));
denominator=sqrt((xb-xa)^2 + (yb-ya)^2);
ptoline=numerator/denominator;
return ptoline;
}
```

File name: function.dist_point_line_2

```
# This function calculates the distance from a point to a line
function getdist(lat0,lon0,lat1,lon1,lat2,lon2,alat,alon){
# lat0,lon0 are the co-ordinates of the point
# lat1,lon1,lat2,lon2 are end of the line
# alat,alon is the center of the line
# step convert lat lon to cartesian coordinates using flat earth approximation
getcar(alat,alon,lat0,lon0,scale) # For the point
x0=x1;
y0=y1;
getcar(alat,alon,lat1,lon1,scale) # For one end of the runway
xa=x1;
ya=y1;
getcar(alat,alon,lat2,lon2,scale) # For other end of the runway
xb=x1;
yb=y1;
# Apply formula
##numerator=abs((xb-xa)*(ya-y0)-(xa-x0)*(yb-ya));
numerator=((xb-xa)*(ya-y0)-(xa-x0)*(yb-ya));
###print x0,y0,xa,ya,xb,yb;
###numerator=((xb-xa)*(ya-y0)-(xa-x0)*(yb-ya));
denominator=sqrt((xb-xa)^2 + (yb-ya)^2);
ptoline=numerator/denominator;
return ptoline;
}
```

File name: function.distairport

```
# This function computes distance between flight location and a fix
# If within the specified threshold, it print the flight id
{dist=gcd(fix_lat,fix_lon,$7,$8);
if(dist<=check_radius){print $0};
## Debuggin test print
##print $0,fix_lat,fix_lon,$7,$8,dist;
}
```

File name: function.distance

```
# This function computes flight track distance
{
if($1!=id){
if(FNR!=1){
print "dummy",id,cum_dis
}; id=$1; lat1=$7; lon1=$8; cum_dis=0} else{
cum_dis=cum_dis+gcd(lat1,lon1,$7,$8);
lat1=$7;
lon1=$8;
}}
```

File name: function.distfromfix1

```
# This function computes distance between flight location and a fix
# If within the specified threshold, it print the flight id
{dist=gcd(fix_lat,fix_lon,$7,$8);
# For RNP check proximity and alignment with the fix
if(procedure=="RNP" && runway=="13C"){
# second fix as reference to capture RNP flows
fix2_lat=41.8263;
fix2_lon=-87.8987;
if(dist<=check_radius){
bearing1=bearing(fix_lat,fix_lon,$7,$8); # Check bearing to see if its parallel to the
line
bearing2=bearing(fix_lat,fix_lon,fix2_lat,fix2_lon); #
diff1=abs(bearing1-bearing2);
# diff2=abs(-132-bearing1);
if(diff1<=3){
# if perpendicular distance is less than some threhold 0.1 NM
dist2=getdist($7,$8,fix_lat,fix_lon,fix2_lat,fix2_lon,fix_lat,fix_lon);
if(abs(dist2)<=.1){if($9<2500 && $9>=2000) print $1}};
}} else{
if(procedure=="RNP"){vthres=100} else{vthres=200};
if(vertical>0){v_diff=abs(vertical-$9); if(dist<=check_radius && v_diff<vthres){print
$1}} else{if(dist<=check_radius){print $1}};
};
## Debuggin test print
##print $0,fix_lat,fix_lon,$7,$8,dist;
}
```

File name: function.distfromfix2

```
# This function computes distance between flight location and a fix
# If within the specified threshold, it print the flight id
{dist=getdist($7,$8,lat1,lon1,lat2,lon2,fix_lat,fix_lon);
##print $7,$8,lat1,lon1,lat2,lon2,fix_lat,fix_lon;
print $0,dist;
## Debuggin test print
##print $0,fix_lat,fix_lon,$7,$8,dist;
}
```

File name: function.distimefix2thresh

```
# This function compute the distance and time from a fix to the runway threshold

{if($1!=x){
if(FNR!=1){
asort(fdisarr);
# Computing distance exactly from the fix line
# Adding additional distance from the fix line to the closest point to the fix line
disfix2thresh=dis2thresh-tdisarr[fdisarr[1]];
timefix2thresh=(time2thresh-ttimearr[fdisarr[1]])/60;
##disfix2thresh=dis2thresh-disnearfix;
##timefix2thresh=(time2thresh-timenearfix)/60;
## debug print
## print x,dis2thresh,disnearfix,disfix2thresh,time2thresh,timenearfix,timefix2thresh;
print x,disfix2thresh,timefix2thresh;
delete tdisarr;
delete ttimearr;
delete fdisarr;
##check=0;
dis2thresh=$11;
##disfix=$12;
time2thresh=$2;};x=$1} else{
###if(check==0){if($12>disfix){disnearfix=dis2thresh; timenearfix=time2thresh; check=1}};
dis2thresh=$11;
###disfix=$12;
time2thresh=$2
tdisarr[$12]=$11;
ttimearr[$12]=$2
fdisarr[$12]=$12;
}}

END{
###disfix2thresh=dis2thresh-disnearfix;
###timefix2thresh=(time2thresh-timenearfix)/60;
### debug print
### print x,dis2thresh,disnearfix,disfix2thresh,time2thresh,timenearfix,timefix2thresh;
###print x,disfix2thresh,timefix2thresh;
asort(fdisarr);
disfix2thresh=dis2thresh-tdisarr[fdisarr[1]];
timefix2thresh=(time2thresh-ttimearr[fdisarr[1]])/60;
print x,disfix2thresh,timefix2thresh;
}
```

File name: function.distimefix2thresh_2

```
# This function compute the distance and time from a fix to the runway threshold
{if($1!=x){
if(FNR!=1){
asort(fdisarr_indx,fdisarr2);
##for(i=1;i<=length(fdisarr2);i++){print i,fdisarr2[i]};
# Compute distance error from perpendicular of fix
add_dist=fdisarr[fdisarr2[1]];
##if(direction=="W"){add_dist=dist_error*-1} else{add_dist=dist_error};
# Compute time error from perpendicular of fix
# Compute velocity
#fix_dist=abs(fdisarr[fdisarr2[1]]-fdisarr[fdisarr2[2]]); # Distance flown around the fix
#fix_time=abs(ttimearr[fdisarr2[1]]-ttimearr[fdisarr2[2]])/60; # Time flown around the
fix
#fix_velocity=fix_dist/fix_time;
#add_time=add_dist/fix_velocity;
add_time=add_dist/4; # 4 nautical miles per minute average terminal velocity close to the
corner post
# Computing distance exactly from the fix line
# Adding additional distance from the fix line to the closest point to the fix line
disfix2thresh=dis2thresh-tdisarr[fdisarr2[1]]+add_dist;
timefix2thresh=((time2thresh-ttimearr[fdisarr2[1]])/60)+add_time;
##disfix2thresh=dis2thresh-disnearfix;
##timefix2thresh=(time2thresh-timenearfix)/60;
## debug print
#print
x,dist_error,add_dist,fix_dist,fix_time,fix_velocity,add_time,dis2thresh,disnearfix,disfi
x2thresh,time2thresh,timenearfix,timefix2thresh;
print x,disfix2thresh,timefix2thresh;
delete tdisarr;
delete ttimearr;
delete fdisarr;
delete fdisarr_indx;
delete fdisarr2;
##check=0;
dis2thresh=$11;
##disfix=$12;
time2thresh=$2;
tdisarr[abs($12)]=$11;
ttimearr[abs($12)]=$2
fdisarr[abs($12)]=$12;
fdisarr_indx[abs($12)]=abs($12);
};x=$1} else{
###if(check==0){if($12>disfix){disnearfix=dis2thresh; timenearfix=time2thresh; check=1}};
dis2thresh=$11;
###disfix=$12;
time2thresh=$2
tdisarr[abs($12)]=$11;
ttimearr[abs($12)]=$2
fdisarr[abs($12)]=$12;
fdisarr_indx[abs($12)]=abs($12);
}}

END{
###disfix2thresh=dis2thresh-disnearfix;
###timefix2thresh=(time2thresh-timenearfix)/60;
### debug print
### print x,dis2thresh,disnearfix,disfix2thresh,time2thresh,timenearfix,timefix2thresh;
###print x,disfix2thresh,timefix2thresh;
asort(fdisarr_indx,fdisarr2);
##for(i=1;i<=length(fdisarr2);i++){print i,fdisarr2[i]};
# Compute distance error from perpendicular of fix
add_dist=fdisarr[fdisarr2[1]];
##if(direction=="W"){add_dist=dist_error*-1} else{add_dist=dist_error};
# Compute time error from perpendicular of fix
# Compute velocity
#fix_dist=abs(fdisarr[fdisarr2[1]]-fdisarr[fdisarr2[2]]); # Distance flown around the fix
#fix_time=abs(ttimearr[fdisarr2[1]]-ttimearr[fdisarr2[2]])/60; # Time flown around the
fix
#fix_velocity=fix_dist/fix_time;
#add_time=add_dist/fix_velocity;
add_time=add_dist/4; # 4 nautical miles per minute average terminal velocity close to the
corner post
# Computing distance exactly from the fix line
# Adding additional distance from the fix line to the closest point to the fix line
disfix2thresh=dis2thresh-tdisarr[fdisarr2[1]]+add_dist;
timefix2thresh=((time2thresh-ttimearr[fdisarr2[1]])/60)+add_time;
##disfix2thresh=dis2thresh-disnearfix;
##timefix2thresh=(time2thresh-timenearfix)/60;
## debug print
# print
x,dist_error,add_dist,fix_dist,fix_time,fix_velocity,add_time,dis2thresh,disnearfix,disfi
x2thresh,time2thresh,timenearfix,timefix2thresh;
print x,disfix2thresh,timefix2thresh;
}
```

File name: function.fb_actype

```
# This function computes fuel burn for each aircraft type
{if($29!=x){
if(FNR!=1){
minmaxmeansd(fb);
print actype,mtow,engine,n,min,max,average,sd
delete fb;
}; x=$29;
actype=$29;
mtow=$33;
engine=$34;
fb[FNR]=$30;} else{
fb[FNR]=$30
}
}
```

File name: function.fb_stats1

