Analysis of Potential Wake Turbulence Encounters in Current and NextGen Flight Operations

Nataliya T. Schroeder

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

Master of Science
In
Civil Engineering

Antonio A. Trani, Chairman
Antoine G. Hobeika
Montasir M. Abbas

January 31, 2011
Blacksburg, VA

Keywords: Wake Turbulence, NextGen, PDARS, Aircraft Spacing, Air Traffic Control, Wake Vortex, Capacity, Wake envelope, Wake encounter

Analysis of Potential Wake Turbulence Encounters in Current and NextGen Flight Operations

Nataliya T. Schroeder

ABSTRACT

Wake vortices pose a threat to a following aircraft, because they can induce a roll and compromise the safety of everyone on board. Caused by a difference in pressure between the upper and the lower part of the wings, these invisible flows of air are a major hazard and have to be avoided by separating theaircraft at considerable distances. One of the known constraints in airport capacity for both departure and arrival operations is the large headway resulting from the wake spacing separation criteria. Reducing wake vortex separations to a safe level between successive aircraft can increase capacity in the National Airspace System (NAS) with corresponding savings indelay times.

One of the main goals of the Wake Encounter Model (WEM) described in this thesis is to assess the outcome from future reduced separation criteria in the NAS. The model has been used to test probable encounters in today's operations, and can also be used to test NextGen scenarios, such as Close Parallel Approaches and reduced in-trail separation flights.

This thesis presents model enhancements to account for aircraft turning maneuvers, giving the wake a more realistic shape. Three major airspaces, New York, Southern California and Atlanta, were analyzed using the original and the enhanced WEM to determine if the enhanced model better represents the conditions in today's operations. Additionally, some analysis on the wake lateral travel for closely spaced runways is presented in this thesis. Finally, some extension tools for post -analysis, such as animation tool and various graphs depicting the interactions between wake pairs were developed.

ACKNOWLEDGEMENTS

First, I would like to thank my advisor, Dr. Trani, for offering me this opportunity to complete my Master's Degree and for guiding and supporting me through the process of completing this thesis. I would also like to thank my committee members, Dr. Hobeika and Dr. Abbas for agreeing to be on my committee.

I would like to thank my husband, Doug Schroeder, for the enormous support he gave me during the time I obtained my degree. Thank you for cleaning, cooking, doing laundry and taking care of me and the dog during this busy time of my life. Thank you for moving to VA so I can complete my Bachelor's and Master's degrees. I love you!

I also want to thank my parents and brother, as well as all of my friends overseas for the support and help. Thank you all for being there for me.

I would like to thank all of my friends from the TISE program, and especially everyone located in Patton Hall. Thank you all for supporting me during these challenging times of this research and having fun during the good times as well. Pooja Dwivedi, Alex Van Dyke, Alan McDonald, Bryan Higgs, Milos Mladenovich, Zain Adam, Maria Rye, and everyone else whose name I forgot to include. Special thanks to Pooja Dwivedi, who was there for me during bad and good times. Without her Graduate school would not have been the same.

I would like to thank Nick Hinze for helping me learn MATLAB, and assisting me with challenging scripts and calculations.

I would like to thank Howard (aka. Boris) for all of the wisdom and support provided during the duration of this research.

Finally, I would like to thank the FAA for sponsoring this research and making it possible for me to complete my Master's Degree. Special thanks to Tom Proeschel and Ed Johnson for their assistance and guidance.

Table of Contents

List of Figures

List of Tables

1. INTRODUCTION

1.1 Background

Air traffic in the National Airspace System (NAS) has increased significantly in the last few decades causing congestions and delays at many airports (Janic 2008). Demand for air transportation will continue to increase and more airports are expected to be capacity-constrained resulting in billions of dollars lost and countless hours of delays (FAA 2002). To increase the efficiency in the air transportation system, we can either build more facilities (airports and runways), or optimize the current operations. The first option may result in an increased capacity initially, but it will eventually become congested again, because we treat the symptom and not the cause of the problem. Also, land availability and environmental concerns are limiting factors to this option. To explain how we can benefit from the second option, we can look at the law of supply and demand. If we can shift the supply curve to the right, we can expect higher quantity (flights) and lower price (delay) as shown on Figure 1 below. The shortage will also decrease, meaning alleviating congestions in the system. An initiative known as Next Generation Air Transportation System (NextGen) lead by Federal Aviation Administration (FAA), has the necessary components to increase capacity of our current National Airspace System (NAS) without compromising its safety. The scope and goals of NextGen, as well as the history behind it are presented in Section1.2.

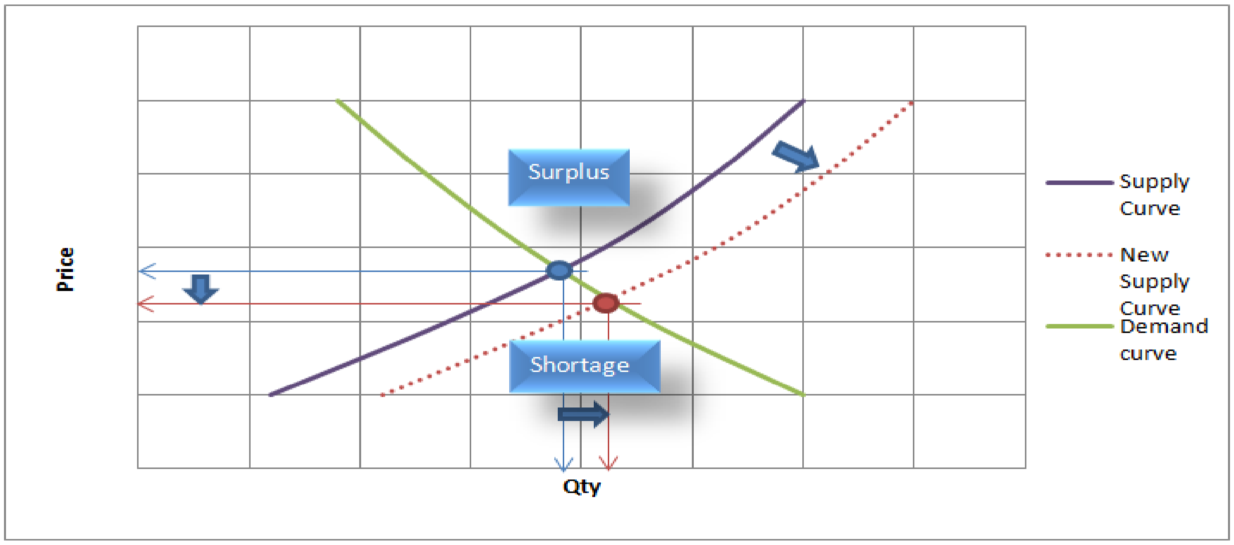


Figure 1 Supply and Demand Law. Supply curve shifts to the right resulting in a new equilibrium point with higher quantity and lower price

Research has proven that one of the main constraints to capacity is the spacing between aircraft used in daily operations (Janic 2008). Separation rules put in place in the 70's to prevent wake vortex encounters caused by the leading aircraft have proven to be safe, but resulted in capacity limitations. The current aircraft spacing are very conservative, because they are static, based on limited empirical observations, and assigned for the worst-case scenario of a wake vortex encounter (Rossow 1996). Reducing the in trail separations requires knowledge of wake vortex behavior, as well as prediction or early warning systems implementation in order to increase capacity, efficiency and safety in the NAS. Table 1 below illustrates the current separation minima between each of the five classes of aircraft based on gross weight. The FAA is considering a procedure known as RECAT, which would introduce more groups and therefore eliminate penalties in separation imposed on airplanes that are just slightly heavier than the upper boundary of the weight category.

Table 1 Current separation minima (nm)(FAA 2010)

Following Aircraft, (nm)	Leading Aircraft				
	SMALL	**LARGE**	**B757**	**HEAVY**	**SUPERHEAVY**
SMALL	2.5*	4	5	6	10
LARGE	2.5*	2.5*	4	5	10
B757	2.5*	2.5*	4	4	10
HEAVY	2.5*	2.5*	4	4	10
SUPERHEAVY	2.5*	2.5*	4	4	10
* 2.5nm separation increases to 3nm if airport has >50 sec runway occupancy time					

Weight Classification		
SMALL	**LARGE**	**HEAVY**
<41,000lb	41,000-300,000 lb	>=300,000 lb

Different topics related to wake turbulence and spacing have been studied not only in the United States, but Europe as well. Busy airports, such as Frankfurt, Germany and Heathrow, UK have been implementing different strategies to mitigate wake vortex and increase capacity. Europe has been working on the SESAR project (Single European Skies ATM Research) since 2004, which is their air traffic control "modernization program." Wake turbulence formation, as well as studies about behavior, warning and detecting is presented in details in Section 1.3.

Additionally, due to wake turbulence constraints Closely Spaced Parallel Runways (CSPR's), which are runways spaced less than 2,500ft, are treated as a single runway under Instrument Meteorological Conditions (IMC) which further reduces the runway capacity at many US and European airports. The FAA Operational Evolution Plan (FAA, 2004) element AW-5.2(Lang

2004) addresses a revision to this separation rule stating the possibility of reducing the distance to only 1000 feet when the leading aircraft is Small or Large with dependent diagonal separation of 1.5nm (Burnham 2001; Lang 2004). Parallel operations are very important because different configurations are present at almost all of the airports serving large cities all over the world. In Europe alone, four of the busiest hubs operate parallel runways, namely, Frankfurt Main (Germany), Paris Charles de Gaulle (France), London Heathrow (UK), and Amsterdam Schiphol airport (The Netherlands). In the US, the busiest airports' inventory includes 28 pairs of CSPRs, 10 pair of intermediate spaced parallel runways, and 28 of far spaced parallel runways(Janic 2008).

1.2 NextGen Overview

The Federal Aviation Administration is calling for a new program called Next Generation Air Transportation system (NextGen), to be put in place by the year 2025 as a result of increased demand for air transportation, and increased flight delays. The inefficiency of the current system is affecting the society by costing millions of dollars and emitting greenhouse gases in the atmosphere. FAA defines the Next Generation Air Transportation System as follows:

"The Next Generation Air Transportation System (NextGen) is a transformation of the National Airspace System (NAS) , including our national system of airports, using 21st century technologies to ensure future safety, capacity and environmental needs are met. NextGen will be realized through coordinated efforts by the FAA and the Departments of Transportation, Defense, Homeland Security, and Commerce, as well as NASA and the White House Office of Science and Technology Policy." (JPDO, 2007)

The three main goals are future safety, capacity and environmental needs(FAA 2009), and some of the key capabilities of the program include the following:

- Network-enabled Information Access
- Performance Based Operations and Services
- Weather Assimilated into Decision Making
- Adaptive Security
- Equivalent Visual Operations
- Positioning, Navigation and Timing Services
- Aircraft Trajectory-Based Operations
- Super-Density Arrival/Departure Operations

In addition, the NextGen concept of operation (CONOPS) specifies that Trajectory-Based Operations "minimize excess separation resulting from today's control imprecision and lack of predictability and enables reduced separation among aircraft, allowing increased capacity" (FAA, 2009). JPDO however defines typical hazards associated with separation reduction, and one of these hazards is wake turbulence.

1.3 Wake Turbulence Overview

Wake turbulence forms as a result of pressure difference above and below the wing of an aircraft. The interaction between the a higher pressured air flow underneath the wing and the lower pressured air above results in a spiraling of the air around the wingtip and the formation of a tornado-like structure. The wake can be either visible or not visible to the following aircraft depending on the atmosphere conditions.

To assure that there is not treat to a following aircraft, a sufficient following distance is imposed to assure the decay and weakened strength of the wake. This is achieved by imposing a separation minima, which has proven to be safe in today's operations because there have not been any accidents directly attributed to wake turbulence in IFR conditions (Burnham 1997).

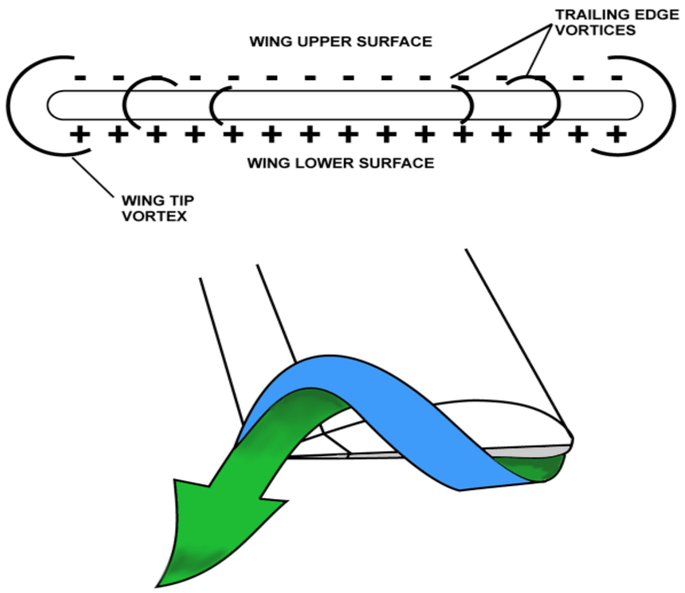


Figure 2 Wake vortices generation (Pendleton 2003, Swol 2009). Used under Fair Use Guidelines

Encountering wake turbulence can cause a significant damage to the aircraft and if the strength is too high, it could result in a deadly accident. Aircraft can encounter turbulence on final approach, during departure, and en route. Solution to wake turbulence encounters during cruise, however will have a minimal, or no impact on capacity of the NAS. Possible encounters situations during final approach are shown below:

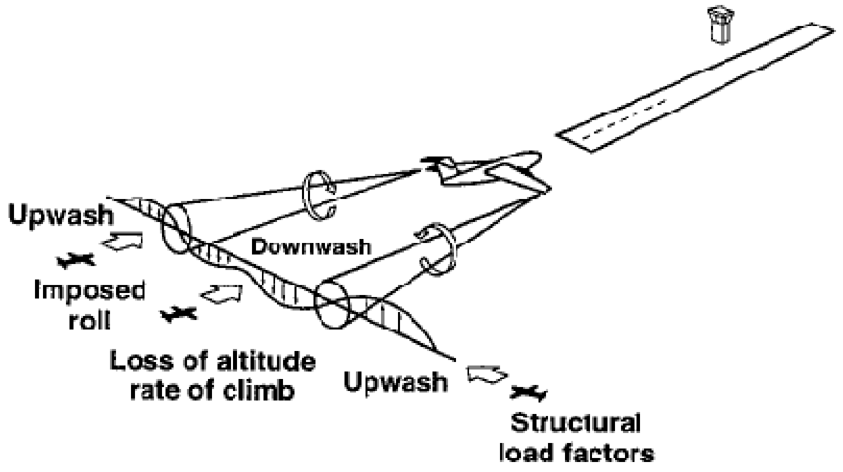


Figure 3 Possible encounters on approach (Rossow, 2000).Used under Fair Use Guidelines

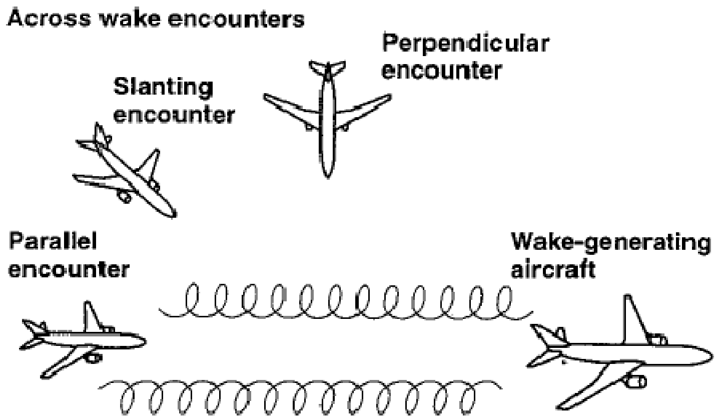


Figure 4 Possible encounters at cruise altitudes (Rossow, 2000). Used under Fair Use Guidelines

Many researchers have tried to find a way to alleviate or detect the wake, by suggesting different flap settings and designs, and using wake turbulence detecting sensors. Other researchers have concentrated on developing computer models used to simulate the dissipation of wake vortices over space and time based on initial given parameters. Wake detection and alleviation strategies are beyond the scope of this paper.

Models such as P2P, D2P, and TDAWP are used to model the wake behavior using aircraft parameters for different atmospheric conditions. Even though initially the wake behavior is influenced by the parameters of the aircraft (mass, wingspan, speed, etc.), wake travel and dissipation are heavily depended on the atmospheric conditions (wind, temperature, etc) (Oconnor and Rutishauser 2001).Three important atmospheric parameters that influence the wake dissipation are the initial wake strength (function of aircraft state parameters), the Eddy Dissipation Rate (EDR), and Brunt-Vaisala Frequency (BVF). These parameters are used in the wake modeling and are described in detail in Chapters 2 and 3.

Even though today's operations are considered safe, there have been some accidents in the past somehow related to wake turbulence. Two of the most significant ones happened in 1994 and 2001 and even though wake turbulence has not been the major cause of the accident, it did play some role. In the first one, all 132 passengers from Flight 427 operated by US Air died when the aircraft crashed on a final approach to Pittsburgh Airport. The airplane encountered wake turbulence, and suffered a failure in the rudder system while attempting to recover from the encounter (National Transportation Safety Board 1999). In the second accident, 265 people lost their lives when Flight 587 operated by American Airlines crashed after encountering the wake of a Boeing 747 operated by Japan Airlines. The cause of this accident was determined to be the overuse of the rudder pedal by the pilot, which resulted in a structural load higher than the design values. This lead to a separation of the vertical stabilizer from the airplane(National Transportation Safety Board 2004).

Recently, there have been two wake turbulence related events without fatalities involved. The first one has been investigated by the Australian Transport Safety Bureau in which a SAAB 340 (twin turboprop aircraft) encountered the wake of an Airbus A380 during a parallel approach in Sydney, Australia on November 3, 2008 (Australian Transport Safety Bureau 2008). The two parallel runways in Sydney are separated by more than 2,500feet and therefore treated as independent. The Airbus A380 was on final approach to runway 34L and about 3.4nm ahead and to the left of the SAAB at the time of the encounter. The SAAB was landing on the parallel runway 34R, which is staggered by about 3,150 feet from runway 24L (see Figure 5). This accident serves as an evidence of the lateral travel of the wake, and shows that even though the runways were separated enough to be considered independent, there are other factors that have to be considered such as cross wind.

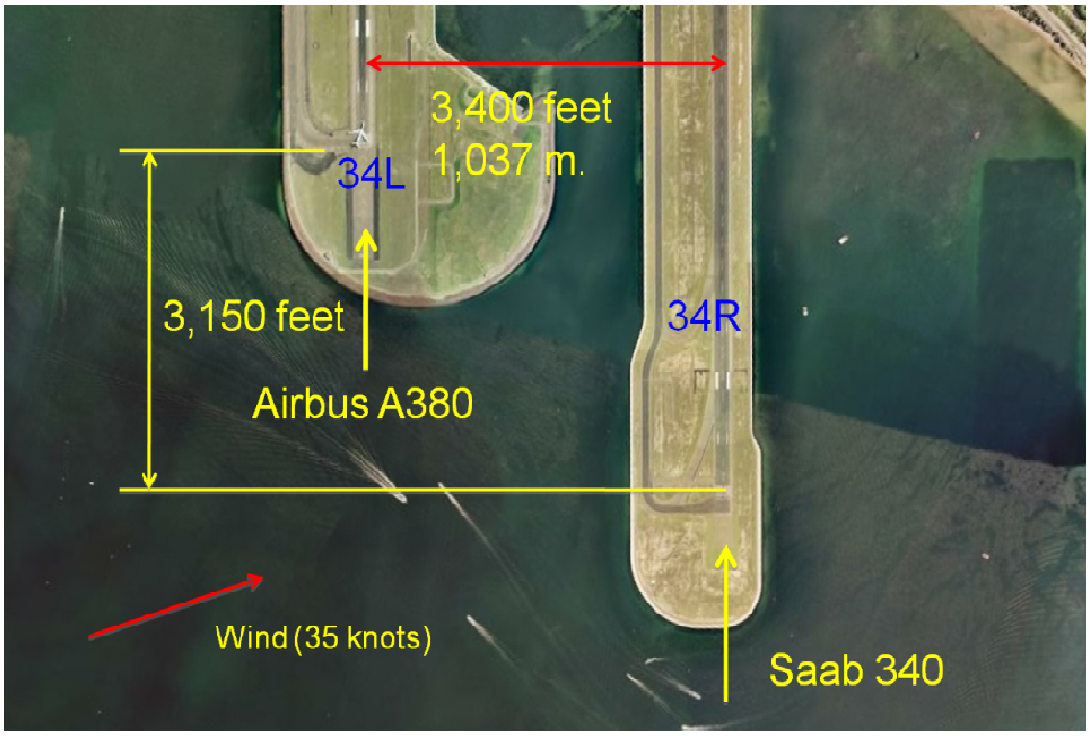


Figure 5 Location of the wake vortex encounter of SAAB 340 (Australian Transport Safety Bureau 2008). Used under Fair Use Guidelines

At the time of the encounter, there was a crosswind of about 35 knots that blew the wake vortices produced by the Airbus A380 into the path of the SAAB 340. At an altitude of 2,400ft above mean sea level (MSL), the SAAB experienced a roll to the left and a loss of altitude of 300-400ft. During the wake encounter, a passenger sustained minor injury but the crew was able to regain control of the airplane and land safely at Sydney (Australian Transport Safety Bureau 2008).

The second encounter has been investigated by the Canadian Transportation Board. On January 10, 2008 an Airbus 319 was en-route from Victoria, British Columbia to Toronto, Ontario and encountered the wake of a Boeing 747-400 (en-route from Hong Kong to Chicago) at an instance where the two aircraft were separated by 10.7nm, far more that the current ICAO and FAA separation. At the time of the encounter, the leading aircraft was at flight level 366, and the encountering one—climbing to flight level 370 (see Figure 6). In this case, even though the separation was much larger that the FAA imposed minima, the atmospheric conditions were favorable for the wake to persist for longer time (Australian Transport Safety Bureau 2008). A cruise speed of 450 knots, 10.7nm separation is equivalent to 85 second headway. Such headway is typical for final approach in trail conditions in NAS.

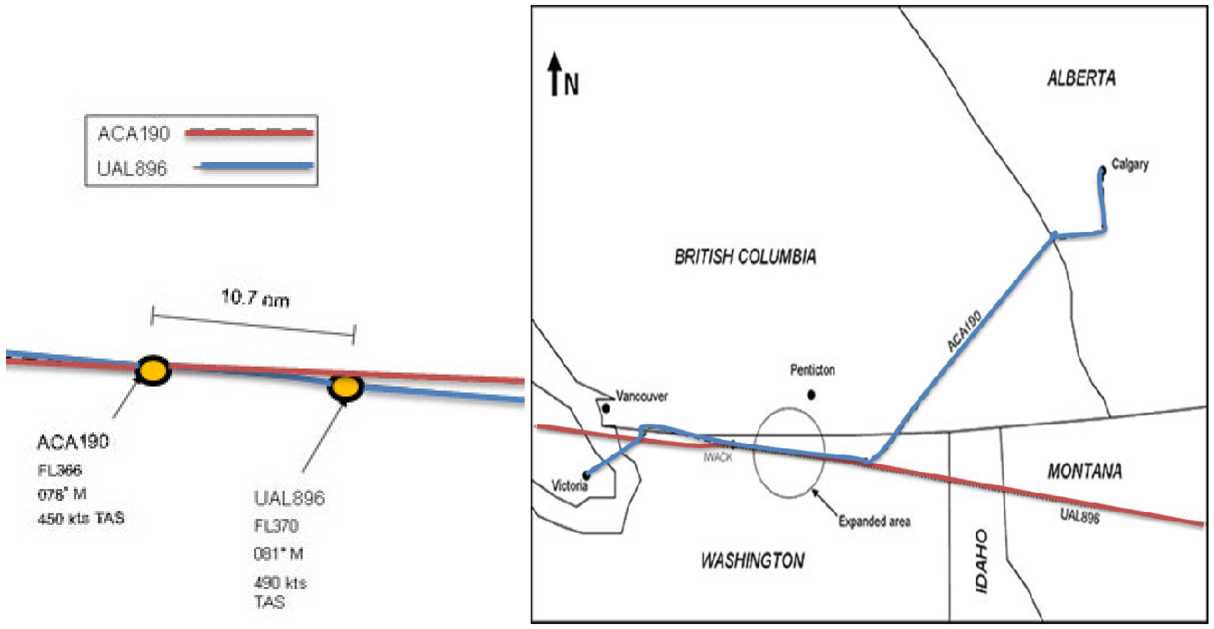


Figure 6 Location of wake vortex encounter of Airbus A319 (Australian Transport Safety Bureau 2008). Used under Fair Use Guidelines

1.4 Scope

The purpose of this study is to improve an already developed wake encounter model (WEM) which can be used to assess how often aircraft are potentially encountering the wakes produced by another aircraft (Swol, 2009). The model can then be used to test different NextGen scenarios and determine problem areas that could result in potential encounters in the future. Encounter probability can also be estimated as the number of wake encounters divided by the total number of aircraft operations. Here, the WEM is used to evaluate current operations with the purpose of establishing a baseline of potential encounters, which can then be used to compare today's safe operations with future scenarios to determine if the likelihood of wake encounterswill increase.

A detailed description of the original Wake Encounter Model is presented in Chapter 2 of this thesis. In summary, the WEM uses a wake envelope (a volume of space formed behind an aircraft where the wake is likely to be) which is determined based on a modification of the TDAWP model, which is also described in Chapter 2. The original WEM modeled the wake as a rigid surface relative to the instantaneous heading of the airplane resulting in some potential encounters that are not physically possible. The model enhancement explained in this thesis include dynamic modeling of the wake, as well as parameter calibrations, additional tools developed as an extension to the model, computing time optimization, and post-model analysis of the potential encounters.

Employing this new envelope design, the enhanced wake encounter model (EWEM) was tested using six days with available radar data at three metroplex locations to simulate typical conditions in the NAS. The model testing estimated the number of potential encounters that could have occurred if we had favorable atmospheric conditions. A potential encounter is defined as an instance when a surrounding aircraft pierces the wake envelope of a leading aircraft. The results were compared to results obtained using the original model and they are presented in Chapter 3. As expected, all of the potential encounters determined by the old model were eliminated because using the improved design. The enhanced model, however determines new instances of potential interactions, mainly for arriving aircraft flying in trail.

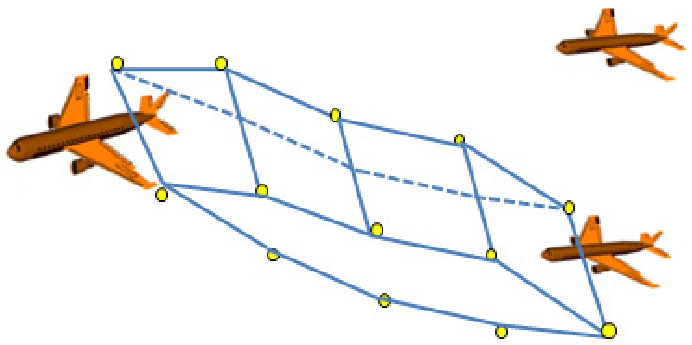


Figure 7 Wake encounter illustration

Finally, the EWEM was used to test a scenario that is going to be important when NextGen is implemented. EWEM is used to assess wake turbulence encounters for different time separations on aircraft pairs landing on the LAX airport closely spaced parallel runways (CSPR).

2. LITERATURE REVIEW

2.1 Overview

A review of the current literature will provide an inside on the findings related to wake turbulence, and can be classified in the following four categories:

- **Wake Turbulence Modeling—** Research in this category focuses on the development of computer models used to simulate the dissipation of wake vortices over space and time based on initial given parameters. This category is important because it matches the scope of this thesis.
- **Wake Turbulence Lateral Travel and Crosswind—** Research in this category focuses on studying the lateral travel of the wake vortices, and how crosswind can affect the wake travelling and dissipation. This category becomes important for closely spaced parallel approaches, which is in the scope of NextGen.
- **Wake Turbulence Effect on Capacity—**Research in this category focuses on understanding wake turbulence allowing for reducing the aircraft separation and increasing capacity in the terminal area.
- **Other Wake Turbulence Related Research such as alleviation strategies, and wake sensing and detecting—**Research in this category includes topics related to wake alleviation techniques, such as wing flap settings and design, development and testing of sensors to detect wake turbulence, etc.

The literature review in this research will focus on the first two categories presented above, because they are directly related to the scope of this thesis. The literature review is organized as follows: Wake Turbulence Modeling is presented in Section 2.2, and wake turbulence lateral travel and crosswind in Section 2.3.

2.2 Wake Turbulence Modeling

A significant number of wake turbulence behavior models, both probabilistic and deterministic have been developed in the recent years. These models include the Greene's model, Probabilistic Two Phase Model (P2P), Deterministic Two Phase Model (D2P), NASA's Aircraft Vortex Spacing System (AVOSS), TASS-driven algorithms for wake prediction (TDAWP), WakeScene model developed by the DLR, as well as the Wake Encounter Model originally developed by Doug Swol (Swol,2009).

2.2.1 Greene's original model and two modified models (Corjon and Poinsot's model, and Sarpkaya's model)

The first wake model measuring the wake vortex decay in the atmosphere was presented by Greene in 1986. The model assumed that the sum of buoyancy force, turbulent decay, and viscous drag reduce the impulse per unit length of wake (wake moment) (Greene 1986; Sarpkaya 2000; Holzapfel 2003). The model, however did not account for ground effect, shear, headwind shear, and turbulent kinetic energy vertical profiles (Sarpkaya 2000).

Corjon and Poinsot enhanced the model in 1996 by adding the ground and the crosswind effects. Additional equation term to account for the advection of the crosswind was added to the original model, and an additional vortex decay model proposed by another researcher was also incorporated. The ratio between the crosswind velocity and the velocity of the vortices was defined as a new term, controlling only the vortex trajectory. The modified code, known as VORTEX, was verified using a Navier-Stokes solver as well as experimental results (Corjon and Poinsot 1996).

Sarpkaya also modified Greene's model by eliminating the viscous drag, and created an empirical model that uses Eddy Dissipation rate instead of kinetic energy to compute the wake decay (Sarpkaya 2000). Therefore in his model, the rate of change of impulse per unit length of vortex is just the sum of the buoyancy force and the turbulent decay. The model was applied to a field data obtained using Lidar in two airports, Memphis and Dallas-Fort Worth, and the results were accurate according to the author. More recently, the wake descend rate was adapted to these field observations by using variable vortex spacing (Sarpkaya, Robins et al. 2001).

The two possible weaknesses of all of these models are , 1) none of the models account for all first-order effects (i.e. aircraft configuration, shear, ground effect, stable stratification), 2) models are deterministic, which means that they do not account for deviations cause by the turbulence's stochastic nature, and uncertainty in aircraft and environmental parameters.

2.2.2 Probabilistic Two Phase Model (P2P)

The P2P model was originally developed by the German Aerospace Laboratory and it is described in detail in two papers written by Holzapfel (Holzapfel 2003; Holzapfel 2006). The probabilistic wake vortex behavior is modeled as a function of aircraft and environmental parameters, and also accounts for wind and ground effects, as well as turbulence and stable stratification. The output of the model predicts the vortex strength and position within a confidence interval—lower and upper bound of the vortex position and circulation (Holzapfel 2003). The P2P model is comprised of two phases of vortex decay—an initial diffusion followed by a rapid decay phase, and it is based on an analytical solution of the Navier-Stokes equation. The decay parameters are "calibrated" to large-eddy simulation (LES) data, resulting in the two-phased decay, as well as in a non-linear relationship between vortex strength and vortex descend. The model is easy to be implemented in a code and provides a probabilistic bounds of the wake locations (Shortle and Jeddi 2007).

Similarly to the Sarpkaya's model, P2P was also verified against wake vortex measurements form the International Airport Memphis (December 1994, august 1995), Dallas Fort Worth (September/October 1997), and the airfield in Oberpfaffenhofen, Germany (April/May 2001) (Holzapfel 2006). The wake vortex data was collected from a Light Detection and Ranging (LIDAR) sensor placed near the runway thresholds, and measurements of both altitude and circulation strength of the wake were taken at that location. The real measurements were compared to outputs obtained from the P2P model generated for the same conditions. Overall, the model predictions were very close to the observed values for stable air conditions, with errors in the range of 5%. In the case of turbulent air, however the model was underestimating the circulation strength for the first couple of seconds of the wake generation.

The P2P model was developed further in 2006 by adding the effect of axial wind and glide-slope angle, and then compared to two field measurements from Tarbes airport, France (Holzapfel 2006). The author concluded that axial-wind and glide –slope angle consideration adjusted the vortex age and descent speed, and the axial wind and wind shear enlarged the envelopes produced by the model.

As the author concluded, the P2P model considers all of the environmental parameters affecting the wake vortex. This combined with the probabilistic nature of the model result in a good model compared to many others because it better captures the wake behavior.

A weakness of the model however is the fact that all validations of the model were performed for landing aircraft on final approach, and no validations was performed for departing aircraft, or aircraft in a higher cruise altitude. A possible explanation of this is that LIDAR sensors were placed near the runway, and data from higher altitude could not be obtained.

2.2.3 Deterministic Two Phase Model (D2P)

The Deterministic Two Phase Model (D2P) is similar to the P2P model, except as the name suggests it uses a deterministic approach. D2P has two phases of decay modeled in a similar fashion as the P2P. The input parameters in the two models are also the same-- atmospheric turbulence, wind, aircraft parameters, etc. The D2P model however computes "exact" wake boundaries based in the specified inputs, rather than providing probabilistic range of values.

The D2P model is the preferred model if the input values are certain, because it provides a mean wake vortex evolution using intermediate decay parameters. If the input data however is less certain, P2P would be the preferred model resulting in better prediction.

D2P model was also verified against field data collected from the Tarbes airport in France, and it resulted in wake circulation strength and wake altitude prediction within 5-10% f the actual measurement (Holzapfel and Steen 2006).

2.2.4 TASS-driven algorithms for wake prediction (TDAWP) model

The TDAWP model was developed by NASA and it is very similar to the D2P/P2P model, because it is also based on numerically solving differential equations which are fitted to an output of the LES model (Shortle 2007). The TDAWP model is a two-equation real time prediction model with a rapid decay phase. The model was derived from the Terminal Area Simulation System (TASS) model, a highly sophisticated model that consists of twelve prognostic equations (Proctor 2004). Ground stresses were also recently included in the model accounting for the interactions of the wake with the ground, also known as "in-ground effect (IGE)." Additionally, the model includes a meteorological framework, which can be simulated either as two- or three-dimensional. Vortex initialization is a function of vortex separation, core size, circulation, and height of the generating aircraft (Proctor 2004). TDAWP uses core radius equal to 5% of the aircraft span, and vortex separation (s) and Circulation (Γ) based on the formulas from aerodynamics and elliptically loaded wing:

$$S = \frac{\pi}{4} B$$

$\Gamma_\infty = \frac{4Mg}{\pi r V_a B}$(Proctor 2004)

Where

S= vortex separation, [m]

B= wing span, [m]

Va= True air speed, [m/s]

M= aircraft mass, [kg]

The two-dimensional version of TDAWP was verified against filed data measurements from Idaho Falls (1990) and Memphis (1994-1995). Seven cases representing different atmospheric conditions were selected from Idaho Falls database, and the model was compared against the data for Boeing 757 and 767 aircraft resulting in agreeable output. Similar results were obtained when the TDAWP model was compared to the measurements obtained for four arriving aircraft in Memphis airport. The prediction of the circulation strength obtained from the model however was not as accurate. The results from one of the runs from Idaho Falls B-757 case are shown below. Upstream positions are indicated by solid lines for simulations and filled symbols for measurements. Downstream positions are indicated by dashed lines for simulations and open symbols for measurements. Right triangles, squares and circles are used to represent Laser Doppler Velocimeter (LDV), Monostatic Acoustic Vortex Sensing System (MAVSS), and tower data. The field data was obtained from Volpe National Transportation System is shown below.

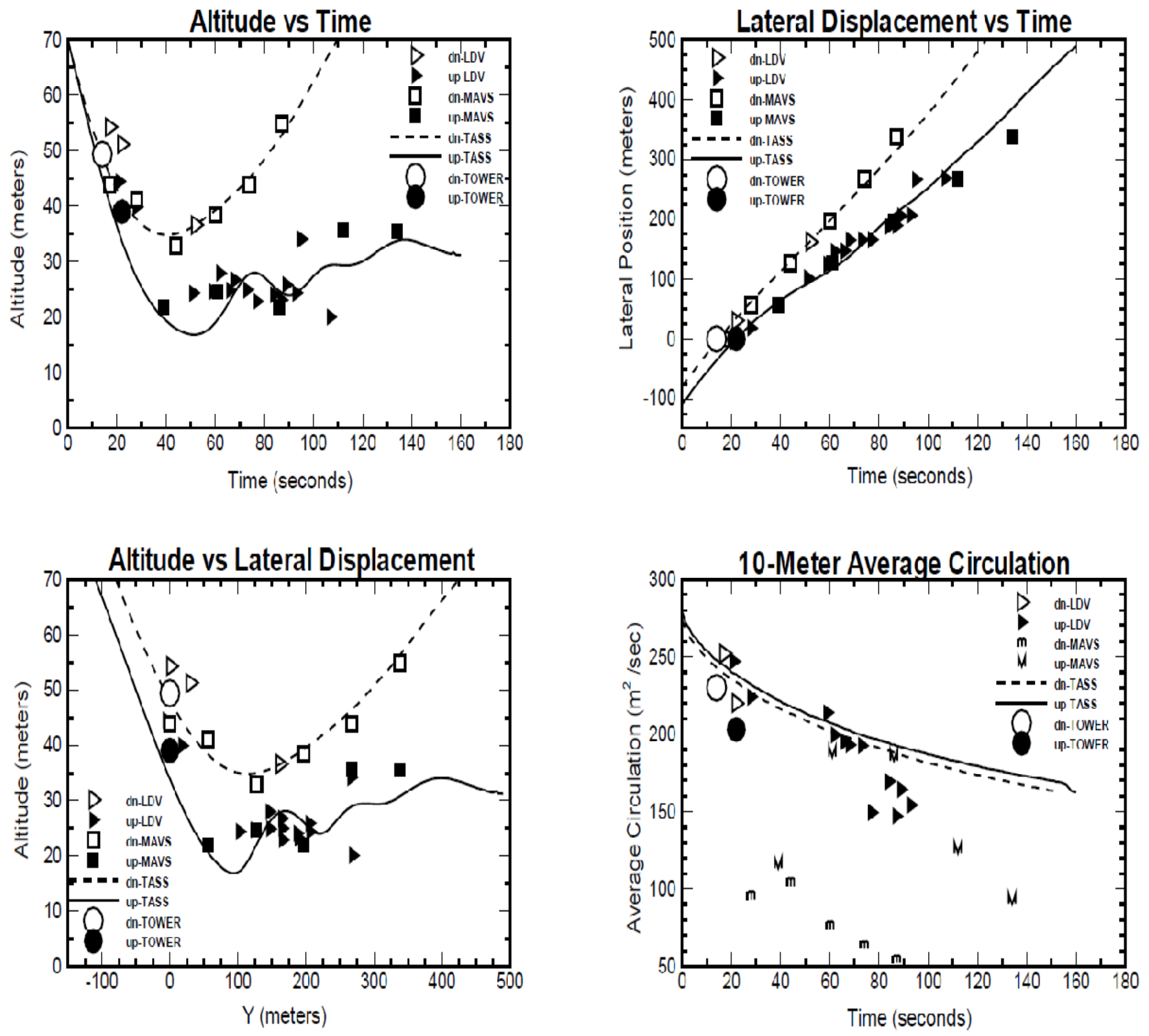


Figure 8 Comparison of TASS results with field data for the Idaho Falls B-757 Run 9 Case. Figures from left to right: Altitude of the vortex track with time, lateral position (relative to tower) vs. time, altitude vs. lateral position, and 10 m circulation vs. time (Proctor, 2004). Used under Fair Use Guidelines

A weakness of this validation is that the sample size compared with the model was quite small. Only small number of flight were selected where the effects of wind shear and ground effect were not present, since the TDAWP model did not originally account for those. Recent enhancement of the TDAWP model however includes the effects of wind shear and ground effect (Proctor, 2010).

2.2.5 Aircraft Vortex Spacing System (AVOSS) Prediction Algorithm (APA)

The Aicraft Vortex Spacing System (AVOSS) was developed by NASA and it estimates the length of time that trailing vortices remain hazard, and uses it to optimize the spacing between aircraft. The circulation decay and the vortex trajectory are computed in a plane perpendicular to the path of the generating aircraft. Crosswind shear effects are not included in the model, and the evolution of vortices is assumed to undergo four phases (Robins 2002). Unlike the P2P/D2P and TDAWP models, APA uses empirical data and forms differential equations modeling the four phases which represent the wake behavior relative to its proximity to the ground. During the first phase, Out of Ground (OGE), atmospheric stratification, wind shear, and turbulence effect influence the evolution of the vortices, and therefore the AVOSS prediction algorithm uses the described earlier Sarpkaya model. The second phase begins as the vortices start to "feel" the effect of the ground. The third phase becomes when the vortices are near the ground, causing the formation of secondary vortices. The last, fourth phase is an extension to phase three when more vortices continue to form near the ground (Robins 2002). APA was also verified against field data from Dallas-Fort Worth International Airport in 1999 and 2000. Over 2000 field measurements from the field were compared to the output generated by the model, resulting in agreement between observed and AVOSS recommended spacing and wake measurements in 99% of the cases (Oconnor and Rutishauser 2001). The small number of measurements that did not agree with the model were taken during strong wind conditions.

Overall, compared to TDAWP and D2P, APA shows less wake descent into stratified environment, and a faster initial decay (TDAWP and D2P have a rapid decay phase). A benefit of the model is that it could better capture the ground effect and better model changes in wake altitude and wake circulation strength (Shortle 2007).

2.2.6 WAKESCENE Model

The original WakeScene Simulation Package (Wake Vortex Scenarios Simulation) was used to model only approaching and landing aircraft. Later, an extension called WakeSceneD was developed, and it was used to model take-offs and departures. The simulation package uses MATLAB to simulate the different scenarios, using different modules to model the generating and following aircraft trajectories based on parameters such as speed and mass, current meteorological conditions, and assess the severity of the probable encounter. Significant research effort has been put in the meteorological database of the model, since wake behavior is heavily influenced by the atmospheric conditions. One-year wind data has been collected from the Frankfurt airport vicinity, and has been verified against 30-year wind climatology (Holzapfel, Frech et al. 2009; Holzapfel, Kladetzke et al. 2009). Aircraft speed profiles were computed using a European simulation called BADA (Base of Aircraft Data) using actual flap and gear settings for each aircraft type.