```
# This function computes and prints stats for fuel burn
{if(FNR==1){start=int($timefield);
print direction""runway""procedure};
if($timefield<=start+inc){
fb1[FNR]=$fbfield} else{
if(length(fb1)>0){
minmaxmeansd(fb1);
if(n>0){con=1.96*(sd/sqrt(n))} else{con=0};
print start+inc,average,sd,n,con;
start=start+inc;
delete fb1} else{
print start+inc,0,0,0,0;
start=start+inc;
delete fb1};if($timefield<=start+inc){
fb1[FNR]=$fbfield} else{
print start+inc,0,0,0,0;
start=start+inc;
delete fb1}}}
```

File name: function.fuelburn_levelsegment

```
# This script get the fuel burn perflight and the total time spent in level  phase for
all approaches.
{if($1!=x){
if(FNR!=1){
if(total_fb=="NA"){print x,"NA","NA","NA","NA","NA","NA","NA";
} else{
# get the cumulative fuel burn closest to the beam of the corner post
asort(fdisarr_indx,fdisarr2);
if(beam_dist<0){
actual_fb=total_fb-cum_fb[fdisarr2[1]];
} else{
actual_fb=total_fb;
};
# print results
print
x,actual_fb,time_level/60,fuel_level,time_nonlevel/60,fuel_nonlevel,time_final/60,fuel_fi
nal;
}
};
x=$1;
# delete all arrays
delete cum_fb;
delete fdisarr_indx;
delete fdisarr2;
delete fdisarr;
# Initialize array and variable
beam_dist=$10;
cum_fb[abs($10)]=$21;
fdisarr[abs($10)]=$10;
fdisarr_indx[abs($10)]=abs($10);
time_level=0;
fuel_level=0;
time_nonlevel=0;
fuel_nonlevel=0;
time_final=0;
fuel_final=0;
z=0; # record after it crosses the last way point on STAR
# Compute time in level phase not decelerating
#
if($7>=final){ # This specified as variable to the function and is the start of the final
approach
if($10>0){z++; if(z>1){if($14>=0){if($17<=-0.1){
time_nonlevel=time_nonlevel+$13;
fuel_nonlevel=fuel_nonlevel+$20;
} else{
time_level=time_level+$13;
fuel_level=fuel_level+$20;
}} else{
time_nonlevel=time_nonlevel+$13;
fuel_nonlevel=fuel_nonlevel+$20;
}}}} else{
time_final=time_final+$13;
fuel_final=fuel_final+$20;}} else{
# record the cumulative fuel burn and distance from corner post
total_fb=$21;
cum_fb[abs($10)]=$21;
fdisarr[abs($10)]=$10;
fdisarr_indx[abs($10)]=abs($10);
# do not record level phase if in cruise phase and decelerating more than .1 m/s^2
if($7>=final){
if($10>0){z++; if(z>1){if($14>=0){if($17<=-0.1){
time_nonlevel=time_nonlevel+$13;
fuel_nonlevel=fuel_nonlevel+$20;
} else{
time_level=time_level+$13;
fuel_level=fuel_level+$20;
}} else{
time_nonlevel=time_nonlevel+$13;
fuel_nonlevel=fuel_nonlevel+$20;
}}}} else{
time_final=time_final+$13;
fuel_final=fuel_final+$20;};

}} END{
if(total_fb=="NA"){print x,"NA","NA","NA","NA","NA","NA","NA"} else{
# get the cumulative fuel burn closest to the beam of the corner post
asort(fdisarr_indx,fdisarr2);
if(beam_dist<0){
actual_fb=total_fb-cum_fb[fdisarr2[1]];
} else{
actual_fb=total_fb;
};

# print results
print
x,actual_fb,time_level/60,fuel_level,time_nonlevel/60,fuel_nonlevel,time_final/60,fuel_fi
nal;}
}
```

File name: function.gcd_haversine

```
# This functin compute gcd between two points on earth
# The two points should be specified in lat lon format
# The radius of earth is assumed to be 3443.89849 nautical miles

function gcd(lat1,lon1,lat2,lon2){rad=3.1416/180; x1=lat1*rad; y1=lon1*rad; x2=lat2*rad;
y2=lon2*rad; r=3443.89849+5; dellat=x1-x2; dellon=y1-y2;
centralangle=2*asin(sqrt(sin(dellat/2)^2 + cos(x1)*cos(x2)*sin(dellon/2)^2));
distance=r*centralangle; return distance}
```

File name: function.gcd_haversine_2

```
# This functin compute gcd between two points on earth
# The two points should be specified in lat lon format
# The radius of earth is assumed to be 3443.89849 nautical miles

function gcd(lat1,lon1,lat2,lon2){
rad=3.1416/180;
x1=lat1*rad;
y1=lon1*rad;
x2=lat2*rad;
y2=lon2*rad;
r=3443.89849;
dellat=x1-x2;
dellon=y1-y2;
a=sin(dellat/2)*sin(dellat/2)+cos(x1)*cos(x2)*sin(dellon/2)*sin(dellon/2);
centralangle=2*atan2(sqrt(a),sqrt(1-a));
distance=r*centralangle;
return distance}
```

File name: function.get_rank_optimal_runway

```
# From the ASPM data, this function get the runway most optimal for the given wind.
# i.e. runway that has the best head wind for a given cross wind threshold
#
{if(NR==FNR){rwy[FNR]=$1} else{
# for each runway compute the cross wind and tail wind
# get wind direction and speed
wind_direction=radian($11);
wind_speed=$12;
for(i=1;i<=length(rwy);i++){
rwy_bearing=radian(rwy[i]*10);
a1=abs(rwy_bearing-wind_direction); # is the difference in wind and rwy bearing
cross_wind=abs(sin(a1)*wind_speed);
if(cross_wind<=20){
tail_wind=cos(a1)*wind_speed;
if(tail_wind>0){
rwy3=rwy[i];
break;
}
####rwy2[tail_wind[i]]=rwy[i];
}
}
#####n=asort(tail_wind);
#####rwy3=rwy2[tail_wind[n]];
print $0,rwy3
}}
```

File name: function.get_track_profile_info

```
# This script compute for each track, at each time step
# change in time
# distance
# altitude
# velocity
# acceleration
{
if(x!=$1){
x=$1;
i=1;
time=$2;
lat=$7;
lon=$8;
alt=$9;
wind_dir=$13;
wind_mag=$14;
print $1,$2,$3,$4,$7,$8,$9,$10,$11,$12,$13,$14,0,0,0,0,0
deltime1=0;
speed1=0;
} else{
deltime2=$2-time;
if(deltime2>=30){
###dist1=gcd(lat,lon,$7,$8)*1852; # Convert to meter
delalt=$9-alt;
# Altitude above ground
alt_g=abs($9-elev);
###speed2=(dist1/deltime2); # in m/s
speed2=airspeed(lat,lon,$7,$8,deltime2,alt_g,$13,$14);
# Implementing simple differential relaxation equation based smoothings
tau=20 # smoothing parameter
dt=deltime2/tau
# alpha=deltime1/(deltime1+deltime2);
#alpha=0 # no smoothning
#if(i>1){alpha=0} else{alpha=0};
# alpha=speed1/(speed1+speed2);
#speed3=alpha*speed1+(1-alpha)*speed2;
# speed3=(speed2+speed1+speed1b+speed1c)/4;
if(i>1){
speed3=(speed1+dt*speed2)/(1+dt);
##speed3=speed2;
delspeed=(speed3-speed1)/deltime2} else{speed3=speed2; delspeed=0}; # in m/s^2
print $1,$2,$3,$4,$7,$8,$9,$10,$11,$12,$13,$14,deltime2,delalt,dist1,speed3,delspeed;
##print
$1,$2,$3,$4,$7,$8,$9,$10,$11,$12,$13,$14,deltime2,delalt,dist1,speed3,delspeed,f_bearing_
rad,w_bearing_rad,w_speed1,w_speed2;
i++;
deltime1=deltime2
speed1=speed3;
time=$2;
lat=$7;
lon=$8;
alt=$9;} else{yy="do nothing"}
}
}
```

File name: function.get_wind_optimal_runway

```
# From the ASPM data, this function get the runway most optimal for the given wind.
# i.e. runway that has the best head wind for a given cross wind threshold
#
{if(NR==FNR){rwy[FNR]=$1} else{
# for each runway compute the cross wind and tail wind
# get wind direction and speed
wind_direction=radian($11);
wind_speed=$12;
# initialize tail wind array
delete tail_wind;
for(i=1;i<=length(rwy);i++){
rwy_bearing=radian(rwy[i]*10);
a1=abs(rwy_bearing-wind_direction); # is the difference in wind and rwy bearing
cross_wind=abs(sin(a1)*wind_speed);
if(cross_wind<=20){
tail_wind[i]=cos(a1)*wind_speed
rwy2[tail_wind[i]]=rwy[i];
}
}
n=asort(tail_wind);
rwy3=rwy2[tail_wind[n]];
print $0,rwy3
}}
```

File name: function.getactual_dist

```
# This function print out the actual arr dep distributiona at chicago metroplex
BEGIN{
bin=15*60;
arr[1]="22L_13C";
arr[2]="22L_Others";
arr[3]="28_13C";
arr[4]="28_Others";
arr[5]="13C_ILS";
arr[6]="13C_RNP";
arr[7]="13C_Visual";
for(i=1;i<=length(arr);i++){
count[i]=0;
if(i==1){
printf "%s\t%s\t%s","Date","Bin",arr[i]} else{
if(i==length(arr)){
printf "\t%s\n",arr[i]} else{
printf "\t%s",arr[i]}}}}

{if(FNR==1){binstart=$2;
binend=binstart+bin};
if($2>=binstart && $2<binend){
for(i=1;i<=length(arr);i++){
if($NF==arr[i]){count[i]++}}} else{
for(i=1;i<=length(arr);i++){
if(i==1){printf "%s\t%s\t%s",strftime("%F",binend,UTC-1),strftime("%T",binend,UTC-
1),count[i]} else{
if(i==length(arr)){printf "\t%s\n", count[i]} else{
```