The software package uses 25 control gates released at predetermined time step along the aircraft trajectory to test for probable wake encounters. Probable encounter is determined by the Hazard-Area model, which checks if the following aircraft penetrates a circular area located along the vortex axis created by the generating aircraft. The encounter severity can be estimated by another software package called VESA, or Vortex Encounter Severity Assessment. A brief summary of WakeScene and WakeSceneD packages is presented in the next paragraph.

2.2.6.1. WakeScene for approach and landing

WakeScene can be used to assess the probability of a wake encounter during approach and landing from a final approach to the runway threshold. The model can also be used to assess the potential capacity gain by adjusting certain parameters and initial conditions (Holzapfel, Frech et al. 2009).

During the simulation generator and following aircraft‘s time, speed, and weight of along the glade path at each of the 25 gates are provided by the Aircraft-Speed Model. The deviations from the nominal path are computed by the Flight-Path Deviation Model and used in the Wake-Vortex Model. The meteorological database uses vertical profiles of wind speed and directions, temperature, and other atmospheric parameters and also feeds them into the Wake-Vortex model. The Wake-Vortex Model is then used to simulate the development, travel and dissipation of the vortices. The distance between the following aircraft and the vortices are computed within each gate by the Hazard-Area Model to determine a possible interaction. Data requiring further evaluation is stored separately and analyzed. The sequence of the model is shown in the following figure.

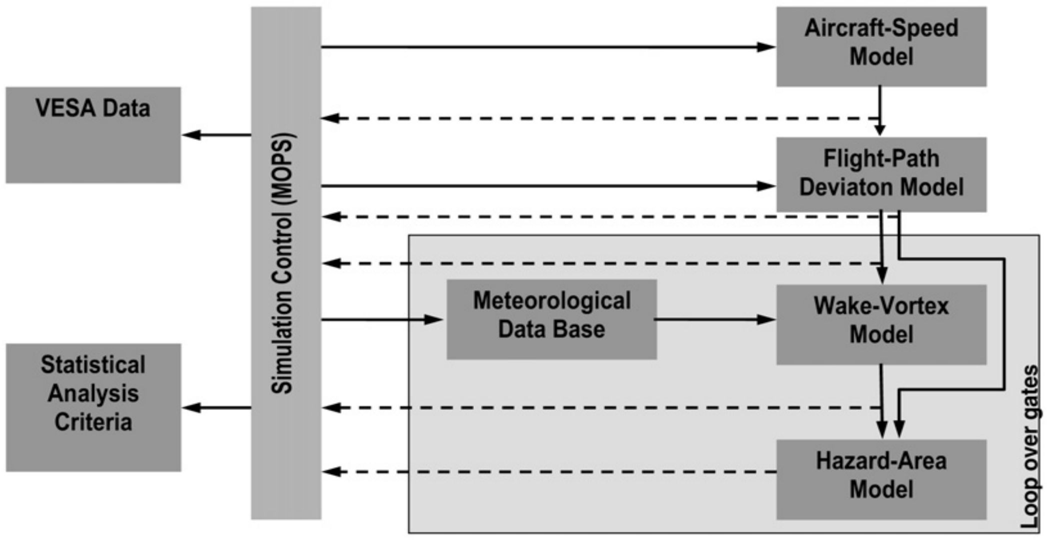


Figure 9WakeScene operational flowchart(Holzapfel, Frech et al. 2009). Used under Fair Use Guidelines

The model was used in a Monte-Carlo simulation using 100,000 approaches of heavy followed by medium aircraft. The results are considered reasonable by the author, and the model validates that most encounters occur at altitude of less than 300 ft bellow the ground and strongest ones at higher altitudes (Holzapfel, Frech et al. 2009).

As of 2009, choices for the Flight-Path Deviation Model in WakeScene are limited to four heavy-weight aircraft as a generating plane, and two medium as following. For a wake prediction model, choices in WakeScene are two—P2P and the AVOSS Prediction algorithm (APA). Both of these models were described earlier in this thesis.

2.2.6.2. WakeSceneD for take-off and departure

WakeScene-D (Wake Vortex Scenarios Simulation for Departures) is used to assess the probability of an encounter during take-off and departure at altitudes from runway altitude to 3,000 ft and different crosswind scenarios (Holzäpfel 2008; Holzapfel, Kladetzke et al. 2009).

Final approaches to a single runway are assessed with the use of 25 gates space by 1/6 nm near the ground, and 1/2nm at a higher altitude. The reason for the different spacing is to better capture the behavior of the vortices near the ground. Because the software considers standard departure path, a curved flight has to be simulated as well. The control gates are inclined at the corresponding flight path angle, and vortices are rotated by the corresponding bank angle (Holzäpfel 2008). Wake-prediction model choices for WakeSceneD are D2P and the

Deterministic wake Vortex model (DVM). The sequence of the model is shown in the following figure.

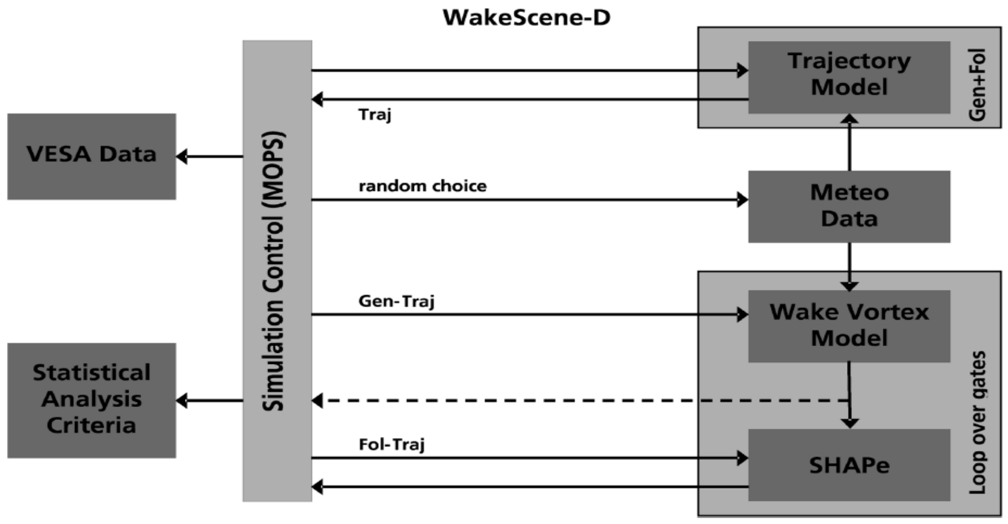


Figure 10 WakeScene-D operational flowchart(Holzapfel, Kladetzke et al. 2009). Used under Fair Use Guidelines

The model has been compared to 20,000 measured departures from Frankfurt airport and agreeable results have been obtained (Holzapfel, Kladetzke et al. 2009).

Some improvements in WakesceneD compared to the original WakeScene model include the use of 6 heavy aircraft types used for vortex generators, and four medium following aircraft, as well as the use of the inclined gates oriented perpendicular to the aircraft heading. The future goal of WakeSceneD however is to determine the appropriate cross wind threshold that will allow for closer aircraft spacing and increased capacity. A weakness of each of the models however is the limiting number of aircraft types that can be used in the simulations.

2.2.7 Wake Encounter Model (WEM)

The Wake encounter Model (WEM) was originally developed by Swol in 2009, and it is similar to WakeScene in the way it operates. The model's benefit compared to others is that WEM can analyze departures and arrival, as well as evaluating potential encounters throughout the terminal area not only near the runway.

PDARS, or PerformanceDataAnalysisandReportingSystem, is obtained from a radar site in a major TRACON. Even though the PDARS data is limited to the terminal area, the data is reported approximately every 5 seconds providing a high sampling rate and high resolution. The PDARS data contains the trajectory of all aircraft in the corresponding TRACON area. More detailed description of this type of radar data is presented in Chapter 3.3 of this thesis. Using the data in the WEM requires that the data is first parsed and saved as a .txt file, which does not change the values in the data but just simplifies it. The WEM operation flowchart is presented in Figure 11 below.

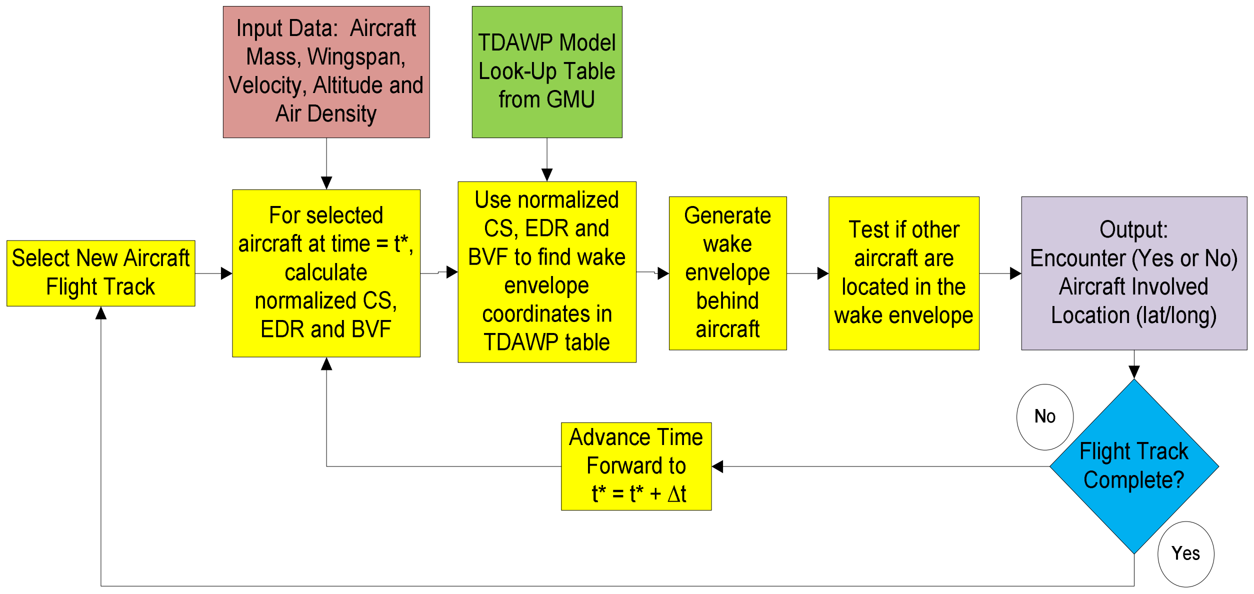
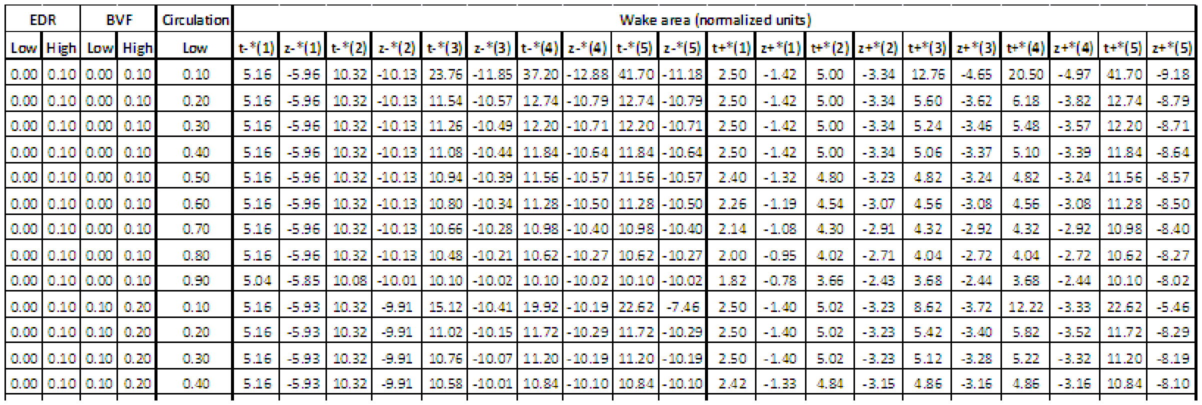

Input Data: Aircraft Mass, Wingspan, Velocity, Altitude and Air Density
TDAWP Model Look-Up Table from GMU
Select New Aircraft Flight Track
For selected aircraft at time = t*, calculate normalized CS, EDR and BVF
Use normalized CS, EDR and BVF to find wake envelope coordinates in TDAWP table
Generate wake envelope behind aircraft
Test if other aircraft are located in the wake envelope
Output: Encounter (Yes or No) Aircraft Involved Location (lat/long)
No
Flight Track Complete?
Yes
Advance Time Forward to t* = t* + Δt

Figure 11 WEM operational chart (Swol 2009)

The operational sequence is determined by a list containing a pair of aircraft obtained by first running a time filter to group the aircraft into leading and following. Then, each leading aircraft is flown along its path to determine if any of the neighboring aircraft can have any potential wake encounters. Two separate files are provided to the model, one with the aircraft parameters, such as mass and wingspan, and the second one a look-up table obtained from researchers at George Mason University (Swol, 2009). The look up table represents values for different wake parameters (Circulation Threshold (or strength), Eddy Dissipation Rate, and Brunt-Vaisala frequency) and it is used to generate a wake envelope representing the area of concern behind each leading aircraft. The table was generated using publicly available data from the TDAWP model described earlier by running the TDAWP Model over range of different values. In the look up table obtained by GMU, the parameter values are normalized with values ranging between 0 and 1. Table 2 below shows a sample of the table.

Table 2 TDAWP model Look up table (created by Dr.John Shortle, George Mason University). Used with Permission



EDR		BVF		Circulation	Wake area (normalized units)																			
Low	High	Low	High	Low	t-*(1)	z-*(1)	t-*(2)	z-*(2)	t-*(3)	z-*(3)	t-*(4)	z-*(4)	t-*(5)	z-*(5)	t+*(1)	z+*(1)	t+*(2)	z+*(2)	t+*(3)	z+*(3)	t+*(4)	z+*(4)	t+*(5)	z+*(5)
0.00	0.10	0.00	0.10	0.10	5.16	-5.96	10.32	-10.13	23.76	-11.85	37.20	-12.88	41.70	-11.18	2.50	-1.42	5.00	-3.34	12.76	-4.65	20.50	-4.97	41.70	-9.18
0.00	0.10	0.00	0.10	0.20	5.16	-5.96	10.32	-10.13	11.54	-10.57	12.74	-10.79	12.74	-10.79	2.50	-1.42	5.00	-3.34	5.60	-3.62	6.18	-3.82	12.74	-8.79
0.00	0.10	0.00	0.10	0.30	5.16	-5.96	10.32	-10.13	11.26	-10.49	12.20	-10.71	12.20	-10.71	2.50	-1.42	5.00	-3.34	5.24	-3.46	5.48	-3.57	12.20	-8.71
0.00	0.10	0.00	0.10	0.40	5.16	-5.96	10.32	-10.13	11.08	-10.44	11.84	-10.64	11.84	-10.64	2.50	-1.42	5.00	-3.34	5.06	-3.37	5.10	-3.39	11.84	-8.64
0.00	0.10	0.00	0.10	0.50	5.16	-5.96	10.32	-10.13	10.94	-10.39	11.56	-10.57	11.56	-10.57	2.40	-1.32	4.80	-3.23	4.82	-3.24	4.82	-3.24	11.56	-8.57
0.00	0.10	0.00	0.10	0.60	5.16	-5.96	10.32	-10.13	10.80	-10.34	11.28	-10.50	11.28	-10.50	2.26	-1.19	4.54	-3.07	4.56	-3.08	4.56	-3.08	11.28	-8.50
0.00	0.10	0.00	0.10	0.70	5.16	-5.96	10.32	-10.13	10.66	-10.28	10.98	-10.40	10.98	-10.40	2.14	-1.08	4.30	-2.91	4.32	-2.92	4.32	-2.92	10.98	-8.40
0.00	0.10	0.00	0.10	0.80	5.16	-5.96	10.32	-10.13	10.48	-10.21	10.62	-10.27	10.62	-10.27	2.00	-0.95	4.02	-2.71	4.04	-2.72	4.04	-2.72	10.62	-8.27
0.00	0.10	0.00	0.10	0.90	5.04	-5.85	10.08	-10.01	10.10	-10.02	10.10	-10.02	10.10	-10.02	1.82	-0.78	3.66	-2.43	3.68	-2.44	3.68	-2.44	10.10	-8.02
0.00	0.10	0.10	0.20	0.10	5.16	-5.93	10.32	-9.91	15.12	-10.41	19.92	-10.19	22.62	-7.46	2.50	-1.40	5.02	-3.23	8.62	-3.72	12.22	-3.33	22.62	-5.46
0.00	0.10	0.10	0.20	0.20	5.16	-5.93	10.32	-9.91	11.02	-10.15	11.72	-10.29	11.72	-10.29	2.50	-1.40	5.02	-3.23	5.42	-3.40	5.82	-3.52	11.72	-8.29
0.00	0.10	0.10	0.20	0.30	5.16	-5.93	10.32	-9.91	10.76	-10.07	11.20	-10.19	11.20	-10.19	2.50	-1.40	5.02	-3.23	5.12	-3.28	5.22	-3.32	11.20	-8.19
0.00	0.10	0.10	0.20	0.40	5.16	-5.93	10.32	-9.91	10.58	-10.01	10.84	-10.10	10.84	-10.10	2.42	-1.33	4.84	-3.15	4.86	-3.16	4.86	-3.16	10.84	-8.10

Since the table was derived using an earlier version of the TDAWP model, the effects of wind and ground effect are not included in the WEM. Additionally, this version of the TDAWP model did not specify the lateral growth of the wake (i.e. wake was modeled as two-dimensional), so the assumption made was that the wake grows laterally at the same rate as is grows vertically.

The operational sequence of the WEM is as follows. At each time step, normalized values of three important atmospheric parameters, namely Circulation Strength (CS), Brunt-Vaisala Frequency (BVF), and Eddy-Dissipation Rate (EDR) are computed and used to determine the corresponding values from the TDAWP look-up table, which were then used to generate a wake envelope behind the aircraft. Following aircraft location (latitude, longitude and altitude) was then tested against the location of the envelope, and if the aircraft pierced the envelope the model stored the data in a separate file as a probable encounter. Because the wake envelope represents the region where the wake is likely to be, if a following aircraft falls inside the envelope it does not necessary means an encounter.

The following describes the formulas used to compute the three important parameters, namely Circulation threshold (CT), Brunt-Vaisala Frequency (BVF), and Eddy-Dissipation Rate (EDR). The Circulation threshold is one of the three parameters used when determining the wake envelope size. The initial circulation strength (Γ_0), which represents the circulation strength at the initial time when the wake leaves the aircraft wing, is calculated using the following formula.

$$\Gamma_0 = \frac{\mathrm{Mg}}{\rho \mathrm{sBV}},$$

Where:

$\mathbf{M}$ = the aircraft mass, [kilograms]

g = the gravitational constant, [9.81 m/s^2]

ρ = the density of air, [kg/m^3]

s = rotational constant, [$\pi/4$]

B = wingspan, [meters]

V = speed, [m/s]

As mentioned earlier, the Look up table utilizes normalized values, therefore, the normalized circulation (Γ^*) is calculated by the following formula:

$$\Gamma^* = \frac{\Gamma}{\Gamma_0}$$

Where:

Γ = circulation threshold (CT) (i.e., the minimum value of wake circulation needed to pose some type of risk to encountering aircraft). CT is parametrically varied in the model.

The second component used in the WEM is the Eddy-Dissipation rate (EDR), which affects the rate of wake dissipation. Usually higher values of EDR indicate faster dissipation.

Normalized EDR (ε^*) can be calculated using the following equation from the TDAWP model:

$$\varepsilon^* = \frac{2\pi\varepsilon^{1/3} b_0^{4/3}}{\Gamma_0}$$

Where:

ε = eddy dissipation rate observed in the atmosphere in the terminal area

$b0$ is the wingspan of the aircraft multiplied by the circulation constant,

Γ_0 = initial circulation

The third and final parameter that affects the wake is the Brunt-Vaisala Frequency (BVF), which measures air buoyancy. Usually higher values cause a wake to dissipate more rapidly. The normalized BVF (N*) is calculated using the following equation:

$$N^* = N * t_0$$

Where:

$t_0 = \frac{2\pi b_0^2}{\Gamma_0}$ = the initial time parameter

N = atmospheric BVF

A weakness of the original Wake Encounter Model however is the way the wake envelope is modeled. The straight 3D rigid piece is attached to the envelope relative only to the current location and not the trajectory travelled by the aircraft. In case of turning maneuver, the envelope would just extend past the turn sometimes resulting in extra potential wake vortex encounter detection. Figure 12 illustrates how the envelope projects behind the aircraft and the realistic wake.

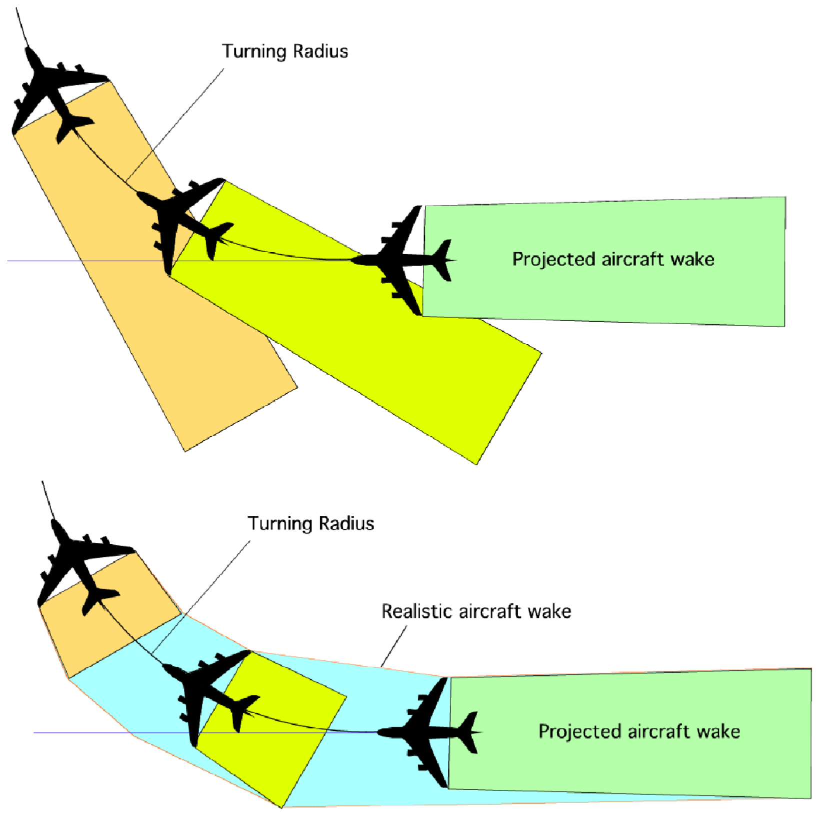


Figure 12 Wake envelope projected behind the aircraft in the WEM (top) vs. realistic wake (bottom)

Another weakness is that the model does not account for cross wind conditions, which have proven to change the behavior of the vortices because sometimes the wind can blow them in the path of a following aircraft in a close runway. This situation becomes very important for operations on closely spaced parallel runways which is in the scope of NextGen. The atmospheric parameter calibrations, as well as model enhancement are presented in the Methodology section of this thesis. Some of the enhancements include modeling the envelope relative to the trajectory, thus being bent as needed, simplification of the input-output of the model, and the development of additional tools.

2.3 Wake Turbulence Lateral Travel and Crosswind

The lateral transport of wake turbulence becomes very critical for operations on closely spaced runways. Hamilton and Proctor (2000) have done some analysis on how crosswind affects the lateral movements of the wake using a large eddy simulation (LES). Additionally, NASA's program Aircraft VOrtex Spacing System (AVOSS) involved a lot of investigations, specifically related to a single runway. Additionally, wake decay and descend can influence wake encounter on a single runway (Burnham, Hallock et al. 2002).

Wake vortex travel near the ground can be broken into four categories—Out of Ground Effect (OGE), Transition into Ground Effect, In Ground Effect(IGE), and On Ground Effect. From these four stages, two are important for this research and they will also be covered in more detail in later chapters. OGE occurs when the wake is located more than one effective wingspan above the ground, and IGE occurs when wake is less than one effective wingspan above the ground (Hamilton 2000; Burnham, Hallock et al. 2002)

Currently, approaches to runways spaced by less than 2,500ft (760m) cannot be performed as independent. Research done by Shilling in 1992 concluded that a crosswind with a magnitude of 2.5m/s is sufficient to transport the wake between the runways separated by 518m at Frankfurt International Airport assuming vortices lifetime of 180s (Schilling 1992; Hamilton 2000). Additional study showed that 90 seconds are required for the vortices to travel the distance between the same runways if wind speed is 4 m/s (Kopp 1994; Hamilton 2000).

An effective wind model that relates crosswind to wake lateral travel was also developed by Burnham et al (2002), and it is used to compute the travel time of vortices between two of the parallel runways at Los Angeles airport and presented in Chapter 4.

3. METHODOLOGY

3.1 Overview

Detailed description of the improvement of the Wake Encounter Model is provided in this chapter. Detailed description of the additional tools developed as extensions to the model are also included in this chapter. Section 3.2 of the Methodology chapter outlines the selected airports and representative days for analysis, and explains the reason for doing so. Section 3.3 includes information about PDARS, and why it was chosen for this analysis. Atmospheric parameters and aircraft parameters affecting the wake behavior are also presented in this section. The last section, Sections 3.4 is dedicated to the operation of the WEM and the output obtained after each run. This section also deals with the modeling of the wake envelope and describes the difference in the original and the improved designs.

3.2 Airport/Airspace and Date Selection

To validate the WEM, Swol selected three major Terminal Radar Approach Control (TRACON) areas were selected, namely Southern California, Atlanta, and New York City area. The TRACON's traffic controllers are responsible for handling arriving and departing flights within 30 to nautical miles (nm) at altitude below 10,000ft. The TRACON area was selected for two reasons, first aircraft separation is usually reduced in the TRACON area compared to the en route spacing, and second many improvements that will come about when NextGen is implemented are related to the terminal area capacity (Swol, 2009).

For each TRACON area three representative days for Instrument Meteorological Conditions (IMC), Marginal Meteorological Conditions (MMC) and Visual Meteorological Conditions (VMC) were selected, and the data is presented in Table 3 below. The date selection was done based on cluster analysis to represent typical conditions for that area.

Table 3 TRACONs and dates selection (Swol, 2009)

	Terminal Area		
	Southern California	**Atlanta**	**New York City**
VMC	6/18/2008	6/26/2008	2/5/2008
MMC	8/13/2008	7/13/2008	6/23/2008
IMC	7/18/2008	2/22/2008	3/31/2008

For consistency, and to compare the results of the improved WEM to the original one, the dates and TRACONs were selected to be the same. Table 4 below illustrates the major airports located in each of the three TRACONs.

Table 4 Airports within each TRACON

NEW YORK CITY TRACON	ATLANTA TRACON	SOUTHERN CALIFORNIATRACON
John F. Kennedy International (JFK) Newark Liberty International (EWR) La Guardia (LGA) Teterboro (TTB) Long Island Islip (ISP)	Hartsfield-Jackson International (ATL)	Los Angeles International (LAX) San Diego International (SAN) Ontario International (ONT), John Wayne Orange County (SNA) Bob Hope Burbank International (BUR) Van Nuys (VNY)

Airport locations within each of the TRACONs are illustrated on Figure 13, Figure 14, and Figure 15, and are shown below.

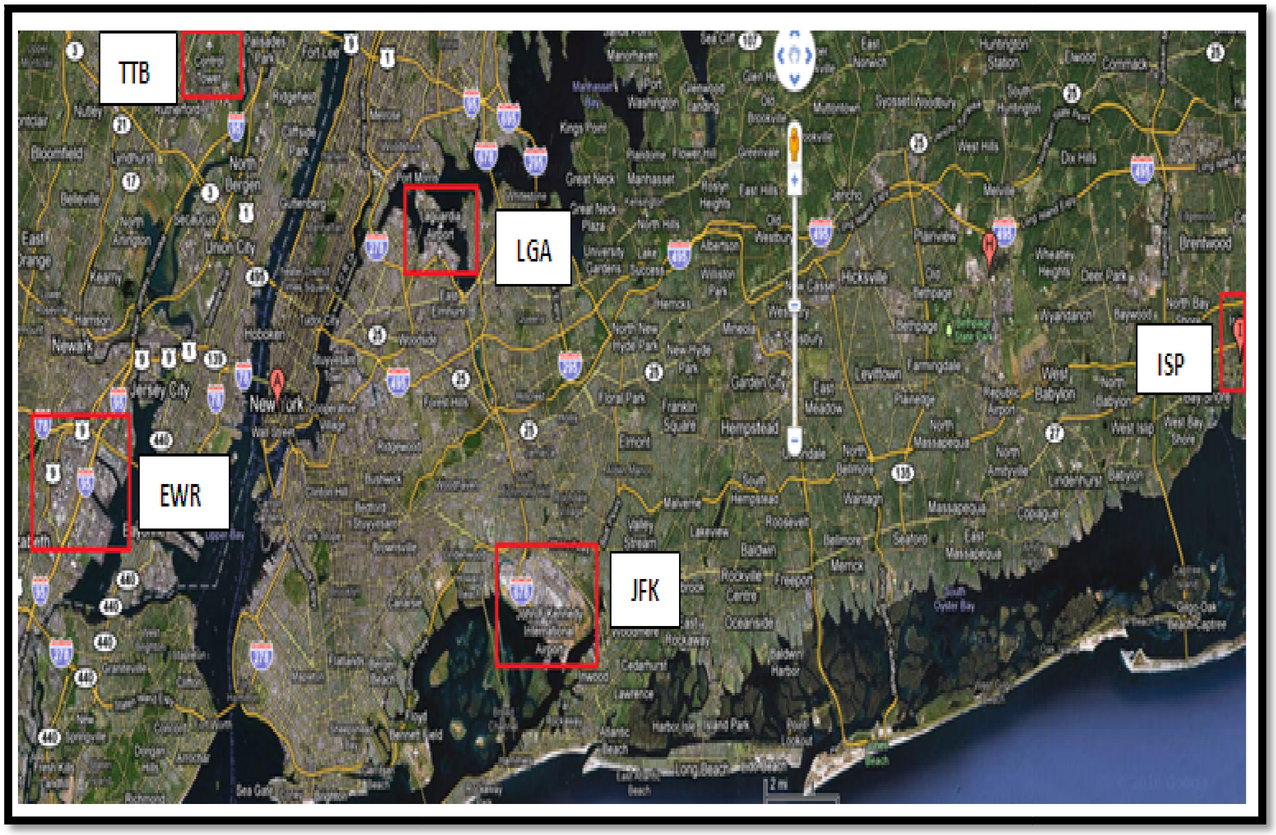


Figure 13 Airports within New York City TRACON

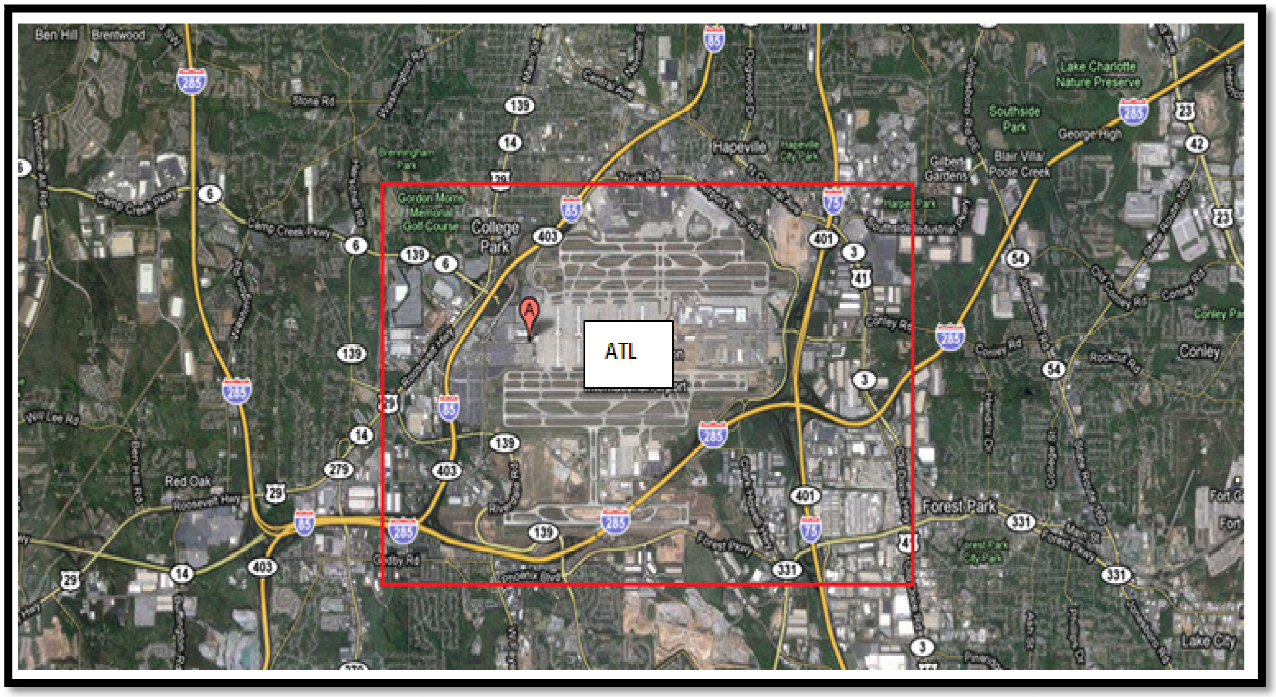


Figure 14 Airports within Atlanta TRACON

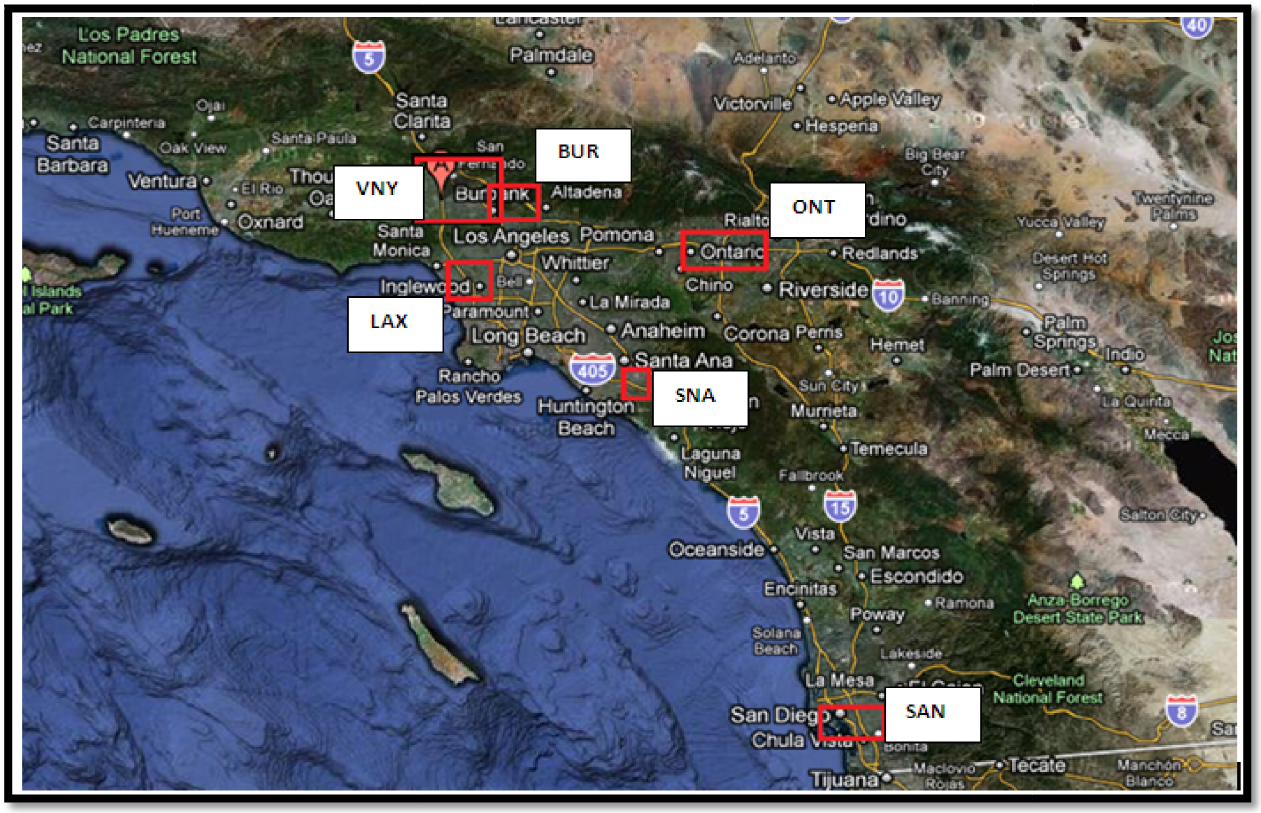


Figure 15 Airports within Southern California TRACON

3.3 PDARS Data Description and Use in the WEM

3.3.1 PDARS Description

PerformanceDataAnalysisandReportingSystem, or PDARS, is collaboration between the FAA office and System Capacity and NASA Aviation Safety Program and it is obtained from a radar site in a major TRACON. The data can be used by the ATC managers to evaluate the overall performance and safety of the air transportation system and evaluate possible problems. The development of the program started in 1997 (den Braven 2003; NASA 2007). Flight data and radar-track data is collected every day from participating TRACONs and processed overnight to ensure quality, and sent to facilities managers as a customized performance report. In 2003, ten ARTCCs, five Terminal Radar Approach Control (TRACONs), and two regional offices and the FAA's ATC Systems Control center located in Herdon, VA had PDARS installed at their sites. As of 2007, all 20 Air Route Control Centers (ARTCCs) in the US have PDARS implemented for operations (den Braven 2003).

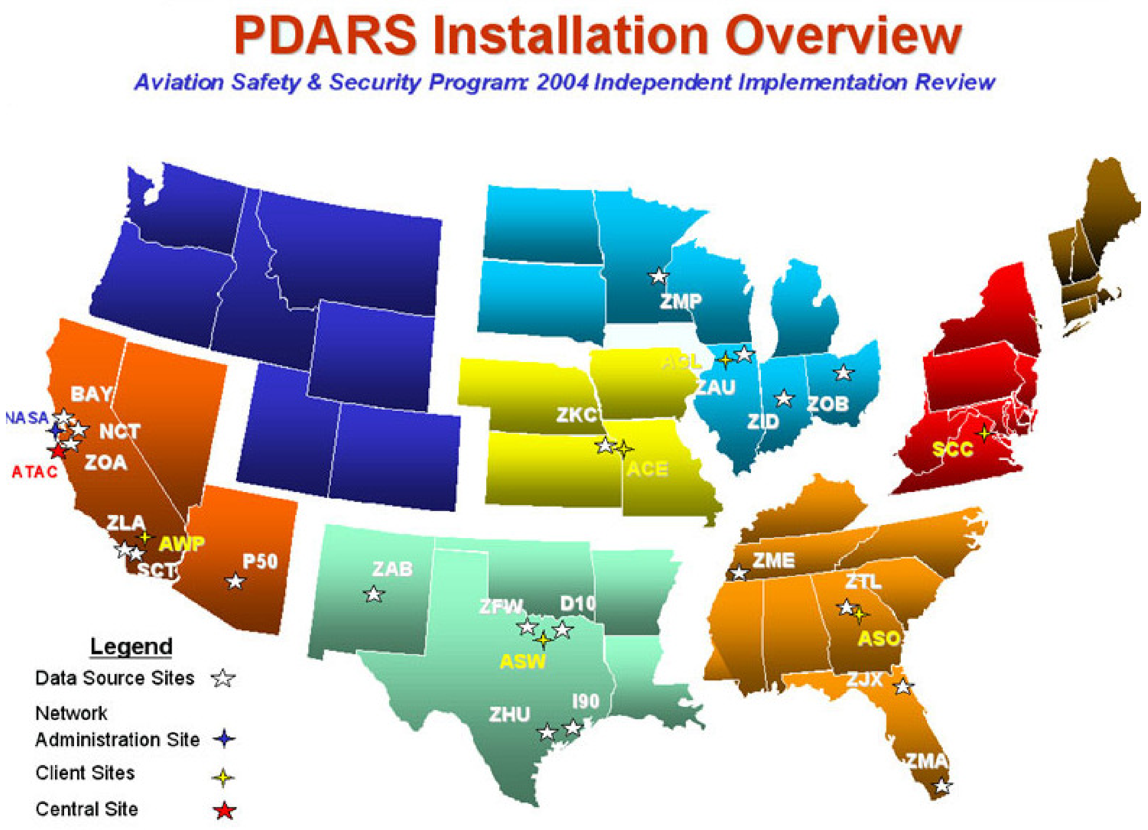


Figure 16 PDARS installation overview in the U.S. as of 2004 (den Braven 2003). Used under Fair Use Guidelines

PDARS continuously collects data from the TRACON's Automatic Radar Terminal System (ARTS) computers and ARTCCs' HOST computers and stores the data in a data management system. Custom reports can then be generated using the data and outputted as Excel-based documents.

Some of the metrics included in PDARS are traffic counts, traffic flows, travel times and distances, in-trail separation, speeds and headings. Some of the benefits related to using PDARS data and not Enhanced Traffic Management Data include the high degree of accuracy, easy access, the data is fully integrated with Microsoft Office, and therefore can be used as an input to MATLAB. Additionally, PDARS data was used in this model because it was already available to use.

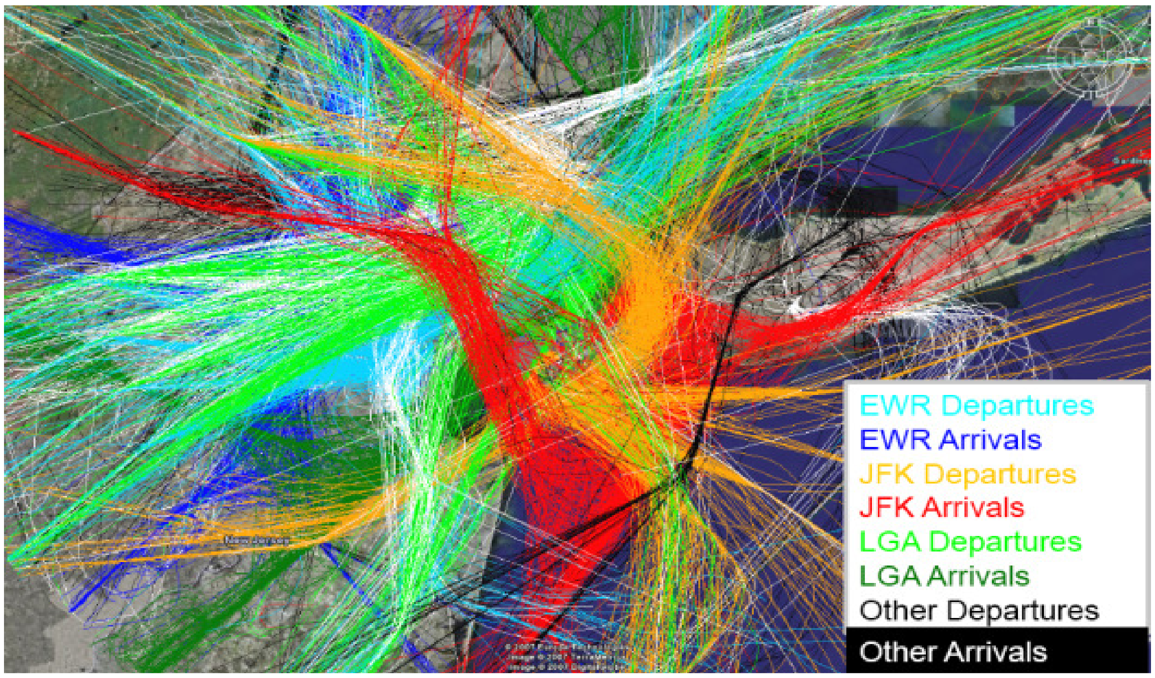


Figure 17 One day of PDARS flight tracks to and from New York City TRACON (ATSL, 2008)

3.3.2 PDARS Use in the WEM

For each of the selected days and TRACONs, PDARS data containing all of the flights was obtained and used as an input to the model. The PDARS data was first ran through a parsing filter to separate only the information needed by the model. Table 5 below illustrates what the raw data looks like before it is parced and used in the model. The parced PDARS file that is provided as an input to the model is presented later in Section 3.4.2.

Table 5 Sample raw PDARS data from New York City TRACON

2.6																							
BirdWatch Analysis Module	4.5.5																						
===																							
52533	2723	721	0/JFK		CAL011	1	B744	JFK	GAY	D	JFK	?											
52533	2723	721	0/JFK	1353	CAL011	1	B744	JFK	GAY	N	-99	-99	340		?	-99	I	J	D	?	-99	?	5
52533	2723	721	3/HPN	1353	CAL011	1	B744	JFK	GAY	N	-99	-99	340		?	-99	I	J	D	?	-99	?	5
52533	2723	721	0/JFK	1350	CAL011	1	40.63518	-73.78761	4.76	1	0.003	0.003	-99	153	104	2004	?	?	?	-99		-99	
52533	2723	721	0/JFK	1350	CAL011	1	40.63221	-73.78042	7.76	3	0.002	0.002	-99	153	104	2004	?	?	?	-99		-99	
52533	2723	721	0/JFK	1350	CAL011	1	40.62907	-73.77326	9.85	1	0.002	0.002	-99	158	106	1807	?	?	?	-99		-99	
52533	2723	721	0/JFK	1350	CAL011	1	40.62801	-73.76911	10.76	2	0.002	0.002	-99	166	94	1237	?	?	?	-99		-99	
52533	2723	721	0/JFK	1350	CAL011	1	40.62786	-73.76476	12.76	4	0.002	0.002	-99	166	78	1593	?	?	?	-99		-99	
52533	2723	721	0/JFK	1350	CAL011	1	40.62761	-73.76026	13.76	10	0.002	0.002	-99	168	80	1564	?	?	?	-99		-99	
52533	2723	721	0/JFK	1350	CAL011	1	40.62745	-73.75134	16.76	1	0.003	0.003	-99	171	77	1564	?	?	?	-99		-99	
52533	2723	721	0/JFK	1350	CAL011	1	40.62742	-73.74665	17.76	9	0.004	0.004	-99	173	76	1332	?	?	?	-99		-99	

3.4 Wake Encounter Model Overview and Enhancements

3.4.1 Wake Envelope Modeling Improvements

3.4.1. 1.Rigid envelope calculation changes

In the new, enhanced Wake Encounter Model a change was made to the formulas used to compute the coordinates of the wake envelope. The original formulas were adding the whole wingspan to all coordinates resulting in much larger envelope. Currently, the model uses half of the wing span on each side of the envelope, resulting in envelope starting width of ¾ of the wingspan, and final width of three times the wingspan. This is consistent with dimensions of a wake envelope given in the literature.

To construct the straight envelope in the enhanced wake encounter model, a function called *generate_straight_envelope* was written in MATLAB using latitude, altitude, longitude, heading, wingspan, GMU look up table, initial time, and aircraft speed as inputs. The complete function file can be found in Appendix B. The function outputs a matrix with twenty points coordinates, i.e. latitude, longitude and altitude coordinates of the wake envelope which are used later to plot the envelope and test the surrounding aircraft for potential interactions. MATLAB Mapping Toolbox is required to run this function because some units are in degrees and some in meters and conversions have to be made between degrees and length units. The coordinates of the first two points of the wake envelope are computed relative to the instantaneous position of the aircraft and attached to the wings. The rest of the points are positioned relative to the first two using the GMU look up table, heading, and speed of the wake generating aircraft.

3.4.1.2. Wake Envelope Design Improvements

As mentioned earlier, the original WEM did not account for the turning maneuvers of the aircraft. The wake envelope was attached as a solid piece behind the aircraft relative to the instantaneous position and heading of the wake-producing aircraft. This design worked well in the instances when the aircraft were travelling in a straight line, but resulted in extra encounters during turns. The original envelope extended to an area where the aircraft had not travelled. The enhancement programmed in the thesis to the design of the wake envelope relative to the trajectory of the aircraft resulted in a more realistic wake envelope modeling. The envelope design was in the enhanced model was revised to turn and move with the aircraft as shown on Figure 18. On the figure, the original straight design is shown in blue, and the revised bent envelope—in magenta and the aircraft trajectory—in black.

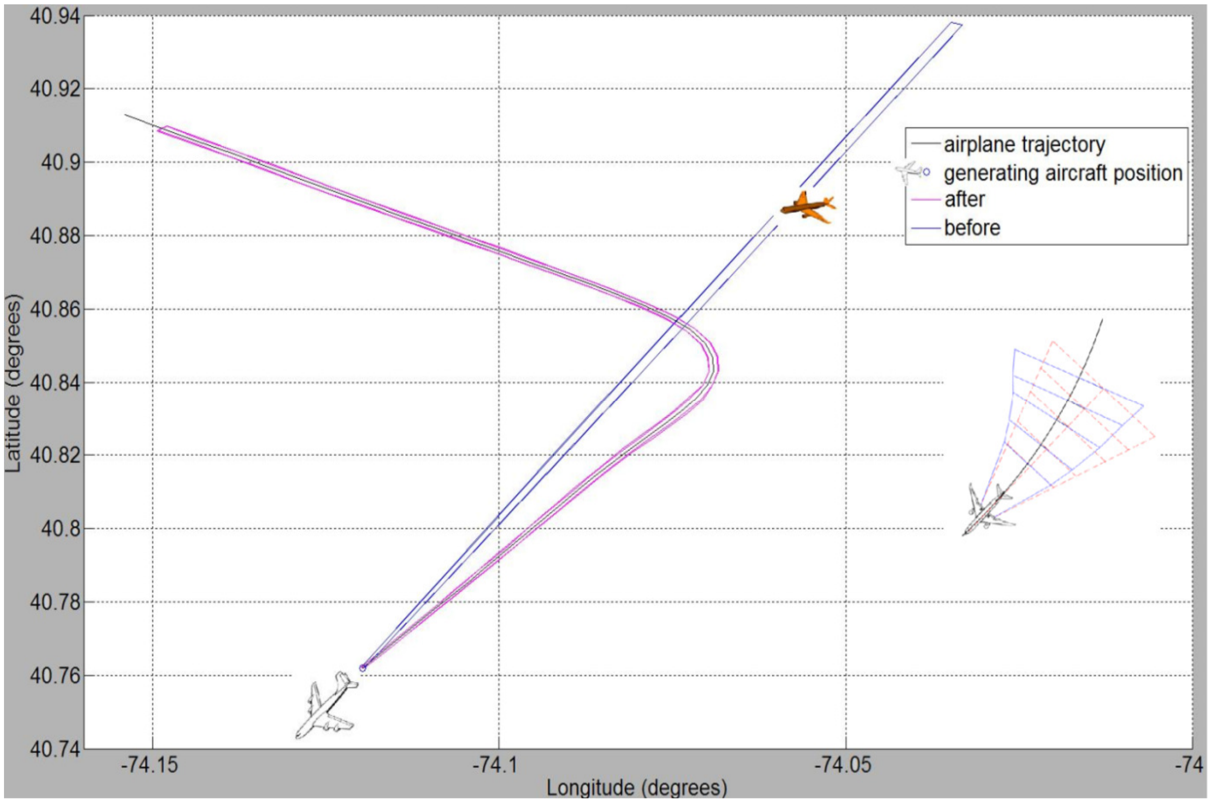


Figure 18 Top view of wake envelope showing original design (blue) versus improved design (magenta). NOT TO SCALE

In order to construct the bent envelope at each instance, the following methodology was used. First, a literature review was done to discover if the envelope can simply be bent along the trajectory. As the vehicle travels along its path we expect that the wake will persist along the trajectory, and no information was found in the literature to prove otherwise. We expect that the wake will decay and sink/rise with time, and this was accounted for in the TDAWP model used to determine the size of the envelope. Second, a straight envelope was constructed and positioned behind the aircraft as if the aircraft was travelling straight and the total length of the wake envelope was calculated to determine the distance we have to go back along the travelled trajectory to position the envelope using the described earlier function file *envelopedimensions.m*. Dimensions of the wake envelope including width, length, and thickness for all available PDARS points along the travelled trajectory were computed using interpolation, and used to determine the new position of each point relative to the trajectory. Thus, the envelope dimensions remained the same but the position was made relative to the travelled trajectory capturing the wake turbulence phenomenon more realistically.

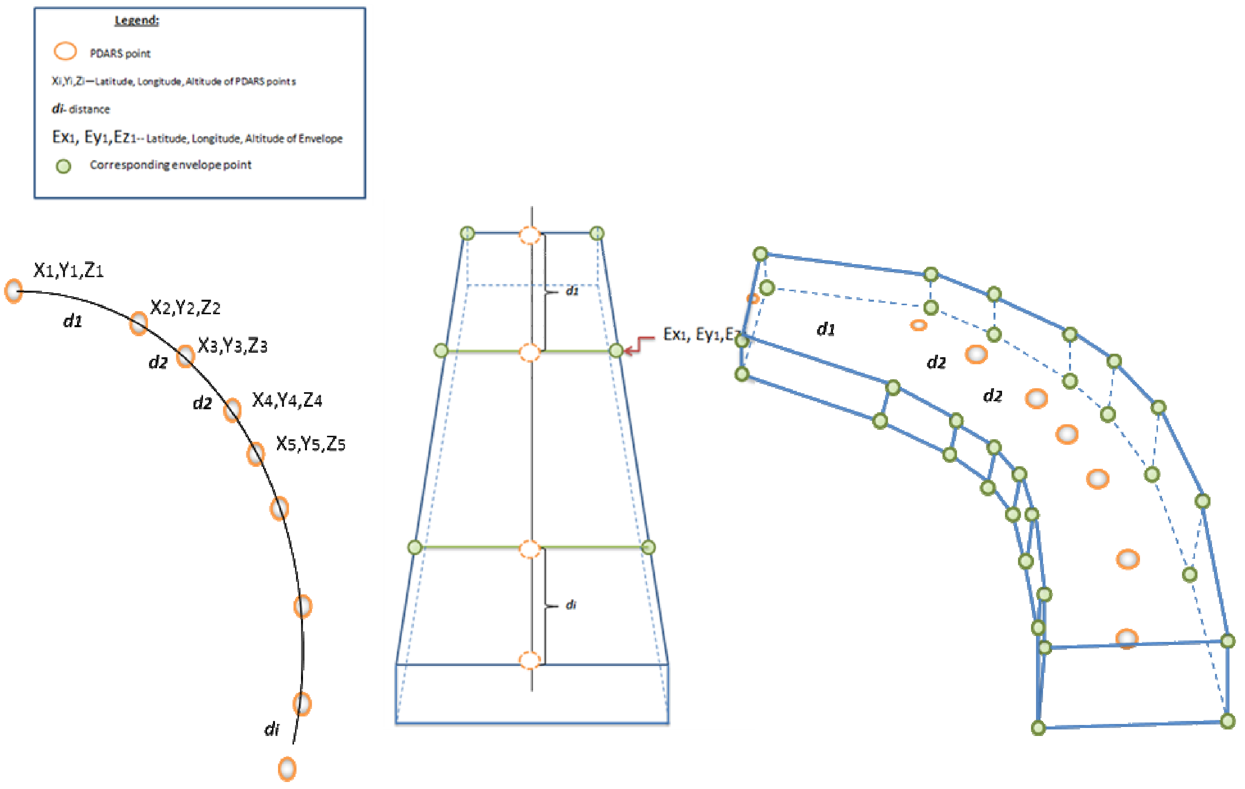

Legend:
PDARS point
Xi,Yi,Zi—Latitude, Longitude, Altitude of PDARS points
di- distance
Ex1, Ey1,Ez1-- Latitude, Longitude, Altitude of Envelope
Corresponding envelope point
X1,Y1,Z1
d1
X2,Y2,Z2
d2
X3,Y3,Z3
d2
X4,Y4,Z4
X5,Y5,Z5
di
di
di
Ex1, Ey1,Ez1
d1
d2
d2

Figure 19 Aircraft trajectory and straight versus bent wake envelope

3.4.2. Wake Encounter Model Input Files

A series of data files are provided to the model as an initial input. The input files can be grouped into two main groups—aircraft characteristics, and operational characteristics. There are two files from the first type of inputs— Aircraft Mass Distribution file, and Aircraft specific parameters (includes aircraft name, manufacturer and id, wake class category, various operating weights (MALW, MTOW, OEW), and wingspan).

The Mass Distribution file consist of mass distribution values that are used for MTOW (maximum take-off weight) and MALW (maximum allowable landing weight) adjustments since aircraft weight will vary based on the distance travelled (arrivals) and distance to be travelled (departures) and PDARS file does not provide mass information (Swol, 2009).

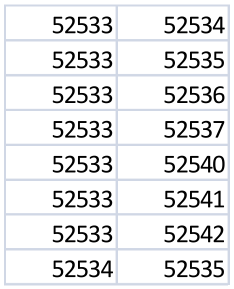
The aircraft specific parameters are provided as an Excel file to the model. 136 different types of aircraft are included in the analysis representing the most commonly flown aircraft in the NAS. The last record, 136, represents a generic aircraft used for the cases when the PDARS flight list does not match the rest of the values. Wake category, weight parameters, manufacturer, and name are some of the parameters included in the aircraft specific file. Table 6 illustrates a sample list of aircraft parameters.

Table 6 Aircraft specific parameters

ANTONOV 124	ANTONOV	AN-124	A124	Heavy	700000	892000	240.4	385000	6780
ANTONOV 225	ANTONOV	AN-225	A225	SuperHeavy	900000	1323000	290.2	385800	9743
A300	AIRBUS INDUSTRIE	A300	A300	Heavy	308600	363760	147.1	199000	2800
A300-600/R/CF/RCF	AIRBUS INDUSTRIE	A300-600	A306	Heavy	308600	363760	147.1	199000	2800
A300B/C/F-100/200	AIRBUS INDUSTRIE	A300B/C	A30B	Heavy	286000	313000	147.1	169890	2800
A310-200C/F	AIRBUS INDUSTRIE	A310-2CF	A310	Heavy	273000	361600	144	178700	2360
A-318	AIRBUS INDUSTRIE	A318	A318	Large	123500	149910	111.1	86000	1320
A319	AIRBUS INDUSTRIE	A319	A319	Large	134500	166500	111.1	89000	1320
A320-100/200	AIRBUS INDUSTRIE	A320-1/2	A320	Large	142200	169800	111.1	92800	1320
A321	AIRBUS INDUSTRIE	A321	A321	Large	171500	206100	111.1	106500	1320
A330-200	AIRBUS INDUSTRIE	A330-200	A332	Heavy	396800	513670	197.8	265730	3890
AIRBUS A330-300	AIRBUS INDUSTRIE	A330-300	A333	Heavy	407900	513670	197.8	274650	3890
A340	AIRBUS INDUSTRIE	A340	A340	Heavy	399000	606270	197.8	287160	3890
A340-200	AIRBUS INDUSTRIE	A340-200	A342	Heavy	399000	606270	197.8	287160	3890
A340-300	AIRBUS INDUSTRIE	A340-300	A343	Heavy	423280	609580	197.8	288500	3890
A340-500	AIRBUS INDUSTRIE	A340-500	A345	Heavy	529100	811300	208.2	376800	4729
A340-600	AIRBUS INDUSTRIE	A340-600	A346	Heavy	571800	811300	208.2	392000	4707
A380-800	AIRBUS INDUSTRIE	A380-800	A380	SuperHeavy	850900	1234600	262	610000	9100
A380-800	AIRBUS INDUSTRIE	A380-800	A388	SuperHeavy	850900	1234600	262	610000	9100
ANTONOV 12	ANTONOV	AN-12	AN12	Large	110000	130000	124.8	62000	1310
ANTONOV AN-22-F	ANTONOV	AN-22	AN22	Heavy	430000	551000	211.2	251330	3713
ANTONOV AN-72/74	ANTONOV	AN-72/74	AN72	Large	65000	72750	84.7	42000	1062
BEECH 1900 A/B/C/D	BEECHCRAFT	BE-1900	B190	Small	16500	17200	58	5533	358
BOEING 717-200	BOEING	B717-200	B712	Large	110000	121000	93.3	68670	1001
BOEING 727-100	BOEING	B727-100	B721	Large	166000	203100	108	89985	1560

From the operational characteristics group, the model requires five different files: List of leaders and followers, PDARS data, PDARS flight list, Look up table, and International Standard Atmosphere (ISA) Density Table. The first file is the output from a time filtering script written by Swol, which outputs a list of leading and following aircraft. A sample of the list of aircraft IDs of leaders and followers is shown in Table 7 below.

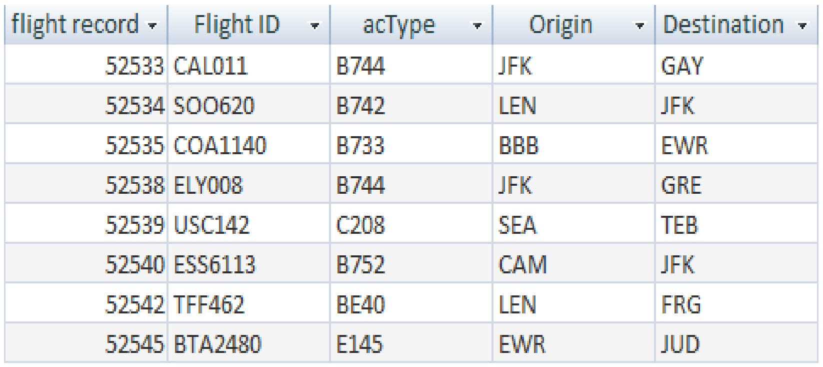
Table 7 List of leading and following aircraft pairs



52533	52534
52533	52535
52533	52536
52533	52537
52533	52540
52533	52541
52533	52542
52534	52535

The PDARS Flight list files and PDARS data originate from the raw PDARS data and include OD pair and aircraft type in the first, and times, coordinates, headings, and speeds in the second. Aircraft ID is the key linking these two files. Table 8 below shows a sample output of the Flights list file, and Table 9 shows sample PDARS data file.

Table 8 Flight list data



flight record	Flight ID	acType	Origin	Destination
52533	CAL011	B744	JFK	GAY
52534	SOO620	B742	LEN	JFK
52535	COA1140	B733	BBB	EWR
52538	ELY008	B744	JFK	GRE
52539	USC142	C208	SEA	TEB
52540	ESS6113	B752	CAM	JFK
52542	TFF462	BE40	LEN	FRG
52545	BTA2480	E145	EWR	JUD

Table 9 Parced and modified PDARS data

flight record	Time	Latitude	Longitude	Altitude	speed	heading
52533	1205901651.73	40.63518	-73.78761	4.76	153	112
52533	1205901660.714	40.632153	-73.78043	7.46	153	112
52533	1205901669.583	40.630466	-73.775032	9.18	155	110
52533	1205901673.996	40.628952	-73.769562	10.98	159	102
52533	1205901678.408	40.628	-73.763746	12.78	163	95
52533	1205901682.838	40.62767	-73.758424	14.36	167	91
52533	1205901691.794	40.627566	-73.752984	15.76	168	91
52533	1205901696.315	40.627496	-73.747436	16.96	171	90
52533	1205901700.86	40.627482	-73.741806	18.16	173	89
52533	1205901705.42	40.627534	-73.736988	18.96	175	89
52533	1205901709.994	40.62761	-73.73206	19.76	177	89
52533	1205901714.606	40.6277	-73.72714	20.76	178	88
52533	1205901719.164	40.627802	-73.722184	21.76	180	88
52533	1205901723.79	40.627912	-73.717194	22.76	181	88

During the data analysis and the enhanced wake modeling two issues were detected with the PDARS data used as an input to the model. The first issue was an inconsistency in the heading values listed in the PDARS data compared to the same values computed using the PDARS latitude and longitude. New heading values were computed and used in the envelope calculations. Additionally, PDARS obtained latitude, longitude and altitudes were smoothened using moving average technique to make the trajectory more realistic. The trajectories plotted with the original PDARS values had many jumps due to the uneven time steps used during data collection. On average, radar data is sampled every 5 to 10 seconds, which is the time required by the radar to make one full 360 degree scan. Figure 20 shows the difference in the trajectory before and after using the moving average values.

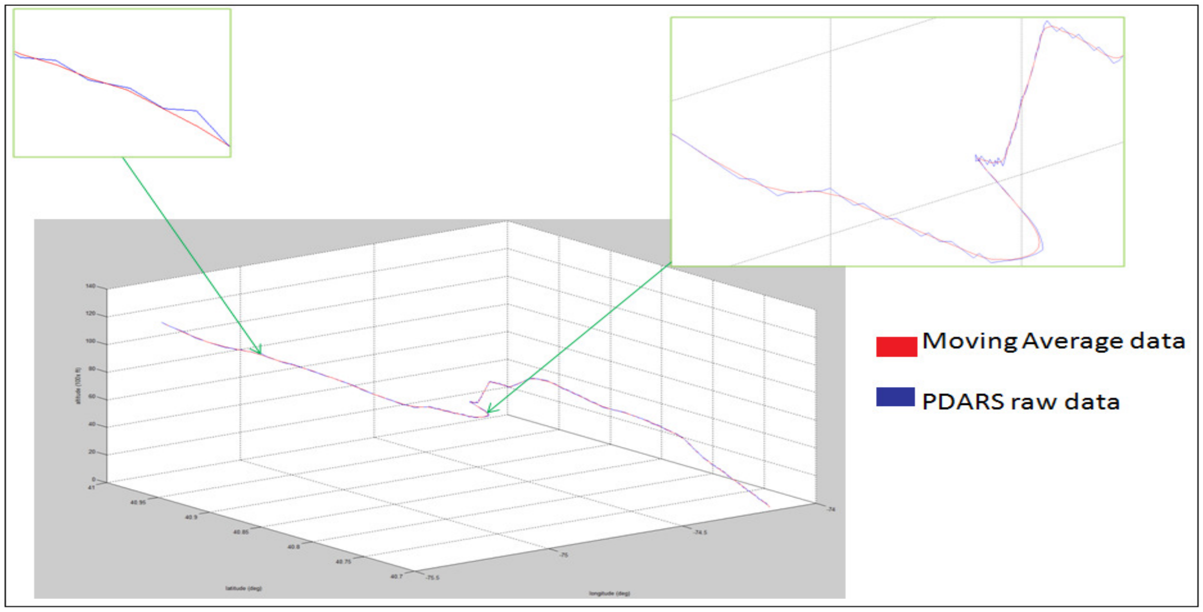


Figure 20 PDARS trajectory using raw data versus smoothed data

The next file, called ISA Density file.xls, consists of density values for different altitudes. ISA conditions are selected as operational, meaning that we have air temperature of 59 degrees Fahrenheit and an air pressure of 29.92” of mercury at sea level (NASA 1976). The circulation strength is dependent on the altitude, and therefore the WEM interpolates based on the current altitude at every time step to find the air density.

Table 10 Density of air versus altitude (ISA)

Altitude (m)	**Air Density (kg/m^3)**
0	1.225
1,000	1.112
2,000	1.007
3,000	0.909
4,000	0.819
5,000	0.736
6,000	0.66
7,000	0.589
8,000	0.525
9,000	0.466
10,000	0.413
11,000	0.364
12,000	0.311
13,000	0.266
14,000	0.227
15,000	0.194

3.4.3. Wake Encounter Model Output Files

An additional change was made to the output files in the Wake Encounter Model. The first version of the model was saving information about a potential encounter in a text format including just information about the IDs of the pair, and whether there were an encounter or not using a binomial variable *wakepiercing* that had the value of 1 for a potential encounter and 0 otherwise. In the recently updated model, the saved information was expanded to include not only the IDs, but the locations of the two aircraft (latitude, longitude, and altitude), speed and heading of the pair, and length of the produced wake envelope in nautical miles. The text file can then be used for further analysis not only in MATLAB, but in Excel, Access and other software because it includes all of the required data in one place eliminating the need to open and review other files.

3.4.4. Wake Encounter Model CPU Time Optimization

The time to simulate a whole day of operations in a TRACON in the original program was anywhere between 20 to 50 hours for the different TRACONs. The script was using only a single core so an attempt was made to utilize MATLAB's Parallel Computing Toolbox. Three cores were used which resulted in a 60% reduction in computing time. Later, the MATLAB script was re-written and the completion time was reduced to approximately 2.5-6 hours even with the added procedures such as the corrected shape and encounter animations and using a single core.

3.4.5. Development of Additional Tools

Additional improvements include two visualization procedures used in the instances when the model determines that there is a potential encounter. The two tools are a three-dimensional plot and an animation tool.

The three-dimensional plot illustrates the position of the wake generator along the PDARS track, and the position of the wake envelope and the potential wake encountering airplane. The plot can be viewed from different angles to better understand the interactions between the wake envelope and the following aircraft.

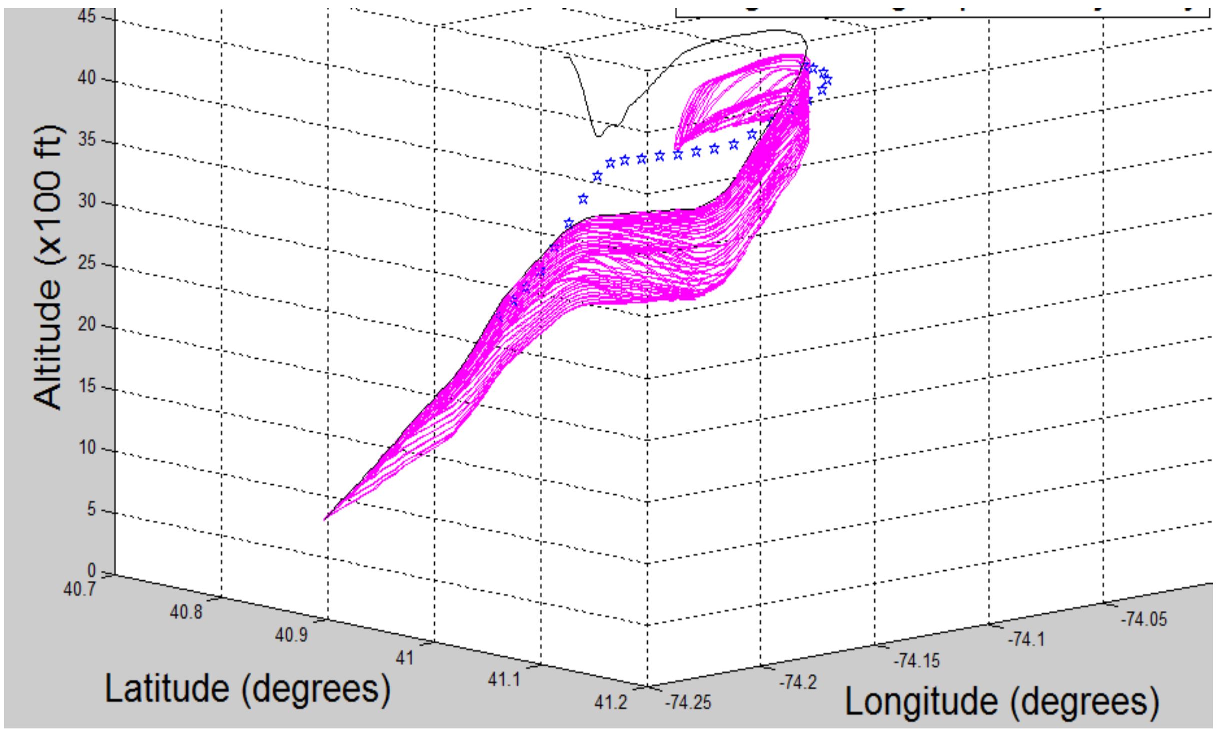


Figure 21 Wake envelope and following aircraft interaction

The animation tool was developed as an extension to the WEM as a way to visualize the interactions between potential wake pairs of aircraft. Even though the design of the animation tool is very simple, it provides important information about the interacting pair, and mainly visualizes the interaction. The tool uses the output text file from the WEM containing the list of all of the potential encounters between aircraft pairs. The first thing that the tool does is to separate the results into pairs of aircraft and extract the times, longitudes, latitudes, and altitudes of each pair. Then, the tool identifies common starting and finishing times for each pair. The times for each aircraft are normalized, and aircraft coordinates are recomputed for a set of common times using interpolation. The time step of the common times can be user defined with a default value of 10 sec. The animation tool then simulates the interaction by flying the two airplanes together. Airport locations can also be plotted on the animation plot for easier identification of origin or destination location. Each pair interaction is animated in three dimensions, and can be rotated by the user to view from different angles. Closest points of approach can also be determined and used for analysis as shown on Figure 22. Relative positions at any given time are known, and can be plotted and compared to the current separation standards. Figure 23 below shows the final stage of the 3D animation of a pair landing at JFK. From the top view of the animation we can see that the pair is landing on the same runway which is important information for the type of potential encounter.

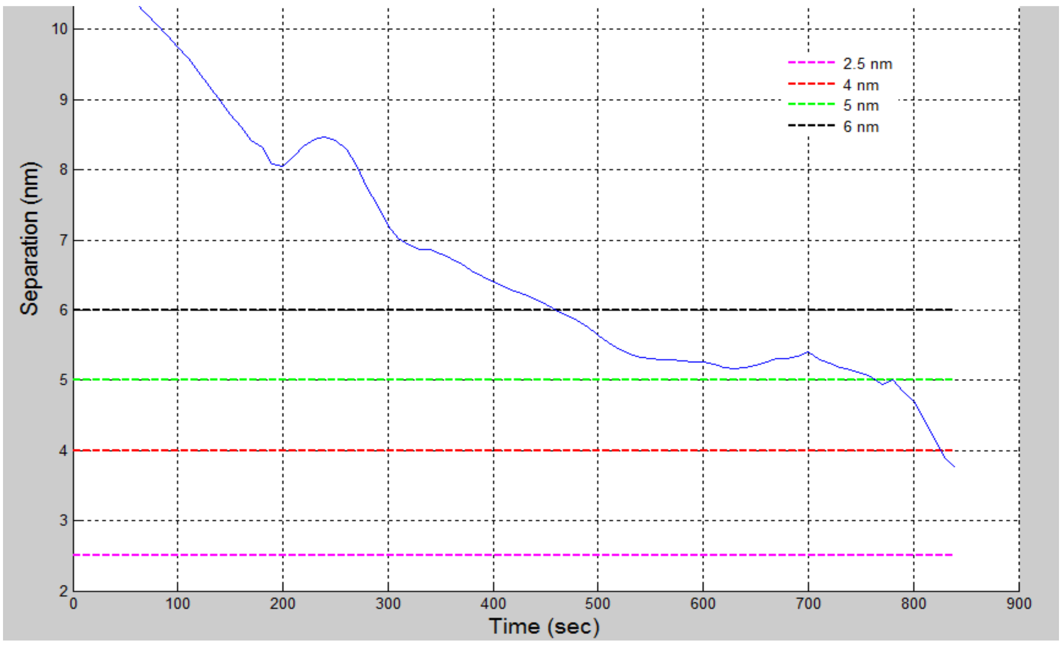


Figure 22 Aircraft pair separation versus time, and current separation minima

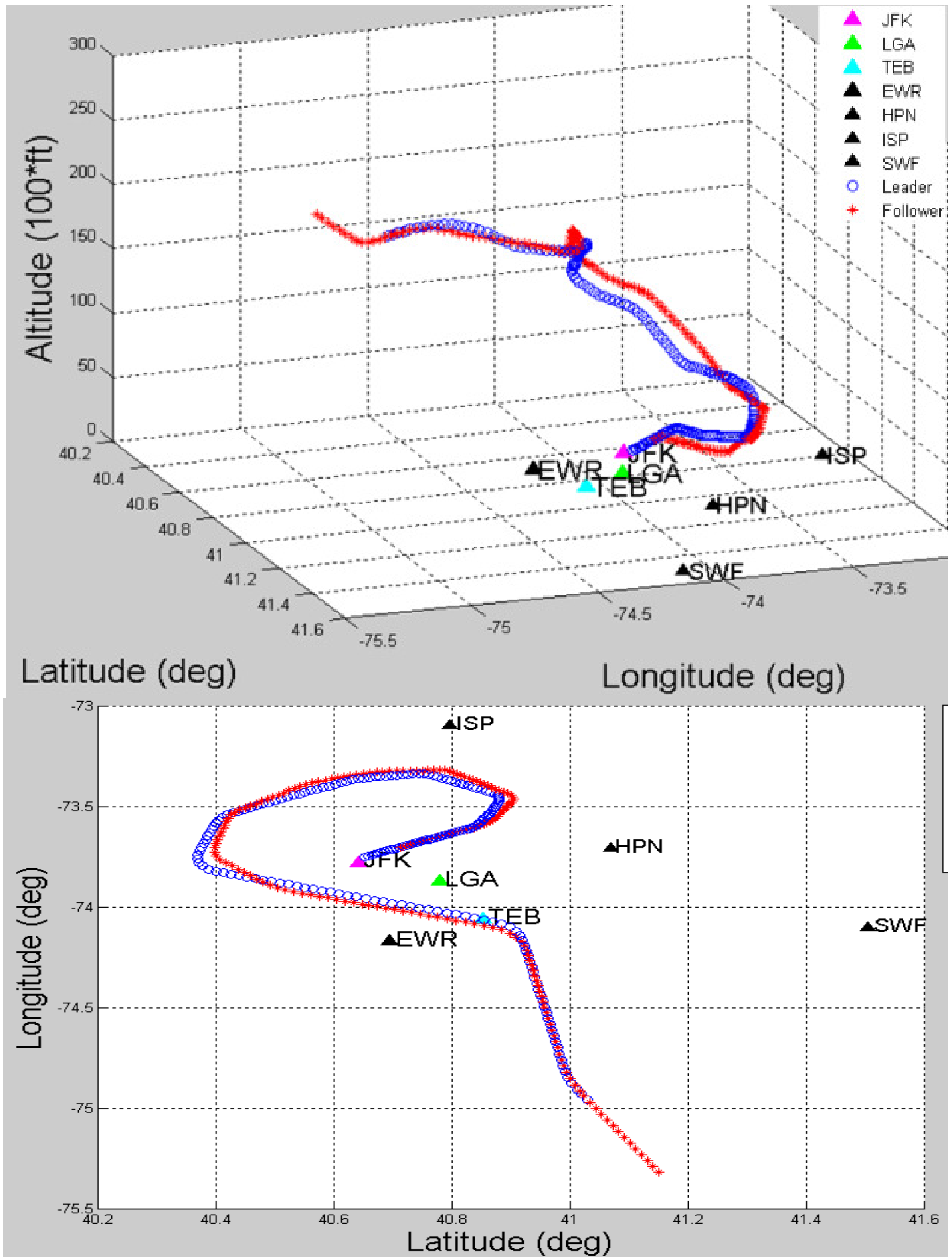


Figure 23 Animation tool showing the trajectories of two aircraft landing in JFK

4. RESULTS AND VALIDATION

4.1 Overall Results and Validations

In order to establish a base line, the WEM was used to analyze the current conditions in three major TRACONs, New York, Atlanta, and Los Angeles for three selected days with IMC, VMC, and MMC conditions at each area, and compared the results to the ones obtained by Swol in the original model. The original results are shown in Table 11 below, where Y represents a potential encounter and blank cell represents no encounter.

Table 11: Results from original model with the wake envelope modeled as a straight solid piece

				Circulation	75	125	175	75	125	175	75	125	175
				EDR	0.0001	0.0001	0.0001	0.0020	0.0020	0.0020	0.0040	0.0040	0.0040
				BVF	0.0005	0.0005	0.0005	0.0100	0.0100	0.0100	0.0200	0.0200	0.0200
TRACON	Date	Number of flights	Conditions	Aircraft Type	Potential Encounter								
NYC	2/5/2008	4,718	VMC	CRJ2	Y	Y	Y						
	3/19/2008	5,153	MMC	Boeing 735	Y	Y	Y	Y					
	6/23/2008	4,765	IMC	Cessna 525	Y	Y	Y						
				Cessna 650	Y	Y	Y	Y	Y		Y		
				Cessna 750	Y	Y		Y					
				J328	Y	Y	Y						
				PA31	Y								
SC	6/18/2008	5,380	VMC	Embraer 145	Y	Y	Y						
	8/13/2008	5,264	MMC	-									
	7/18/2008	5,453	IMC	Embraer 120	y								
ATL	6/26/2008	3,867	VMC	-									
	7/13/2008	2,740	MMC	Boeing 744	Y	Y	Y						
	2/22/2008	3,175	IMC	-									

As mentioned earlier, because the envelope was modeled as a rigid piece behind the airplane relative to the instantaneous position of the aircraft the model was over predicting the potential encounters. After correcting the envelope shape, the same flights were re-tested for the same values of EDR, BVF, and circulation strength (CS) and neither resulted in a potential encounter. The enhanced model, however determined new instances of potential encounters mostly arrivals and mostly in-trail. Figure 24 illustrated a few instances when the old model will flag the interaction between the airplanes as a potential encounter, and the enhanced one will not. On the figure, the generating aircraft trajectory is shown in black, the original envelope design extending straight, shown in blue, the enhanced envelope following the trajectory—in magenta, and the following aircraft shown as black asterisks.

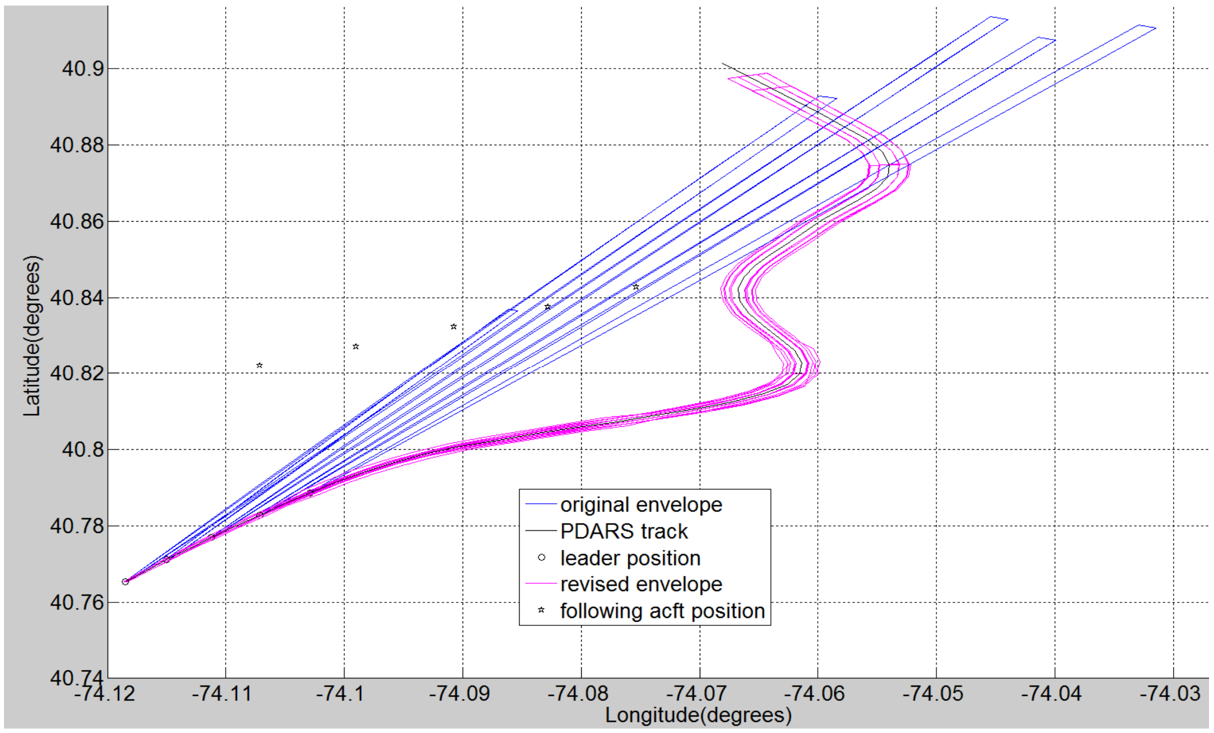


Figure 24 Improved wake envelope modeling shows no potential encounters

The results obtained from the enhanced wake encounter model are shown in Table 12 As we can see from the table, there are more potential encounters for the low values of the atmospheric parameters, which represent a slowly decaying wake. As the values increase, all potential encounters are eliminated.

Table 12 Results from MATLAB runs of WEM for all TRACONS

			Circulation	**125**	**125**	**125**	**175**	**175**	**175**
			EDR	**0.0001**	**0.0020**	**0.0040**	**0.0001**	**0.0020**	**0.0040**
			BVF	**0.0005**	**0.0100**	**0.0200**	**0.0005**	**0.0100**	**0.0200**
TRACON	**Date**	**Number of flights**	**Conditions**	**Total Number of Potential Encounters**					
NYC	2/5/2008	4718	VMC	6	1	0	6	0	0
	6/23/2008	5153	MMC	11	2	0	11	0	0
	3/19/2008	4765	IMC	12	0	0	9	0	0
SC	6/18/2008	5380	VMC	16	2	0	9	0	0
	8/13/2008	5264	MMC	12	3	0	7	0	0
	7/18/2008	5453	IMC	11	2	0	7	0	0
ATL	6/26/2008	3867	VMC	17	1	0	5	0	0
	7/13/2008	2740	MMC	12	0	0	5	0	0
	2/22/2008	3175	IMC	3	1	0	0	0	0

A weakness related to a function used in the WEM, however was discovered that requires further improvement. To determine if there is a potential encounter, the model employs a MATLAB function called “inhull” which tests if a point lies within a convex shape. Since the wake envelope is non-convex, the function will sometimes flagged an instance as an encounter when the point is outside of the envelope. This issues is currently resolved using the three dimensional plots of the potential encounters, where the user can inspect the plot of the interaction. An example of a “false” encounter flag is shown on Figure 25.