```
printf "\t%s",count[i]}}; count[i]=0};
binstart=binend;
binend+=bin;
for(i=1;i<=length(arr);i++){
if($NF==arr[i]){count[i]++}}}}
END{
for(i=1;i<=length(arr);i++){
if(i==1){printf "%s\t%s\t%s",strftime("%F",binend,UTC-1),strftime("%T",binend,UTC-
1),count[i]} else{
if(i==length(arr)){printf "\t%s\n", count[i]} else{
printf "\t%s",count[i]}}}}
```

File name: function.getcartesian

```
# This function converts lat lon to cartesian co-ordinates using flat earth approximation
# lat1 and lon1 are co-ordinates of the airport
# lat2 and lon2 are the lat lon to be converted
function getcar(lat1,lon1,lat2,lon2){
lat0=lat1-.2;
lon0=lon1-.2;
x1=((lon2-lon0)*cos(lat1*0.0174532925))*60; # Unit nautical miles
y1=(lat2-lat0)*60; # Unit nautical miles
}
```

File name: function.getcartesian_2

```
# This function converts lat lon to cartesian co-ordinates using flat earth approximation
# lat1 and lon1 are co-ordinates of the airport, used to interpolate
# lat2 and lon2 are the lat lon to be converted
# scale decides the distance of the origin from the airport center
# Larger the scale lesser the accuracy
# e.g scale can be .2 for co-ordinates inside the airport, and upto 4 for considering
whole area of NOP data
function getcar(lat1x,lon1x,lat2x,lon2x,scale){
lat0x=lat1x-scale;
lon0x=lon1x-scale;
x1=((lon2x-lon0x)*cos(lat1x*0.0174532925))*60; # Unit nautical miles
y1=(lat2x-lat0x)*60; # Unit nautical miles
}
```

File name: function.getcolor

```
# This function gets the color code for the specified color
function getcolor(color){
arr["indianred"]="ff5c5ccd";
arr["salmon"]="ff7280fa";
arr["red"]="ff0000ff";
arr["crimson"]="ff3c14dc";
arr["firebrick"]="ff2222b2";
arr["darkred"]="ff00008b";
arr["pink"]="ffcbc0ff";
arr["hotpink"]="ffb469ff";
arr["deeppink"]="ff9314ff";
arr["mediumvioletred"]="ff8515c7";
arr["palevioletred"]="ff9370db";
arr["coral"]="ff507fff";
arr["tomato"]="ff4763ff";
arr["orangered"]="ff0045ff";
arr["orange"]="ff00a5ff";
arr["gold"]="ff00d7ff";
arr["yellow"]="ff00ffff";
arr["peachpuff"]="ffb9daff";
arr["khaki"]="ff8ce6f0";
```

```
arr["darkkhaki"]="ff6bb7bd";
arr["thistle"]="ffd8bfd8";
arr["plum"]="ffdda0dd";
arr["violet"]="ffee82ee";
arr["orchid"]="ffd670da";
arr["magenta"]="ffff00ff";
arr["mediumpurple"]="ffdb7093";
arr["blueviolet"]="ffe22b8a";
arr["darkviolet"]="ffd30094";
arr["darkorchid"]="ffcc3299";
arr["darkmagenta"]="ff8b008b";
arr["purple"]="ff800080";
arr["indigo"]="ff82004b";
arr["darkslateblue"]="ff8b3d48";
arr["slateblue"]="ffcd5a6a";
arr["greenyellow"]="ff2fffad";
arr["lime"]="ff00ff00";
arr["limegreen"]="ff32cd32";
arr["lightgreen"]="ff90ee90";
arr["mediumspringgreen"]="ff9afa00";
arr["seagreen"]="ff578b2e";
arr["green"]="ff008000";
arr["darkgreen"]="ff006400";
arr["yellowgreen"]="ff32cd9a";
arr["olivedrab"]="ff238e6b";
arr["olive"]="ff008080";
arr["darkolivegreen"]="ff2f6b55";
arr["darkseagreen"]="ff8fbc8f";
arr["lightseagreen"]="ffaab220";
arr["darkcyan"]="ff8b8b00";
arr["teal"]="ff808000";
arr["cyan"]="ffffff00";
arr["paleturquoise"]="ffeeeeaf";
arr["aquamarine"]="ffd4ff7f";
arr["turquoise"]="ffd0e040";
arr["darkturquoise"]="ffd1ce00";
arr["cadetblue"]="ffa09e5f";
arr["steelblue"]="ffb48246";
arr["lightsteelblue"]="ffdec4b0";
arr["lightblue"]="ffe6d8ad";
arr["skyblue"]="ffebce87";
arr["deepskyblue"]="ffffbf00";
arr["dodgerblue"]="ffff901e";
arr["cornflowerblue"]="ffed9564";
arr["royalblue"]="ffe16941";
arr["blue"]="ffff0000";
arr["darkblue"]="ff8b0000";
arr["wheat"]="ffb3def5";
arr["burlywood"]="ff87b8de";
arr["tan"]="ff8cb4d2";
arr["rosybrown"]="ff8f8fbc";
arr["sandybrown"]="ff60a4f4";
arr["goldenrod"]="ff20a5da";
arr["darkgoldenrod"]="ff0b86b8";
arr["peru"]="ff3f85cd";
arr["chocolate"]="ff1e69d2";
arr["saddlebrown"]="ff13458b";
arr["sienna"]="ff2d52a0";
arr["brown"]="ff2a2aa5";
arr["maroon"]="ff000080";
arr["white"]="ffffffff";
arr["lightgray"]="ffd3d3d3";
arr["silver"]="ffc0c0c0";
arr["darkgray"]="ffa9a9a9";
arr["gray"]="ff808080";
arr["slategray"]="ff908070";
arr["darkslategray"]="ff4f4f2f";
```

```
arr["black"]="ff000000";
if(arr[color]!=""){hexcode=arr[color]} else{hexcode=arr["red"]};
return hexcode}
```

File name: function.getconfigstats

```
# This function computes for each day
# 1. Duration of runway configuration
# 2. Stats on meteorological conditions (ceiling, visibilty, wind angle, wind speed)
# 2. Airport Arrival Rate and Airport Departure Rate
BEGIN{FS="\t"}
{
#print FNR;
if(FNR==1){
config=$12"|"$13;
hour=$4
airport=$1;
year=substr($2,1,4);
month=substr($2,5,2);
day=$3;
time_start=$4"|"$5;
mc[1]=0; ceiling[1]=0; wind_angle[1]=0; wind_speed[1]=0; aar[1]=0; adr[1]=0};

if(config!=$12"|"$13 || abs(hour-$4)>1){
# Compute stats and print output
rec=length(mc);
# print rec;
if(rec>1){
# Get MC count
IMC=0; VMC=0
for(i=1; i<=length(mc);i++){if(mc[i]=="I"){IMC++} else{VMC++}};
# print IMC, VMC;
# get min max mean and std of ceiling
if(length(ceiling)>1){
minmaxmeansd(ceiling);
# print min, max, average, sd;
# for(no in ceiling){print no, ceiling[no]};
ceilingmin=min;
ceilingmax=max;
ceilingmean=average;
ceilingsd=sd} else{
ceilingmin=ceiling[1];
ceilingmax=ceiling[1];
ceilingmean=ceiling[1];
ceilingsd=0};
# get min max mean and std of visibility
minmaxmeansd(visibility);
# print min, max, average, sd;
visibilitymin=min;
visibilitymax=max;
visibilitymean=average;
visibilitysd=sd;
# get min max mean and std of wind_angle
minmaxmeansd(wind_angle);
# print min, max, average, sd;
wind_anglemin=min;
wind_anglemax=max;
wind_anglemean=average;
wind_anglesd=sd;
# get min max mean and std of wind_speed
minmaxmeansd(wind_speed);
wind_speedmin=min;
wind_speedmax=max;
wind_speedmean=average;
wind_speedsd=sd;
# get min max mean and std of aar
```

```
minmaxmeansd(aar);
aarmin=min;
aarmax=max;
aarmean=average;
aarsd=sd;
# get min max mean and std of adr
minmaxmeansd(adr);
#print min, max, average, sd;
adrmin=min;
adrmax=max;
adrmean=average;
adrsd=sd;
# print output
print
airport,year,month,day,time_start,time_end,config,VMC*100/rec"|"IMC*100/rec,ceilingmin"|"
ceilingmax"|"ceilingmean"|"ceilingsd,visibilitymin"|"visibilitymax"|"visibilitymean"|"vis
ibilitysd,wind_anglemin"|"wind_anglemax"|"wind_anglemean"|"wind_anglesd,wind_speedmin"|"w
ind_speedmax"|"wind_speedmean"|"wind_speedsd,aarmin"|"aarmax"|"aarmean"|"aarsd,adrmin"|"a
drmax"|"adrmean"|"adrsd;
} else{
xxxx="donothing"};

# Record new set of data
config=$12"|"$13;
hour=$4;
airport=$1;
year=substr($2,1,4);
month=substr($2,5,2);
day=$3
time_start=$4"|"$5;
# delete all array
delete mc; delete ceiling; delete wind_angle; delete wind_speed; delete aar; delete adr;
# initialize array
mc[1]=0; ceiling[1]=0; wind_angle[1]=0; wind_speed[1]=0; aar[1]=0; adr[1]=0;
# start recording into arrays again
i=1;
mc[i]=$6; # meteorological conditions
if($7!="na"){ceiling[i]=$7}; # ceiling
if($8!="-"){visibility[i]=$8}; # visibility
if($10!="-"){wind_angle[i]=$10}; # wind angle
if($11!="-"){wind_speed[i]=$11}; # wind speed
aar[i]=$16; # airport arrival rate
adr[i]=$17; # airport departure rate