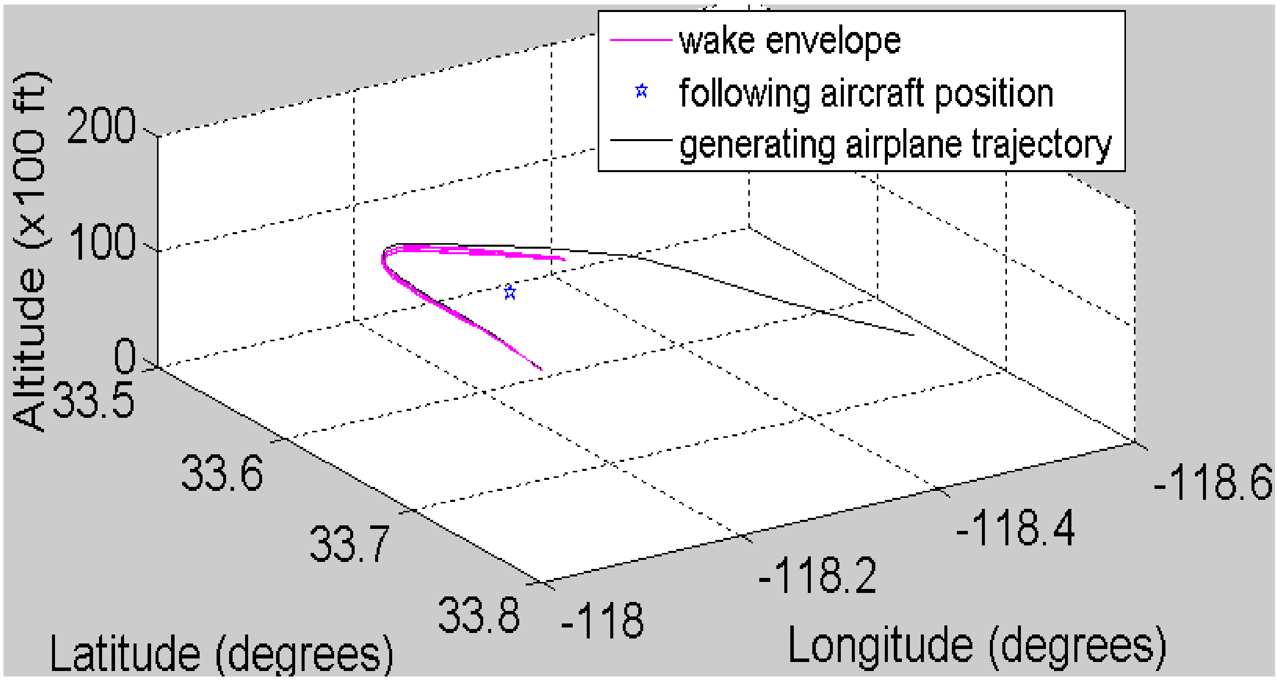


Figure 25 Instances of non-convex flag weakness

4.2 Qualitative Analysis using Aviation Safety Reports

NASA’s Aviation Safety Reporting System or ASRS is a voluntary database, where the aviation personnel (i.e. pilots, flight attendants, etc) can voluntary report safety issues. The information submitted is anonymous and non-punitive, and can be useful for policy development, human factor studies, and training (NASA 2011). The database is publicly available online, and it can be beneficial in the calibration of the Wake Encounter Model because it includes narratives from flight crews that have experienced wake turbulence. The report includes a brief narrative, aircraft information for both aircraft, location of the actual encounter, wind information, and other valuable parameters such as magnitude of the encounter. During the period of August, 2007 to October 1, 2010 there were 129 structured callbacks related to wake vortex completed and entered in the database. 109 out of the 129 were passenger operations. Pilots were also asked to report the severity of the wake encounter grouping it in the following four categories: Light, Moderate, Severe and Extreme. The results can be seen on Figure 26. Reports were mainly for moderate and severe turbulence, and one encounter was reported as extreme.

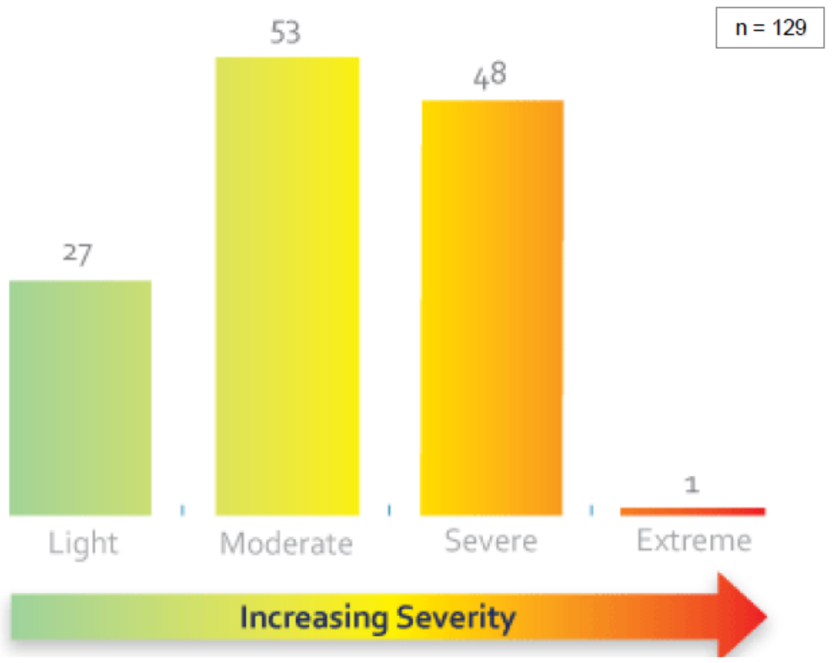


Figure 26 Pilot assessment of the wake encounter magnitude for the 129 call backs (Taube 2010). Used under Fair Use Guidelines

Figure 27 illustrates the breakdown of the encounters per reported location, and type of operation. We can conclude that about 68% of the encounters occurred in the terminal area, and another 68% of them happened during arrivals. Additionally, LAX has claimed 18% of the total number of encounters during arrivals for the period of interest, the highest number of all other airports.

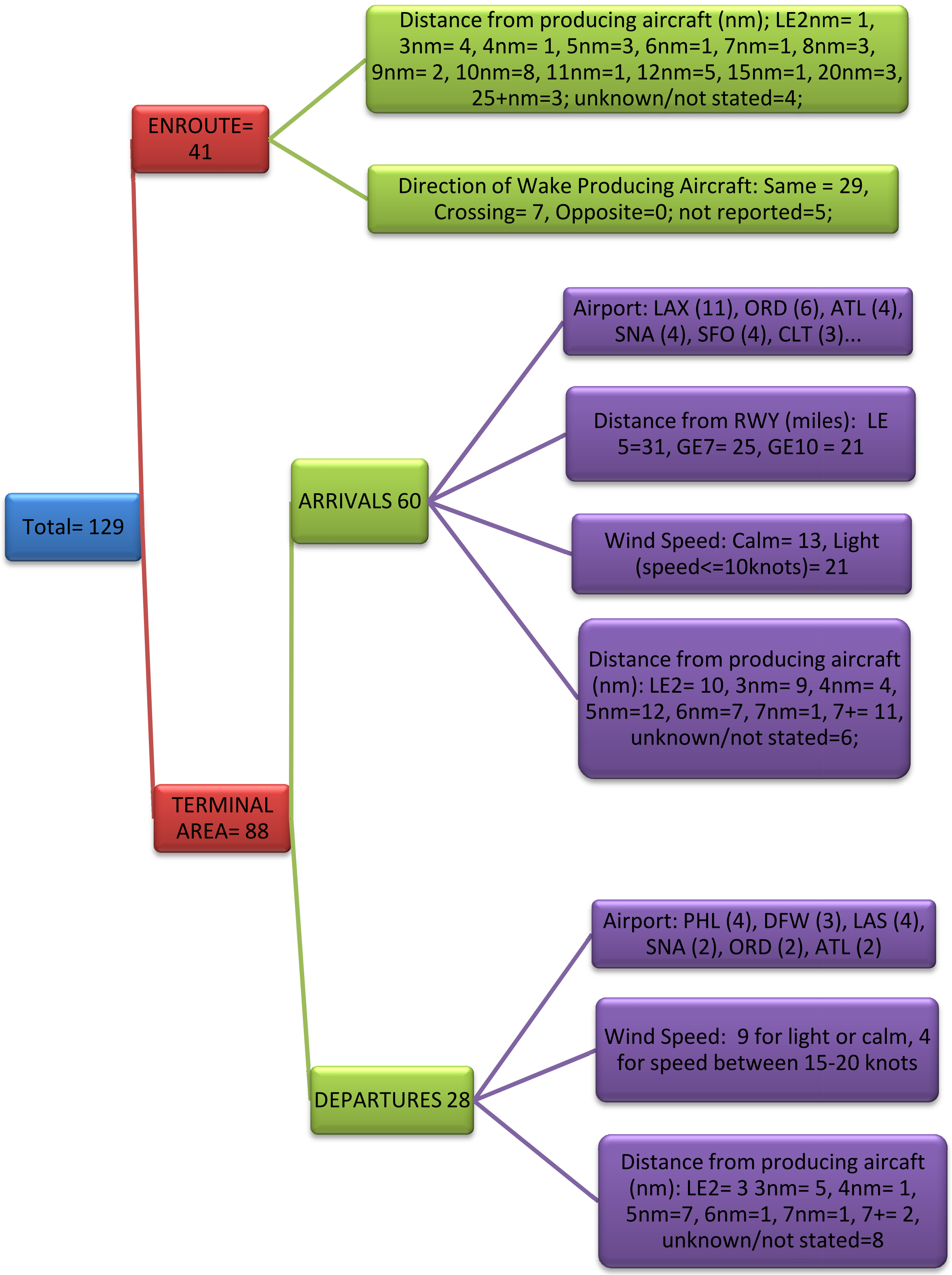

Total= 129
ENROUTE= 41
Distance from producing aircraft (nm); LE2nm= 1, 3nm= 4, 4nm= 1, 5nm=3, 6nm=1, 7nm=1, 8nm=3, 9nm= 2, 10nm=8, 11nm=1, 12nm=5, 15nm=1, 20nm=3, 25+nm=3; unknown/not stated=4;
Direction of Wake Producing Aircraft: Same = 29, Crossing= 7, Opposite=0; not reported=5;
TERMINAL AREA= 88
ARRIVALS 60
Airport: LAX (11), ORD (6), ATL (4), SNA (4), SFO (4), CLT (3)...
Distance from RWY (miles): LE 5=31, GE7= 25, GE10 = 21
Wind Speed: Calm= 13, Light (speed<=10knots)= 21
Distance from producing aircraft (nm): LE2= 10, 3nm= 9, 4nm= 4, 5nm=12, 6nm=7, 7nm=1, 7+= 11, unknown/not stated=6;
DEPARTURES 28
Airport: PHL (4), DFW (3), LAS (4), SNA (2), ORD (2), ATL (2)
Wind Speed: 9 for light or calm, 4 for speed between 15-20 knots
Distance from producing aircaft (nm): LE2= 3 3nm= 5, 4nm= 1, 5nm=7, 6nm=1, 7nm=1, 7+= 2, unknown/not stated=8

Figure 27 Breakdown of the reports based on the encounter location

Additionally, Taube (2010) had grouped the encounters per altitude of occurrence showing that for both arrivals and departures the majority occur at altitudes below 5,000ft. Figure 28illustrates the altitudes for encountering and producing aircraft during arrivals, and Figure 29altitudes during departures.

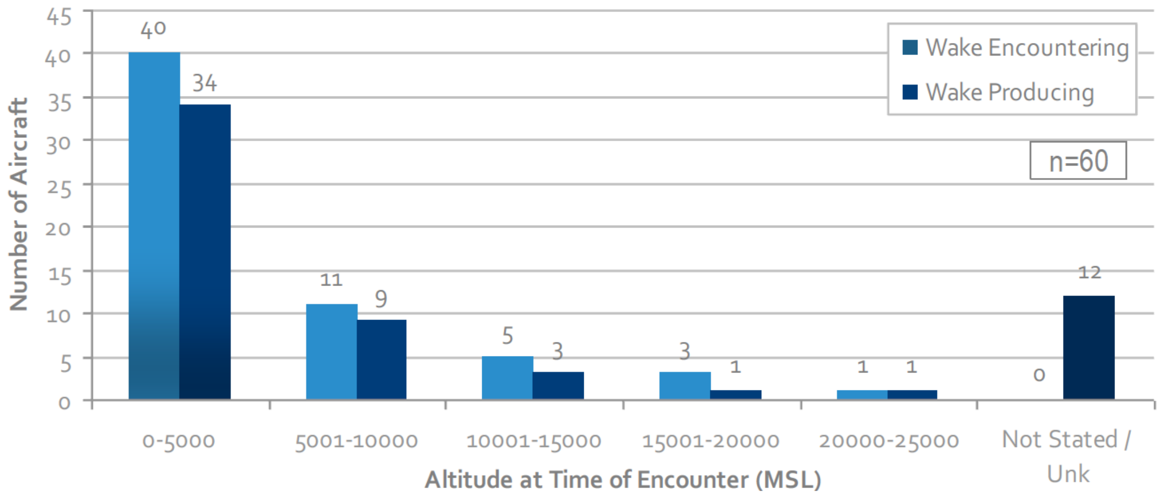


Figure 28 Altitude of wake encountering and wake producing aircraft - Arrival (Taube, 2010). Used under Fair Use Guidelines

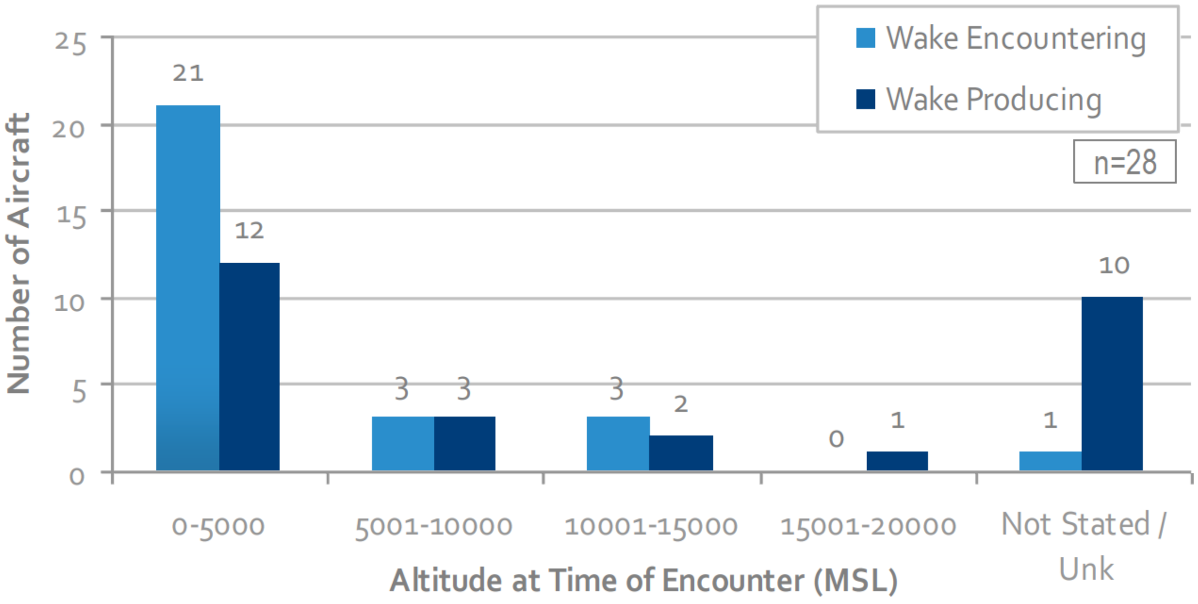
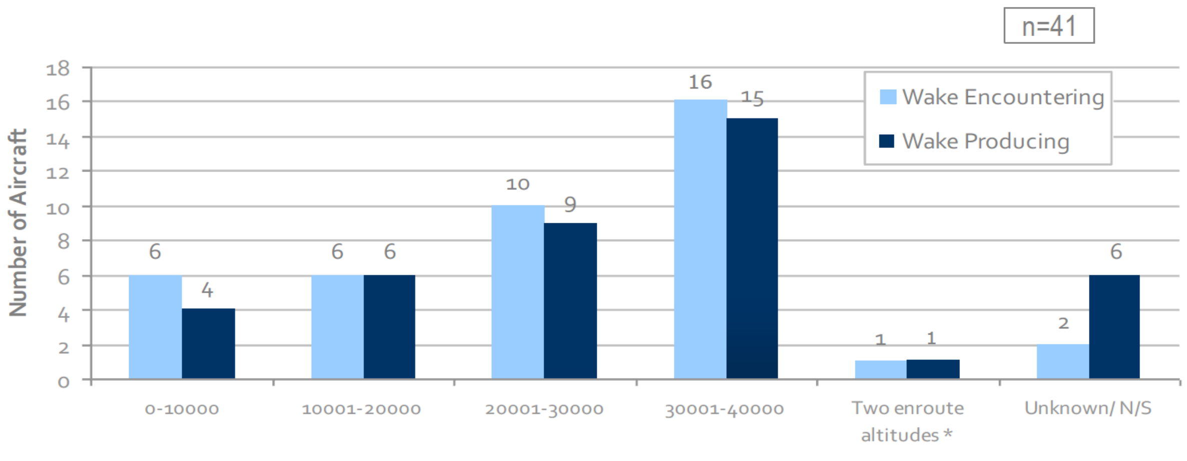


Figure 29 Altitude of wake encountering and wake producing aircraft - Departure (Taube, 2010). Used under Fair Use Guidelines

For en route operations, the majority of the reported wake encounters occurred at altitudes between 30,000 and 40,000 feet as seen inFigure 30. The second largest group of altitudes where encounters were reported is between 20,000 and 30,000ft.



*1 aircraft encountered wake at 2 altitudes-- 10,500 MSL and 25,500 feet MSL (aircraft producing wake was 4,000 feet higher for each encounter)

Figure 30 Altitude of Wake Encountering and Wake Producing Aircraft – En route (Taube, 2010). Used under Fair Use Guidelines

The presented results from the ASRS reports related to type of operations are consistent with the finding from WEM. For example, looking at South California TRACON, the model predicted the following potential encounters shown in Table 13.

Table 13 Results from WEM for Southern California TRACON

			Circulation	125	125	125	175	175	175
			EDR	0.0001	0.0020	0.0040	0.0001	0.0020	0.0040
			BVF	0.0005	0.0100	0.0200	0.0005	0.0100	0.0200
TRACON	Date	Number of flights	Conditions	Total Number of Potential Encounters					
SC	6/18/2008	5380	VMC	16	2	0	9	0	0
	8/13/2008	5264	MMC	12	3	0	8	0	0
	7/18/2008	5453	IMC	11	2	0	7	0	0

We can then look further at the results from the three days of operations and find the following breakdowns for the potential encounters grouped by meteorological conditions combinations:

Table 14 Encounters per operation type for Southern California TRACON three days of flights

	All 3 Days of SCT Operations					
	EDR 0001 BVF 005 CS 125		EDR 0001 BVF 005 CS 175		EDR 002 BVF 01 CS 125	
	Count	Percent value	Count	Percent value	Count	Percent value
Departure	12	31%	11	46%	4	57%
Arrival	26	67%	12	50%	3	43%
Enroute	1	3%	1	4%	0	0%

We can conclude that there are more potential encounters during arrivals than during departures, except in the last case where the values are slightly higher for departures. This conclusion agrees with the results from the ASRS reports presented earlier in this chapter.

Additionally, if we want to look at the instances in the South California TRACON when the WEM determines a potential encounter and LAX is either origin or destination, we can obtain the following Table 15:

Table 15 Instances when LAX was either origin or destination

	Count of LAX	Total number of events	Percent value
EDR 0001 BVF 005 CS 125	29	39	74%
EDR 0001 BVF 005 CS 175	21	24	88%
EDR 002 BVF 01 CS 125	6	7	86%

LAX is either origin or destination over 75% of the time. To qualitatively compare the result from ASRS however, we will need data from the other airports listed as top locations of wake encounters.

Lastly, the ASRS measures the consequences of wake encounters, and some of the categories are presented inFigure 31. Even though in the majority of time there were no significant consequences reported, there were instances where physical injuries were reports, as well as temporarily loss of aircraft control. This further stresses the importance of wake turbulence in the safety of flight operations.

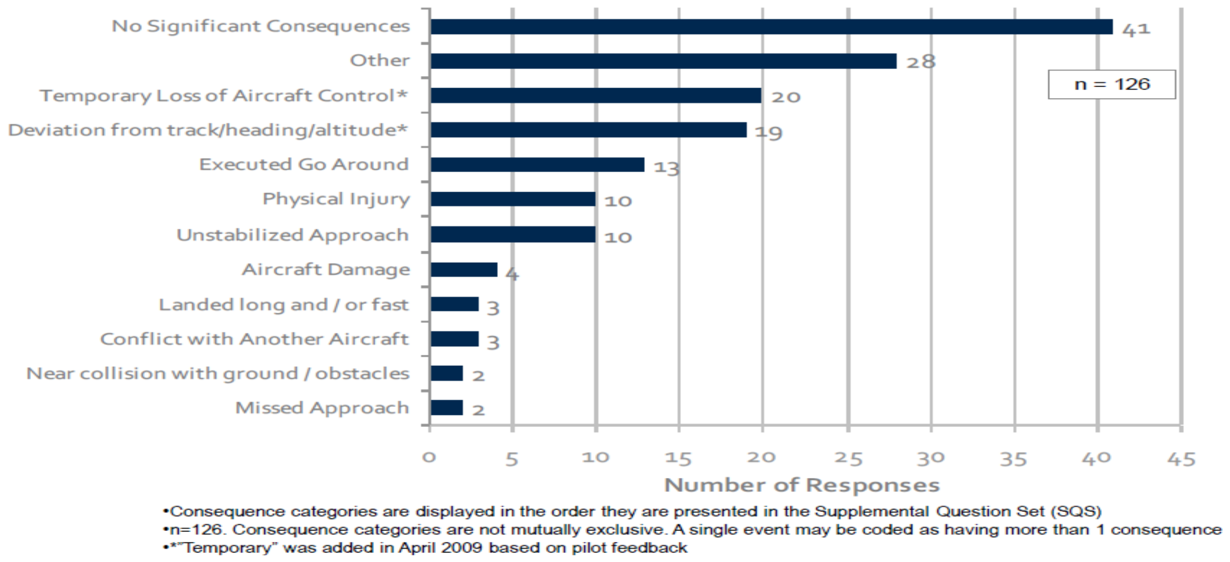


Figure 31 Consequences of WVE (Taube 2010). Used under Fair Use Guidelines

4.3 EDR and BVF values analysis

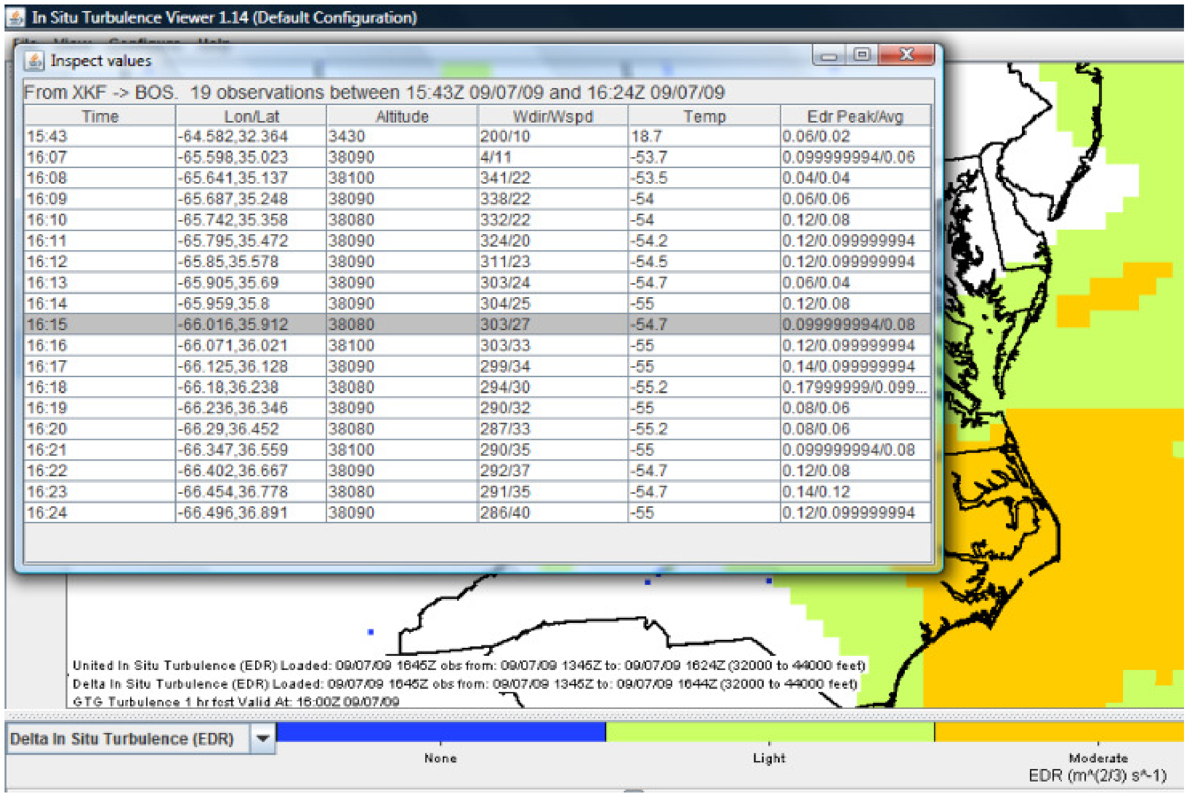
4.3.1 Current Turbulence Conditions Observations

As described in the Literature Review section, the EDR and BFV values were obtained from a paper by Riddick and Hinton. Suggestion for future research given by Swol included a recommendation of further analysis of these parameters.

Additional Literature Review was performed to compare the values of BVF and EDR used in the model and recommended in the literature. As already mentioned, these parameters are not readily available from any weather of meteorological databases.

Currently, they are two types of turbulence condition observations, namely in situ observations and automated vertical accelerometer (AVAR) observations. The in situ observations require a software installation on the aircraft and are based on a transformation of the observed vertical acceleration. The measurements obtained from the second type are independent of aircraft characteristics and motions, and are based on a transformation of the observed vertical acceleration. Currently, however, only pilot reports (PIREPs) and some AVAR observations have been used for verification of current and new systems for forecasting turbulence conditions (Brown 2000).

In the past few years, the National Center for Atmospheric Research (NCAR) has developed a technology to use with existing aircraft performance systems to derive an automated turbulence measurement (in terms of EDR values) as part of the FAA Turbulence Product Development Team (PDT). Delta and United Airlines are two of the airlines that have been participating in the program for automatic turbulence reports, where EDR values were measured at different altitudes by special software installed at the flying aircraft. The measurement is “real time, scientific, objective aircraft independent.” As of October 2009, all Delta Boeing 737, and about 100 of United Airlines have been equipped with EDR measuring devices, and the program is expanding to include more airlines and airplanes. Some of the pick EDR values found in one of the reports range from 0.02 to about 0.18 $m^{2/3}s^{-1}$. Figure 32illustrates the 19 in situ values measured on a flight to Boston Logan airport on September 7, 2009 used in the example (Delta Airlines 2009).


In Situ Turbulence Viewer 1.14 (Default Configuration)
Inspect values
Delta In Situ Turbulence (EDR)
None
Light
Moderate
EDR (m^(2/3) s^-1)

From XKF -> BOS. 19 observations between 15:43Z 09/07/09 and 16:24Z 09/07/09

Time	Lon/Lat	Altitude	Wdir/Wspd	Temp	Edr Peak/Avg
15:43	-64.582,32.364	3430	200/10	18.7	0.06/0.02
16:07	-65.598,35.023	38090	4/11	-53.7	0.09999994/0.06
16:08	-65.641,35.137	38100	341/22	-53.5	0.04/0.04
16:09	-65.687,35.248	38090	338/22	-54	0.06/0.06
16:10	-65.742,35.358	38080	332/22	-54	0.12/0.08
16:11	-65.795,35.472	38090	324/20	-54.2	0.12/0.09999994
16:12	-65.85,35.578	38090	311/23	-54.5	0.12/0.09999994
16:13	-65.905,35.69	38090	303/24	-54.7	0.06/0.04
16:14	-65.959,35.8	38090	304/25	-55	0.12/0.08
16:15	-66.016,35.912	38080	303/27	-54.7	0.09999994/0.08
16:16	-66.071,36.021	38100	303/33	-55	0.12/0.09999994
16:17	-66.125,36.128	38090	299/34	-55	0.14/0.09999994
16:18	-66.18,36.238	38080	294/30	-55.2	0.17999999/0.099...
16:19	-66.236,36.346	38090	290/32	-55	0.08/0.06
16:20	-66.29,36.452	38080	287/33	-55.2	0.08/0.06
16:21	-66.347,36.559	38100	290/35	-55	0.09999994/0.08
16:22	-66.402,36.667	38090	292/37	-54.7	0.12/0.08
16:23	-66.454,36.778	38080	291/35	-54.7	0.14/0.12
16:24	-66.496,36.891	38090	286/40	-55	0.12/0.09999994

United In Situ Turbulence (EDR) Loaded: 09/07/09 1645Z obs from: 09/07/09 1345Z to: 09/07/09 1624Z (32000 to 44000 feet)
Delta In Situ Turbulence (EDR) Loaded: 09/07/09 1645Z obs from: 09/07/09 1345Z to: 09/07/09 1644Z (32000 to 44000 feet)
GTG Turbulence 1 hr fcst Valid At: 16:00Z 09/07/09

Figure 32 Screen capture of In Situ Turbulence Viewer with EDR values (Delta Airlines 2009). Used under Fair Use Guidelines

The units of EDR, however, used in the WEM are different than the units in the above example. The WEM uses m^2/s^3, and the values for EDR in these units are 0.0001, 0.002, and 0.004. Table 16 summarizes the conversions between the two types of Total Kinetic energy (TKE) units.

Table 16 EDR unit conversion

EDR [m^(2/3)/s]	EDR (WEM) [m^2/s^3]
0.02	0.00001
0.03	0.00003
0.04	0.00006
0.05	0.0001
0.06	0.0002
0.13	0.002
0.16	0.004

The presentation describing Delta airline's in situ measurements however does not mention what values are considered moderate and what values are considered light (weak) turbulence, i.e. not turbulence standards have been established.

However, another research effort has been done as part of the FAA Aviation Weather Research Program, to outline the relationship between AVAR observation and PIREPs. In the report an EDR value of 0.05 is listed as a no (or minimal) turbulence, and nearly all of the values shown in the report for the continental United States fall into that category (Takacs 2004).

Additionally, Figure 33illustrates a 24-hour measurement of in situ EDR values over the continental United States. Figure 33includes color codingfor EDR values with units of $m^{2/3}s^{-1}$. The top plot in Figure 33is from UAL B757-200 aircraft, and the one at the bottom is from DAL 737-800 aircraft. As seen in the figure, the in situ values are mostly below 0.1 $m^{2/3}s^{-1}$.

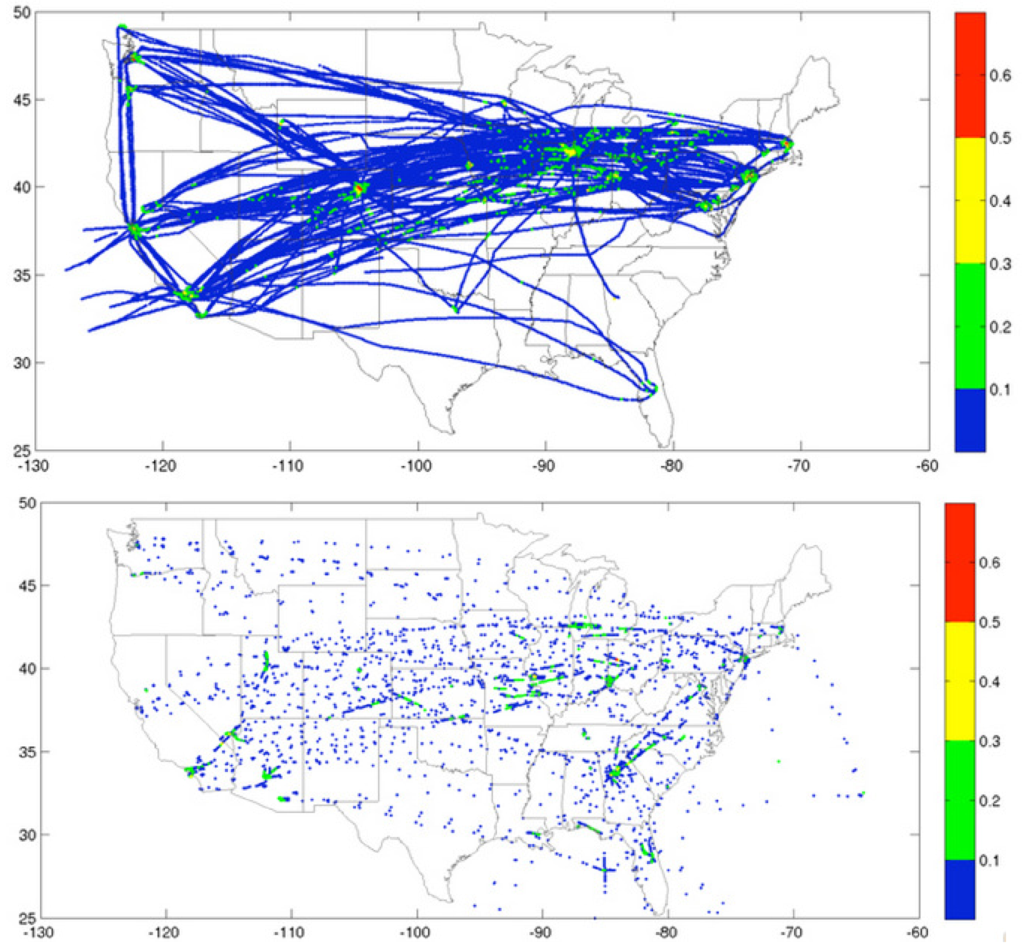


Figure 33 Coverage of 24 hour period of EDR measurements. Top figure: UAL B757-200 with one minute routine reporting. Bottom figure: DAL B737-800 with event-based EDR reporting plus 15-min routing reporting (National Center for Atmospheric Research 2011). Used under Fair Use Guidelines

PIREPs, or pilot reports, are textual messages sent by the pilots and include a verbal description of the severity of the experienced turbulence during flight, such as smooth, moderate, severe, or extreme. The value is then converted to a numerical scale from 0 to 8, where 0 corresponds to no turbulence, and 8 corresponds to extreme turbulence. A weakness of this way of reporting is the "subjective nature of the intensities reported and the imprecise time and location of the reported encounter" (Takacs 2004). PIREPs can be accessed from the Aviation Digital Data Service (ADDS) of the National Oceanic and Atmospheric Administration's (NOAA's) National Weather Service website (NOAA's National Weather Service 2010). PIREP of turbulence for JFK airport from December, 28 2010 is shown inFigure 34.

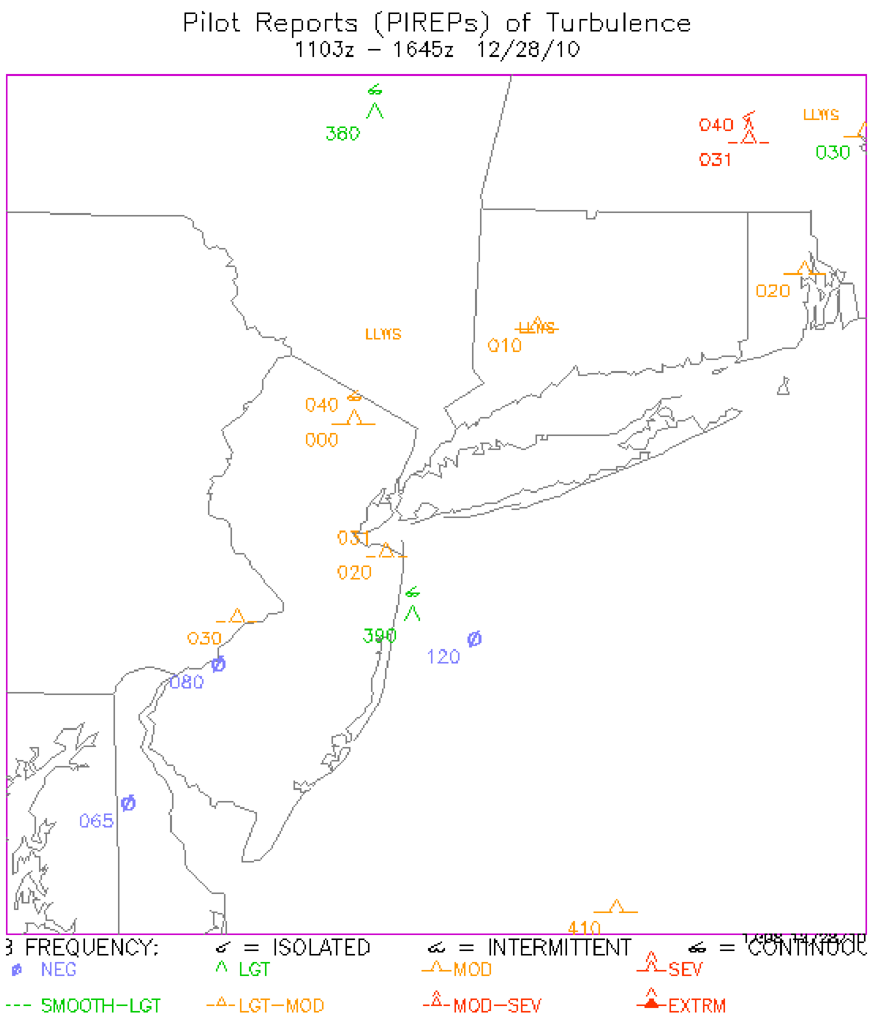


Figure 34 JFK past 12 hours

Additionally a Java tool is available at the same website. The Java tool can be used to interactively load and show an overview of PIREPs for a specific area as shown onFigure 35. The user can specify the type of report, i.e. turbulence or icing, and select a time frame from the time selector. The interface is user friendly and the results are provided quickly.

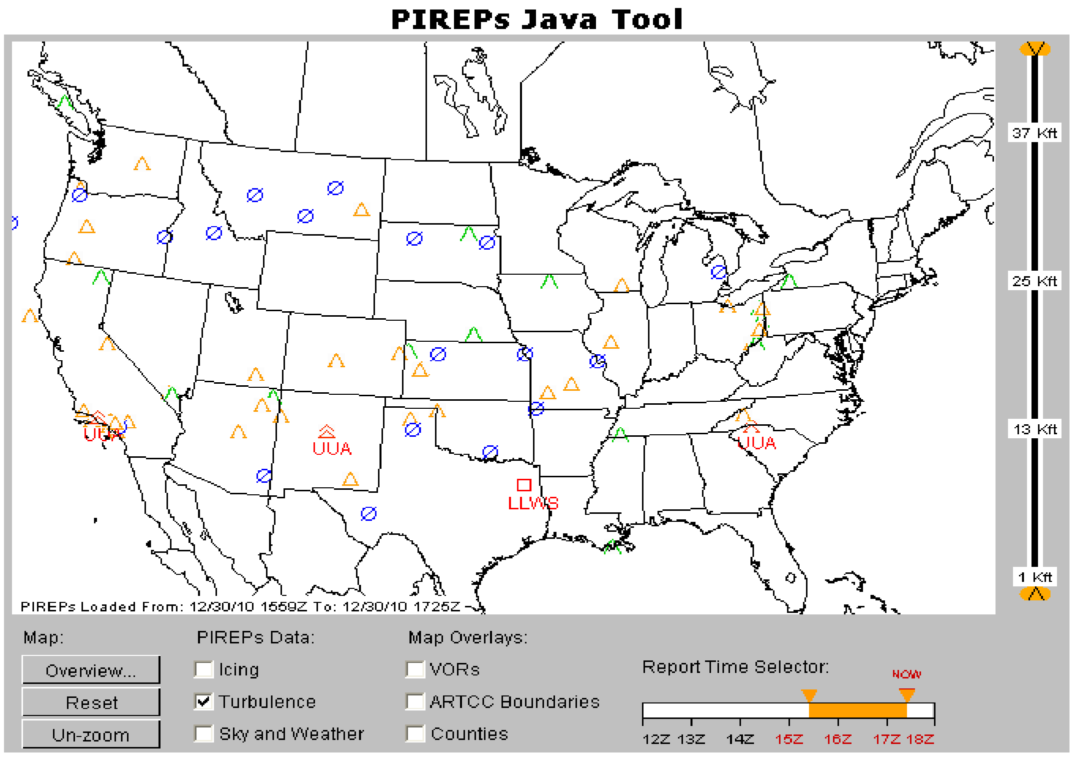


Figure 35 PIREPs Java tool interface

4.3.2 EDR Analysis for WEM

As part of the analysis and calibration of the Enhanced Wake Encounter Model, some qualitative statistical analysis was performed for the different EDR values. The purpose of this analysis was to select the best parameters of EDR and BVF for the EWEM for future use. Using the PDARS data and values of wake envelope lengths predicted by the EWEM, a series of histograms of the distributions were generated. A representative aircraft was selected from each of the wake categories and ran through the EWEM model. The following critical aircraft were selected in the analysis: Airbus A380-800 (Super heavy), Boeing 747-400 (Heavy), Boeing 737-800 (Large), SF340 (Small). Arbitrarily, PDARS data from New York TRACON dated 03/19/2008 was selected for the analysis. All flights with the corresponding critical aircraft were divided into groups and using the EWEM model, wake envelopes were produced for all available data points along the PDARS tracks available for the day selected. The values of wake envelope length aredifferent for the different values of EDR, BVF and circulation threshold. For each critical aircraft a total of six histograms were generated, corresponding to values of EDR= 0.0001, 0.02, 0.04 m^2/s^3, BVF= 0.0005, 0.02, 0.04 s^{-1}, and CS=125, 175 m/s^2, and the values for the envelope length of each were compared to the current separation criteria to determine which set of parameters results in envelope lengths closer to the current separations.

The general conclusion is that for all critical aircraft the values that result in envelope lengths closer to the current in-trail separation conditions are: EDR= 0.04, BVF= 0.2, and circulation threshold =175. For example, the histograms for the Airbus A380, we observe few values of envelope lengths of about 15-16nm, but the majority fall under 10nm which is the current spacing required when following a super heavy aircraft. The minimum separation between Boeing 747 and a following aircraft is 4, 5 and 6nm for Heavy, Large and Small, respectively. Again the last case of parameters results in lengths closer to the current separations, although there are some instances of lengths exceeding 10-12nm for higher altitudes. Similar conclusions can be made for Boeing 757, Boeing 737 and SF-340. All histograms can be found in Appendix B.

4.4 CSPR, Crosswind and Lateral Travel of the Wake

4.4.1 Wake Vortex Tracking Algorithm for Multiple Aircraft

Switzer et al (2010), has proposed an algorithm for tracking multiple vortex systems. This algorithm can play an important role for operational scenarios such as Closely Spaced Parallel Runways (CSPR). The described methodology uses the concept of a region-of-interest (ROI) for each of the involved aircraft and then looks at the individual ROI. A limitation of this methodology is the assumption that a single vortex of same rotation exists in the given area.

Switzer et al (2010) have shown two analyses—one for a single aircraft, and one for multiple aircraft. For both cases, a value of $0.01m^2/s^3$ EDR was selected to characterize the atmospheric boundary layer. Additionally, an in-ground effect (IGE) is considered for the analysis using Large Eddy simulation (LES) wake vortex dataset. A vortex is considered to be in ground effect when the altitude is less than one initial vortex separation, b_0(Hamilton 2000). When the wake is more than one wingspan above the ground, it is said to be in out of ground (OGE) (Burnham, Hallock et al. 2002). A schematic of the vortices generated by two parallel aircraft spaced about 400 ft (122m) is shown inFigure 36. For simplicity, both are assumed to be Boeing 747 aircraft.

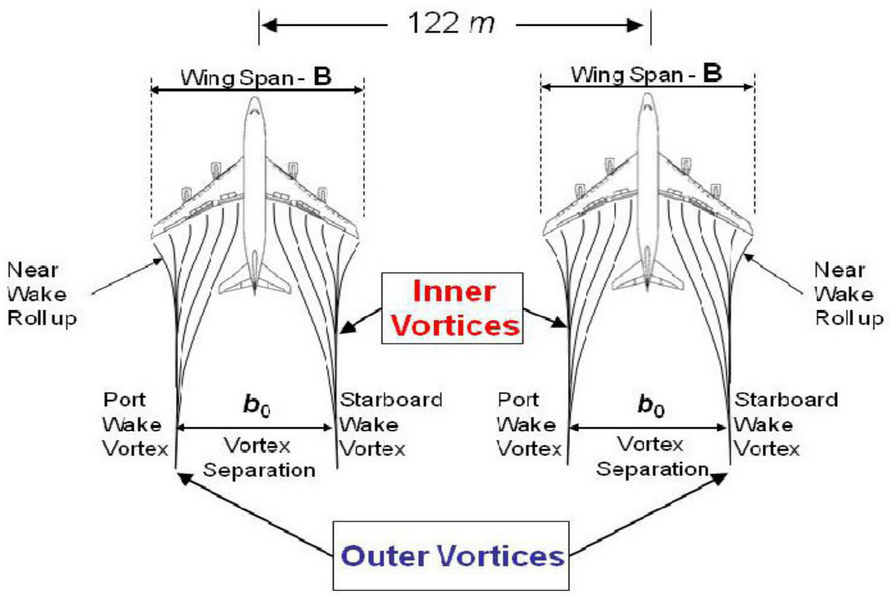


Figure 36 Schematic of wake vortices generated by parallel aircraft (Switzer and Proctor 2010). Used under Fair Use Guidelines

The ROI is represented as a box with a length of the initial vortex spacing denoted by b_0. Figure 37illustrates the ROI of the two closely spaced Boeings 747 aircraft with a wingspan of 63.66m. The initial vortex separation b_0 for B747 is estimated to be 50m. The starboard vortex of the left aircraft and the port vortex of the right aircraft have a different position because of the presence of the in-ground effect causing the vortex to rise.

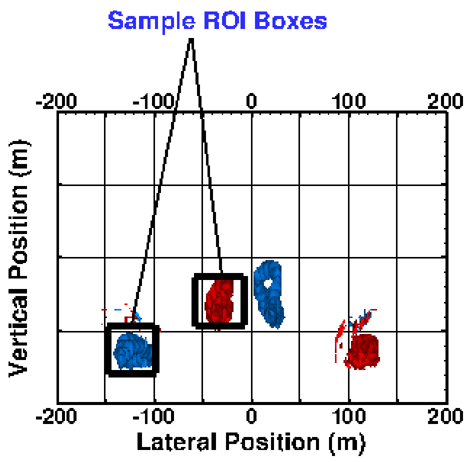


Figure 37 ROI boxes for two closely positioned B747's. Port (left) and starboard (right) vortices are shown in blue and red respectively (Switzer and Proctor 2010). Used under Fair Use Guidelines

The initial lateral position of the left and right vortices of a single Boeing 747 are positioned at 25m and 75 m, respectively; and the flight path at 50m. The positions of ROI as well as the non-ROI approach are shown in Figure 38for both lateral and vertical positions.

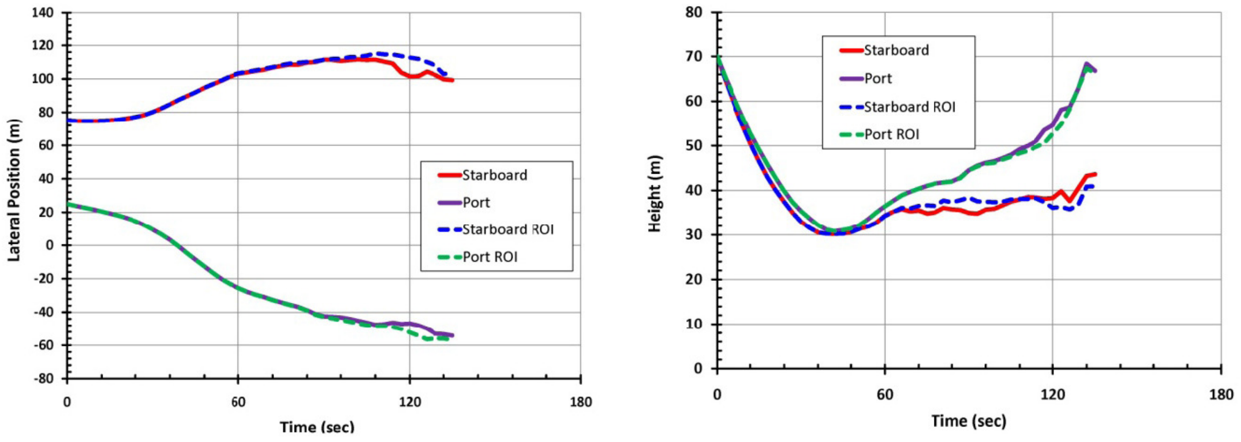


Figure 38 Lateral position and height versus time for a single aircraft (Switzer and Proctor, 2010). Used under Fair Use Guidelines

The analysis for 2 aircraft is similar to the single aircraft, and the ROI's are presented inFigure 39. IGE was considered as well under the assumption that the pair of B747 is flying in close ground proximity. Lateral position and height of the wakes are shown below.

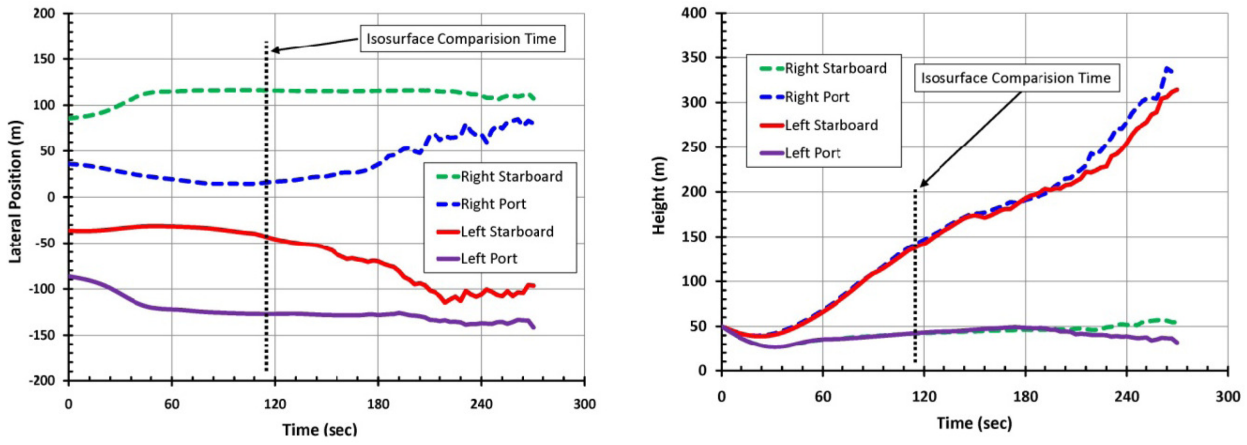


Figure 39 Lateral position and height versus time for two parallel aircraft (Switzer and Proctor, 2010). Used under Fair Use Guidelines

4.4.2 CSPR and Wind Effects

Los Angeles International Airport (LAX) is an example where we have two sets of closely spaced runways. Currently, independent operations cannot be performed on runways spaced below 2,500ft. Some analysis was performed for runways 25R and 25L at LAX. The runways are spaced by about 800ft. Using a MATLAB script written by Dr. Trani (ATSL,2010), flights from

08/13/2008 were separated for individual runways. The script was modified so flights landing on 25L and 25R separated by 30 seconds were identified as parallel operations. Times were then varied between close parallel approach pairs to see if the wake will affect the following aircraft if it is closer to the wake generator.

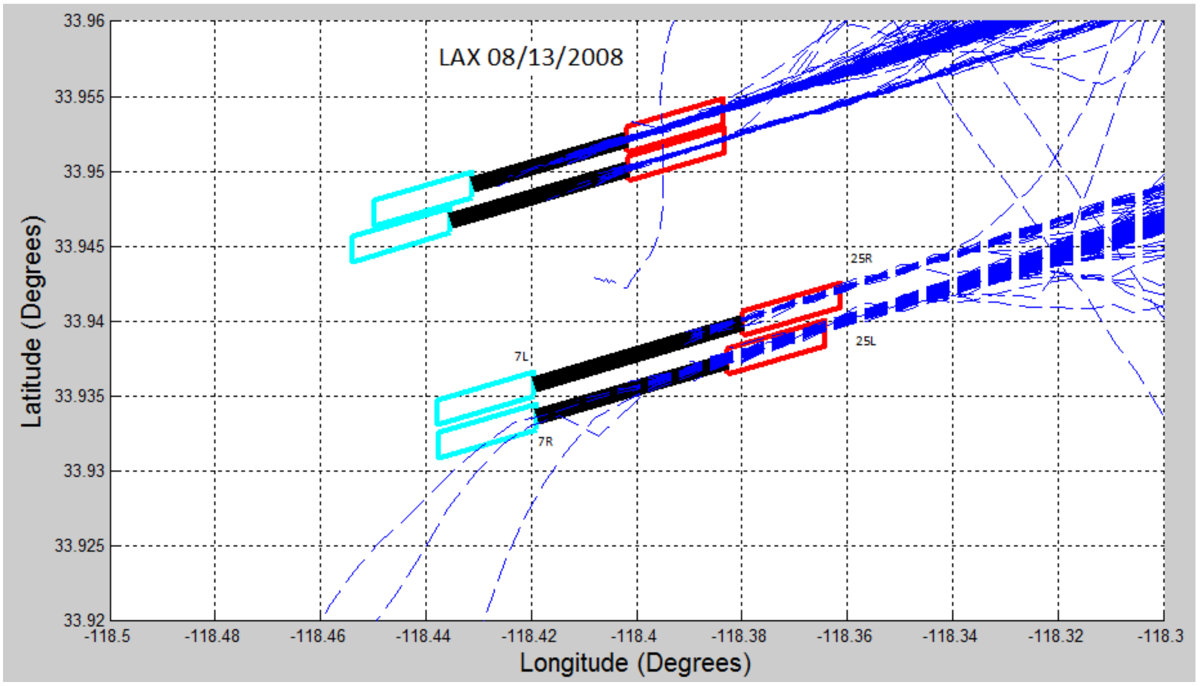


Figure 40 Trajectories at LAX runways for 08/13/2008

Based on the analysis described in Section 4.4.1, however, we can conclude that in the case of LAX the second aircraft will not experience wake turbulence from the wake generator, because of the geometry of the wake and the spacing of the runways, unless there is a cross wind. The wake envelope of an aircraft grows laterally to a width of three times the wingspan. Therefore an aircraft with an effective wingspan of 50m will have a final envelope length of 150m, which is less than the separation between the runways. Anecdotal evidence exist showing that wind is a factor in many wake vortex encounters at low altitudes. One example is the wake encounter described in Section 1.3, where a SAAB 340 encountered the wake from an Airbus A380 in a 35-knot crosswind condition. Additionally, some of the reports from the ASRS call-back database indicate cross wind during the wake encounter. The following is an example narrative from a MD-88 that encountered wake vortex from a Boeing 747 landing on parallel runways 24R and 24L at LAX.

"WE WERE CLRED VISUAL APCH TO RWY 24R AT LAX. F0 WAS PLT FLYING. ATC HAD CLRED A B747 TO LAND ON RWY 24L. WE HAD A VISUAL ON THE ARPT AND WITH THE B747. ATC DID

ISSUE A CAUTION ABOUT WAKE TURBULENCE FROM THE PRECEDING B747. THE PF DID A GOOD JOB BEING AWARE OF THE WAKE TURB AND FLEW SLIGHTLY HIGH TO STAY ABOVE AND SLOWED THE ACFT TO GAIN ADDED DISTANCE. EVERYTHING WAS GREAT UNTIL 100 FT AGL. WE ENCOUNTERED VERY RAPID ROLLS RIGHT THEN ABRUPTLY L. WE INITIATED A GAR TO ESCAPE THE WAKE TURB AND USED MAX PWR DOING SO. WE BOTH HAVE A GREAT AWARENESS OF THE CONSEQUENCES OF THIS UNSEEN HAZARD. WE DID ALL WE COULD TO AVOID IT. ***THE PREVAILING WIND MUST HAVE BLOWN THE B747 WAKE DIRECTLY ON OUR LNDG RWY****. I INTEND TO GET A FULL 5 MILES BEHIND THIS TYPE HEAVY EVEN IF THE ACFT IS LNDG ON A PARALLEL. "*(NASA 2011)

An additional narrative from another Boeing 737, that experienced wake turbulence from a leading Boeing 747 landing on the same parallel runways as in the previous example is presented below.

"WE WERE GIVEN A VISUAL CLRNC TO RWY 24R, WITH A B747 4-5 MI AHEAD FOR RWY 24L. ***THE WINDS WERE ABOUT 240 DEGS AT 11 KTS, WHICH HELPED TO BLOW THE WAKE INTO OUR PATH.*** *WE ROLLED R THEN L, RECOVERED, AND CLBED TO 2000 FT FOR VECTORS FOR ANOTHER APCH. LESSONS LEARNED: I WILL NOT FOLLOW A B747 TO A PARALLEL RWY IN ANY CONDITIONS, UNLESS I AM WELL ABOVE HIS PATH AND CAN CONFIRM HIS ALT PATH. JUST BECAUSE IT IS LEGAL SEPARATION, DOES NOT MAKE IT RIGHT. I'LL USE BETTER JUDGEMENT AND EXTEND ANOTHER 3 MI AT LEAST REQUIRED SEPARATION BTWN LARGE AND SMALLER ACFT MUST BE GREATER IF THE LARGER ACFT IS AHEAD AND GOING TO PARALLEL, BUT CLOSE RWYS."*(NASA 2011)

In both cases, as well as in the example involving the SAAB described in Section 1.3, wind played a significant role in the encounter. A brief analysis of wake travel due to wind based on the literature review is presented in the following section. However, future research should try to identify and analyze the effect of crosswind to the wake lateral behavior in greater detail.

4.4.3 Wake Travel Time calculations

To compute the time required for the vortex to travel between parallel runways, a methodology given by Burnham et al (2002) was used. A simple formula relating time, velocity and distance can be used to calculate the travel time. Suggested values for an effective crosswind were 6, 10, and 20 knots, and the formula to compute the travel time is as follows:

$t=D/v$, Where:

D=distance between parallel runways (feet)

v= effective crosswind (used 6, 10 and 20 knots)

t= travel time for a vortex to reach parallel runway (s)

This formula can be applied to LAX, where the separation between runways 25R and 25L is 800ft (244m), and between runways 24R and 25L is about 700ft (213m). The results are presented in Table 17:

Table 17 Wake travel times between parallel runways for different wind speeds

Runways 24R and 24L	
v (knots)	t(sec)
6	69
10	41
20	21

Runways 25R and 25L	
v (knots)	t(sec)
6	79
10	47
20	24

The travel time, however can be reduced by four important parameters (Burnham et al 2002), which are as follows:

1. The initial position of the wake is closer than D. It should be reduced by b'= $b_0/2$.
2. If the vortices are near the ground, the in-ground effect can increase the lateral motion of the wake by a value of v_i, or as much as 4knots.
3. The following aircraft is required to keep a safety distance d from the leader, which can be assumed to be 100 ft (30m).
4. The lateral navigation errors, d_n, from both aircraft have to also be added in the equation. This value can be assumed as 50ft (15m).

The result from adding all these corrections is the following formula:

$$t = \frac{D - \frac{b\prime}{2} - d - 2d_n}{v + v_i}$$

Where,
D= runway separation (m)
b'= effective wingspan (m)
d= distance between leading and following aircraft (m)
d_n= lateral navigation error (m)
v= effective crosswind (knots)
v_i= in-ground effect (knots)

Assuming, b'= 150ft= 46m, we obtain the following results for when in-ground effect is considered and when it is not considered (see Table 18).

Table 18 Adjusted wake travel times between parallel runways for different wind speeds

Runways 24R and 24L		
v (knots)	t(sec)	t(sec)
6	25	42
10	18	25
20	11	13
	IGE	OGE

Runways 25R and 25L		
v (knots)	t(sec)	t(sec)
6	31	52
10	22	31
20	13	16
	IGE	OGE

Therefore, for runways 25L and 25R, the required time for the wake to travel from one runway to the other is about 11 seconds when the effective crosswind is 11 knots and in-ground effect is considered.

4.5 In-trail Separation Reduction

It is shown that 0.5nm decrease in in-trail separation can increase capacity by 4% for a single runway (Swol, 2009). A simple calculation was done for two example flight pairs, one arrival and one departure, which were randomly selected. The first flight pair included two Boeing 737-800's departing LAX. The second pair included a Boeing 777-200 followed by an Embraer 145 landing at LAX. The minimum speed was computed for the leading aircraft of each potential wake pair. Using the basic kinematic formulas the time required to travel 0.5nm was computed. Then, the WEM run for the selected day was repeated using the new times for the following aircraft, reduced by the computed delta t to reduce the spacing between successive aircraft. The count of the instances when the following aircraft had pierced the envelope of the leading one were compared between the baseline (original run) and the run with reduced separation and are presented in Table 19.

Table 19 Results from base separation versus reduced separation for two flights

Leading Aircraft	Following Aircraft	Type of operation	Number of time piercing occurs	
			Original Separation	Reduced Separation
B737-800	B737-800	Departure	1	4
B777-200	E145	Arrival	2	5

The conclusion, based on the presented example is that we observe more potential encounters in the reduced separation case than in the original case. For the departure example, we observed a single instance of a potential encounter, while in the case of reduced separation we observed four. Similarly, for the arrival case, there were two instances when the Embraer pierced the envelope produced by the Boeing 777, and the number went up to 5 when separation was decreased. Therefore, based on these two examples we can conclude that in these two cases the likelihood of wake encounter increased with the decrease in spacing.

5. CONCLUSION

The most important objective of this thesis was to improve the Wake Encounter Model (WEM) developed at the Air Transportation Lab at Virginia Tech. One of the enhancements included making the wake envelope position relative to the travelled trajectory, rather than relative to the instantaneous position of the aircraft.Other enhancements included reduction in execution time and development of tools for post-model analysis. To verify the Enhanced Wake Encounter Model, the same model runs for the same dates as the original model were repeated, and results compared. Because the original model was computationally inefficient and adding the extra computation to correct the envelope shape and location added additional execution time, another goal was to revise and optimize the MATLAB script used for the model. Another objectiveincludes the development of visualization tools that could be easily used to inspect the instances determined by the WEM as potential encounters. In addition, some statistical analysis related to the distribution of the wake envelope length based on different values of EDR and BVF was shown in this thesis. The purpose of this analysis was to determine the best parameters of the model for future use. Finally, some analysis was presented related to close parallel operations showing the time required for a wake vortex to travel from one runways to another using a crosswind model found in the literature. The overall results of the model were also compared qualitatively to the ASRS report database of real-life reported wake encounters.

5.1 WEM Improvements and Results

One of the many benefits of the EWEM compared to earlier research efforts and models is the ability to test for potential wake turbulence encounters across the entire terminal airspace. Additionally, many of the values used in the model can be parametrically varied allowing for user-defined operations. EWEM is able to capture more realistically the interactions between airplanes and the way wake travels when wind is not present compared to the earlier WEM.

The results from the baseline runs from both, the original and enhanced models, indicated low numbers of potential encounters, which could translate in conclusion that wake encounters in current flight operations are rare. Most of the potential wake encounters occurred at low values of circulation threshold, EDR and BVF values which correspond to turbulence level close to the atmospheric turbulence. As the values of the atmospheric parameters increased the number of potential wake encounters was eliminated, which is consistent with the condition we have in the NAS.

The potential encounters determined by the original wake encounter model were not confirmed after the enhancement, because they were all piercing the wake envelope in areas extending past the travelled path of the leading aircraft. The enhanced model however determined new instances of potential encounters, mostly during arrivals, and mostly for in-trail operations. These results are consistent with the wake turbulence reports from pilots for operation types (i.e. arrivals), and airport locations (i.e. LAX having the highest number of reported encounters).

A capacity gain can likely be achieved by reducing the separations between aircrafts, which is one of the objectives of NextGen. Research has shown a capacity increase at a single runway by about 4%, for a 0.5nm reduction in separation down to a minimum of 2.5nm. This translates into delay reduction of 20-25% for an airport at 90% capacity (Swol, 2009). Additionally, airports with parallel runways can also achieve some capacity gains by implementing procedures such as closely spaced parallel approaches. More research however has to be conducted to determine the impact of close parallel separation reductions without increasing the likelihood of wake turbulence encounters.

5.2 Computational Time Improvements

MATLAB was used as a computational tool for the original WEM developed by Swol (2009). As part of the improvement of the model, the original code was re-written and further optimized. Originally, the model was extracting information about position and parameters multiple times for the same aircraft. After the improvement, the execution time was only 1/8 of the original time value even though many more operations were added.

5.3 WEM Extension Tools Development

Additional tools were developed as part of the model enhancements. These tools include:a) animation tool, b) plot of the close point of approach, and c) plot of the interaction of the following aircraft with the wake envelope of the wake generator. These model enhancements provide important information about the interaction between the aircraft pair and also help to verify the results predicted by the model.

6. RECOMMENDATION FOR FUTURE RESEARCH

1. Wind Effects

The current version of the Enhanced Wake Encounter Modeldoes not account for the wind effect. Therefore, future research should add the effects of cross wind and tail wind to the EWEM. Wind can have positive as well as negative influence on the vortices. Sometimes strong winds can cause the wake vortices to dissipate much quicker, but sometimes crosswinds can blow the vortices of one aircraft into the path of another. The second scenario is important when performing approaches on close parallel runways. Some examples of wake encounters due to crosswind were presented in Chapter 4.4.

2. In-ground Effect (IGE)

Similarly, in-ground effects are very important since it has been determined that interactions with the ground cause the vortices to roll and rise. Therefore, the WEM should include the IGE when the aircraft is within one effective wingspan distance from the ground.

Dr. Fred Proctor, the developer of the TDAWP model used in the Wake Encounter Model described here, has recently enhanced the TDAWP model to account for both wind and IGE. Using this updated model, we can solve two of our recommendations.Other ways of accounting for these two phenomena can be incorporated as an alternative.

3. In-trail Separations

The main goal of NextGen is to increase capacity without compromising safety; therefore future research should be geared toward finding the optimal spacing that will not increase the likelihood of potential wake encounters. A simple calculation was presented earlier for two example flight pairs, one arrival and one departure, which were randomly selected. The count of the instances when the following aircraft had pierced the envelope of the leading one were compared between the today's separation run and the run with 0.5 nm reduced separation. The conclusion is that we observe more potential encounters in the reduced separation case than in the original case. However, future research should look at more data and should calculate the time reduction more precisely in order to verify the conclusion. Additionally, future research should look at different distances as well as incorporate weather in the decision-making process, which is another NextGen objective.

4. Function used for wake piercing improvement

Currently, the model uses the MATLAB function “inhull” to check if an aircraft pierces the wake envelope produced by another aircraft. A limitation of the function is that it works for convex shapes. The wake envelope shape however is non-convex. Under some conditions, the model determines a potential encounter at an instance when the aircraft is near the envelope, but it does not pierce it. Currently, these instances are eliminated during the phase when the model plots the potential encounters. Further research should seek an improved function that will work with non-convex shapes.

5. More computational time optimization

The model can be further optimized by using the parallel computing capabilities of MATLAB and running it on multiple processors. Currently, the code determines the unique leaders and loops over each leader to find the following aircraft and performs the computations. Since the calculations within the loop are independent, this could be easily converted to a “parfor” loop using multiple CPU’s to perform the iterations. Some of the variables used in the code, however have to be converted to sliced variables in order for the parallel computing to work. Additionally, some variables are overwritten at each loop so they have to be changed as well. The concept or parallel computing is simple, once these issues with the variables are solved. Once the model is revised, using a computer with three CPUs for example, will allow the user to check three leaders for potential encounters simultaneously reducing the execution time by approximately a third.

REFERENCES

1. Australian Transport Safety Bureau , A. (2008). Wake Turbulence event Sydney Airport, NSW 3 November 2008. ATSB Transport Safety Report.
2. Brown, B. G. M., Jennifer L.; Sharman, Robert ; Vogt, Jamie & Henderson, Judy (2000). USE OF AUTOMATED OBSERVATIONS FOR VERIFICATION OF TURBULENCE FORECASTS Report to the FAA.
3. Burnham, D. C., J. N. Hallock, et al. (2002). "Wake turbulence limits on paired approaches to parallel runways." Journal of Aircraft**39**(4): 630-637.
4. Burnham, D. C., Hallock, J. N. and Greene, G. C., (2001). "INCREASING AIRPORT CAPACITY WITH MODIFIED IFR APPROACH PROCEDURES FOR CLOSE-SPACED PARALLEL RUNWAYS." Air Traffic Control Quarterly**9**(1): 45-58.
5. Burnham, D. C., Hallock, J.N. (1997). WAKE VORTEX SEPARATION STANDARDS: ANALYSIS METHODS. D. V. N. T. S. Center. Cambridge, MA, DOT Volpe National Transportation Systems Center.
6. Corjon, A. and T. Poinsot (1996). "Vortex model to define safe aircraft separation distances." Journal of Aircraft**33**(Compendex): 547-553.
7. Delta Airlines (2009). Eddy Dissipation Rate (EDR) Automatic Turbulence Reports. NBAA Friends and Partners in Aviation Weather Forum 2009. Orlando, FL**:** 12.
8. den Braven, W. S., John (2003) "Concept and Operation of the Performance Data Analysis and Reporting System (PDARS)."
9. FAA, N. I. a. I. O. (2009) "FAA's NextGen Implementation Plan 2009."
10. FAA, U. S. D. o. T. (2002). Airport Capacity Benchmarking Report 2001.
11. FAA, U. S. D. o. T. (2010). Aeronautical Information Manual. Official Guide to Basic Flight Information and ATC Procedures, FAA.
12. Greene, G. C. (1986). "APPROXIMATE MODEL OF VORTEX DECAY IN THE ATMOSPHERE." Journal of Aircraft**23**(Compendex): 566-573.
13. Hamilton, D. W. P., Fred H. (2000). WAKE VORTEX TRANSPORT IN PROXIMITY TO THE GROUND 19th Digital Avionics Systems Conference. Philadelphia, Pennsylvania.
14. Holzapfel, F. (2003). "Probabilistic two-phase wake vortex decay and transport model." Journal of Aircraft**40**(2): 323-331.
15. Holzapfel, F. (2006). "Probabilistic two-phase aircraft wake-vortex model: Further development and assessment." Journal of Aircraft**43**(3): 700-708.
16. Holzapfel, F., M. Frech, et al. (2009). "Aircraft wake vortex scenarios simulation package - WakeScene." Aerospace Science and Technology**13**(Compendex): 1-11.
17. Holzapfel, F., J. Kladetzke, et al. (2009). "Aircraft wake vortex scenarios simulation package for takeoff and departure." Journal of Aircraft**46**(Compendex): 713-717.
18. Holzäpfel, F., Kladetzke,J., Amelsberg,S., Lenz,H., Schwarz,C., De Visscher, I. (2008). AIRCRAFT WAKE VORTEX SCENARIOS SIMULATION FOR TAKE-OFF AND DEPARTURE. 26th INTERNATIONAL CONGRESS OF THE AERONAUTICAL SCIENCES. Anchorage, Alaska, AIAA.
19. Janic, M. (2008). "Modelling the capacity of closely-spaced parallel runways using innovative approach procedures." Transportation Research Part C: Emerging Technologies**16**(6): 704-730.
20. Kopp, F. (1994). "Doppler Lidar Investigation of Wake Vortex Transport Between Closely Space parallel Runways." AIAA **32**: 805-810.

21. Lang, S., Domino, D. A., Mundra, A. and Bodoh, C., (2004). ATC FEASIBILITY OF POTENTIAL NEAR-TERM WAKE TURBULENCE PROCEDURES. 23rd Digital Avionics Systems Conference. **1:** 5.B.4-1--5.B.4-15.
22. NASA (2007) "PDARS - Performance Data Analysis and Reporting System."
23. NASA. (2011). "ASRS Database Online." Retrieved January 13, 2011, from http://asrs.arc.nasa.gov/search/database.html.
24. National Center for Atmospheric Research, N. (2011). "In Situ Turbulence." Retrieved January 24, 2011, from http://www.ral.ucar.edu/projects/in_situ/.
25. National Transportation Safety Board, N. (1999). Aircraft Accident Report: Uncontrolled Descent and Collision with Terrain - USAir Flight 427. Washington, DC. **NTSB/AAR-99/01**.
26. National Transportation Safety Board, N. (2004). Aircraft Accident Report: In-Flight Separation of Vertical Stabilizer - American Airlines Flight 587. NTSB/AAR-04/04. Washington, DC.
27. NOAA's National Weather Service. (2010). "Aviation Weather Center: Aviation Digital Data Service (ADDS)." Retrieved January 24, 2011, from http://aviationweather.gov/adds/pireps/.
28. Oconnor, C. J. and D. K. Rutishauser (2001). Enhanced Airport Capacity Through Safe, Dynamic Reductions in Aircraft Separation: NASA's Aircraft VOrtex Spacing System (AVOSS). United States**:** 15p.
29. Proctor, F. H. (2004). Numerical Simulation of Wake Vortices Measured During the Idaho Falls and Memphis Field Programs. United States**:** 24p.
30. Robins, R. E. a. D., D. P., (2002). NWRA AVOSS WAKE VORTEX PREDICTION ALGORITHM VERSION 3.1.1. Bellevue, WA, Northwest Research Associates.
31. Rossow, V. J. (1996). "Wake-vortex separation distances when flight-path corridors are constrained." Journal of Aircraft**33**(3): 539-546.
32. Sarpkaya, T. (2000). "New model for vortex decay in the atmosphere." Journal of Aircraft**37**(1): 53-61.
33. Sarpkaya, T., R. E. Robins, et al. (2001). "Wake-vortex eddy-dissipation model predictions compared with observations." Journal of Aircraft**38**(Compendex): 687-692.
34. Schilling, V. K. (1992). "Motion and Decay of Trailing Vortices Within the Atmospheric Surface Layer." Beitr. Phys. Atmos**65**: 157-169.
35. Shortle, J. F. and B. G. Jeddi (2007). "Using multilateration data in probabilistic analysis of wake vortex hazards for landing aircraft." Transportation Research Record(2007): 90-96.
36. Shortle, J. F. a. J., B. G., (2007). PROBABILISTIC ANALYSIS OF WAKE VORTEX HAZARDS FOR LANDING AIRCRAFT USING MULTILATERATION DATA. TRB Annual Meeting. Washington, DC.
37. Switzer, G. F. P., Fred H. ; Ahmad, Nashat N.;LimonDuparcmeur, Fanny M. (2010). An Improved Wake Vortex Tracking Algorithm for Multiple Aircraft. AIAA Atmospheric Flight Mechanics Conference. Toronto, Canada, NASA Technical Reports Server (NTRS): 9.
38. Swol, D. (2009). Simulation-Based Analysis of Wake Turbulence Encounters in Current Flight Operations. Civil and Environmental Engineering. Blacksburg, VA, Virginia Polytechnic Institute and State University. Master of Science: 92.
39. Takacs, A. H., Lacey ; Hueftle, Robert; Brown, Barbara ; Holmes, Anne (2004). Using in situ Eddy Dissipation Rate (EDR) Observations for Turbulence Forecast Verification. Friends/Partners in Aviation Weather. NTSB Conference Center – Washington, D.C.

40. Taube, E. (2010). ASRS Reports on Wake Vortex Encounters. Global Wake Vortex Conference II. San Diego, CA.
41. Virginia Tech Air Transportation Systems Laboratory (2008). http://128.173.204.63. Accessed on January 24, 2011.

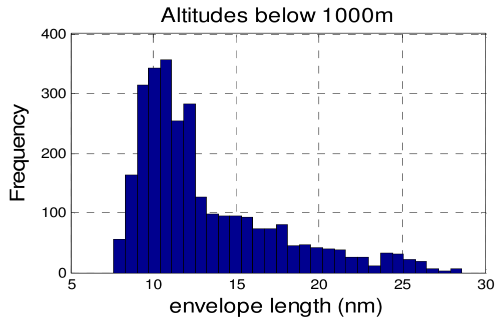
APPENDIX A: ACRONYMS

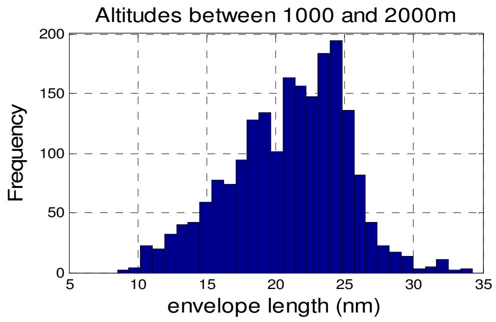
Appendix A includes a list of frequently used acronyms in this document.

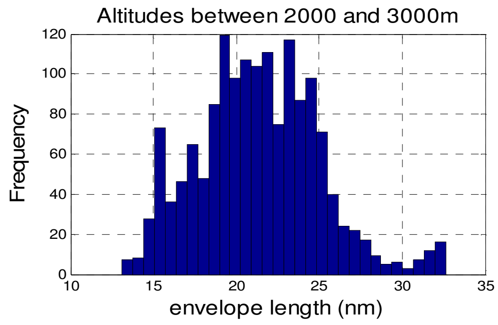
ATC – Air Traffic Control
ATL – Atlanta Hartsfield-Jackson International Airport
AVOSS – Aircraft Vortex Spacing System
BVF – Brunt-Vaisala Frequency
CONOPS – Concepts of Operation
CT—Circulation Threshold
CSPR—Closely Spaced Parallel Runways
D2P—Deterministic Two Phase Model
EDR – Eddy Dissipation Rate
EWR – Newark Liberty International Airport
FAA – Federal Aviation Administration
GMU – George Mason University
ICAO – International Civil Aviation Organization
IMC – Instrument Meteorological Conditions
ISA – International Standard Atmosphere
JFK – John F. Kennedy International Airport
JPDO – Joint Planning and Development Office
LAX – Los Angeles International Airport
LES – Large Eddy Simulation
LIDAR – Light Detection and Ranging
LGA – La Guardia Airport
MEM – Memphis International Airport
MMC – Marginal Meteorological Conditions
NAS – National Airspace System
NASA – National Aeronautics and Space Administration
NextGen—Next Generation Air Transportation Systems
nm—Nautical Mile
PDARS – Performance Data and Analysis Reporting System
P2P—Probabilistic Two Phase Model
SPR—Spaced Parallel Runways
TASS – Terminal Area Simulation System
TDAWP – TASS Driven Algorithm for Wake Prediction
TEB – Teterboro Airport
TRACON – Terminal Radar Approach Control
VMC – Visual Meteorological Conditions
WEM – Wake Encounter Model