} else{
hour=$4;
# Record data to compute stats
time_end=$4"|"$5; # end time
i++;
mc[i]=$6; # meteorological conditions
if($7!="na"){ceiling[i]=$7}; # ceiling
if($8!="-"){visibility[i]=$8}; # visibility
if($10!="-"){wind_angle[i]=$10}; # wind angle
if($11!="-"){wind_speed[i]=$11}; # wind speed
aar[i]=$16; # airport arrival rate
adr[i]=$17; # airport departure rate
}} END{
# Compute stats and print output
rec=length(mc);
# print rec;
if(rec>1){
# Get MC count
IMC=0; VMC=0
for(i=1; i<=length(mc);i++){if(mc[i]=="I"){IMC++} else{VMC++}};
# print IMC, VMC;
# get min max mean and std of ceiling
if(length(ceiling)>1){
```

```
minmaxmeansd(ceiling);
# print min, max, average, sd;
# for(no in ceiling){print no, ceiling[no]};
ceilingmin=min;
ceilingmax=max;
ceilingmean=average;
ceilingsd=sd} else{
ceilingmin=ceiling[1];
ceilingmax=ceiling[1];
ceilingmean=ceiling[1];
ceilingsd=0};
# get min max mean and std of visibility
minmaxmeansd(visibility);
# print min, max, average, sd;
visibilitymin=min;
visibilitymax=max;
visibilitymean=average;
visibilitysd=sd;
# get min max mean and std of wind_angle
minmaxmeansd(wind_angle);
# print min, max, average, sd;
wind_anglemin=min;
wind_anglemax=max;
wind_anglemean=average;
wind_anglesd=sd;
# get min max mean and std of wind_speed
minmaxmeansd(wind_speed);
wind_speedmin=min;
wind_speedmax=max;
wind_speedmean=average;
wind_speedsd=sd;
# get min max mean and std of aar
minmaxmeansd(aar);
aarmin=min;
aarmax=max;
aarmean=average;
aarsd=sd;
# get min max mean and std of adr
minmaxmeansd(adr);
#print min, max, average, sd;
adrmin=min;
adrmax=max;
adrmean=average;
adrsd=sd;
# print output
print
airport,year,month,day,time_start,time_end,config,VMC*100/rec"|"IMC*100/rec,ceilingmin"|"
ceilingmax"|"ceilingmean"|"ceilingsd,visibilitymin"|"visibilitymax"|"visibilitymean"|"vis
ibilitysd,wind_anglemin"|"wind_anglemax"|"wind_anglemean"|"wind_anglesd,wind_speedmin"|"w
ind_speedmax"|"wind_speedmean"|"wind_speedsd,aarmin"|"aarmax"|"aarmean"|"aarsd,adrmin"|"a
drmax"|"adrmean"|"adrsd;
} else{
xxxx="donothing"}}
```

File name: function.getdistribution

```
# This function generates distribution of track mile and track time for each flow
#
BEGIN{
delete arr;
delete arr2;
delete dist;
##print runway, direction,procedure;
}
{
if($22==runway && $23==direction && $24==procedure){
```

```
arr[FNR]=$fieldno;}
} END{
##for (no in arr){print no, arr[no]};
if(length(arr)>50){
n=asort(arr,arr2);
start=0
for(i=1; i<=n; i++){
if(arr2[i]>start && arr2[i]<=start+bin){
dist[start" "start+bin]++
} else{
start=start+bin;
while(arr2[i]>start+bin){
dist[start" "start+bin]=0;
start=start+bin;};
dist[start" "start+bin]++}};
for(no in dist){print no, dist[no], dist[no]/n};
#for(no in arr2){print no, arr2[no]};
}}
```

File name: function.geteq

```
# given two lat lon, this script computes the equation of line for the two points
{if($1==airport && $2$3~runway){
e1lat=$4+$5/60;
e1lon=-1*($6+$7/60);
getcar(lat,lon,e1lat,e1lon);
e1x=x1;
e1y=y1;
e2lat=$8+$9/60;
e2lon=-1*($10+$11/60);
getcar(lat,lon,e2lat,e2lon);
e2x=x1;
e2y=y1;
m=(e2y-e1y)/(e2x-e1x);
c=e1y-m*e1x;
print airport,runway,m,c;
}}
```

File name: function.geteq_2

```
# given two lat lon, this script computes the equation of line for the two points
{if($1==airport){
getcar(lat,lon,$8,$9);
e1x=x1;
e1y=y1;
getcar(lat,lon,$10,$11);
e2x=x1;
e2y=y1;
m=(e2y-e1y)/(e2x-e1x);
c=e1y-m*e1x;
printf "%s %s %s %f %f\n", airport,$6,$7,m,c;
}}
```

File name: function.getfbrwyconfig1

```
# This function average fuel burn per flight for a given runway configuration
BEGIN{
if(flow_scope>1){
delete condition;
delete runway;
delete app;
delete east;
delete west;
}
}{
if(NR==FNR){
# record fuel burn per flow
condition[FNR]=$1;
runway[FNR]=$2;
app[FNR]=$3;
east[FNR]=$4;
west[FNR]=$5;
} else{
# Filter out valid flows
# Initialize arr
### if(FNR<20){
delete rwy_index;
delete app_index;
delete fb_index;
delete fb_actual;
j=1;
for(i=1;i<=length(runway);i++){
# while the data provide the runway information, the information on the type of approach
has to be determined
# The type of approach is decided based on fuel burn ranking
# Based on the east west flow ratio compute average fuel burn for each approach type
avg_fb=$16*east[i]+(1-$16)*west[i]; # average fuel burn

# record fuel for all possible approaches given the meteorological conditions
if($7==condition[i]){
rwy_index[avg_fb]=runway[i]; # record runway info as function of fuel burn
app_index[avg_fb]=app[i]; # record app information as a function of fuel burn
fb_index[avg_fb]=avg_fb; # record fuel burn as a function of runway
}


# record fuel burn for actual configuration and meteorological conditions
check=0
if($17==runway[i] && $7==condition[i]){
fb_actual[j]=avg_fb
app_actual[avg_fb]=app[i];
check=1;
j++; # increment index
} else{if(check==0){
fb_actual[j]=999; # If runway is not in the list
app_actual[999]=999;
}}
}

# For each scope get the average runway fuel burn for actual runway configuration
asort(fb_actual,fb_actual_2);
actual_fb=fb_actual_2[1];
actual_app=app_actual[actual_fb];


# get optimal runway and the corresponding fuel burn
if(flow_scope>1){
asort(fb_index,fb_index_2);

wind_direction=radian($11);
wind_speed=$12;
for(i=1;i<=length(fb_index_2);i++){
rwy_bearing=radian(rwy_index[fb_index_2[i]]*10);
a1=abs(rwy_bearing-wind_direction); # is the difference in wind and rwy bearing
cross_wind=abs(sin(a1)*wind_speed);
if(cross_wind<=20){
tail_wind=cos(a1)*wind_speed;
if(tail_wind>0){
opt_fb=fb_index_2[i];
opt_rwy=rwy_index[opt_fb];
opt_app=app_index[opt_fb];
break;
}}}}

# Print results
if(flow_scope==1){
print $0,actual_fb; # for scope1
} else{
print $0,actual_app,actual_fb,opt_rwy,opt_app,opt_fb; # for scope 2 and 3
##if(flow_scope==3){
##for(no in rwy_index){print no, rwy_index[no],app_index[no]};
##for(no in fb_index_2){print no, fb_index_2[no]};
##}
};

##} else {exit}
}
}
```

File name: function.getflowstats

```
# This function prints out flow stats
#
BEGIN{
delete arr;
##print runway, direction,procedure;
}
{
if(level==1){
if($22==runway){
arr[FNR]=$fieldno}} else{
if(level==2){
if($22==runway && $24==procedure){
arr[FNR]=$fieldno}} else{
if(level==3){
if($22==runway && $23==direction && $24==procedure){
arr[FNR]=$fieldno}}}}} END{
##for (no in arr){print no, arr[no]};
if(length(arr)>0){
minmaxmeansd(arr);
print direction,runway,procedure,n,min,max,average,sd;
}}
```

File name: function.getheader

```
# This function get the header information of a file
{if(FNR==1){for(i=1;i<=NF;i++){print i,$i}} else{exit}}
```

File name: function.getlatlon

```
# This function converts cartesian coordinates to lat lon
# lat1 and lon1 are co-ordinates of the airport, used to interpolate
# lat2 and lon2 are the lat lon to be converted
# scale decides the distance of the origin from the airport center
# e.g scale can be .2 for co-ordinates inside the airport, and upto 4 for considering
whole area of NOP data
function getlatlon(lat1x,lon1x,x1,y1,scale){
lat0x=lat1x-scale;
lon0x=lon1x-scale;

lon2x=(x1/(cos(lat1x*0.0174532925)*60))+lon0x; # Unit degrees
lat2x=(y1/60)+lat0x # Unit degrees
}
```

File name: function.getrunway

```
# This function assigns runway to each track co-ordinate
{if(NR==FNR){runway=$2; m=$3; c=$4} else{
if($10=="D"){flt_bearing=bearing($7,$8,$17,$18)}
else{flt_bearing=bearing($17,$18,$7,$8)}; # get bearing information for flight track
if(flt_bearing<0){flt_bearing=flt_bearing+360}; # convert to 360 degreed
if(runway~"L" || runway~"R" || runway~"C"){rwy_bearing=substr(runway,1,length(runway)-
1)*10} else{rwy_bearing=runway*10}; # Get runway bearing
diff_bearing=flt_bearing-rwy_bearing;
if(diff_bearing<0){diff_bearing=diff_bearing*-1};
getcar(lat,lon,$7,$8);
upper=y1-m*x1-c-up;
lower=y1-m*x1-c+down;
if(upper<0 && lower>0 && diff_bearing<=25){
print $1,runway,m,c,y1,x1,upper,lower,flt_bearing} else{print
$1,"NA",m,c,y1,x1,upper,lower,flt_bearing}
}}
```

File name: function.getrunway_2