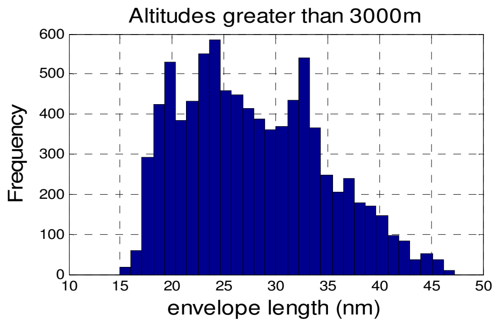
APPENDIX B: HISTOGRAMS

Airbus 380: **A380__20080319_0001_0005_125**

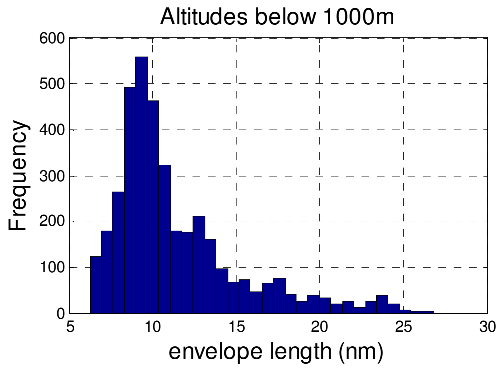


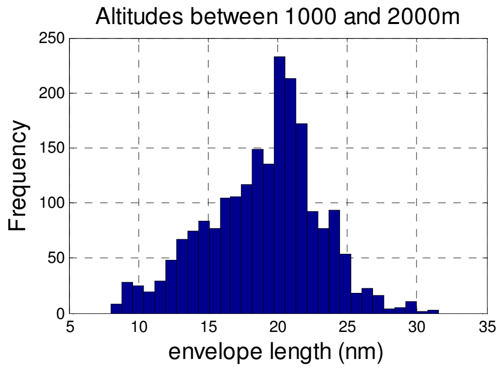


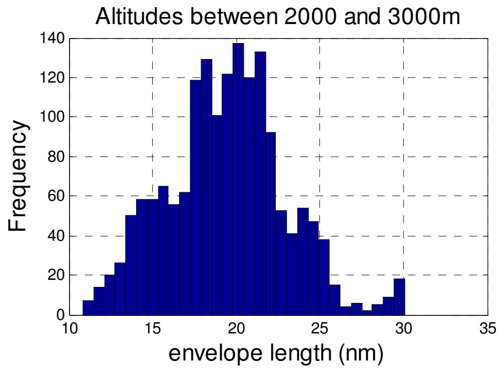


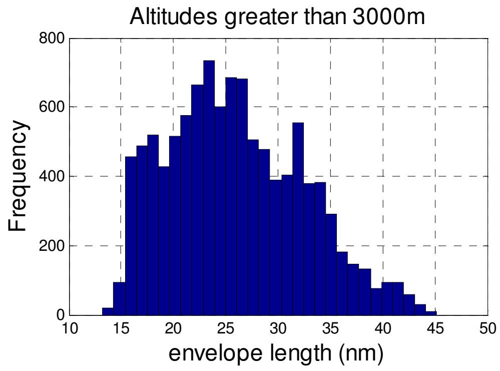


A380__20080319_0001_0005_175

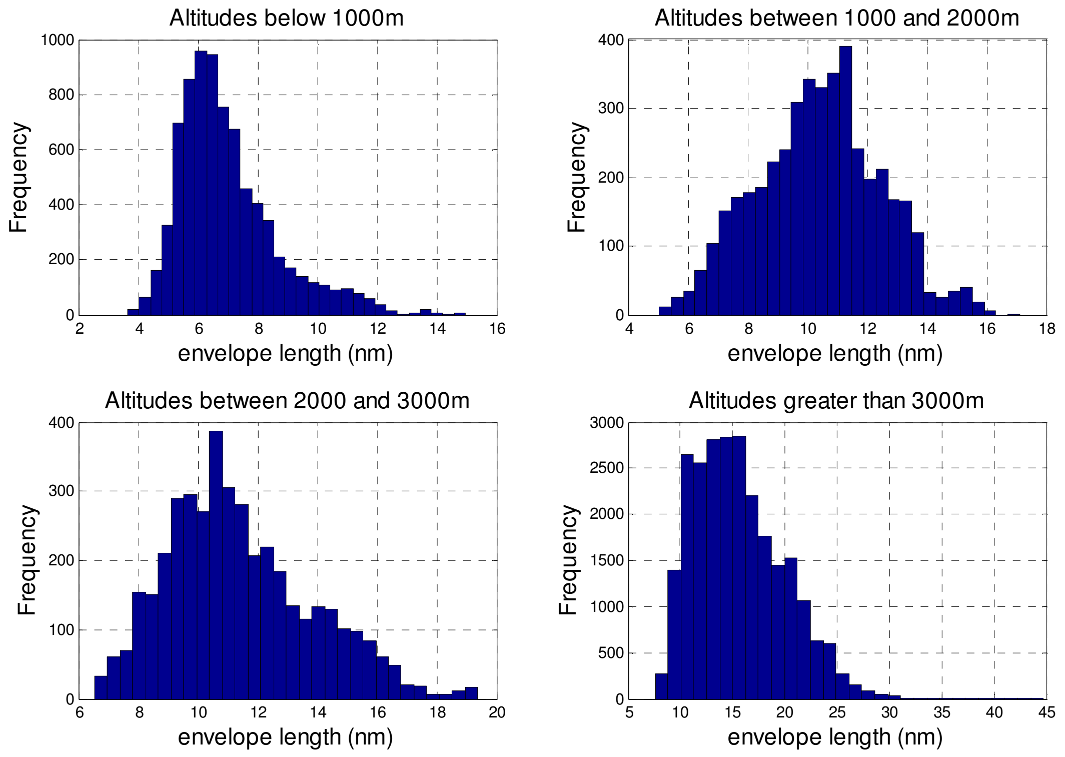




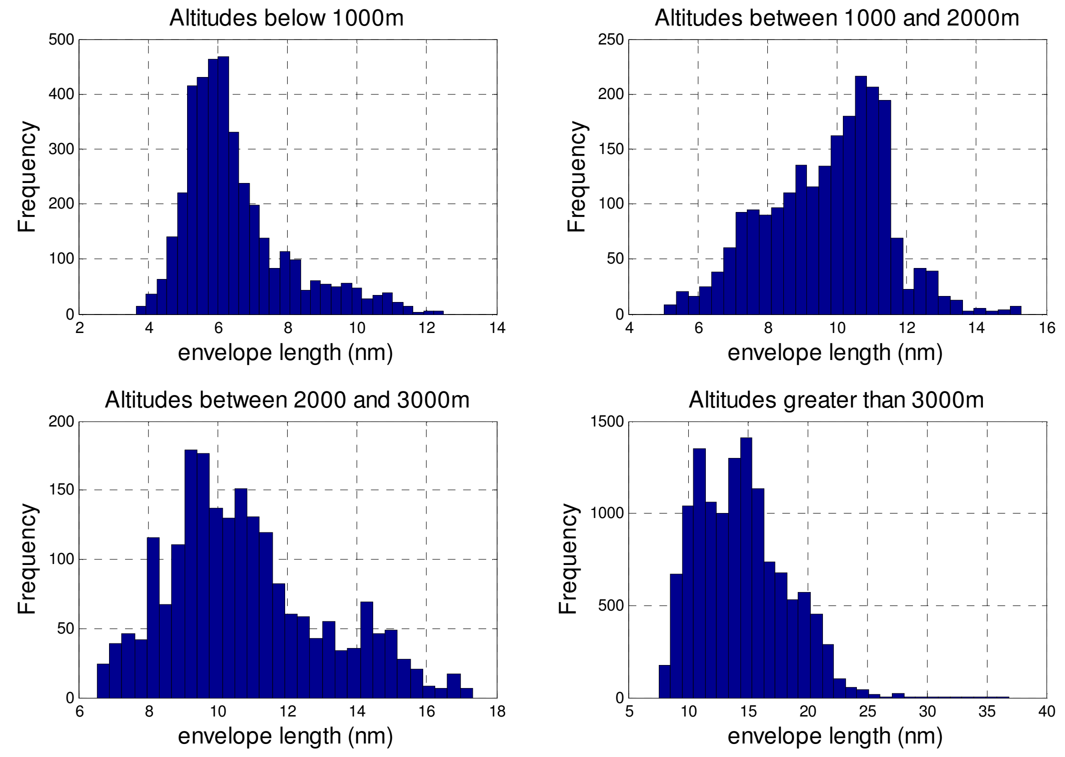




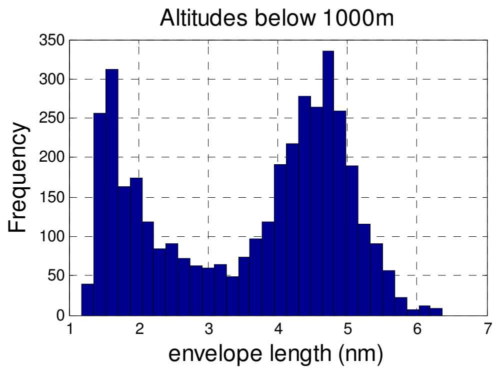
A380__20080319_002_01_125

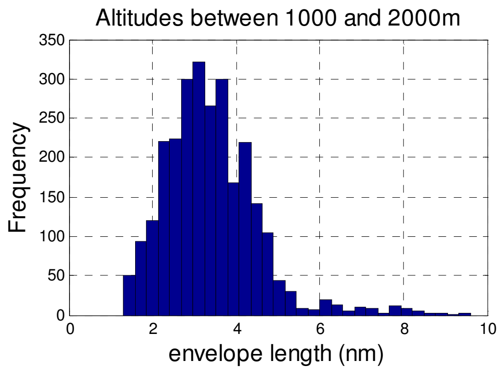


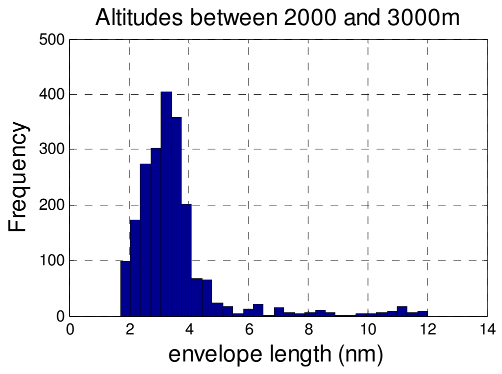
A380__20080319_002_01_175

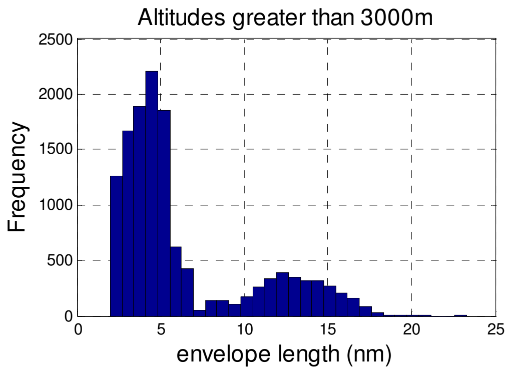


A380__20080319_004_02_125

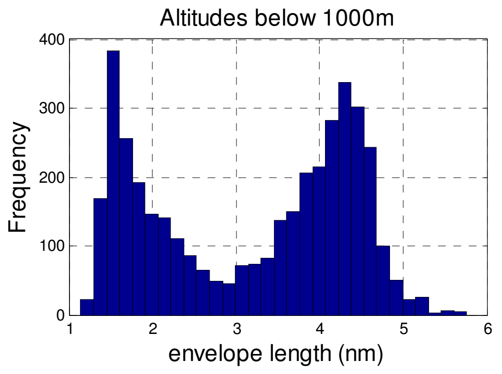


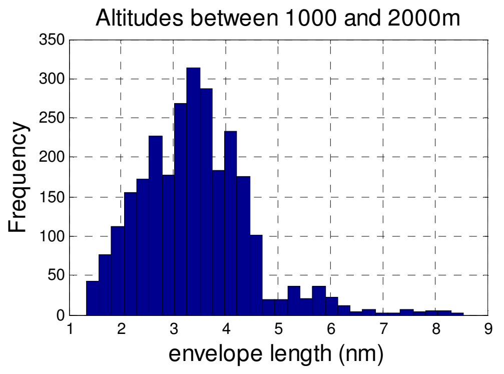


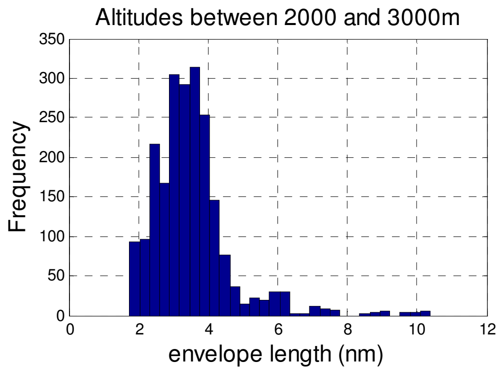


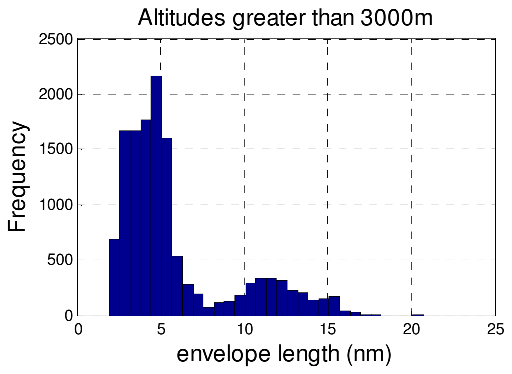


A380__20080319_004_02_175.txt

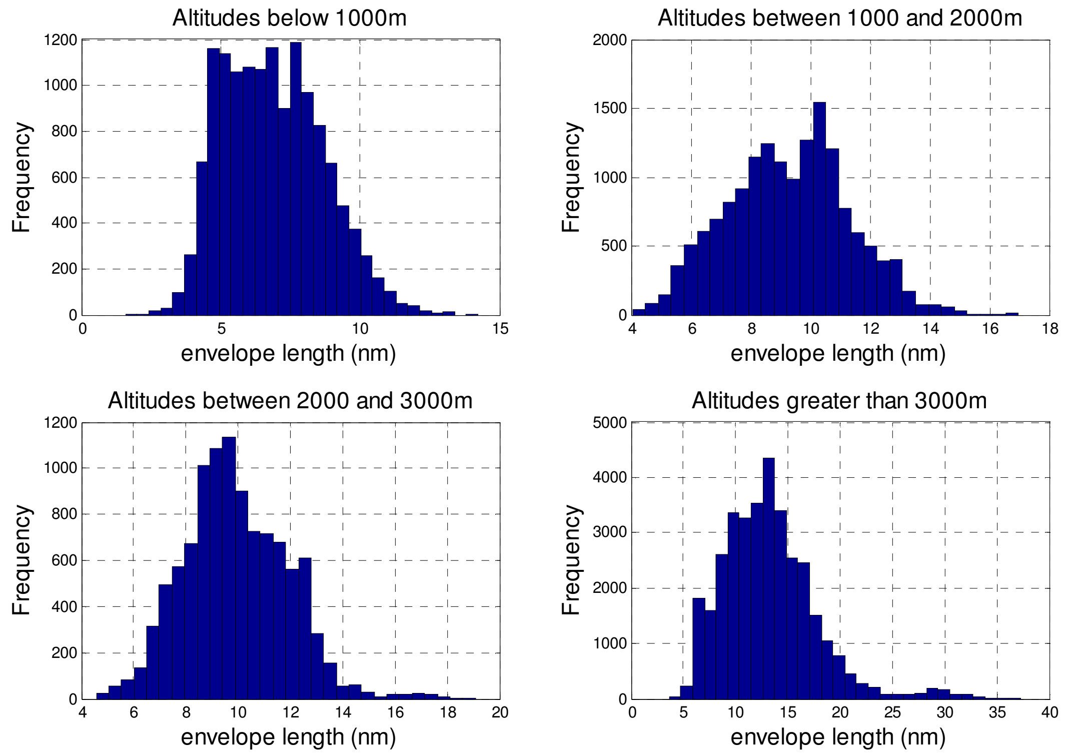




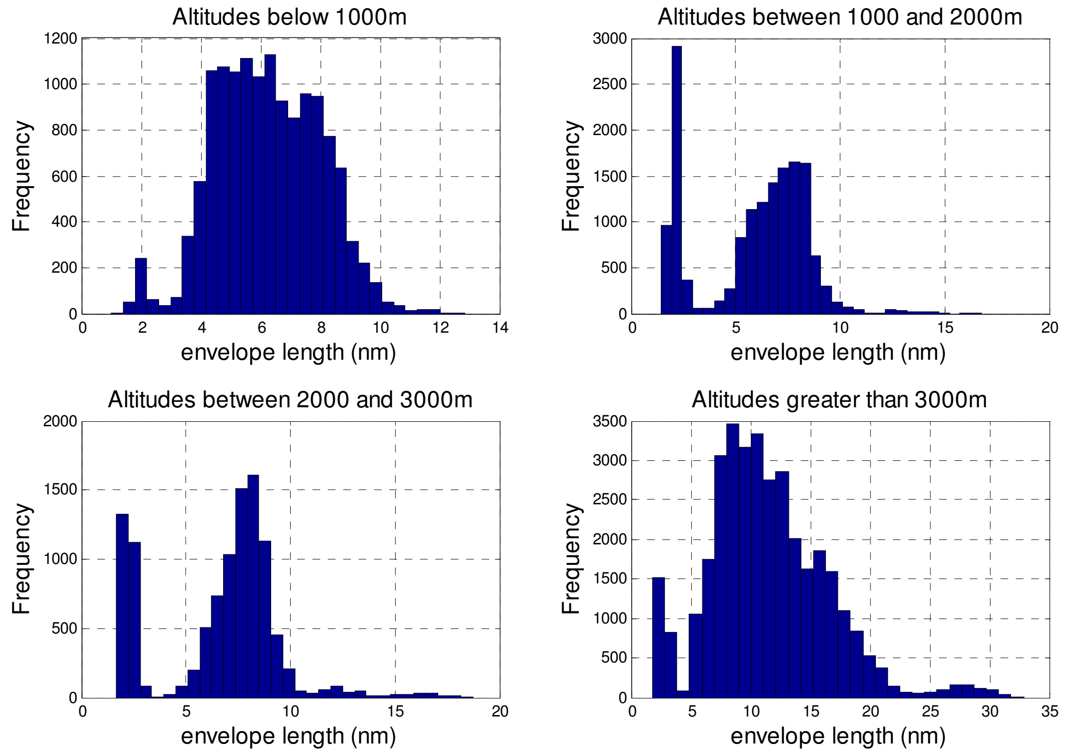




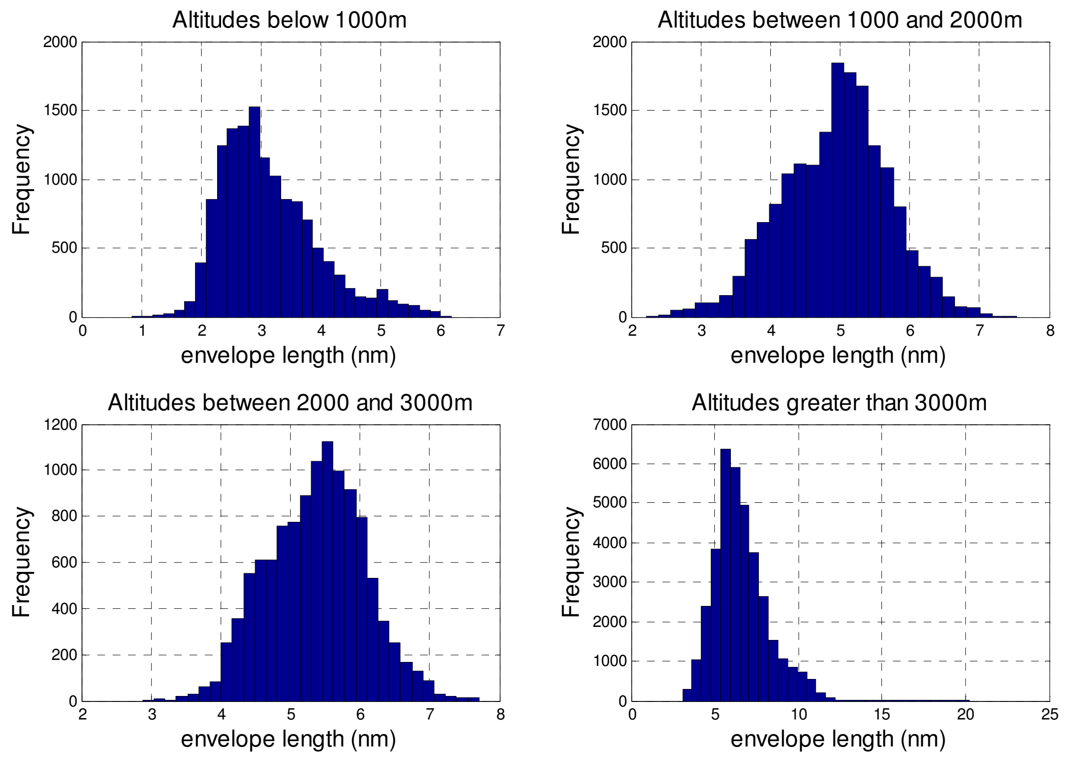
Boeing 757: **B757_20080319_0001_0005_125**



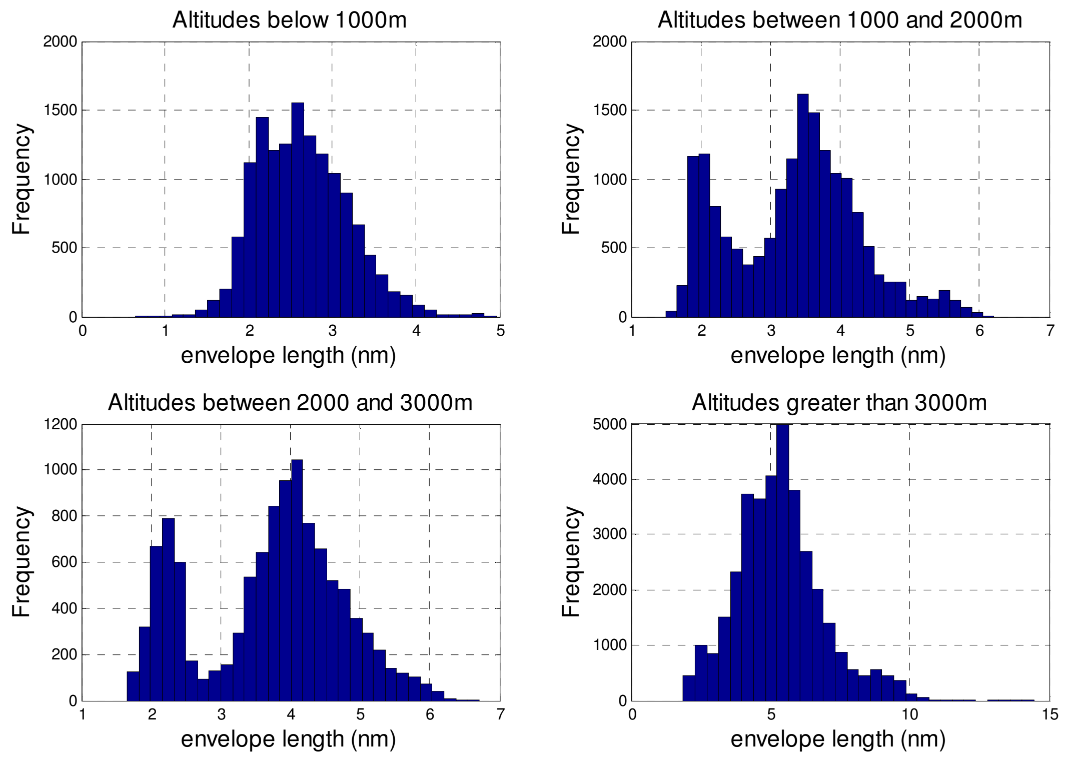
B757_20080319_0001_0005_175



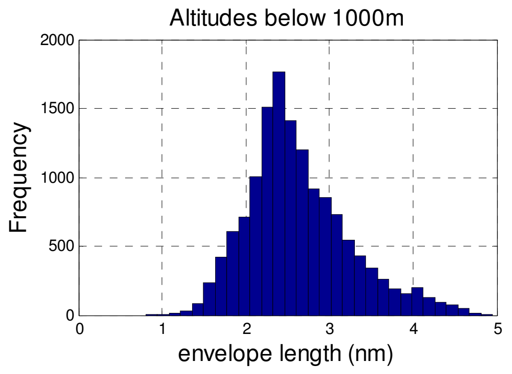
B757_20080319_002_01_125

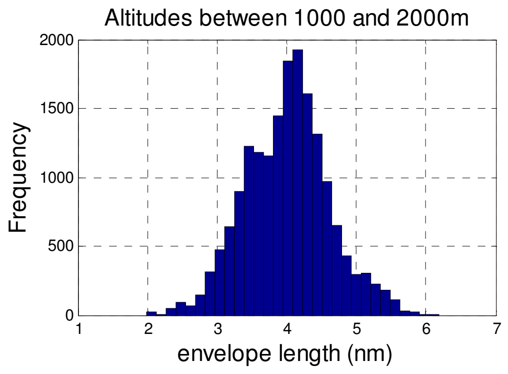


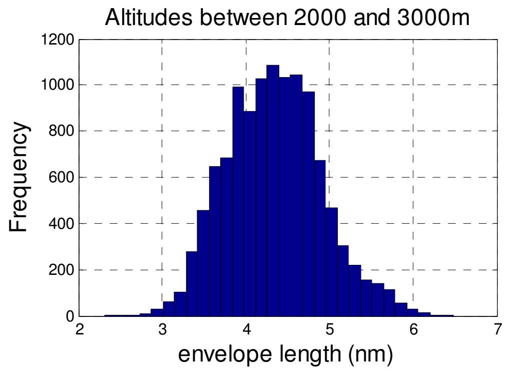
B757_20080319_002_01_175

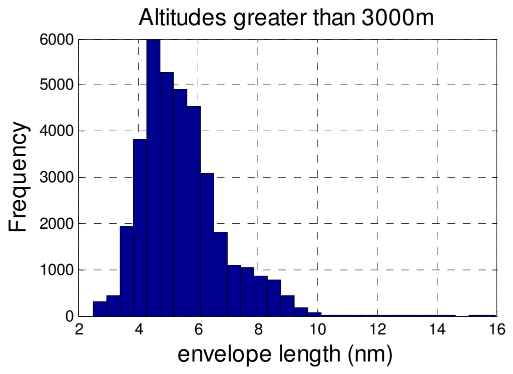


B757_20080319_004_02_125

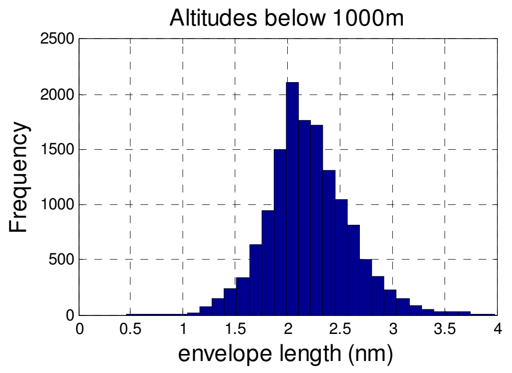


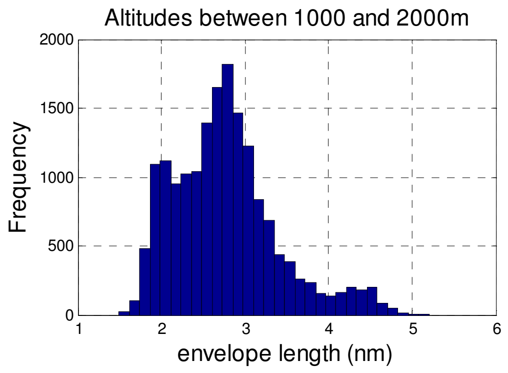


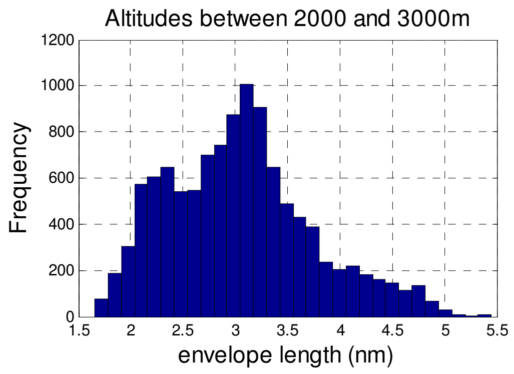


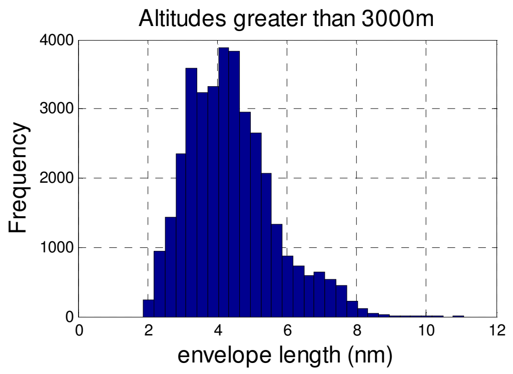


B757_20080319_004_02_175

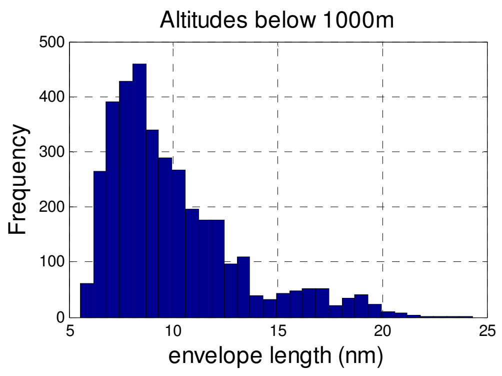


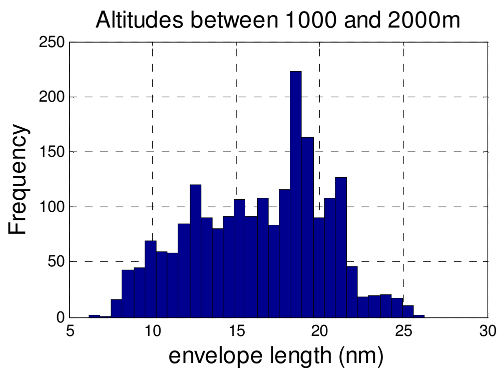


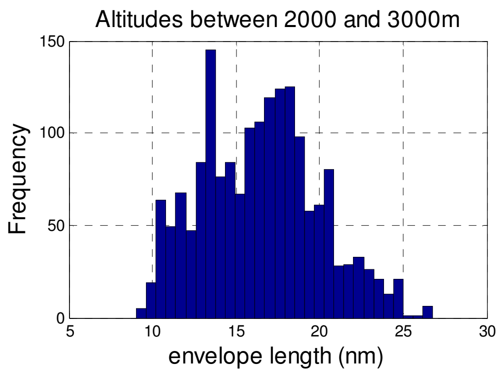


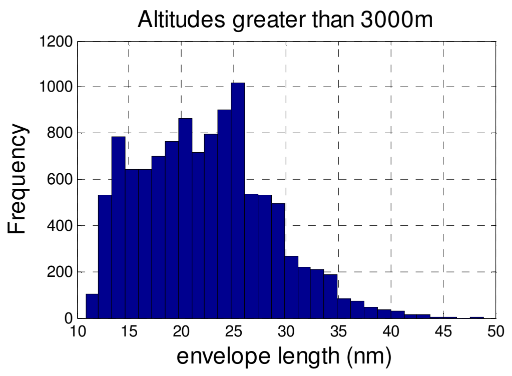


Boeing 747: **B747__20080319_0001_0005_125**

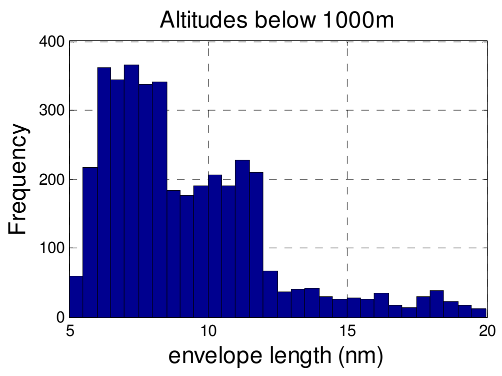


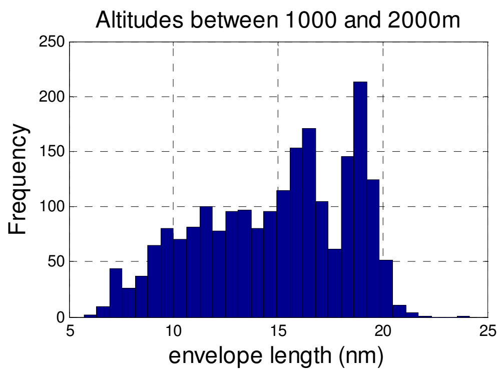


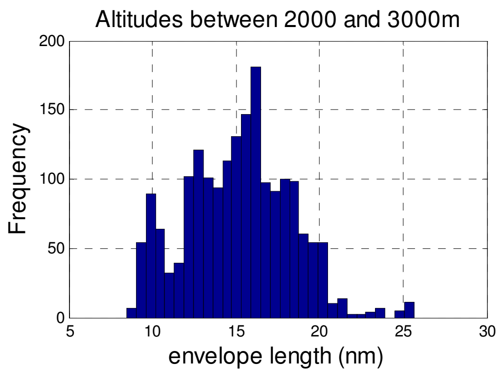


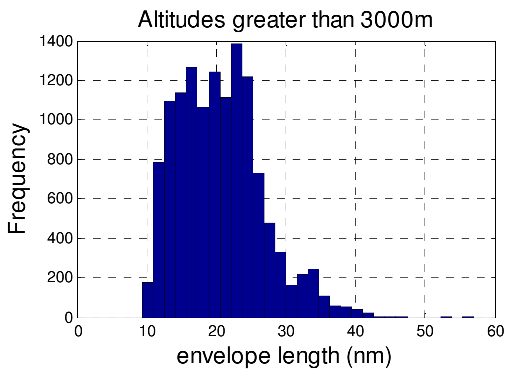


B747__20080319_0001_0005_175

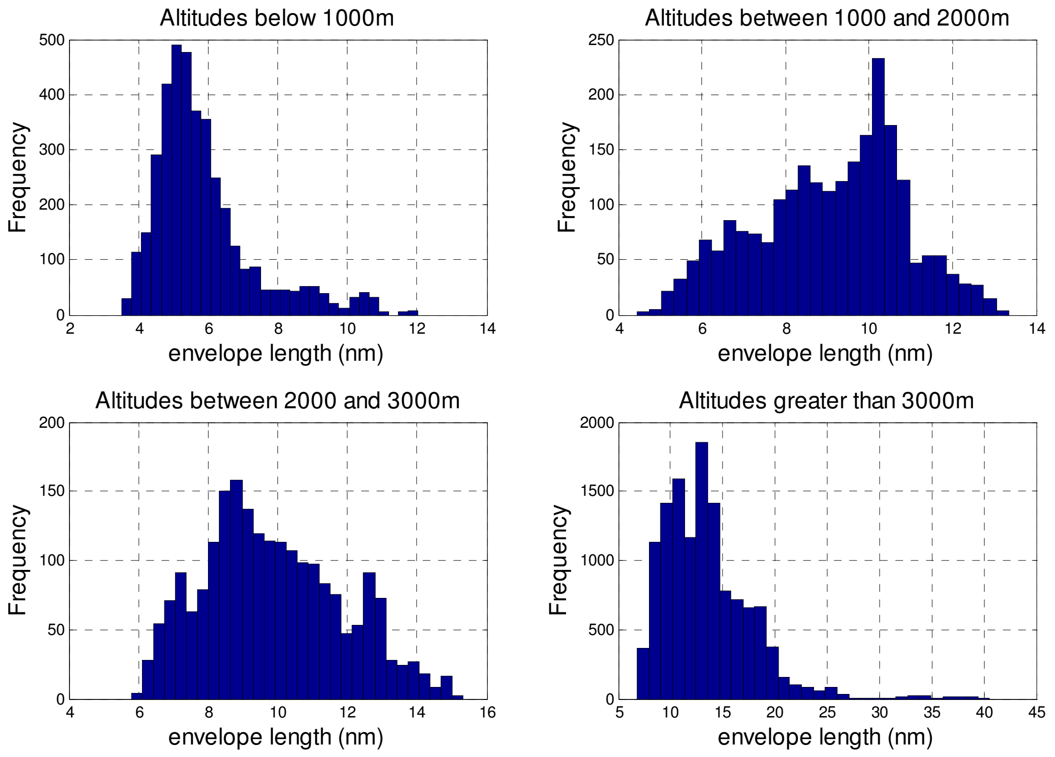




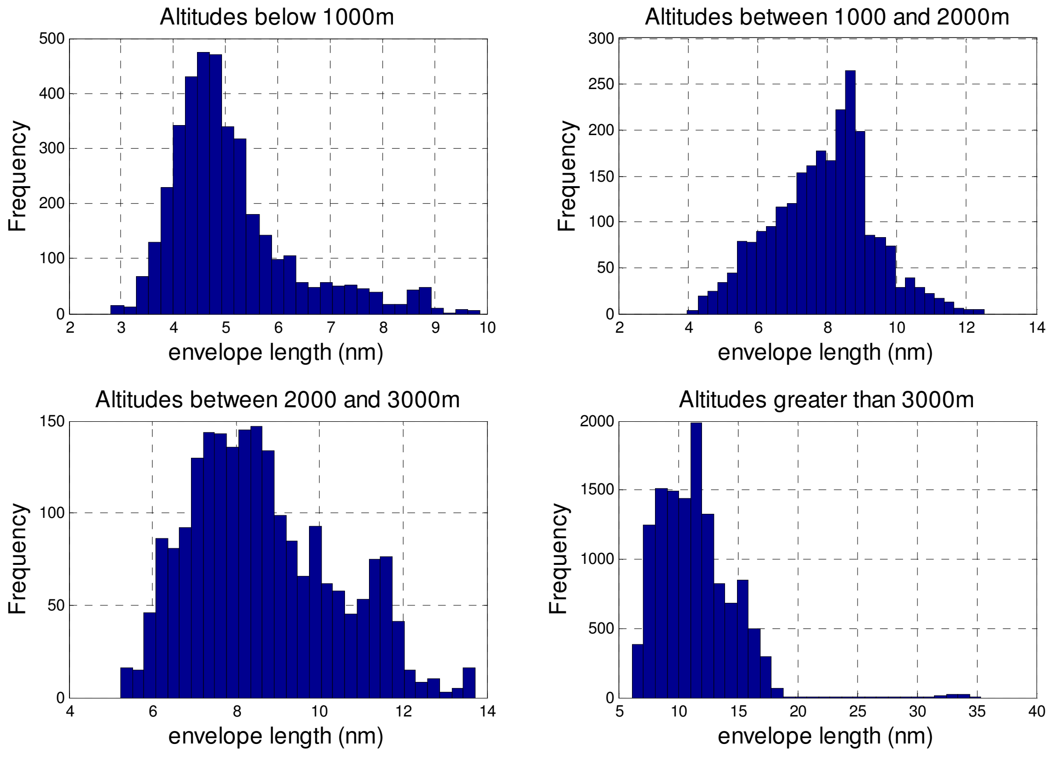




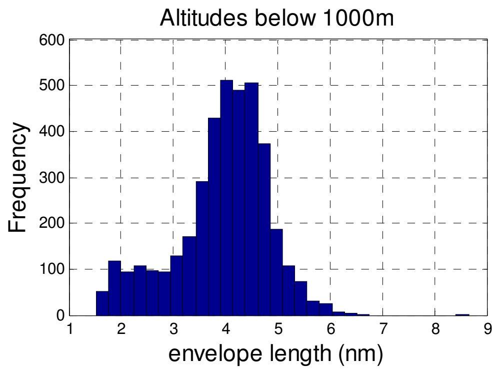
B747__20080319_002_01_125

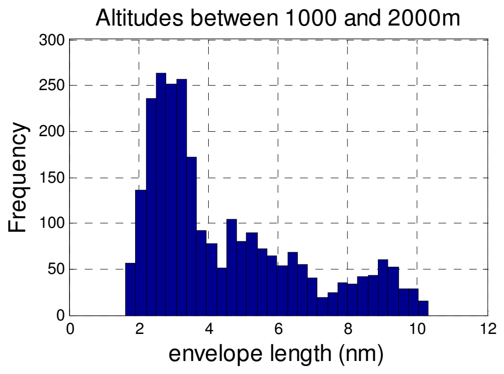


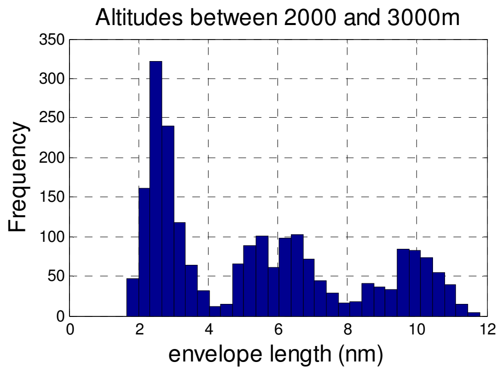
B747__20080319_002_01_175

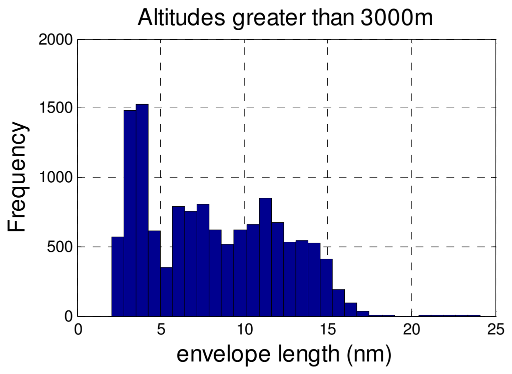


B747__20080319_004_02_125

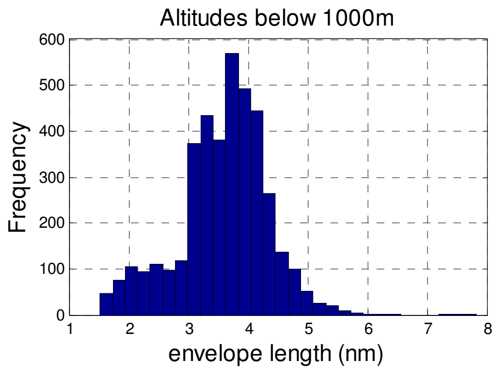


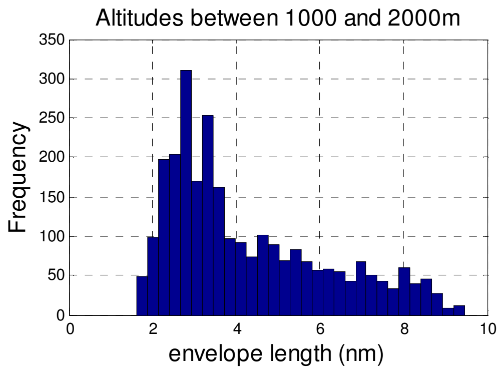


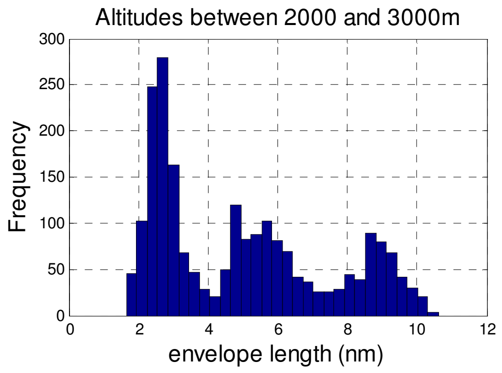


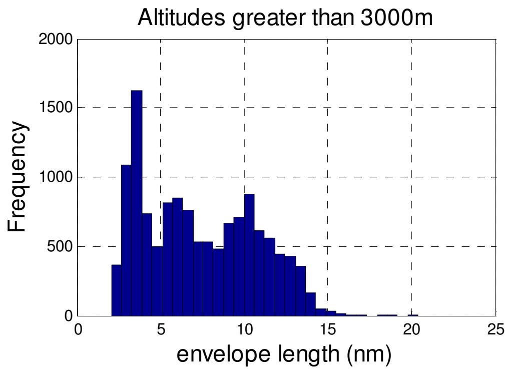


B747__20080319_004_02_175

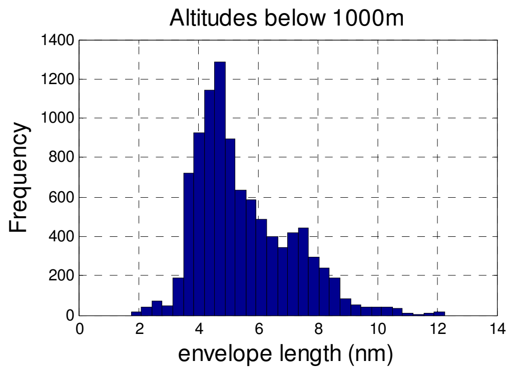


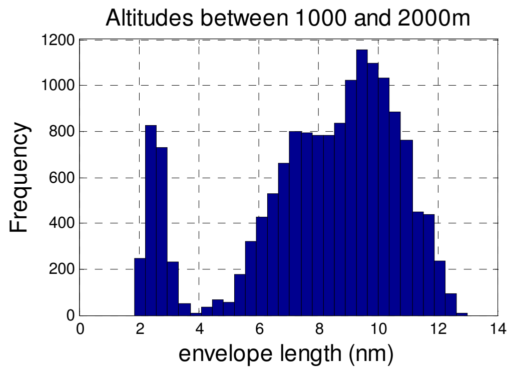


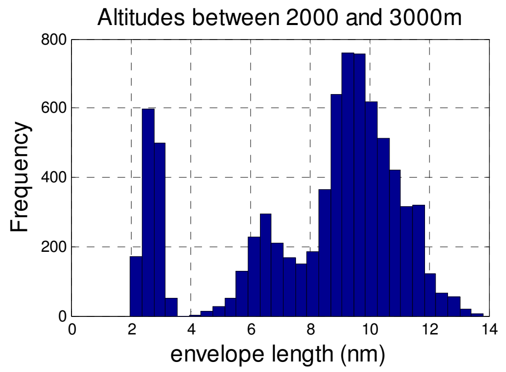


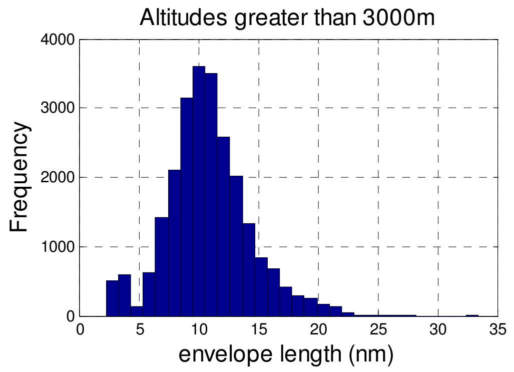


Boeing 737: **B737_20080319_0001_0005_125**

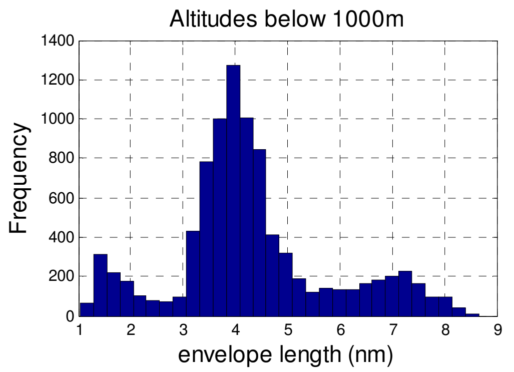


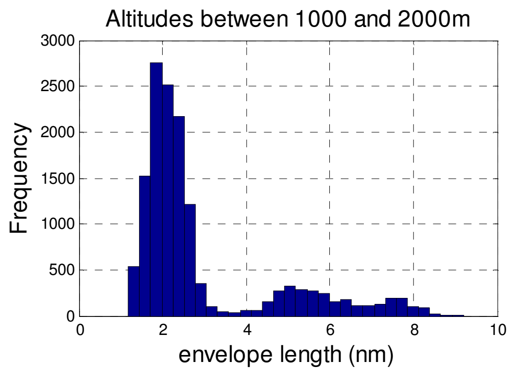


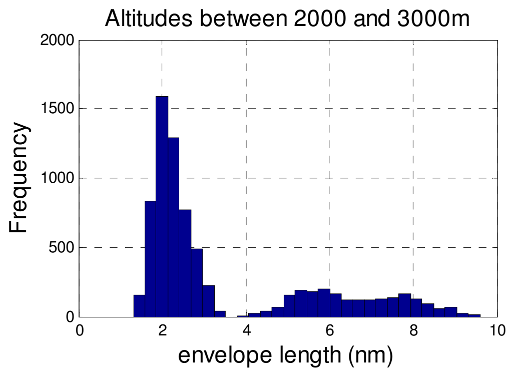


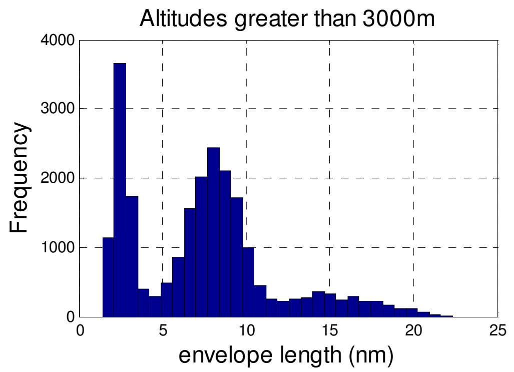


B737_20080319_0001_0005_175

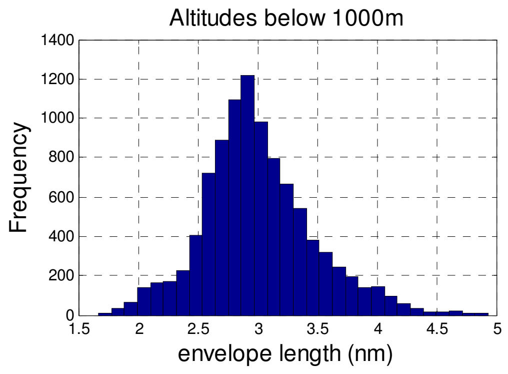


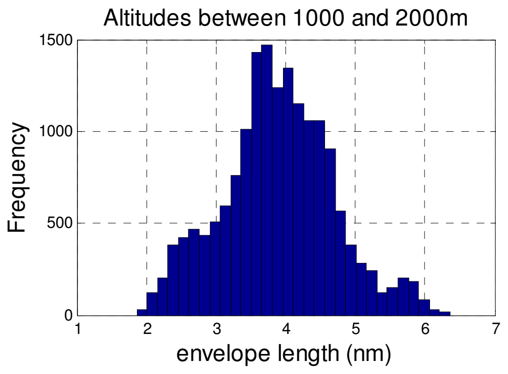


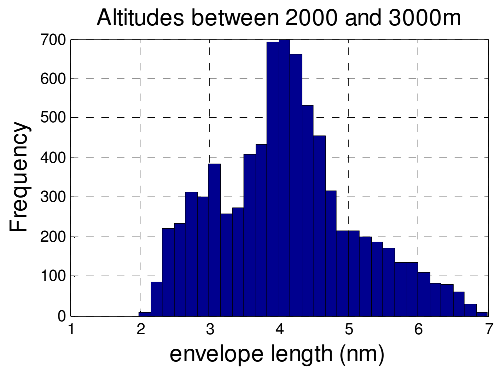


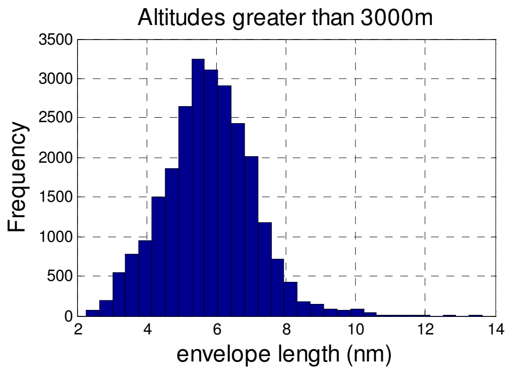


B737_20080319_002_01_125

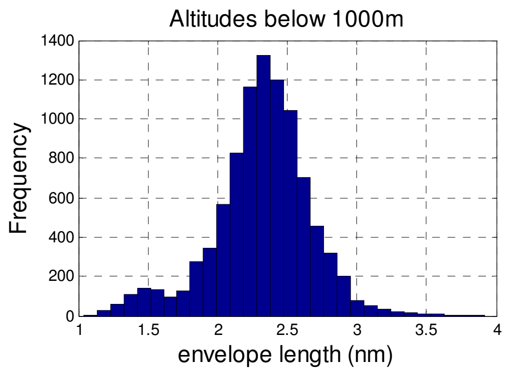


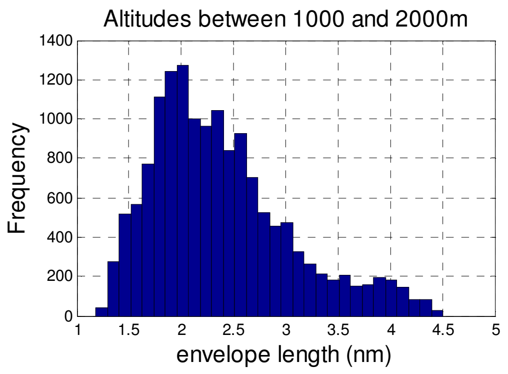


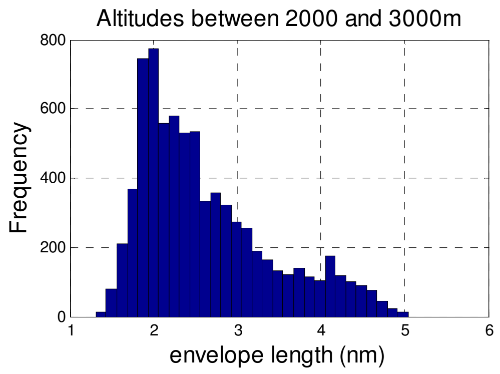


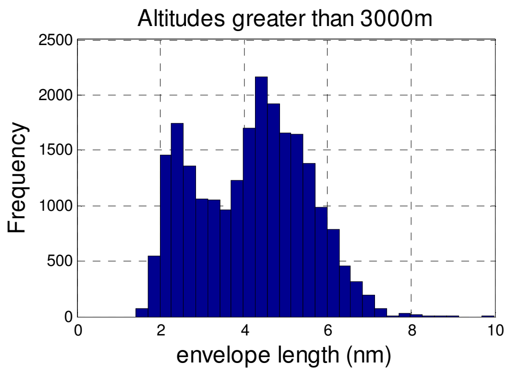


B737_20080319_002_01_175

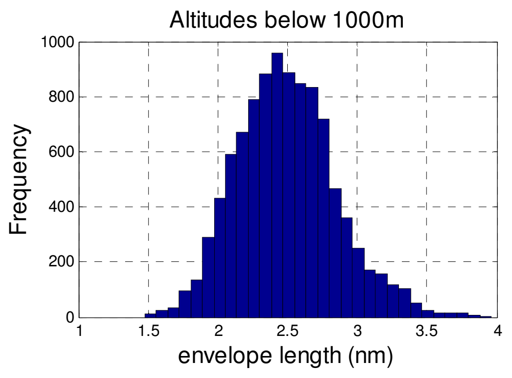


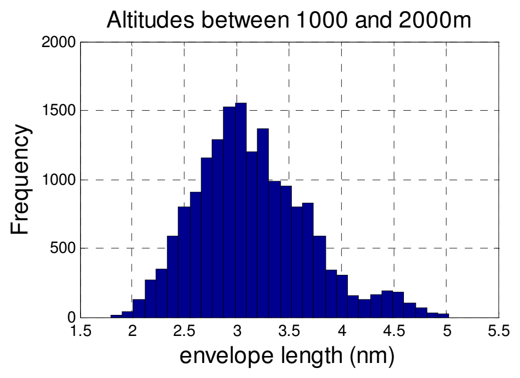


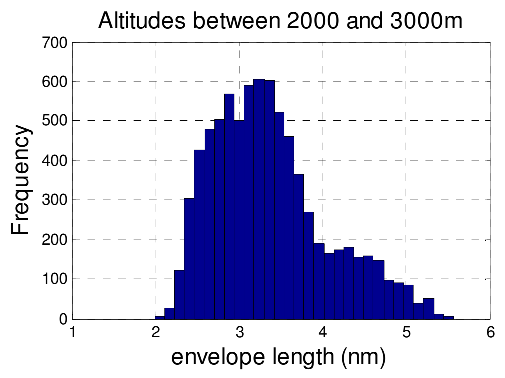


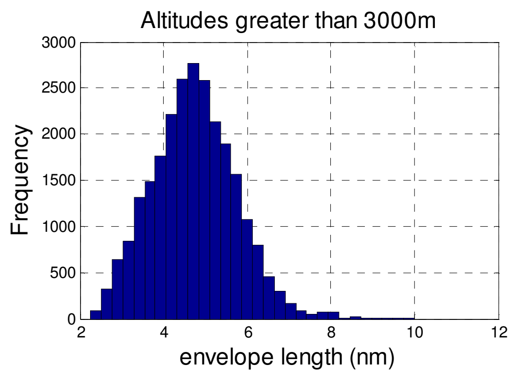


B737_20080319_004_02_125

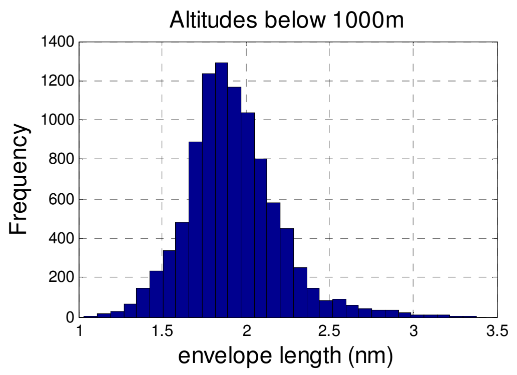


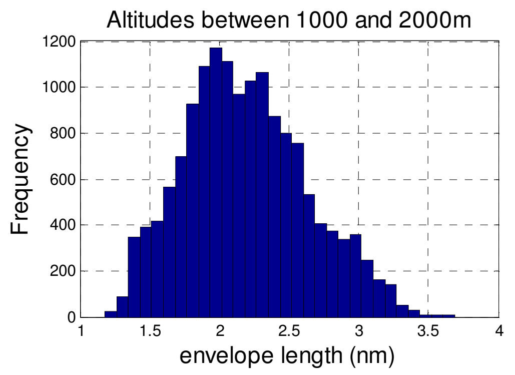


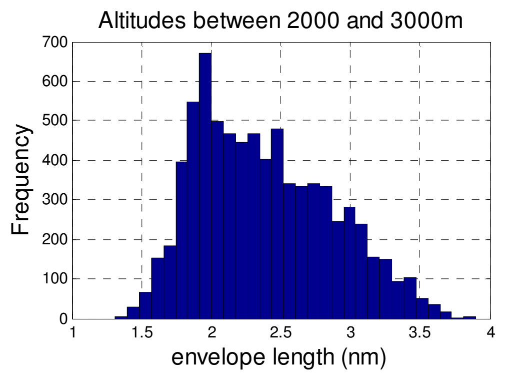


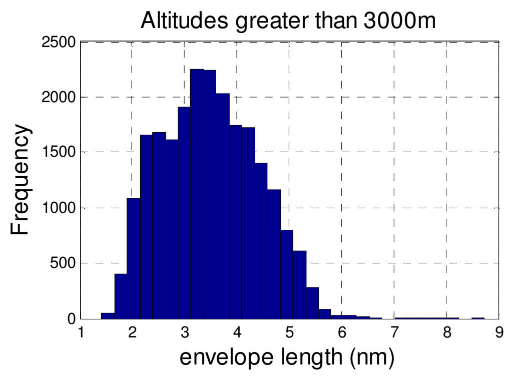


B737_20080319_004_02_175

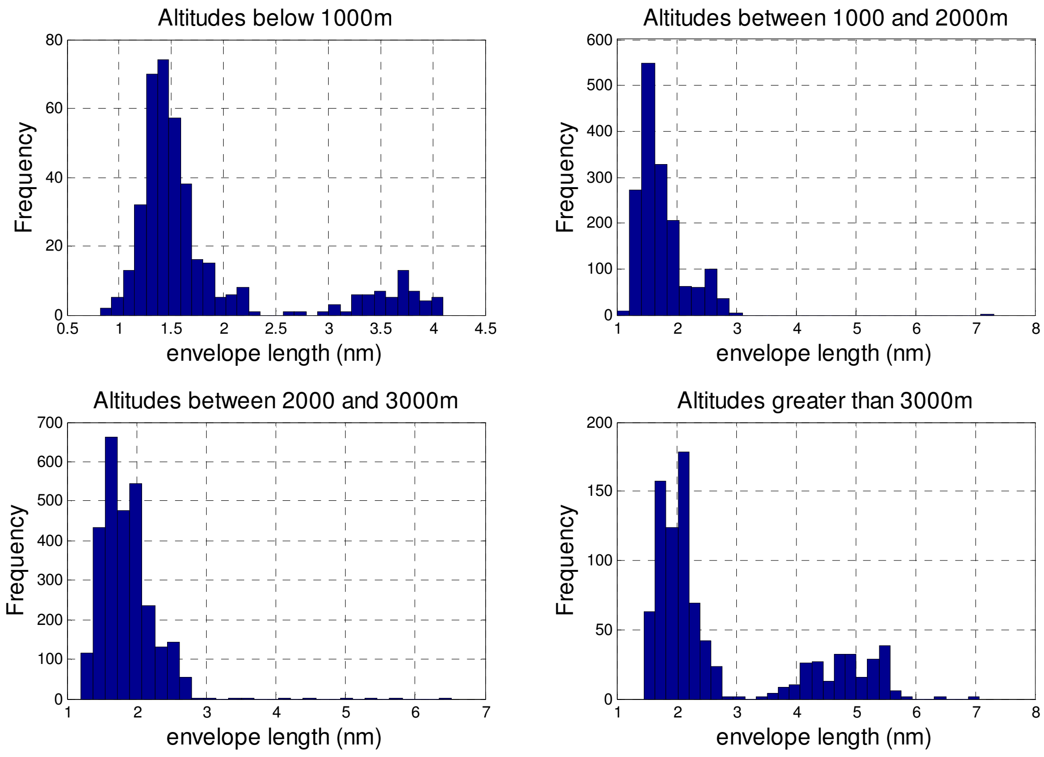




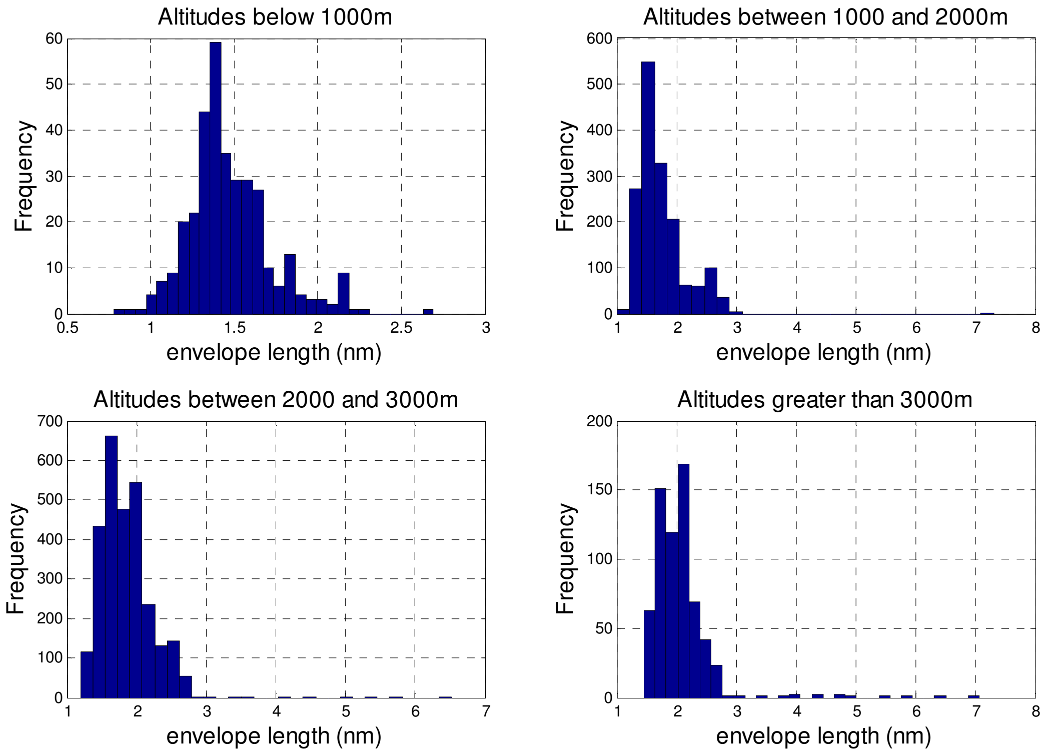




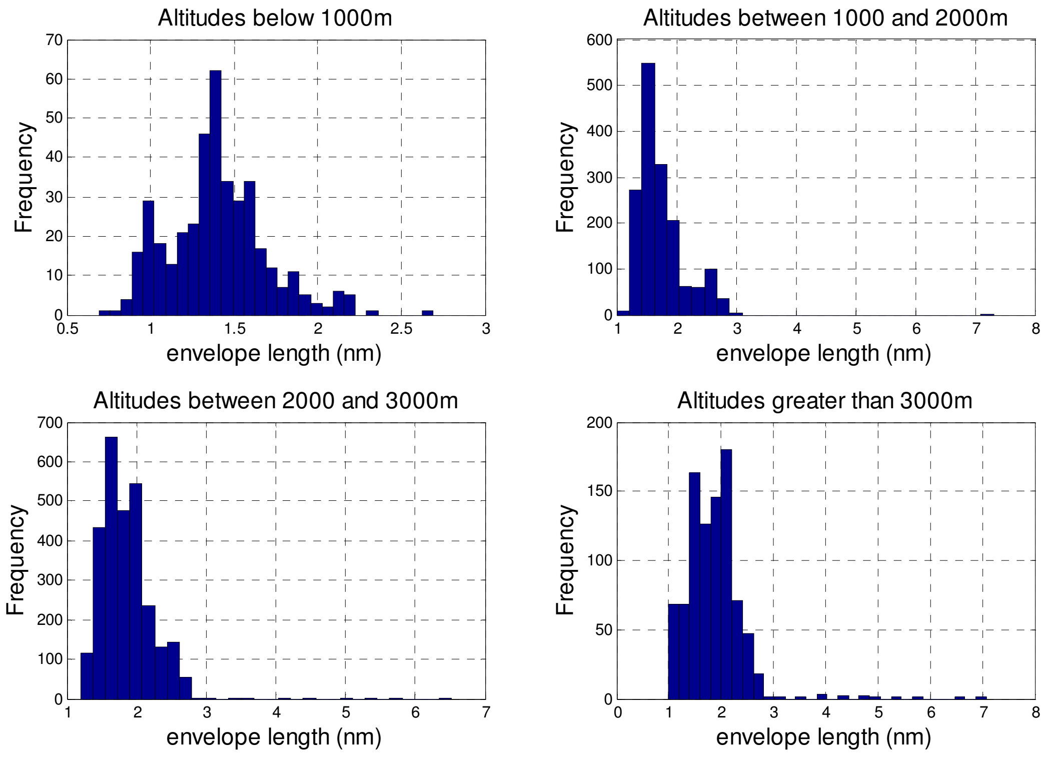
SF 340: **SF340__20080319_0001_0005_125**



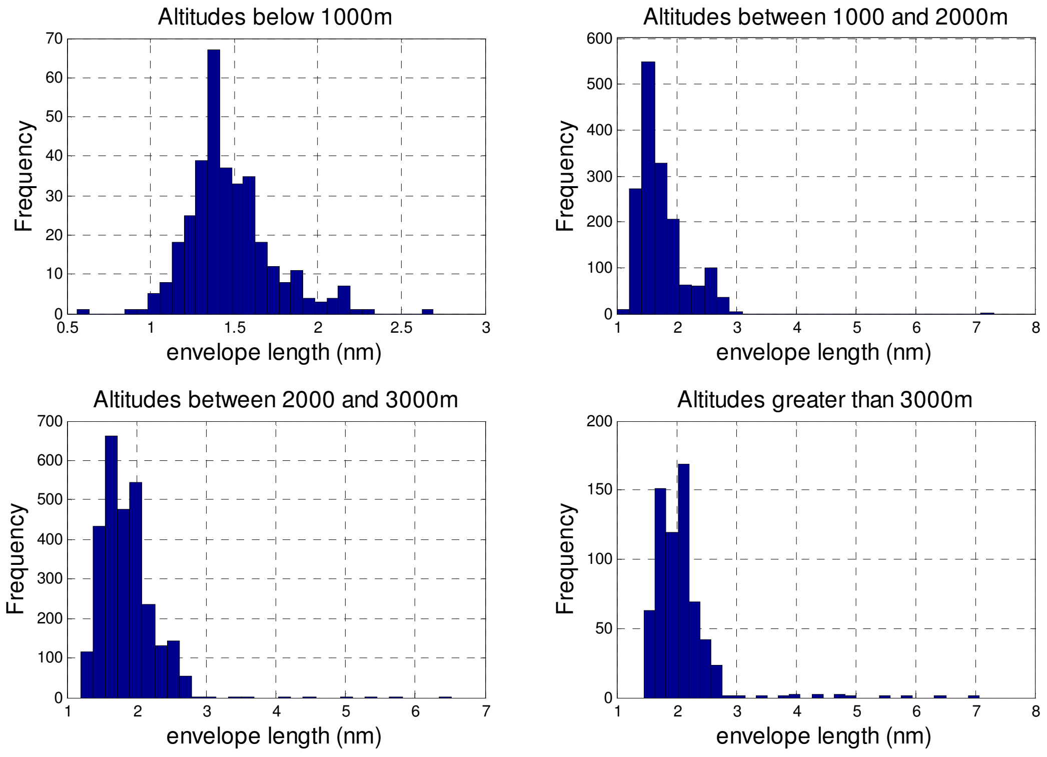
SF340__20080319_0001_0005_175



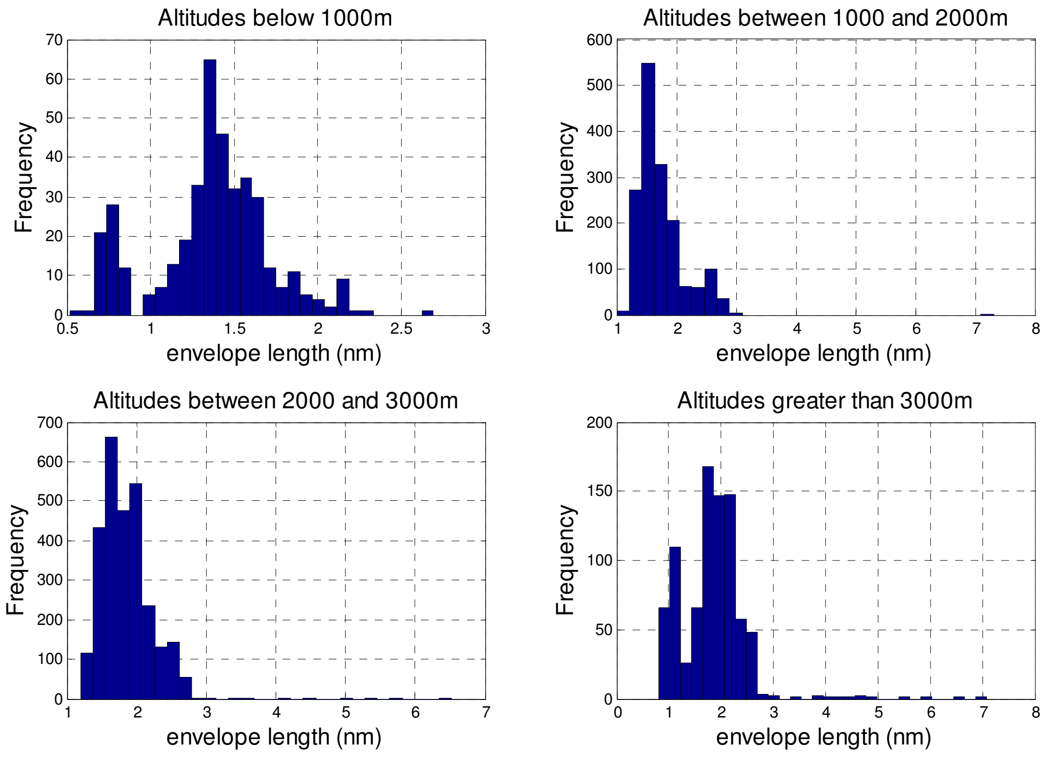
SF340__20080319_002_01_125



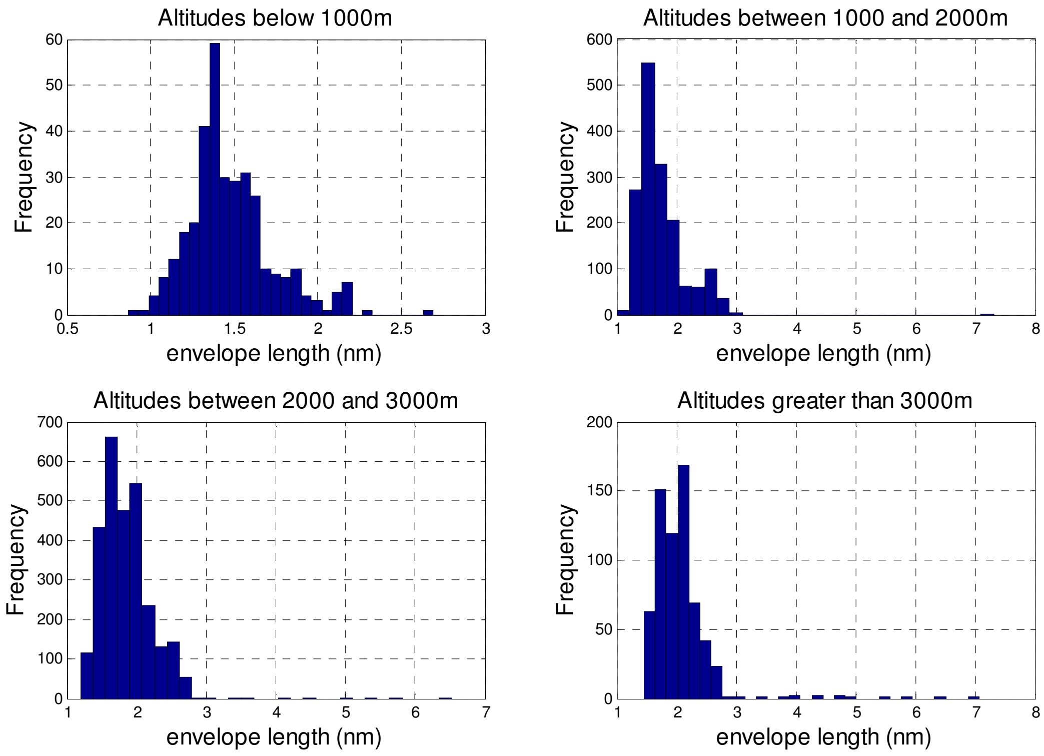
SF340__20080319_002_01_175



SF340__20080319_004_02_125



SF340__20080319_004_02_175



APPENDIX C: MATLAB FILES

MATLAB Scripts

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%Wake Encounter Model—Atlanta Area 20080222Main File

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%Wake Encounter Model
%Model to determine if an aircraft pierces a wake envelope
%parsed PDARS list containing flight paths of each aircraft
clear all;clc
%Nataliya Schroeder, Nov16, 2010
%%------------------------------------------------------------------------
%Import all required files to run the WEM
%Import ISA Density table with values for density based on altitude
ISADensitytable = xlsread('ISADensityTable.xls');
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%
%List of aicraft produced by the Time Filter Script
fid0 = fopen('A80_FilteredData_20080222.txt' , 'r');
timefilteredrecords = textscan(fid0, '%d%d', 'delimiter', ',');
%Field 1: filtered lead record
%Field 2: filtered follower record
filteredleadrecord     = timefilteredrecords{1};
filteredfollowerrecord = timefilteredrecords{2};
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%
%Records for each aircraft obtained from the PDARS parced file which has
%smoothed latitude, longitude and altitude
%fid1 = fopen('A80_PDARS_20080222.txt' ,'r');
fid1 = fopen('A80_PDARS_20080222_smoothed.txt' ,'r');
flighttrackdata = textscan(fid1, '%f%f%f%f%f%f%f', 'delimiter', ',');
%Field 1: record number
%Field 2: time
%Field 3: latitude, [degrees]
%Field 4: longitude, [degrees]
%Field 5: altitude, [100x ft]
%Field 6: groundspeed , [knots]
%Field 7: heading , [degrees from magnetic North]
recordnum   = flighttrackdata{1};
time        = flighttrackdata{2};
latitude    = flighttrackdata{3};
longitude   = flighttrackdata{4};
altitude    = flighttrackdata{5}; %in 100 ft
groundspeed = flighttrackdata{6};
heading     = flighttrackdata{7};
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%
%list used to match actype with recordnum; total number of flights
fid2 = fopen('A80PDARFlightList_20080222.txt', 'r');
flightlist = textscan(fid2, '%f%s%s%s%s','delimiter', ',');
%Field 1: flight recs
%Field 2: flight id
%Field 3: actypes
%Field 4: departure airport
```

```
%Field 5: arrival airportt
flightrecs = flightlist{1};
actypes    = flightlist{3};
deptarpt   = flightlist{4};
arrvarpt   = flightlist{5};
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%Table provided by GMU used to build the wake envelope for wake vortex
%generating aircraft
wakenormtable = xlsread('DummyEnvelopeTable.xls', 'A3:X731');
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%aircraft information (mass, wingspan, etc)
fid3 = fopen('exportAircraft_PDARS.csv', 'r');
masswingspanlookup = textscan(fid3, '%s%s%s%s%s%f%f%f%f%f', 'delimiter', ',');
%Field 1: aircraft name
%Field 2: aircraft manufacturer
%Field 3: acid_long
%Field 4: acid
%Field 5: wakeclass
%Field 6: malw
%Field 7: mtow
%Field 8: wspan
%Field 9: oew
acid      = masswingspanlookup{4};
wakeclass = masswingspanlookup{5};
malw      = masswingspanlookup{6};
mtow      = masswingspanlookup{7};
wspan     = masswingspanlookup{8};
oew       = masswingspanlookup{9};
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%Table containing departures and arrivals mean weights per aicraft class
fid4 = fopen('AircraftMassDistribution.csv', 'r');
acftmassdistro = textscan(fid4, '%s', 'delimiter', ',');
acftdistro = xlsread('AircraftMassDistribution.xls', 'B3:E6');
%Field 1: acft class
%Field 2: dept mean
%Field 3: dept sd
%Field 4: arr mean
%Field 5: arr sd
acftclass = acftmassdistro{1};
fclose all;
recordlength = length(filteredleadrecord); %length of available records from the PDARS file. Multiple records per
individual aircraft. Will be used

%%
%CONSTANTS and PREDETERMINED VALUES---CAN BE CHANGED BY USER
g = 9.81;%gravitational constant (m/s^2)
time_separation            = 0.1; %sec
distance_separation_nm     = 10;  %nm
distance_separation_deg    = nm2deg(10); %conversion factor to degrees
vertical_separation        = 40; %in 100x ft
assigned_num_of_points_back=80; %To ensure at least 15-20nm distance, we go 80 back along the track (envelope
forms behind)
```

```
%Atmospheric Parameters

% EDR = 0.0001;
% BVF = 0.0005;
% CirculationThreshhold = 125; %M^2/S
%
% EDR = 0.0001;
% BVF = 0.0005;
% CirculationThreshhold = 175;

EDR = 0.002;
BVF = 0.01;
CirculationThreshhold = 125;

% EDR = 0.002;
% BVF = 0.01;
% CirculationThreshhold = 175;

% EDR = 0.004;
% BVF = 0.02;
% CirculationThreshhold = 175;

% EDR = 0.004;
% BVF = 0.02;
% CirculationThreshhold = 125;

%strength value is predetermined (m^2/s)
%0.0001, 0.002 , 0.004
%0.0005, 0.01; 0.02
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%
%FIND UNIQUE LEADING AIRCRAFT FROM THE TIME FILTER
list_of_leaders  = double(filteredleadrecord); %all leader records
list_of_followers= double(filteredfollowerrecord); %all follower records
Unique_Leaders   = unique(list_of_leaders); %Find just the unique leaders


tic

counter=0;
for i=1:size(Unique_Leaders,1)

   random_number=randn(1,1); %random number to be used to adjust weight

   if mod(i, 100) == 0
      disp(['Number of flights processed: ', num2str(i)]);
   end %mod(i, 100) == 0

   leadindex              = find(Unique_Leaders(i) == recordnum);%find in PDARS file
   indeces_all_followers  = find(Unique_Leaders(i) ==list_of_leaders);
   recordnum_all_followers=list_of_followers(indeces_all_followers);% in timefilter

   if numel(leadindex)==0 %some records are missing in the PDARS data
      continue
   end
```

```
    recordnumleader  = Unique_Leaders(i);
    leadlength       = length(leadindex);
    time_leading     = time(leadindex);% seconds since 1970
    latitude_leading = latitude(leadindex); %[degrees]
    longitude_leading=longitude(leadindex);%[degrees]
    altitude_leading = altitude(leadindex);%[x100ft]
    groundspeed_leading_knots= groundspeed(leadindex);% knots
    groundspeed_leading_ms=groundspeed_leading_knots*.514; %converts from knots to m/s
    heading_leading= heading(leadindex);
    leader_index_in_Flight_list_file= find(Unique_Leaders(i) ==flightrecs);
    departureairport= deptarpt(leader_index_in_Flight_list_file);
    arrivalairport= arrvarpt(leader_index_in_Flight_list_file);
    actype= actypes(leader_index_in_Flight_list_file);

    %if loop used for when actype doesn't match those in table
    if  strmatch(actype,acid) > 0
        acidindex = strmatch(actype,acid);
        %Possible error if the aircraft type is listed as 'CRJ'
        %because there are three possible CRJ's in the
        %table,and the model selects all three
        if size (acidindex,1)>1
            acidindex= acidindex(1,:);
        end %if size (acidindex,1)>1
    else
        acidindex = 136;
    end %if  strmatch(actypes{actypeindex},acid) > 0

    wakecategory = wakeclass(acidindex);
    wakeclassindex = strmatch(wakecategory, acftclass);
    wingspan = wspan(acidindex)*0.3048; %converts ft to meters
    meanlandingweight = oew(acidindex) + min(1, acftdistro(wakeclassindex, 3))*(malw(acidindex)-
oew(acidindex));
    meandepartureweight = 1.3*oew(acidindex) + min(1, acftdistro(wakeclassindex, 1))*(mtow(acidindex)-
1.3*oew(acidindex));

    %uses departure airport code to determine if a flight is a departure or arrival
    if (strcmp(departureairport, 'ATL') == 1)  || (strcmp(departureairport, 'FTY') == 1)  || (strcmp(departureairport,
'PDK') == 1)
        acmass = (meandepartureweight + random_number(1,1)*meandepartureweight*acftdistro(wakeclassindex,
2))*0.454; %converts lbs to kilograms
    else
        acmass = (meanlandingweight + random_number(1,1)*meanlandingweight*acftdistro(wakeclassindex,
4))*0.454; %converts lbs to kilograms
    end %(strcmp(departureairport, 'ATL') ...



    %Extract information for all followers of the "i"-leader
    num_of_followers = length(recordnum_all_followers);

    for q= 1:num_of_followers
        %Extract information for each follower of the "i"-leader
        recordnumfollower=  recordnum_all_followers(q);
        followindex= find(recordnumfollower == recordnum); %find in PDARS file
        time_following     = time(followindex);% seconds since 1970
        latitude_following = latitude(followindex); %[degrees]
        longitude_following=longitude(followindex);%[degrees]
        altitude_following = altitude(followindex);%[x100ft]
        groundspeed_following_knots= groundspeed(followindex);% knots
        groundspeed_following_ms=groundspeed_following_knots*.514; %converts from knots to m/s
        heading_following= heading(followindex);
        followlength = length(followindex);

        follower_index_in_Flight_list_file= find(recordnumfollower ==flightrecs);
        departureairport_follower= deptarpt(follower_index_in_Flight_list_file);
        arrivalairport_follower= arrvarpt(follower_index_in_Flight_list_file);
        actype_follower= actypes(follower_index_in_Flight_list_file);

        for j = 1:2:leadlength

          for k = 1:2:followlength

            leadtime     = time_leading(j);%[seconds since 1970]
            leadlatitude = latitude_leading(j);%[degrees]
            leadlongitude= longitude_leading(j);%[degrees]
            leadaltitude = altitude_leading(j);%[x100ft]

            followtime      = time_following(k);%[seconds since 1970]
            followlatitude  = latitude_following(k);%[degrees]
            followlongitude = longitude_following(k);%[degrees]
            followaltitude  = altitude_following(k);%[x100ft]


            %this loop compares chosen flight against other flights and filters out
            %tho)se flights we want to create wake envelopes for
            if (abs(leadtime - followtime) <= time_separation)...
    && (distance('gc',[leadlongitude,leadlatitude],[followlongitude,followlatitude]) <= distance_separation_deg)...
    %warning distance in degrees (0.1332 = 8 miles)
    && (abs( leadaltitude - followaltitude) <= vertical_separation)

              %once possible interaction filtered, establish basic info about
              %the wake producing aircraft (lat, long, alt, gs, actype, mass)
              aircraftposition = [followlongitude,followlatitude,followaltitude]; %[deg,deg,100ft]
              followheading=heading(followindex(k));

              wakeheading =  heading(leadindex(j)); %[degrees from true North]
              wakelongitude = leadlongitude;%[degrees]
              wakelatitude  = leadlatitude;%[degrees]
              wakealtitude =  leadaltitude*100*0.3048;  %converts from 100's feet to meters
              lead_position= [wakelongitude,wakelatitude,leadaltitude];%[degrees,degrees, 100ft]
              wakegroundspeed = groundspeed(leadindex(j))*.514; %converts from knots to m/s
              rho = densitylookup(ISADensitytable,wakealtitude); %altitude has to be in meters

              %this section will look up actype, mass, wingspan, etc values of all the
              %wakecandidate aircraft, these values will then be used to determine
              %normalized values of circulation, EDR and BVF
              CirculationInitial = (acmass*g)/(rho*pi/4*wingspan*wakegroundspeed);  %M^2/S
              CirculationNorm = CirculationThreshhold / CirculationInitial;
              SpanInitial = pi/4*wingspan;
              EDRnorm = (2*pi*EDR^(1/3)*(SpanInitial).^(4/3))/CirculationInitial;
              TimeInitial = (2*pi*SpanInitial.^2)/CirculationInitial;
              BVFNorm = BVF * TimeInitial;

              %this section uses normalized circulation, EDR and BVF to lookup
              %normalized x,y,z coordinates of the wake envelope from
              %the GMU table
              circmin = (CirculationNorm  >= wakenormtable(: ,5));
              circmax = (CirculationNorm  < wakenormtable(: , 6));
              EDRmin = (EDRnorm >= wakenormtable(: ,1));
              EDRmax = (EDRnorm < wakenormtable(: , 2));
              BVFmin = (BVFNorm >= wakenormtable(: , 3));
              BVFmax = (BVFNorm < wakenormtable(: , 4));

              %gives row where normalized values are located
              %if statement covers very small acft whose values will exceed the
              %table provided by GMU (i.e. very tiny wakes)
              if (CirculationNorm >= 1.00) || (EDRnorm >= 0.90) || (BVFNorm >= 0.90)
                 finalindex = 729;
              else
                 finalindex = find(EDRmin == 1  &  EDRmax == 1 & BVFmin == 1  &  BVFmax == 1 & circmin ==
    1 & circmax == 1);
              end %(CirculationNorm >= 1.00) || (EDRnorm >= 0.90) || (BVFNorm >= 0.90)


              envelope_starting_point=lead_position; %[degrees,degrees, 100ft]
              acft_lat  = envelope_starting_point (1,2);
              %acft_long = envelope_starting_point (1,1);
              acft_alt  = envelope_starting_point (1,3);%[100ft]

    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    %%%%%%%%%%%%%%%%%%%%
              % FUNCTION "generate_straight_envelope" will construct the straight envelope using GMU
              % Look-Up Table and TDAWP Model
              envelopecoordinates =
    generate_straight_envelope(wakelatitude,wakealtitude,wakeheading,wakelongitude,wingspan,wakenormtable,finali
    ndex,TimeInitial,wakegroundspeed);

    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    %%%%%%%%%%%%%%%%%%%%
              % FUNCTION envelopedimensions--computes the dimensions of the straight
              % envelope so they can be used later to construct the bent envelope
              [altLB,altUB, distLLLB, distLRLB,
    distLLUB,distLRUB,halfdist,halfdistUB,envelope_cum_distLLLB,envelope_cum_distLRLB,envelope_cum_distLL
    UB,envelope_cum_distLRUB] = envelopedimensions(envelopecoordinates);

    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    %%%%%%%%%%%%%%%%%%%%
              index_start = find( acft_alt==altitude_leading & acft_lat==latitude_leading);%find the corresponding
    point along the PDARS track
              index_end   = max(index_start-assigned_num_of_points_back,1); %Last point cannot be less than the
    first PDARS point
              num_of_points_back= index_start-index_end;
              %%
              if num_of_points_back>5
                 %We need minimum of 5 PDARS points travelled already to construct the
                 %envelope
                 use_this_lat_to_compute_track_length  = latitude_leading(index_end:index_start,1); %[degrees](261
    to 341 index)
                 use_this_long_to_compute_track_length = longitude_leading(index_end:index_start,1);%[degrees]
                 use_this_alt_to_compute_track_length  = altitude_leading(index_end:index_start,1);

                 use_this_lat_to_plot_envelope  =flipud(use_this_lat_to_compute_track_length); %[degrees] (from
    341 to 261)
                 use_this_long_to_plot_envelope =flipud(use_this_long_to_compute_track_length);%[degrees]
                 use_this_alt_to_plot_envelope  =flipud(use_this_alt_to_compute_track_length);%

                 % Compute the distance between each sets of points along the PDARS
                 % trajectory, then compute the cummulative distance and compare to the
                 % actual envelope lenght, then select the actual number of points
                 compute_dist_between_each_point= zeros(num_of_points_back,1);
                 for r= 1:num_of_points_back
                   compute_dist_between_each_point (r,1)= distance(use_this_lat_to_compute_track_length(r,1),
    use_this_long_to_compute_track_length(r,1),
    use_this_lat_to_compute_track_length(r+1,1),use_this_long_to_compute_track_length(r+1,1));
                 end %for r= 1:num_of_points_back
                 dist_between_each_point= compute_dist_between_each_point;
                 %Flip the matrix so the end of the trajectory is the beggining of the
                 %envelope and compute the cummulative distances
                 dist_between_each_point_flipped= flipud(dist_between_each_point); %Flip it so it goes in the
    direction of the envelope-backwords
                 %from 341--index start to 261--index_end

                 compute_cum_dist_PDARS=zeros(num_of_points_back,1);
                 %Now compute the cummulative distance starting at pt.0 (going backwards on track 346-241)
                 compute_cum_dist_PDARS(1,1)=  dist_between_each_point_flipped (1,1);
                 for l=2:num_of_points_back
                   compute_cum_dist_PDARS(l,1)= compute_cum_dist_PDARS(l-1,1)+
    dist_between_each_point_flipped(l,1);
                 end % for l=2:num_of_points_back
                 cum_dist_PDARS= compute_cum_dist_PDARS;
                 %First find the last point of the PDARS track that will be used
                 final_index_PDARS = find (envelope_cum_distLLLB (5,1)<cum_dist_PDARS (:,1),1,'first');

                 if final_index_PDARS>5 % was 0 and it worked good

                   %Interpolate along the envelope centerline to find the corresponding width
                   %at the available PDARS locations
                   cum_dist_PDARS_for_envelope= cum_dist_PDARS(1:final_index_PDARS,1);
                   usable_lat = use_this_lat_to_plot_envelope (1:final_index_PDARS,1);
                   usable_long= use_this_long_to_plot_envelope(1:final_index_PDARS,1);
                   usable_alt= use_this_alt_to_plot_envelope(1:final_index_PDARS,1);