```
# This function assigns runway to each track id
# BEGIN{delete runway1; delete runway2; delete lat1; delete lon1; delete lat2; delete
lon2; delete dist}
{if(NR==FNR){runway1[FNR]=$1; runway2[FNR]=$2; lat1[FNR]=$3; lon1[FNR]=$4; lat2[FNR]=$5;
lon2[FNR]=$6} # read the runway coordinates to an array
else{
# Caculate the distance of the last two track hit from the runway centerline
if(FNR!=1){delete dist};
for(i=1;i<=length(runway1);i++){
if(runway1[i]=="13C" || runway1[i]=="31C" || runway2[i]=="13R" || runway2[i]=="31L"){
# The runway 13C and 13R are too close to each other
# Therefore distance of only the last point is computed and not the cumulative distance
# This approach shows better results
if(arr==1){
dis_pnt2=getdist($17,$18,lat1[i],lon1[i],lat2[i],lon2[i],lat,lon);
dist[dis_pnt2]=i} else{
dis_pnt1=getdist($7,$8,lat1[i],lon1[i],lat2[i],lon2[i],lat,lon);
dist[dis_pnt1]=i;}}
# For all other runways the cumulative distance of last two point is considered
 else{
dis_pnt1=getdist($7,$8,lat1[i],lon1[i],lat2[i],lon2[i],lat,lon);
dis_pnt2=getdist($17,$18,lat1[i],lon1[i],lat2[i],lon2[i],lat,lon);
dist[dis_pnt1+dis_pnt2]=i;
}};
# for(no in dist){print no, dist[no]};
# get the runway index with the minimum distance
n=asorti(dist,dist2);
rwy_index=dist[dist2[1]];
# print rwy_index;

# Calculate the bearing of flight track
flt_bearing=bearing($7,$8,$17,$18); # get bearing information for flight track
if(flt_bearing<0){flt_bearing=flt_bearing+360}; # convert to 0 - 360 degrees format
#print flt_bearing;
# Get runway bearing
if(runway1[rwy_index]~"L" || runway1[rwy_index]~"R" || runway1[rwy_index]~"C"){
rwy1_bearing=substr(runway1[rwy_index],1,length(runway1[rwy_index])-1)*10;
rwy2_bearing=substr(runway2[rwy_index],1,length(runway2[rwy_index])-1)*10} else{
rwy1_bearing=runway1[rwy_index]*10;
rwy2_bearing=runway2[rwy_index]*10};
#print runway1[rwy_index],runway2[rwy_index],rwy1_bearing,rwy2_bearing;

# Compare bearing to track with bearing of runway
if(abs(rwy1_bearing-flt_bearing)<30){print $0,runway1[rwy_index]}
else{if(abs(rwy2_bearing-flt_bearing)<30){print $0,runway2[rwy_index]} else{print
$0,"NA"}};
}}
#}
```

File name: function.goaround_holding

```
# Step1: Calculate bearing between two consecutive points
# Step2: Calcute change in bearing
# Step3: Add up change in bearing
# Step4: If above threshold then classify as go around or holding pattern
{if($1!=x){if(FNR==1){
# Initialize variables
x=$1;
lat1=$7;
lon1=$8;
sumangle1=0;
sumangle2=0;
newbearing=0;
y=1;
print $0,"Start","Start","Start","Start","Start";
} else{
x=$1;
lat1=$7;
lon1=$8;
sumangle1=0;
sumangle2=0
newbearing=0;
y=1;
print $0,"Start","Start","Start","Start","Start";}
#Compute the bearing at each point
#Compute the change in turn angle at each point
#Sum the change in angle both of negative and positive angles
} else{y++; if(y<=2){
newbearing1=bearing(lat1,lon1,$7,$8);
if(newbearing1<0){newbearing1=newbearing1+360};
lat1=$7;
lon1=$8;
print $0,newbearing1,"Start","Start","Start","Start";
} else{
newbearing2=bearing(lat1,lon1,$7,$8);
if(newbearing2<0){newbearing2=newbearing2+360};
turnangle=newbearing2-newbearing1;
if(abs(turnangle)>180){if(turnangle<0){turnangle=360-abs(turnangle)}
else{turnangle=abs(turnangle)-360}};
if(turnangle<0){
sumangle1=sumangle1+turnangle;
} else{
sumangle2=sumangle2+turnangle;
};
newbearing1=newbearing2;
lat1=$7;
lon1=$8;
print $0,newbearing2,turnangle,sumangle1,sumangle2,abs(sumangle1+sumangle2);
}
}}
END{
newbearing2=bearing(lat1,lon1,$7,$8);
if(newbearing2<0){newbearing2=newbearing2+360};
turnangle=newbearing2-newbearing1;
if(abs(turnangle)>180){if(turnangle<0){turnangle=360-abs(turnangle)}
else{turnangle=abs(turnangle)-360}};
if(turnangle<0){
sumangle1=sumangle1+turnangle;
} else{
```

```
sumangle2=sumangle2+turnangle;
};
newbearing1=newbearing2;
lat1=$7;
lon1=$8;
print $0,newbearing2,turnangle,sumangle1,sumangle2,abs(sumangle1+sumangle2);
}
```

File name: function.holdingstats

```
# This function computes mean and standard deviation for holding metrics like time, track
length and fuel burn
#
BEGIN{
delete htime;
delete hdis;
delete hfb;
}
{
if(actype=="all"){
htime[FNR]=$5;
hdis[FNR]=$6;
hfb[FNR]=$7;
} else{
if($2~actype){
htime[FNR]=$5;
hdis[FNR]=$6;
hfb[FNR]=$7;
}
}


} END{
n=length(htime);
mean_htime=mean(htime);
sd_htime=std(mean_htime,htime);
mean_hdis=mean(hdis);
sd_hdis=std(mean_hdis,hdis);
mean_hfb=mean(hfb);
sd_hfb=std(mean_hfb,hfb);
print actype,n,mean_htime,sd_htime,mean_hdis,sd_hdis,mean_hfb,sd_hfb;
}
```

File name: function.kmlfile_1

```
# This function produces a kml file for track data, to be viewed on google earth

BEGIN{
colorcode=getcolor(linecolor); # get kml color code for line color specified
# print header
print "<?xml version=\42""1.0\42 encoding=\42UTF-8\42?>";
print "<kml xmlns=\42http://www.opengis.net/kml/2.2\42
xmlns:gx=\42http://www.google.com/kml/ext/2.2\42
xmlns:kml=\42http://www.opengis.net/kml/2.2\42
xmlns:atom=\42http://www.w3.org/2005/Atom\42>";

# print document name and style information
print "<Document id=\42doc1\42>";
print "\t<name>"filename"</name>";
print "\t<visibility>0</visibility>";
print "\t<open>1</open>";
# Specific line style and color
print "\t<Style id=\42style1\42>";
# Icon style
print "\t\t<IconStyle>";
print "\t\t\t<color>"colorcode"</color>";
print "\t\t\t<scale>0.6</scale>";
print "\t\t\t<heading>0</heading>";
print "\t\t\t<Icon>";
print "\t\t\t\t<href>airplane.png</href>";
print "\t\t\t</Icon>";
print "\t\t</IconStyle>";
# Lable style
print "\t\t<LabelStyle>";
print "\t\t\t<color>0000ffff</color>";
print "\t\t\t<scale>0.8</scale>";
print "\t\t</LabelStyle>";
# Line style
print "\t\t<LineStyle>";
print "\t\t\t<color>"colorcode"</color>";
print "\t\t\t<width>4</width>";
print "\t\t</LineStyle>";
# Poly style
print "\t\t<PolyStyle>";
print "\t\t\t<color>ff00ffff</color>";
print "\t\t\t<fill>0</fill>";
# print "\t\t\t<fill>1</fill>";
print "\t\t</PolyStyle>"
print "\t</Style>";
# Create a folder for the track information
print "\t<Folder id=\42Flight Tracks\42>";
print "\t\t<name>"filename"</name>";
print "\t\t<visibility>0</visibility>";
}
# Body of the script, print the track information
{if($1!=x){x=$1;if(FNR==1){
print "\t\t<Placemark id=\42"$1"_"$2"\42>";
print "\t\t\t<name>"$1"</name>";
print "\t\t\t<visibility>0</visibility>";
print "\t\t\t<description>"$1"</description>";
print "\t\t\t<styleUrl>#style1</styleUrl>";
print "\t\t\t<gx:Track kml:id=\42null\42>";
print "\t\t\t\t<extrude>1</extrude>";
print "\t\t\t\t<altitudeMode>absolute</altitudeMode>";} else{

print "\t\t\t</gx:Track>";
print "\t\t</Placemark>";
print "\t\t<Placemark id=\42"$1"_"$2"\42>";
print "\t\t\t<name>"$1"</name>";
print "\t\t\t<visibility>0</visibility>";
print "\t\t\t<description>"$1"</description>";
print "\t\t\t<styleUrl>#style1</styleUrl>";
print "\t\t\t<gx:Track kml:id=\42null\42>";
print "\t\t\t\t<extrude>1</extrude>";
print "\t\t\t\t<altitudeMode>absolute</altitudeMode>";
}} else{
print "\t\t\t\t<when>"$5"T"$6"Z</when>";
print "\t\t\t\t<gx:coord>"$8","$7","$9*.3048"</gx:coord>";
}}
END{
print "\t\t\t</gx:Track>";
print "\t\t</Placemark>";
print "\t</Folder>";
print "</Document>";
print "</kml>"}
```

File name: function.kmlfile_2

```
# This function produces a kml file for track data, to be viewed on google earth
```

```
BEGIN{
FS="\t";
colorcode1=getcolor(linecolor1); # get kml color code for line color specified
colorcode2=getcolor(linecolor2); # get kml color code for line color specified
# print header
print "<?xml version=\42""1.0\42 encoding=\42UTF-8\42?>";
print "<kml xmlns=\42http://www.opengis.net/kml/2.2\42
xmlns:gx=\42http://www.google.com/kml/ext/2.2\42
xmlns:kml=\42http://www.opengis.net/kml/2.2\42
xmlns:atom=\42http://www.w3.org/2005/Atom\42>";

# print document name and style information
print "<Document id=\42doc1\42>";
print "\t<name>"filename"</name>";
print "\t<visibility>0</visibility>";
print "\t<open>1</open>";
# Specific line style and color
print "\t<Style id=\42style1\42>";
# Line style
print "\t\t<LineStyle>";
print "\t\t\t<color>"colorcode1"</color>";
print "\t\t\t<width>5</width>";
print "\t\t</LineStyle>";
print "\t</Style>";
# Specific line style and color
print "\t<Style id=\42style2\42>";
# Line style
print "\t\t<LineStyle>";
print "\t\t\t<color>"colorcode2"</color>";
print "\t\t\t<width>4</width>";
print "\t\t</LineStyle>";
print "\t</Style>";
# Create a folder for the track information
print "\t<Folder id=\42Flight Tracks\42>";
print "\t\t<name>"filename"</name>";
print "\t\t<visibility>0</visibility>";
}
# Body of the script, print the track information
{if(NF==2){if(FNR==1){
print "\t\t<Placemark id=\42"$1"_"$2"\42>";
print "\t\t\t<name>"$2"</name>";
print "\t\t\t<visibility>0</visibility>";
if($1=="STAR"){
print "\t\t\t<styleUrl>#style1</styleUrl>"} else{print
"\t\t\t<styleUrl>#style2</styleUrl>"};
print "\t\t\t<LineString>";
print "\t\t\t\t<tessellade>1</tessellade>";
print "\t\t\t\t<altitudeMode>clampToGround</altitudeMode>";
print "\t\t\t\t<coordinates>"} else{
print "\t\t\t\t</coordinates>";
print "\t\t\t</LineString>";
print "\t\t</Placemark>";
print "\t\t<Placemark id=\42"$1"_"$2"\42>";
print "\t\t\t<name>"$2"</name>";
print "\t\t\t<visibility>0</visibility>";
if($1=="STAR"){
print "\t\t\t<styleUrl>#style1</styleUrl>"} else{print
"\t\t\t<styleUrl>#style2</styleUrl>"};
print "\t\t\t<LineString>";
print "\t\t\t\t<tessellade>1</tessellade>";
print "\t\t\t\t<altitudeMode>clampToGround</altitudeMode>";
print "\t\t\t\t<coordinates>"
}} else{
print "\t\t\t\t\t"$3","$2;
}}
END{
print "\t\t\t\t</coordinates>"
print "\t\t\t</LineString>";
print "\t\t</Placemark>";
print "\t</Folder>";
print "</Document>";
print "</kml>"}
```

File name: function.kmlfile_3

```
# This function produces a kml file for track data, to be viewed on google earth

BEGIN{
FS="\t";
colorcode1=getcolor(linecolor1); # get kml color code for line color specified
colorcode2=getcolor(linecolor2); # get kml color code for line color specified
# print header
print "<?xml version=\42""1.0\42 encoding=\42UTF-8\42?>";
print "<kml xmlns=\42http://www.opengis.net/kml/2.2\42
xmlns:gx=\42http://www.google.com/kml/ext/2.2\42
xmlns:kml=\42http://www.opengis.net/kml/2.2\42
xmlns:atom=\42http://www.w3.org/2005/Atom\42>";

# print document name and style information
print "<Document id=\42doc1\42>";
print "\t<name>"filename"</name>";
print "\t<visibility>0</visibility>";
print "\t<open>1</open>";
# Specific line style and color
print "\t<Style id=\42style1\42>";
# Icon style
print "\t\t<IconStyle>";
print "\t\t\t<color>"colorcode1"</color>";
print "\t\t\t<scale>0.5</scale>";
print "\t\t\t<Icon>";
print "\t\t\t\t<href>http://maps.google.com/mapfiles/kml/shapes/open-diamond.png</href>";
print "\t\t\t</Icon>";
print "\t\t</IconStyle>";
# Lable style
print "\t\t<LabelStyle>";
print "\t\t\t<color>ffffffff</color>";
print "\t\t\t<scale>0.8</scale>";
print "\t\t</LabelStyle>";
# Line style
print "\t\t<LineStyle>";
print "\t\t\t<color>"colorcode1"</color>";
print "\t\t</LineStyle>";
print "\t</Style>";
# Specific line style and color
print "\t<Style id=\42style2\42>";
# Icon style
print "\t\t<IconStyle>";
print "\t\t\t<color>"colorcode2"</color>";
print "\t\t\t<scale>0.8</scale>";
print "\t\t\t<Icon>";
print "\t\t\t\t<href>http://maps.google.com/mapfiles/kml/shapes/open-diamond.png</href>";
print "\t\t\t</Icon>";
print "\t\t</IconStyle>";
# Lable style
print "\t\t<LabelStyle>";
print "\t\t\t<color>ff00ffff</color>";
print "\t\t\t<scale>0.8</scale>";
print "\t\t</LabelStyle>";
# Line style
print "\t\t<LineStyle>";
print "\t\t\t<color>"colorcode2"</color>";
print "\t\t</LineStyle>";
print "\t</Style>";
# Create a folder for the track information
print "\t<Folder id=\42Flight Tracks\42>";
print "\t\t<name>"filename"</name>";
print "\t\t<visibility>0</visibility>";
}
# Body of the script, print the track information
{
print "\t\t<Placemark id=\42"$1"_"$2"\42>";
print "\t\t\t<name>"$2"</name>";
print "\t\t\t<visibility>0</visibility>";
if($1=="FIX1"){
print "\t\t\t<styleUrl>#style1</styleUrl>"} else{print
"\t\t\t<styleUrl>#style2</styleUrl>"};
print "\t\t\t<Point>";
print "\t\t\t\t<altitudeMode>clampToGround</altitudeMode>";
print "\t\t\t\t<coordinates>";
print "\t\t\t\t\t"$5","$4;
print "\t\t\t\t</coordinates>";
print "\t\t\t</Point>";
print "\t\t</Placemark>"}
END{
print "\t</Folder>";
print "</Document>";
print "</kml>"}
```

File name: function.mean

```
# This function computes mean for a given get of values
function mean(arr){zz1=0; yy1=0; for(no in arr){zz1+=arr[no]; yy1++}; out=zz1/yy1; return
out}
```

File name: function.mean_sd_1

```
# This functin computes the mean and std of one variable
# input is of the format
# gawk -v recno1=5 -v fields=124 -f function.mean -f function.std -f function.mean_sd_1
inputfile
# recno1 is the field for which the mean and variance is to be computed
# fields are the field number of the unique identifier over which the mean and sd is to
be computed
# the input file should be sorted by the fields of the unique idenfier
{for(i=1;i<=length(fields);i++){if(i==1){fld1=$substr(fields,i,1)} else{fld1=fld1"
"$substr(fields,i,1)}};
if(fld1!=fld2)
{if(FNR!=1){
mean_recno1=mean(arr_recno1);
std_recno1=std(mean_recno1,arr_recno1);
print fld2,count,mean_recno1,std_recno1};
fld2=fld1;
delete arr_recno1; arr_recno1[FNR]=$recno1;count=0;count++}
 else{arr_recno1[FNR]=$recno1; count++}} END{
mean_recno1=mean(arr_recno1);
std_recno1=std(mean_recno1,arr_recno1);
print fld2,count,mean_recno1,std_recno1}
```

File name: function.min_max_mean_sd_3

```
function minmaxmeansd(arr1){
n=asort(arr1, arr2);
min=arr2[1];
max=arr2[n];
average=mean(arr1);
sd=std(average,arr1);
}
```

File name: function.nonbadacoefficient

```
# This function gets the BADA coefficient for a non BADA actype by matching it with the
BADA actype MTOW
#
{if(NR==FNR){if($1!="#"){actype[FNR]=$1; MTOW[FNR]=$19; enginetype[FNR]=$20;
record[$1]=$0}} else{if($1!="#"){
if(FNR!=1){delete arr1; delete arr2};
# Compute the weight difference between the unknown actype and BADA actypes
for(i=1;i<=length(actype);i++){
if($3==enginetype[i]){
arr1[abs(MTOW[i]-$2)]=abs(MTOW[i]-$2);
arr2[abs(MTOW[i]-$2)]=actype[i]} else{
yy=="do nothing"}};
# assign the aircraft with the smallest difference in MTOW
if(length(arr1)>0){
asort(arr1);
# substitute actype
subtype=arr2[arr1[1]];
print record[subtype],$1;
}}}}
```

File name: function.opsperrwy

```
# This function compute ops per runway for a given day, for given timebins
{if(NR==FNR){
time1[FNR]=$1;
time2[FNR]=$2;
bin1[FNR]=$3;
bin2[FNR]=$4;
i=1;
count=0;}
else{
if($2>bin1[i] && $2<=bin2[i]){
count++;
if($NF!="NA"){arr[$NF]++};
} else{if(count>0 && i<=length(time1)){
printf "%s %s ", time1[i],count;
for(no in arr){printf "%s(%s)", no,arr[no]};
printf "\n";
i++;
count=0;
count++;
delete arr;
if($NF!="NA"){arr[$NF]++};
}}}} END{
if(count>0 && i<=length(time1)){
printf "%s", count;
for(no in arr){printf " %s(%s)", no,arr[no]};
printf "\n";
}}
```

File name: function.pointinrectangle

```
# This function check if a point is within a rectangle
BEGIN{
# Get the center of the reactangle as reference
alat=(lat1+lat2+lat3+lat4)/4;
alon=(lon1+lon2+lon3+lon4)/4;
# Convert all the four vertices coordinated to cartersian
# vertex 1
getcar(alat,alon,lat1,lon1,scale)
xx1=x1;
yy1=y1;
# vertex 2
getcar(alat,alon,lat2,lon2,scale)
xx2=x1;
yy2=y1;
# vertex 3
getcar(alat,alon,lat3,lon3,scale)
xx3=x1;
yy3=y1;
# vertex 4
getcar(alat,alon,lat4,lon4,scale)
xx4=x1;
yy4=y1;
} {
# Convert the track hit to cartersian point
getcar(alat,alon,$7,$8,scale);
px1=x1;
py1=y1;

# Check if the point if within the rectangle
xa=xx1; ya=yy1; xb=xx2; yb=yy2; x0=px1; y0=py1;
check1=((xb-xa)*(ya-y0)-(xa-x0)*(yb-ya));

xa=xx2; ya=yy2; xb=xx3; yb=yy3; x0=px1; y0=py1;
check2=((xb-xa)*(ya-y0)-(xa-x0)*(yb-ya));

xa=xx3; ya=yy3; xb=xx4; yb=yy4; x0=px1; y0=py1;
check3=((xb-xa)*(ya-y0)-(xa-x0)*(yb-ya));

xa=xx4; ya=yy4; xb=xx1; yb=yy1; x0=px1; y0=py1;
check4=((xb-xa)*(ya-y0)-(xa-x0)*(yb-ya));

if(check1>0 && check2>0 && check3>0 && check4>0){print $0,fix}
}
```

File name: function.radian

```
# This function converts degress to radian
function radian(value){return value*.0174532925}
```

File name: function.reflectrotate

```
# This function reflect and rotate tracks
#
{
# convert to cartersian
# track points
getcar(alat,alon,$7,$8,scale);
track_x=x1;
track_y=y1;
# runway coordinates
getcar(alat,alon,rwylat1,rwylon1,scale);
rwy_x1=x1;
rwy_y1=y1;
getcar(alat,alon,rwylat2,rwylon2,scale);
rwy_x2=x1;
rwy_y2=y1;

# get angle to rotate
angle_rotate=(baserwy*10)-(newrwy*10);
theta=radian(angle_rotate);
# reflect the co-ordinate with respect to the runway
if(reflect==1){
# get the line vector (runway vector)
l_x=rwy_x1-rwy_x2;
l_y=rwy_y1-rwy_y2;
# do the dot products
ldotl=l_x*l_x+l_y*l_y;
vdotl=track_x*l_x+track_y*l_y;
scalar=2*(vdotl/ldotl);
l_x2=scalar*l_x;
l_y2=scalar*l_y;
track_x2=l_x2-track_x;
track_y2=l_y2-track_y;
} else{
track_x2=track_x;
track_y2=track_y;
}

# rotate the co-ordinate with respect to the center of the airport
if(rotate==1){
track_x3=track_x2*cos(theta)-track_y2*sin(theta);
track_y3=track_x2*sin(theta)+track_y2*cos(theta);
} else{
track_x3=track_x2;
track_y3=track_y2;
}

# convert back to lat lon
getlatlon(alat,alon,track_x3,track_y3,scale);

# print results
$7=lat2x;
$8=lon2x;

}1
```

File name: function.stats_fld

```
# This function computes stats for a given field
{
if($field1!="NA" && $field1!="" && $field1>0 && $25!="GA"){level1[FNR]=$field1};
if($field2!="NA" && $field2!="" && $field2>0 && $25!="GA"){level2[FNR]=$field2};
if($field3!="NA" && $field3!="" && $field3>0 && $25!="GA"){level3[FNR]=$field3};
} END{
minmaxmeansd(level1);
print field1,n,min,max,average,sd;
minmaxmeansd(level2);
print field2,n,min,max,average,sd;
minmaxmeansd(level3);
print field3,n,min,max,average,sd
}
```

File name: function.std

```
# This function computes the standard deviation for a given set of values
function std(mean, arr){zz2=0; xx2=0; yy2=0; for(no in arr){xx2=(arr[no]-mean)^2;
zz2+=xx2; yy2++}; if(yy2==1){out=0} else{out=sqrt(zz2/(yy2-1))}; return out}
```

A5. List of Output Files

The code outputs .kml file for visualization purpose. These can directly be opened in google earth. In addition of the kml file the code generates the following files for further analysis.

File name: NOP_trackmile_stats
This file contains the track mile (NM) statistics for the various flow at MDW.

```
# Direction Runway Approach Trackcount min max mean sd
E 13C ILS 798 31.7929 71.2796 48.184 5.31704
W 13C ILS 568 33.7186 55.3428 36.2614 1.96889
E 13C RNP 87 17.6211 65.4578 44.9136 5.36458
W 13C RNP 147 32.0912 48.8617 32.8135 1.50635
E 13C Visual 1026 21.9659 55.6934 32.5769 4.08759
W 13C Visual 861 29.7395 43.704 33.6472 1.95794
E 13L Visual 8 24.3209 35.0471 30.0363 4.32471
W 13L Visual 9 30.9194 37.6006 33.7229 2.16911
E 22L Visual 840 17.4906 35.2995 20.3877 1.25408
W 22L Visual 650 34.3452 79.2656 46.3623 5.20962
E 22R Visual 70 18.2388 22.5888 19.7552 0.978345
W 22R Visual 56 35.7856 64.6551 47.0014 5.89336
E 31C ILS 1467 16.1936 68.4902 16.5589 1.43939
W 31C ILS 387 36.6948 68.5526 47.7468 5.52442
E 31C Visual 345 16.1346 52.5936 16.7711 2.48091
W 31C Visual 987 32.4715 75.7561 43.2969 4.66297
E 31R Visual 5 16.4091 16.7405 16.599 0.15527
W 31R Visual 2 41.776 47.2662 44.5211 3.88216
E 4L Visual 48 22.2426 39.9566 30.4561 4.67899
W 4L Visual 50 28.1452 38.8384 29.7292 2.03106
E 4R ILS 390 23.8985 59.4301 33.5865 5.52063
W 4R ILS 729 28.0203 34.5688 29.0588 0.743759
E 4R Visual 1181 19.307 55.6927 28.7352 4.17167
W 4R Visual 564 27.9691 60.3833 29.4635 2.23428
E NA SA 21 16.5882 35.5931 18.8162 5.24879
W NA SA 22 37.0307 90.664 52.1715 14.1722
```

File name: NOP_tracktime_stats
Same as above. but contains the track time (min) statistics

File name: NOP_fuelburn_stats
Same as above. but contains the fuel burn (kg) statistics for all aircraft type at MDW

File name: NOP_fuelburn_B737s_stats
Same as above. but contains the fuel burn (kg) statistics for B73's aircraft type at MDW

File name: MDW_all_flows_mean_fb
Contains fuel burn stats for current and hypothesized flows at MDW.

```
# Direction Runway Approach Trackcount Min_B73 Max_B73 Mean_B73 SD_B73 Mean_allactype
E 31C ILS 961 53.127 788.616 115.872 38.8616 97.5021
E 4R ILS 285 57.974 930.643 321.796 149.524 270.78
E 13C ILS 520 172.912 834.801 408.605 122.028 343.826
E 4R RNP 147 22.9637 346.371 161.826 50.6269 136.171
E 22L RNP 147 31.8219 361.966 164.355 47.0182 138.299
E 13C RNP 87 203.659 663.64 353.56 93.15 297.508
E 31C Visual 215 49.6781 359.002 117.225 39.7691 98.6406
E 22L Visual 606 51.0552 361.814 153.177 48.2601 128.893
```

```
E 4R Visual 842 50.4657 954.648 209.826 95.5988 176.561
E 13C Visual 673 87.3725 503.376 242.576 78.56 204.119
W 4R ILS 548 68.6093 303.365 168.468 40.1448 141.76
W 13C ILS 416 123.264 456.273 244.843 65.1689 206.027
W 31C ILS 295 107.859 757.759 345.702 126.672 290.896
W 31C RNP 147 37.903 363.542 190.18 45.0639 160.03
W 13C RNP 144 110.97 396.475 201.655 46.1235 169.685
W 22L RNP 87 253.767 858.877 428.14 104.799 360.264
W 4R Visual 423 71.8874 508.45 166.787 51.8071 140.345
W 13C Visual 615 84.5284 458.807 200.548 56.2973 168.754
W 31C Visual 734 85.6216 876.96 276.491 96.4799 232.657
W 22L Visual 495 96.3357 910.564 298.171 94.0693 250.9
```

File name: temp_flight_count_fb_by_year2
This file contains total fuel burn statistics at MDW for all aircraft; shown in table 27

File name: temp_flight_count_fb_by_year2_swa
This file contains total fuel burn statistics at MDW for all Southwest Airlines; shown in table 27

File name: temp_ORD_MCC_stats
This file contains delay statistics shown in table 12.

File name: temp_ORD_nonMCC_stats
This file contains delay statistics shown in table 13.

File name: MDW_13C_holding_stats
This file contains holding stats for MDW arrivals to 13C.

```
actype count mean_htime sd_htime mean_hdis sd_hdis mean_hfb sd_hfb
all 83 18.3725 9.08 79.7546 38.6666 346.27 255.237
B73 42 16.5913 7.89984 72.3806 35.812 458.325 238.319
```

REFERENCES

A4A Cost Index. (2012). A4A Quarterly Cost Index: U.S. Passenger Airlines. Retrieved July 31, 2013, from http://www.airlines.org/Pages/A4A-Quarterly-Cost-Index-U.S.-Passenger-Airlines.aspx

AhmadBeygi, S., Bromberg, E., Elliott, M., Lewis, T., & Sud, V. (2013). Capability-aware traffic flow management for metroplex environment. *Integrated Communications, Navigation and Surveillance Conference (ICNS), 2013*, 1–22. doi:10.1109/ICNSurv.2013.6548647

Air Transport World. (2012). IndiGo A320 makes India's first RNP approach landing | ATWOnline. Retrieved November 29, 2012, from http://atwonline.com/aircraft-engines-components/news/indigo-a320-makes-india-s-first-rnp-approach-landing-0625

Airservices Australia. (2008). *Brisbane Green*. Retrieved from http://www.geaviation.com/systems/products-and-services/performance-based-navigation/pdf/Brisbane_Green_-_Final.pdf

ASPM System Overview. (2012). ASPM System Overview - ASPMHelp. Retrieved November 25, 2012, from http://aspmhelp.faa.gov/index.php/ASPM_System_Overview

Atkins, S. (2008). Observation and measurement of metroplex phenomena. In *Digital Avionics Systems Conference, 2008. DASC 2008. IEEE/AIAA 27th* (p. 3–E). Retrieved from http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=4702816

Aviation Today. (2002). Horizon's Push for RNP. Retrieved from http://www.aviationtoday.com/av/commercial/Horizons-Push-for-RNP_12686.html#.UnmQ3fnbNsI

Aviation Today. (2012a). Etihad Airways Commits to RNP. Retrieved from http://www.aviationtoday.com/av/issue/departments/scan/Scan_76592.html#.UnmT3vnbNsI

Aviation Today. (2012b). JetBlue to Fly RNP AR Approaches into JFK.

BTS. (2012). About BTS | Bureau of Transportation Statistics. Retrieved September 26, 2013, from http://www.rita.dot.gov/bts/about/

Chatterji, G. B. (2011). Fuel Burn Estimation Using Real Track Data. In *11th AIAA ATIO Conference*. Retrieved from about:home

Clarke, J.-P., Ren, L., Schleicher, D., Crisp, D. L., Gutterud, R., Thompson, T., … Lewis, T. B. (2011). Characterization of and Concepts for Metroplex Operations. Retrieved from http://www.aviationsystemsdivision.arc.nasa.gov/publications/2011/NASA-CR-2011-216414.pdf

Daniel, R. (2010). Awk by example. Retrieved October 30, 2013, from http://www.ibm.com/developerworks/linux/library/l-awk1/index.html

DeLaurentis, D. A., & Ayyalasomayajula, S. (2010). *Analysis of dependencies and impacts of metroplex operations*. Retrieved from http://trid.trb.org/view.aspx?id=1097955

Devlin, C., Mills, M., Porter, S., & Sprong, K. (2012). Applications and Benefits of RNP Approaches in the United States National Airspace System. American Institute of Aeronautics and Astronautics, “In Press.”

Donaldson, A. D., & Hansman, R. J. (2011). *Improvement of Terminal Area Capacity in the New York Airspace* (No. Report No. ICAT-2011-4). MIT International Center for Air Transportation (ICAT), Department of Aeronautics & Astronautics, Massachusetts Institute of Technology.

Dorfman, S., Daily, J., Gonzalez, T., & Kondo, G. S. (2012). NAS-wide vertical profile analysis: Level segments in arrival and departure flows. In *Integrated Communications, Navigation and Surveillance Conference (ICNS), 2012* (pp. D6–1). Retrieved from http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=6218390

Enriquez, M. (2013). Identifying Temporally Persistent Flows in the Terminal Airspace via Spectral Clustering. The MITRE Corporation.

Euro Control. (2012a). AIRE programme moving forward with the Mint project | SESAR. Retrieved November 29, 2012, from http://www.sesarju.eu/news-press/news/aire-programme-moving-forward-mint-project--345

Euro Control. (2012b). Base of Aircraft Data (BADA) v3.8 | EUROCONTROL. Retrieved November 25, 2012, from http://www.eurocontrol.int/services/bada

Euro Control. (2012c). SKYbrary - Altimetry System Error. Retrieved November 1, 2013, from http://www.skybrary.aero/index.php/Altimetry_System_Error

FAA. (2005a). Advisory Circular 150/5325-4B. Retrieved from http://www.co.beaufort.sc.us/departments/Transportation/hhi-airport/documents/RunwayLengthRequirements.pdf

FAA. (2005b). Business Case Analysis Report for O'hare Modernization Program. Retrieved from http://www.faa.gov/airports/airport_development/omp/Planning/media/OMPbusCaseFinal.pdf

FAA. (2009). Facility Orientation Guide, Juneau Air Traffic Control Tower.

FAA. (2012a). Airport Categories - Airports. Retrieved October 18, 2013, from http://www.faa.gov/airports/planning_capacity/passenger_allcargo_stats/categories/

FAA. (2012b). FAA Historical Chronology, 1926-1996. Retrieved from http://www.faa.gov/about/media/b-chron.pdf

FAA. (2012c). Fact Sheet – Aviation Safety Information Analysis and Sharing (ASIAS) System. Retrieved September 24, 2013, from http://www.faa.gov/news/fact_sheets/news_story.cfm?newsId=13732

FAA. (2012d). NFDC Mission. Retrieved July 23, 2013, from https://nfdc.faa.gov/xwiki/bin/view/NFDC/NFDC+Mission

FAA. (2012e). OAPM Environmental-Metroplexes. Retrieved August 29, 2013, from http://oapmenvironmental.com/oapm.html

FAA. (2012f). Terminal Area Forecast Summary.

FAA. (2012g, February 9). Aeronautical Information Manual. Retrieved from http://www.faa.gov/air_traffic/publications/atpubs/aim/

FAA. (2012h, February 27). *Best Equipped - Best Served Operational Incentives: Top 10 Candidate Scenarios*. Retrieved from http://www.faa.gov/about/office_org/headquarters_offices/apl/environ_policy_guidance/2012meeting/media/BEBS%20Public%20Meeting_3-13-12.pdf

FAA. (2012i, August). The Business Case of Next Generation Air Transportation System.

Gariel, M., Clarke, J.-P., & Feron, E. (2007). A Dynamic I/O Model for TRACON Traffic Management. In *AIAA Guidance, Navigation and Control Conference and Exhibit* (Vols. 1-0). American Institute of Aeronautics and Astronautics. Retrieved from http://dx.doi.org/10.2514/6.2007-6551

GE Aviation. (2011, February). GE Aviation’s PBN Services Professional Resume, Version1.

Hughes, D. (2008, December 15). RNP with a Difference. *Aviation Week & Space Technology*, *169*(23). Retrieved from http://web.ebscohost.com/ehost/detail?sid=645f80d3-bb29-4c3b-aef7-5983702acbac%40sessionmgr112&vid=3&hid=119&bdata=JnNpdGU9ZWhvc3QtbGl2ZQ%3d%3d#db=a9h&AN=35903014

Jamet, D. (2011, August). Smoothing Out GPS and Heat Rate Monitor Data. Retrieved from http://mytourbook.sourceforge.net/mytourbook/files/mytourbook_smooth_11.8.2.pdf

Jeppesen Briefing Bulletin. (2005, August). Retrieved from http://ww1.jeppesen.com/documents/aviation/notices-alerts/hubwatch/BriefingBullentins/bulletin_jep05-03_RNP.pdf

JPDO. (2007). Concept of Operations for the Next Generation Air Transportaion System. Retrieved from http://www.jpdo.gov/library/nextgen_v2.0.pdf

Laskey, K. B., Xu, N., & Chen, C.-H. (2012). Propagation of delays in the national airspace system. *arXiv preprint arXiv:1206.6859*. Retrieved from http://arxiv.org/abs/1206.6859

Levy, B. S. (2003). Track averages and traffic pattern characterization for airspace analysis. In *Digital Avionics Systems Conference, 2003. DASC’03. The 22nd* (Vol. 1, p. 1–E). Retrieved from http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=1245811

Martin, J. (2009, July 29). Testimony of Captain Jeff Martin. Southwest Airlines Co. Retrieved from http://republicans.transportation.house.gov/Media/file/TestimonyAviation/2009-07-29-Martin1.pdf

NTC. (2013). North Texas Commission FAQ. Retrieved September 13, 2013, from http://www.ntc-dfw.org/ntcfaq.html

Oaks, R. ., & Ryan, H. . (2010). Prototype Implementation and Concept Validation of a 4-D Trajectory Fuel Burn Model Application. In *AIAA Guidance, Navigation and Control Conference*. Toronto. Retrieved from about:home

Panofsky, H. ., & Dutton, J. . (1984). *Atmospheric Turbulence*. Wiley&Sons.

Patterson, J., Noel, G. ., Senzig, D. ., Roof, C. ., & Fleming, G. . (2009). Analysis of Departure and Arrival Profiles Using Real-Time Aircraft Data. *Journal of Aircraft*, *46*(4), 1095–1103.

Post, J., Wells, M., Bonn, J., & Ramsey, P. (2011). Financial Incentives for NextGen Avionics. Retrieved from http://www.atmseminar.org/seminarContent/seminar9/papers/43-Post-Final-Paper-3-31-11.pdf

Ray, E. (2013). Performance Based Navigation in the US. *Air Traffic Technology International*, 44–48.

Senzig, D. A., Fleming, G. G., & Iovinelli, R. J. (2009). Modeling of Terminal-Area Airplane Fuel Consumption. *Journal of Aircraft*, *46*(4), 1089–1093. doi:10.2514/1.42025

U.S. Department of Commerce. (2012). GDP by Metropolitan Area. Retrieved from http://www.bea.gov/iTable/drilldown.cfm?reqid=70&stepnum=11&AreaTypeKeyGdp=5&GeoFipsGdp=XX&ClassKeyGdp=naics&ComponentKey=200&IndustryKey=1&YearGdp=2012&YearGdpBegin=-1&YearGdpEnd=-1&UnitOfMeasureKeyGdp=Levels&RankKeyGdp=1&Drill=1&nRange=5

United States Census Bureau. (2012). 2012 Population Estimates. Retrieved from http://www.census.gov/popest/data/metro/totals/2012/tables/CBSA-EST2012-01.csv

Vempati, L., & Ramadani, A. (2012). Is it RNAV, RNP or Conventional? In *AIAA Modeling and Simulation Technologies Conference* (Vols. 1-0). American Institute of Aeronautics and Astronautics. Retrieved from http://dx.doi.org/10.2514/6.2012-4486

BIOGRAPHY

Akshay Belle received his B.S in Mechanical Engineering from Anna University, India in 2005 and M.S in Operations Research from George Mason University, Fairfax, Virginia in 2012. He worked as a Graduate Research Assistant for six year (2008-2013) at the Center for Air Transportation Systems Research (CATSR), George Mason University (GMU). His expertise includes performing safety analysis, operational efficiency analysis, and Next Generation Performance Based Navigation performance analysis. He has developed, implemented and tested a system/framework that uses high fidelity 4-D track data along with aerodynamic fuel burn models and weather data to accurately quantify the cost and benefits of terminal arrival flows. He also has built simulation models using standard industry simulation tools such as NASA's Future ATM Concept Evaluation Tool (FACET) and Euro Control's Re-organized ATM Mathematical Simulator (RAMS), to perform safety-capacity and system wide performance analyses. His research interests include design and analysis of future concepts for next generation air transportation system, fossil fuel management for a sustainable future and theories of origin of universe and beyond. His email address is akshaykr2@gmail.com.