                   %Compute the azimuths between all needed points (will be less than 80)
                   compute_azimuth_between_each_point=zeros(final_index_PDARS-1,1); %prealocate
                   for n= 1:(final_index_PDARS-1)
                     compute_azimuth_between_each_point(n)= azimuth(usable_lat(n+1,1), usable_long(n+1,1),
    usable_lat(n,1),usable_long(n,1));
                     %compute_azimuth_between_each_point(n)= azimuth(use_this_lat_to_plot_envelope(n+1,1),
    use_this_long_to_plot_envelope(n+1,1), use_this_lat_to_plot_envelope(n,1),use_this_long_to_plot_envelope(n,1));
                   end %for n= 1:(final_index_PDARS-1)
                   azimuth_between_each_point= compute_azimuth_between_each_point;
```

```
                    projected_point_coord_LLB=zeros(final_index_PDARS-1,3);
                    projected_point_coord_RLB=zeros(final_index_PDARS-1,3);
                    projected_point_coord_LUB=zeros(final_index_PDARS-1,3);
                    projected_point_coord_RUB=zeros(final_index_PDARS-1,3);

                    %Interpolate along the envelope width to find the corresponding value
                    for m=1:(final_index_PDARS-1)
                      %LOWER BOUND (LB)
                      %The indeces will be from 1 to 5 (or 4 if 4 and 5 coincide)
                      Index= find (cum_dist_PDARS_for_envelope (m,1)<envelope_cum_distLLLB (:,1),1, 'first');
                      if Index==0
                        continue
                      end

                      Distance_to_add (m,1)= cum_dist_PDARS_for_envelope(m,1)*(halfdist(Index+1,1)-
halfdist(1,1))/envelope_cum_distLLLB(Index,1);
                      Distance_to_project(m,1) =Distance_to_add (m,1)+  halfdist(1,1);

                      %reckon==Point at specified azimuth, range on sphere or ellipsoid
                      [latoutL,lonoutL]= reckon(usable_lat(m,1), usable_long(m,1),Distance_to_project(m,1),
(azimuth_between_each_point(m,1)-90));
                      projected_point_coord_LLB(m,1)= (latoutL);
                      projected_point_coord_LLB(m,2)= (lonoutL);
                      [latoutR,lonoutR]= reckon(usable_lat(m,1), usable_long(m,1),Distance_to_project(m,1),
(azimuth_between_each_point(m,1)+90));
                      projected_point_coord_RLB(m,1)= (latoutR);
                      projected_point_coord_RLB(m,2)= (lonoutR);
                      %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
                      %UPPER BOUND
                      %The indeces will be from 1 to 5 (or 4 if 4 and 5 coincide)
                      IndexUB= find (cum_dist_PDARS_for_envelope (m,1)<envelope_cum_distLLUB (:,1),1,
'first');
                      if IndexUB==0
                        continue
                      end
                      Distance_to_add_UB (m,1)= cum_dist_PDARS_for_envelope(m,1)*(halfdistUB
(IndexUB+1,1)-halfdistUB(1,1))/envelope_cum_distLLUB(Index,1);
                      Distance_to_project_UB(m,1) =Distance_to_add_UB (m,1)+  halfdistUB(1,1);
                      %reckon==Point at specified azimuth, range on sphere or ellipsoid
                      [latoutLUB,lonoutLUB]= reckon(usable_lat(m,1),
usable_long(m,1),Distance_to_project_UB(m), (azimuth_between_each_point(m,1)-90));
                      projected_point_coord_LUB(m,1)= (latoutLUB);
                      projected_point_coord_LUB(m,2)= (lonoutLUB);
                      %
                      [latoutRUB,lonoutRUB]= reckon(usable_lat(m,1),
usable_long(m,1),Distance_to_project_UB(m), (azimuth_between_each_point(m,1)+90));
                      projected_point_coord_RUB(m,1)= (latoutRUB);
                      projected_point_coord_RUB(m,2)= (lonoutRUB);
                    end  %for m=1:(final_index_PDARS-1)


                    % save a vector to plot it for LB
                    projected_coord_RLB= projected_point_coord_RLB;
                    projected_coord_LLB =projected_point_coord_LLB;
                    % save a vector to plot it for UB
                    projected_coord_RUB= projected_point_coord_RUB;
                    projected_coord_LUB =projected_point_coord_LUB;


                    %Interpolate the altitudes. We need the cummulative distance starting from
                    %0
                    PDARS_altitute= usable_alt(1:final_index_PDARS-1,1);
                    envelope_cum_distLLLB_rev=[0;envelope_cum_distLLLB];
                    envelope_cum_distLLUB_rev=[0;envelope_cum_distLLUB];

                    record_length= length(cum_dist_PDARS_for_envelope)-1; %we want 1 less record cause it will
be NAN
                    compute_altitudeLB =  interp1q(envelope_cum_distLLLB_rev
,altLB,cum_dist_PDARS_for_envelope);
                    altitude_to_subtract_LB= compute_altitudeLB(1,1);
                    altitudeLB= compute_altitudeLB(1:record_length,:);
                    altitudeLB_final= altitudeLB+ PDARS_altitute-altitude_to_subtract_LB;

                    compute_altitudeUB =  interp1q(envelope_cum_distLLUB_rev
,altUB,cum_dist_PDARS_for_envelope);
                    altitude_to_subtract_UB= compute_altitudeUB(1,1);
                    altitudeUB= compute_altitudeUB(1:record_length,:);
                    altitudeUB_final= altitudeUB+ PDARS_altitute-altitude_to_subtract_UB;
                    altitudeUB_final(1,1)=PDARS_altitute(1,1);

                    projected_coord_LLB (:,3) = altitudeLB_final;
                    projected_coord_RLB (:,3) = altitudeLB_final;
                    projected_coord_LUB(:,3) = altitudeUB_final;
                    projected_coord_RUB(:,3) = altitudeUB_final;

                    projected_coord_RUB(record_length,3)= projected_coord_RLB(record_length,3);
                    projected_coord_LUB(record_length,3)= projected_coord_LLB(record_length,3);

                    projected_coord_RLB_flipped= flipud(projected_coord_RLB);
                    Lower_surface= [projected_coord_LLB;
                       projected_coord_RLB_flipped];
                    projected_coord_RUB_flipped= flipud(projected_coord_RUB);
                    Upper_surface= [projected_coord_LUB
                       projected_coord_RUB_flipped];
                    bentenvelope= [Lower_surface
                       Upper_surface];

                    %need aircraft position in the correct order--longitude,latitude, altitude
                    aircraftposition_corrected = [aircraftposition(1,2),aircraftposition(1,1),aircraftposition(:,3)];

                    if size((bentenvelope),1)>5
                       wakepiercing = inhull(aircraftposition_corrected, bentenvelope); %[deg deg 100ft]
                    end %if size((bentenvelope),1)>5


                    length_of_envelope_nm= deg2nm(envelope_cum_distLLUB_rev(6,:));

                    fid5 = fopen ('A80_20080222_EDR002_BVF001_CS125_1210.txt','a');
                    fprintf(fid5, '%d,%f,%f,%f,%f,%f,%d,%f,%f,%f,%f,%f,%d \n', recordnumleader,wakelongitude,
wakelatitude,leadaltitude,
wakegroundspeed,wakeheading,recordnumfollower,aircraftposition(1,1),aircraftposition(1,2),aircraftposition(1,3),fo
llowheading,length_of_envelope_nm,wakepiercing);
```

```
                    status = fclose(fid5);

                    %plots
                    if wakepiercing==1

                      %                  length_of_envelope= deg2nm(envelope_cum_distLLUB_rev(6,:));
                      %                  wake_pair(i).distance= length_of_envelope;

                      figure



                      counter=  counter+1;
                      axes;
                      set(gca,'FontSize',20)
                      actype1= cell2mat(actype);
                      actype2= cell2mat(actype_follower);
                      Title1 = sprintf('%s - %s ',actype1,actype2);
                      leaderOD= cell2mat(departureairport);
                      followerOD= cell2mat(departureairport_follower);
                      Title2 = sprintf('%s - %s , %s  - %s' ,leaderOD,followerOD);
                      title({Title1,Title2},'Fontsize',24)


                      %
                      hold on
                      plot3(bentenvelope(:,2), bentenvelope(:,1),bentenvelope(:,3),'m')

plot3(aircraftposition_corrected(:,2),aircraftposition_corrected(:,1),aircraftposition_corrected(:,3),'pentagram')
                      xlabel('Longitude (degrees)','fontsize',20)
                      ylabel('Latitude (degrees)','fontsize',20)
                      zlabel('Altitude (x100 ft)','fontsize',20)
                      plot3(use_this_long_to_plot_envelope,use_this_lat_to_plot_envelope,
use_this_alt_to_plot_envelope,'k')

                      legend('wake envelope','following aircraft position','generating airplane trajectory')
                      grid on
                      set(legend,'FontSize',16);

                      Myfilename= strcat('result_',num2str(counter));
                     saveas(gcf,Myfilename,'fig')

                    end % if wakepiercing==1
                  end %final_index_PDARS>5
                end %if num_of_points_back>5
              end % if (abs(leadtime - followtime) <= 0.1)...
            end %for k = 1:2:followlength
          end %for j = 1:2:leadlength
  end%for q=1: length(num_of_followers)
end %for i= 1:size(Unique_Leaders,1)
toc


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```
Functions

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%
Function File generate_straight_envelope used to create the straight envelope at every time step.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%

function[envelopecoordinates] =
generate_straight_envelope(wakelatitude,wakealtitude,wakeheading,wakelongitude,wingspan,wakenormtable,finali
ndex,TimeInitial,wakegroundspeed)
wakeheadingLong = sin(deg2rad(wakeheading))*TimeInitial*wakegroundspeed*0.001; %projection of the vector in
X directions--along wings
wakeheadingLat  = cos(deg2rad((wakeheading+180)))*TimeInitial*wakegroundspeed*0.001; %+180 for y3, y5...-
180 for y4, y6...


s= pi/4; %s= rotational constant; Used everywhere where wingspan is used; Converts the circular motion to a line
projection

wingspan_km=wingspan*.001; % wingspan converted from m to km



%x axis is along the Longitudinal direction
x1 = deg2km(wakelongitude) + sin(deg2rad((wakeheading-90)))*s*wingspan_km/2;
x2 = deg2km(wakelongitude) + sin(deg2rad((wakeheading+90)))*s*wingspan_km/2;
x3  = x1 - (wakeheadingLong * wakenormtable(finalindex,7) - sin(deg2rad((wakeheading-
90)))*(wakenormtable(finalindex,18)-wakenormtable(finalindex,8))*s*wingspan_km/2);
x4  = x2 - (wakeheadingLong * wakenormtable(finalindex,7) -
sin(deg2rad((wakeheading+90)))*(wakenormtable(finalindex,18)-
wakenormtable(finalindex,8))*s*wingspan_km/2);
x5  = x1 - (wakeheadingLong * wakenormtable(finalindex,9) - sin(deg2rad((wakeheading-
90)))*(wakenormtable(finalindex,20)-wakenormtable(finalindex,10))*s*wingspan_km/2);
x6  = x2 - (wakeheadingLong * wakenormtable(finalindex,9) -
sin(deg2rad((wakeheading+90)))*(wakenormtable(finalindex,20)-
wakenormtable(finalindex,10))*s*wingspan_km/2);
x7  = x1 - (wakeheadingLong * wakenormtable(finalindex,11)- sin(deg2rad((wakeheading-
90)))*(wakenormtable(finalindex,22)-wakenormtable(finalindex,12))*s*wingspan_km/2);
x8  = x2 - (wakeheadingLong * wakenormtable(finalindex,11)-
sin(deg2rad((wakeheading+90)))*(wakenormtable(finalindex,22)-
wakenormtable(finalindex,12))*s*wingspan_km/2);
x9  = x1 - (wakeheadingLong * wakenormtable(finalindex,13)- sin(deg2rad((wakeheading-
90)))*(wakenormtable(finalindex,24)-wakenormtable(finalindex,14))*s*wingspan_km/2);
x10 = x2 - (wakeheadingLong * wakenormtable(finalindex,13)-
sin(deg2rad((wakeheading+90)))*(wakenormtable(finalindex,24)-
wakenormtable(finalindex,14))*s*wingspan_km/2);
x11 = x1 - (wakeheadingLong * wakenormtable(finalindex,15)- sin(deg2rad((wakeheading-
90)))*(wakenormtable(finalindex,24)-wakenormtable(finalindex,14))*s*wingspan_km/2);
x12 = x2 - (wakeheadingLong * wakenormtable(finalindex,15)-
sin(deg2rad((wakeheading+90)))*(wakenormtable(finalindex,24)-
wakenormtable(finalindex,14))*s*wingspan_km/2);
x13 = x1 - (wakeheadingLong * wakenormtable(finalindex,23)- sin(deg2rad((wakeheading-
90)))*(wakenormtable(finalindex,24)-wakenormtable(finalindex,14))*s*wingspan_km/2);
x14 = x2 - (wakeheadingLong * wakenormtable(finalindex,23)-
sin(deg2rad((wakeheading+90)))*(wakenormtable(finalindex,24)-
wakenormtable(finalindex,14))*s*wingspan_km/2);
x15 = x1 - (wakeheadingLong * wakenormtable(finalindex,21)- sin(deg2rad((wakeheading-
90)))*(wakenormtable(finalindex,22)-wakenormtable(finalindex,12))*s*wingspan_km/2);
```

```
x16 = x2 - (wakeheadingLong * wakenormtable(finalindex,21)-
sin(deg2rad((wakeheading+90)))*(wakenormtable(finalindex,22)-
wakenormtable(finalindex,12))*s*wingspan_km/2);
x17 = x1 - (wakeheadingLong * wakenormtable(finalindex,19)- sin(deg2rad((wakeheading-
90)))*(wakenormtable(finalindex,20)-wakenormtable(finalindex,10))*s*wingspan_km/2);
x18 = x2 - (wakeheadingLong * wakenormtable(finalindex,19)-
sin(deg2rad((wakeheading+90)))*(wakenormtable(finalindex,20)-
wakenormtable(finalindex,10))*s*wingspan_km/2);
x19 = x1 - (wakeheadingLong * wakenormtable(finalindex,17)- sin(deg2rad((wakeheading-
90)))*(wakenormtable(finalindex,18)-wakenormtable(finalindex,8))*s*wingspan_km/2);
x20 = x2 - (wakeheadingLong * wakenormtable(finalindex,17)-
sin(deg2rad((wakeheading+90)))*(wakenormtable(finalindex,18)-
wakenormtable(finalindex,8))*s*wingspan_km/2);

%y axis is along the Latitudinal direction
y1  = deg2km(wakelatitude) + cos(deg2rad((wakeheading-90)))*s*wingspan_km/2;
y2  = deg2km(wakelatitude) + cos(deg2rad((wakeheading+90)))*s*wingspan_km/2;
y3  = y1 + wakeheadingLat * wakenormtable(finalindex,7)+ cos(deg2rad((wakeheading-
90)))*(wakenormtable(finalindex,18)-wakenormtable(finalindex,8))*s*wingspan_km/2;
y4  = y2 + wakeheadingLat * wakenormtable(finalindex,7)+
cos(deg2rad((wakeheading+90)))*(wakenormtable(finalindex,18)-wakenormtable(finalindex,8))*s*wingspan_km/2;
y5  = y1 + wakeheadingLat * wakenormtable(finalindex,9)+ cos(deg2rad((wakeheading-
90)))*(wakenormtable(finalindex,20)-wakenormtable(finalindex,10))*s*wingspan_km/2;
y6  = y2 + wakeheadingLat * wakenormtable(finalindex,9)+
cos(deg2rad((wakeheading+90)))*(wakenormtable(finalindex,20)-
wakenormtable(finalindex,10))*s*wingspan_km/2;
y7  = y1 + wakeheadingLat * wakenormtable(finalindex,11)+ cos(deg2rad((wakeheading-
90)))*(wakenormtable(finalindex,22)-wakenormtable(finalindex,12))*s*wingspan_km/2;
y8  = y2 + wakeheadingLat * wakenormtable(finalindex,11)+
cos(deg2rad((wakeheading+90)))*(wakenormtable(finalindex,22)-
wakenormtable(finalindex,12))*s*wingspan_km/2;
y9  = y1 + wakeheadingLat * wakenormtable(finalindex,13)+ cos(deg2rad((wakeheading-
90)))*(wakenormtable(finalindex,24)-wakenormtable(finalindex,14))*s*wingspan_km/2;
y10 = y2 + wakeheadingLat * wakenormtable(finalindex,13)+
cos(deg2rad((wakeheading+90)))*(wakenormtable(finalindex,24)-
wakenormtable(finalindex,14))*s*wingspan_km/2;
y11 = y1 + wakeheadingLat * wakenormtable(finalindex,15)+ cos(deg2rad((wakeheading-
90)))*(wakenormtable(finalindex,24)-wakenormtable(finalindex,14))*s*wingspan_km/2;
y12 = y2 + wakeheadingLat * wakenormtable(finalindex,15)+
cos(deg2rad((wakeheading+90)))*(wakenormtable(finalindex,24)-
wakenormtable(finalindex,14))*s*wingspan_km/2;
y13 = y1 + wakeheadingLat * wakenormtable(finalindex,23)+ cos(deg2rad((wakeheading-
90)))*(wakenormtable(finalindex,24)-wakenormtable(finalindex,14))*s*wingspan_km/2;
y14 = y2 + wakeheadingLat * wakenormtable(finalindex,23)+
cos(deg2rad((wakeheading+90)))*(wakenormtable(finalindex,24)-
wakenormtable(finalindex,14))*s*wingspan_km/2;
y15 = y1 + wakeheadingLat * wakenormtable(finalindex,21)+ cos(deg2rad((wakeheading-
90)))*(wakenormtable(finalindex,22)-wakenormtable(finalindex,12))*s*wingspan_km/2;
y16 = y2 + wakeheadingLat * wakenormtable(finalindex,21)+
cos(deg2rad((wakeheading+90)))*(wakenormtable(finalindex,22)-
wakenormtable(finalindex,12))*s*wingspan_km/2;
y17 = y1 + wakeheadingLat * wakenormtable(finalindex,19)+ cos(deg2rad((wakeheading-
90)))*(wakenormtable(finalindex,20)-wakenormtable(finalindex,10))*s*wingspan_km/2;
y18 = y2 + wakeheadingLat * wakenormtable(finalindex,19)+
cos(deg2rad((wakeheading+90)))*(wakenormtable(finalindex,20)-
wakenormtable(finalindex,10))*s*wingspan_km/2;
```

```
y19 = y1 + wakeheadingLat * wakenormtable(finalindex,17)+ cos(deg2rad((wakeheading-
90)))*(wakenormtable(finalindex,18)-wakenormtable(finalindex,8))*s*wingspan_km/2;
y20 = y2 + wakeheadingLat * wakenormtable(finalindex,17)+
cos(deg2rad((wakeheading+90)))*(wakenormtable(finalindex,18)-wakenormtable(finalindex,8))*s*wingspan_km/2;

%wake altitude-- wakealtitude is given as meters--must
%multiply by 0.001 to convert to km
z1 = wakealtitude*.001;
z2 = wakealtitude*.001;
z3 = z1 + wakenormtable(finalindex,8)*s*wingspan_km;
z4 = z2 + wakenormtable(finalindex,8)*s*wingspan_km;
z5 = z1 + wakenormtable(finalindex,10)*s*wingspan_km;
z6 = z2 + wakenormtable(finalindex,10)*s*wingspan_km;
z7 = z1 + wakenormtable(finalindex,12)*s*wingspan_km;
z8 = z2 + wakenormtable(finalindex,12)*s*wingspan_km;
z9 = z1 + wakenormtable(finalindex,14)*s*wingspan_km;
z10 = z2 + wakenormtable(finalindex,14)*s*wingspan_km;
z11 = z1 + wakenormtable(finalindex,16)*s*wingspan_km;
z12 = z2 + wakenormtable(finalindex,16)*s*wingspan_km;
z13 = z1 + wakenormtable(finalindex,24)*s*wingspan_km;
z14 = z2 + wakenormtable(finalindex,24)*s*wingspan_km;
z15 = z1 + wakenormtable(finalindex,22)*s*wingspan_km;
z16 = z2 + wakenormtable(finalindex,22)*s*wingspan_km;
z17 = z1 + wakenormtable(finalindex,20)*s*wingspan_km;
z18 = z2 + wakenormtable(finalindex,20)*s*wingspan_km;
z19 = z1 + wakenormtable(finalindex,18)*s*wingspan_km;
z20 = z2 + wakenormtable(finalindex,18)*s*wingspan_km;

%creates envelope used for the inhull test
envelopecoordinates_km = [x1, y1, z1;
    x2, y2, z2;
    x3, y3, z3;
    x4, y4, z4;
    x5, y5, z5;
    x6, y6, z6;
    x7, y7, z7;
    x8, y8, z8;
    x9, y9, z9;
    x10, y10, z10;
    x11, y11, z11;
    x12, y12, z12;
    x13, y13, z13;
    x14, y14, z14;
    x15, y15, z15;
    x16, y16, z16;
    x17, y17, z17;
    x18, y18, z18;
    x19, y19, z19;
    x20, y20, z20];

envelopecoordinates(:,1)= km2deg(envelopecoordinates_km(:,1)); %convert longitude to degrees
envelopecoordinates(:,2)= km2deg(envelopecoordinates_km(:,2)); %convert latitude to degrees
envelopecoordinates(:,3) = envelopecoordinates_km(:,3)/100/0.3048/.001; %convert from km to 100ft

end
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
Function envelopedimensions used to calculate the dimensions of the straight envelope which are then used in the
bent envelope design.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

function [altLB,altUB, distLLLB, distLRLB,
distLLUB,distLRUB,halfdist,halfdistUB,envelope_cum_distLLLB,envelope_cum_distLRLB,envelope_cum_distLL
UB,envelope_cum_distLRUB] = envelopedimensions(envelopecoordinates)
%Function that computes the envelope dimensions that will be used to bend the envelope
  %%
%Lower Bound Computation: 1 3 5 7 9 11 12  10 8 6 4 2
%Lower Left
%hypothenuse between 1-3, 3-5, 5-7, 9-11

halfdist1_2  = distance (envelopecoordinates (1,2), envelopecoordinates(1,1), envelopecoordinates (2,2),
envelopecoordinates(2,1))/2;
halfdist3_4  = distance (envelopecoordinates (3,2), envelopecoordinates(3,1), envelopecoordinates (4,2),
envelopecoordinates(4,1))/2;
halfdist5_6  = distance (envelopecoordinates (5,2), envelopecoordinates(5,1), envelopecoordinates (6,2),
envelopecoordinates(6,1))/2;
halfdist7_8  = distance (envelopecoordinates (7,2), envelopecoordinates(7,1), envelopecoordinates (8,2),
envelopecoordinates(8,1))/2;
halfdist9_10 = distance (envelopecoordinates (9,2), envelopecoordinates(9,1), envelopecoordinates (10,2),
envelopecoordinates(10,1))/2;
halfdist11_12= distance (envelopecoordinates (11,2), envelopecoordinates(11,1), envelopecoordinates (12,2),
envelopecoordinates(12,1))/2;

halfdist= [halfdist1_2
        halfdist3_4
        halfdist5_6
        halfdist7_8
        halfdist9_10
        halfdist11_12];

hypothenuse1_3  = distance(envelopecoordinates(1,2), envelopecoordinates(1,1), envelopecoordinates(3,2),
envelopecoordinates(3,1));
hypothenuse3_5  = distance(envelopecoordinates(3,2), envelopecoordinates(3,1), envelopecoordinates(5,2),
envelopecoordinates(5,1));
hypothenuse5_7  = distance(envelopecoordinates(5,2), envelopecoordinates(5,1), envelopecoordinates(7,2),
envelopecoordinates(7,1));
hypothenuse7_9  = distance(envelopecoordinates(7,2), envelopecoordinates(7,1), envelopecoordinates(9,2),
envelopecoordinates(9,1));
hypothenuse9_11 = distance(envelopecoordinates(9,2), envelopecoordinates(9,1), envelopecoordinates(11,2),
envelopecoordinates(11,1));

hypothenuseLL= [hypothenuse1_3
           hypothenuse3_5
           hypothenuse5_7
           hypothenuse7_9
           hypothenuse9_11];

%Find distance along the centerline of the straight envelope
dist13 = sqrt(abs((hypothenuse1_3)^2- (halfdist3_4-halfdist1_2)^2));
dist35 = sqrt(abs((hypothenuse3_5)^2- (halfdist5_6-halfdist3_4)^2));
```

```
dist57 = sqrt(abs((hypothenuse5_7)^2- (halfdist7_8-halfdist5_6)^2));
dist79 = sqrt(abs((hypothenuse7_9)^2- (halfdist9_10-halfdist7_8)^2));
dist911= sqrt(abs((hypothenuse9_11)^2-(halfdist11_12-halfdist9_10)^2));

distLLLB= [dist13
        dist35
        dist57
        dist79
        dist911];

altLB(:,1)= [envelopecoordinates(1,3)
         envelopecoordinates(3,3)
         envelopecoordinates(5,3)
         envelopecoordinates(7,3)
         envelopecoordinates(9,3)
         envelopecoordinates(11,3)]; %in 100 ft



altUB(:,1)= [envelopecoordinates(1,3)
         envelopecoordinates(19,3)
         envelopecoordinates(17,3)
         envelopecoordinates(15,3)
         envelopecoordinates(13,3)
         envelopecoordinates(11,3)]; %in 100 ft




%Lower Right
%distance between 3-4, 4-6, 6-8, 8-10, 10-12
%hypo 2-4 ,4-6, 6-8, 8-10, 10-12
hypothenuse2_4   = distance(envelopecoordinates (2,2), envelopecoordinates(2,1), envelopecoordinates(4,2),
envelopecoordinates (4,1));
hypothenuse4_6   = distance(envelopecoordinates (4,2), envelopecoordinates(4,1), envelopecoordinates(6,2),
envelopecoordinates (6,1));
hypothenuse6_8   = distance(envelopecoordinates (6,2), envelopecoordinates(6,1), envelopecoordinates(8,2),
envelopecoordinates (8,1));
hypothenuse8_10  = distance(envelopecoordinates (8,2), envelopecoordinates(8,1), envelopecoordinates(10,2),
envelopecoordinates (10,1));
hypothenuse10_12 = distance(envelopecoordinates (10,2), envelopecoordinates(10,1), envelopecoordinates(12,2),
envelopecoordinates (12,1));

hypothenuseLR= [hypothenuse2_4
           hypothenuse4_6
           hypothenuse6_8
           hypothenuse8_10
           hypothenuse10_12];

dist24  = sqrt(abs((hypothenuse2_4)^2- (halfdist3_4-halfdist1_2)^2));
dist46  = sqrt(abs((hypothenuse4_6)^2- (halfdist5_6-halfdist3_4)^2));
dist68  = sqrt(abs((hypothenuse6_8)^2- (halfdist7_8-halfdist5_6)^2));
dist810 = sqrt(abs((hypothenuse8_10)^2- (halfdist9_10-halfdist7_8)^2));
dist1012= sqrt(abs((hypothenuse10_12)^2- (halfdist11_12-halfdist9_10)^2));

distLRLB= [dist24
```

```
        dist46
        dist68
        dist810
        dist1012];



%%
 %Upper Bound Computation:% 1,19 17 15 13 11 12 14 16 18 20 2
%Lower Left
%hypothenuse between 1-19, 19-17, 17-15, 15-13,13-11


%halfdist1_2  = distance (envelopecoordinates (1,2), envelopecoordinates(1,1), envelopecoordinates (2,2),
envelopecoordinates(2,1))/2;
halfdist19_20 = distance (envelopecoordinates (19,2), envelopecoordinates(19,1), envelopecoordinates (20,2),
envelopecoordinates(20,1))/2;
halfdist17_18 = distance (envelopecoordinates (17,2), envelopecoordinates(17,1), envelopecoordinates (18,2),
envelopecoordinates(18,1))/2;
halfdist15_16 = distance (envelopecoordinates (15,2), envelopecoordinates(15,1), envelopecoordinates (16,2),
envelopecoordinates(16,1))/2;
halfdist13_14 = distance (envelopecoordinates (13,2), envelopecoordinates(13,1), envelopecoordinates (14,2),
envelopecoordinates(14,1))/2;
%halfdist11_12= distance (envelopecoordinates (11,2), envelopecoordinates(11,1), envelopecoordinates (12,2),
envelopecoordinates(12,1))/2;
%  if  halfdist13_14>halfdist11_12
%      halfdist13_14=halfdist11_12;
%  end

halfdistUB= [halfdist1_2
          halfdist19_20
          halfdist17_18
          halfdist15_16
          halfdist13_14
          halfdist11_12];


%1-19, 19-17, 17-15, 15-13,13-11
hypothenuse1_19  = distance(envelopecoordinates (1,2),  envelopecoordinates (1,1),  envelopecoordinates(19,2),
envelopecoordinates (19,1));
hypothenuse19_17 = distance(envelopecoordinates (19,2), envelopecoordinates (19,1), envelopecoordinates(17,2),
envelopecoordinates (17,1));
hypothenuse17_15 = distance(envelopecoordinates (17,2), envelopecoordinates (17,1), envelopecoordinates(15,2),
envelopecoordinates (15,1));
hypothenuse15_13 = distance(envelopecoordinates (15,2), envelopecoordinates (15,1), envelopecoordinates(13,2),
envelopecoordinates (13,1));
hypothenuse13_11 = distance(envelopecoordinates (13,2), envelopecoordinates (13,1), envelopecoordinates(11,2),
envelopecoordinates (11,1));

hypothenuseLL_UB= [hypothenuse1_19
              hypothenuse19_17
              hypothenuse17_15
              hypothenuse15_13
              hypothenuse13_11];
```

```
dist119  = sqrt(abs((hypothenuse1_19)^2- (halfdist19_20-halfdist1_2)^2));
dist1917 = sqrt(abs((hypothenuse19_17)^2- (halfdist17_18-halfdist19_20)^2));
dist1715 = sqrt(abs((hypothenuse17_15)^2- (halfdist15_16-halfdist17_18)^2));
dist1513 = sqrt(abs((hypothenuse15_13)^2- (halfdist13_14-halfdist15_16)^2));
dist1311 = sqrt(abs((hypothenuse13_11)^2- (halfdist11_12-halfdist13_14)^2));

distLLUB= [dist119
        dist1917
        dist1715
        dist1513
        dist1311];

%Upper Bound
%Lower Right
%hypothenuse between 2-20, 20-18, 18-16,16-14,14-12

hypothenuse2_20  = distance(envelopecoordinates (2,2),  envelopecoordinates (2,1),  envelopecoordinates(20,2),
envelopecoordinates (20,1));
hypothenuse20_18 = distance(envelopecoordinates (20,2), envelopecoordinates (20,1), envelopecoordinates(18,2),
envelopecoordinates (18,1));
hypothenuse18_16 = distance(envelopecoordinates (18,2), envelopecoordinates (18,1), envelopecoordinates(16,2),
envelopecoordinates (16,1));
hypothenuse16_14 = distance(envelopecoordinates (16,2), envelopecoordinates (16,1), envelopecoordinates(14,2),
envelopecoordinates (14,1));
hypothenuse14_12 = distance(envelopecoordinates (14,2), envelopecoordinates (14,1), envelopecoordinates(12,2),
envelopecoordinates (12,1));

hypothenuseLR_UB= [hypothenuse2_20
            hypothenuse20_18
            hypothenuse18_16
            hypothenuse16_14
            hypothenuse14_12];


dist220  = sqrt(abs((hypothenuse2_20)^2- (halfdist19_20-halfdist1_2)^2));
dist2018 = sqrt(abs((hypothenuse20_18)^2- (halfdist17_18-halfdist19_20)^2));
dist1816 = sqrt(abs((hypothenuse18_16)^2- (halfdist15_16-halfdist17_18)^2));
dist1614 = sqrt(abs((hypothenuse16_14)^2- (halfdist13_14-halfdist15_16)^2));
dist1412 = sqrt(abs((hypothenuse14_12)^2- (halfdist11_12-halfdist13_14)^2));

distLRUB= [dist220
        dist2018
        dist1816
        dist1614
        dist1412];

   %%
%Find the cummulative distances from the starting point of the envelope for LB and UB
envelope_cum_distLLLB (1,1)= distLLLB(1,1)  ;
envelope_cum_distLRLB (1,1)= distLRLB(1,1)  ;
envelope_cum_distLLUB (1,1)= distLLUB(1,1)  ;
envelope_cum_distLRUB (1,1)= distLRUB(1,1)  ;
for j=2:length(distLLLB)
   envelope_cum_distLLLB (j,1) = envelope_cum_distLLLB (j-1,1)+distLLLB(j,1);
   envelope_cum_distLRLB (j,1) = envelope_cum_distLRLB (j-1,1)+distLRLB(j,1);
   envelope_cum_distLLUB (j,1) = envelope_cum_distLLUB (j-1,1)+distLLUB(j,1);
```

```
    envelope_cum_distLRUB (j,1) = envelope_cum_distLRUB (j-1,1)+distLRUB(j,1);
end



end



%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%

Function inhull—MATLAB forum
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%
function in = inhull(testpts,xyz,tess,tol)
% inhull: tests if a set of points are inside a convex hull
% usage: in = inhull(testpts,xyz)
% usage: in = inhull(testpts,xyz,tess)
% usage: in = inhull(testpts,xyz,tess,tol)
%
% arguments: (input)
%  testpts - nxp array to test, n data points, in p dimensions
%       If you have many points to test, it is most efficient to
%       call this function once with the entire set.
%
%  xyz - mxp array of vertices of the convex hull, as used by
%       convhulln.
%
%  tess - tessellation (or triangulation) generated by convhulln
%       If tess is left empty or not supplied, then it will be
%       generated.
%
%  tol - (OPTIONAL) tolerance on the tests for inclusion in the
%       convex hull. You can think of tol as the distance a point
%       may possibly lie outside the hull, and still be perceived
%       as on the surface of the hull. Because of numerical slop
%       nothing can ever be done exactly here. I might guess a
%       semi-intelligent value of tol to be
%
%         tol = 1.e-13*mean(abs(xyz(:)))
%
%       In higher dimensions, the numerical issues of floating
%       point arithmetic will probably suggest a larger value
%       of tol.
%
%       DEFAULT: tol = 0
%
% arguments: (output)
%  in  - nx1 logical vector
%        in(i) == 1 --> the i'th point was inside the convex hull.
%
% Example usage: The first point should be inside, the second out
%
%  xy = randn(20,2)
%  tess = convhull(xy(:,1),xy(:,2));
%  testpoints = [ 0 0; 10 10];
%  in = inhull(testpts,xyz,tess)
%
% in =
%     1
%     0
%
% A non-zero count of the number of degenerate simplexes in the hull
% will generate a warning (in 4 or more dimensions.) This warning
% may be disabled off with the command:
%
%   warning('off','inhull:degeneracy')
%
% See also: convhull, convhulln, delaunay, delaunayn, tsearch, tsearchn
%
% Author: John D'Errico
% e-mail: woodchips@rochester.rr.com
% Release: 3.0
% Release date: 10/26/06

% get array sizes
% m points, p dimensions
p = size(xyz,2);
[n,c] = size(testpts);
if p ~= c
  error 'testpts and xyz must have the same number of columns'
end
if p < 2
  error 'Points must lie in at least a 2-d space.'
end

% was the convex hull supplied?
if (nargin<3) || isempty(tess)
  tess = convhulln(xyz);
end
[nt,c] = size(tess);
if c ~= p
  error 'tess array is incompatible with a dimension p space'
end

% was tol supplied?
if (nargin<4) || isempty(tol)
  tol = 0;
end

% build normal vectors
switch p
  case 2
    % really simple for 2-d
    nrmls = (xyz(tess(:,1),:) - xyz(tess(:,2),:)) * [0 1;-1 0];

    % Any degenerate edges?
    del = sqrt(sum(nrmls.^2,2));
    degenflag = (del<(max(del)*10*eps));
    if sum(degenflag)>0
      warning('inhull:degeneracy',[num2str(sum(degenflag)), ...
        ' degenerate edges identified in the convex hull'])

      % we need to delete those degenerate normal vectors
      nrmls(degenflag,:) = [];
      nt = size(nrmls,1);
    end
  case 3
    % use vectorized cross product for 3-d
    ab = xyz(tess(:,1),:) - xyz(tess(:,2),:);
    ac = xyz(tess(:,1),:) - xyz(tess(:,3),:);
    nrmls = cross(ab,ac,2);
    degenflag = repmat(false,nt,1);
  otherwise
    % slightly more work in higher dimensions,
    nrmls = zeros(nt,p);
    degenflag = repmat(false,nt,1);
    for i = 1:nt
      % just in case of a degeneracy
      nullsp = null(xyz(tess(i,2:end),:) - repmat(xyz(tess(i,1),:),p-1,1))';
      if size(nullsp,1)>1
        degenflag(i) = true;
        nrmls(i,:) = NaN;
      else
        nrmls(i,:) = nullsp;
      end
    end
    if sum(degenflag)>0
      warning('inhull:degeneracy',[num2str(sum(degenflag)), ...
        ' degenerate simplexes identified in the convex hull'])

      % we need to delete those degenerate normal vectors
      nrmls(degenflag,:) = [];
      nt = size(nrmls,1);
    end
end

% scale normal vectors to unit length
nrmllen = sqrt(sum(nrmls.^2,2));
nrmls = nrmls.*repmat(1./nrmllen,1,p);

% center point in the hull
center = mean(xyz,1);

% any point in the plane of each simplex in the convex hull
a = xyz(tess(~degenflag,1),:);

% ensure the normals are pointing inwards
dp = sum((repmat(center,nt,1) - a).*nrmls,2);
k = dp<0;
nrmls(k,:) = -nrmls(k,:);

% We want to test if:  dot((x - a),N) >= 0
% If so for all faces of the hull, then x is inside
% the hull. Change this to dot(x,N) >= dot(a,N)
aN = sum(nrmls.*a,2);
```

```
% test, be careful in case there are many points
in = repmat(false,n,1);

% if n is too large, we need to worry about the
% dot product grabbing huge chunks of memory.
memblock = 1e6;
blocks = max(1,floor(n/(memblock/nt)));
aNr = repmat(aN,1,length(1:blocks:n));
for i = 1:blocks
  j = i:blocks:n;
  if size(aNr,2) ~= length(j),
    aNr = repmat(aN,1,length(j));
  end
  in(j) = all((nrmls*testpts(j,:)' - aNr) >= -tol,1)';
end




%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%Animation Tool
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%Latitude, Longitude, Altitude of airports in NY area
JFK= [40.6397, -73.7789,0];
LGA= [40.7772, -73.8726,0];
TEB= [40.8501,-74.0608,0];
EWR= [40.6925,-74.1687,0];
HPN= [41.0669444,-73.7075556,0];
ISP= [40.79525,-73.1002222,0];
SWF= [41.5041111,-74.1048333,0];

%Open the encounters list WHICH IS THE OUTPUT OF THE WAKE ENCOUNTER MODEL
% fid5 = fopen ('N90_20080319_EDR002_BVF01_CS125_rev.txt','r');
fid5 = fopen ('N90_20080319_EDR0001_BVF0005_CS175_rev.txt','r');
% fid5 = fopen ('N90_20080319_EDR0001_BVF0005_CS125_rev.txt','r');
% fid5 = fopen ('N90_20080319_EDR0001_BVF0005_CS125_rev.txt','r');
% fid5 = fopen ('N90_20080319_EDR0001_BVF0005_CS125_2rev.txt','r');
%fid5 = fopen ('N90_20080319_EDR0001_BVF0005_CS125_1rev.txt','r');
acft_pairs=textscan(fid5,'%f%f%f%f%f%f%f%f%f%f%f%f', 'delimiter',',');
% Field 1: recordnumleader
% Field 2: wakelongitude
% Field 3: wakelatitude
% Field 4: leadaltitude
% Field 5: wakegroundspeed
% Field 6: wakeheading
% Field 7: recordnumfollower
% Field 8: aircraftposition(1,1)
% Field 9: aircraftposition(1,2)
% Field 10: aircraftposition(1,3)
% Field 11: followheading
% Field 12: wakepiercing

all_leading_id = acft_pairs{1};
```

```
all_folowing_id= acft_pairs{7};
wakepiercing=  acft_pairs{12};
fclose all;


%%

TimeStep= 10; %this can be changed by user
%It's the time step for the common times of the two pairs

%inspect output file and extract the indeces where piercing occurs for lead
%and following acft
indeces_where_piercing_occurs= find (wakepiercing==1);
leading_id  = all_leading_id(indeces_where_piercing_occurs);
folowing_id = all_folowing_id(indeces_where_piercing_occurs);

wakerecordlength = size(leading_id,1);
%length of available records from the wake file. Multiple records per individual aircraft. Will be used
unique_leading_aircraft= unique(leading_id);
unique_following_aircraft= unique(folowing_id);
Total_number_of_Encounters= unique(leading_id);

%Need to extract PDARS information for all pairs--

 for i= 1: size(unique_leading_aircraft,1)
index_for_WEM(i)= find(unique_leading_aircraft(i)==filteredleadrecord &
unique_following_aircraft(i)==filteredfollowerrecord);



leading_index_to_use_for_PDARS_data= find(unique_leading_aircraft(i)==recordnum);

index_to_find_leading_acType= find(unique_leading_aircraft(i)==flightrecs);
leading_acType= actypes(index_to_find_leading_acType);
leading_acType_rev= cell2mat(leading_acType);

index_to_find_following_acType= find(unique_following_aircraft(i)==flightrecs);
following_acType= actypes(index_to_find_following_acType);
following_acType_rev= cell2mat(following_acType);

 %PDARS info for leading aircraft
leading_recordnum   = recordnum (leading_index_to_use_for_PDARS_data);
leading_time        = time(leading_index_to_use_for_PDARS_data);
leading_latitude    = latitude (leading_index_to_use_for_PDARS_data);
leading_longitude   = longitude(leading_index_to_use_for_PDARS_data);
leading_altitude    = altitude (leading_index_to_use_for_PDARS_data);
leading_groundspeed = groundspeed(leading_index_to_use_for_PDARS_data);
leading_heading     = heading(leading_index_to_use_for_PDARS_data);

following_index_to_use_for_PDARS_data= find(unique_following_aircraft (i)==recordnum);
%PDARS info for following aircraft
following_recordnum   = recordnum (following_index_to_use_for_PDARS_data);
following_time        = time(following_index_to_use_for_PDARS_data);
following_latitude    = latitude (following_index_to_use_for_PDARS_data);
following_longitude   = longitude(following_index_to_use_for_PDARS_data);
following_altitude    = altitude (following_index_to_use_for_PDARS_data);
```

```
following_groundspeed = groundspeed(following_index_to_use_for_PDARS_data);
following_heading     = heading(following_index_to_use_for_PDARS_data);

min_of_leading_time=min(leading_time);
max_of_leading_time= max(leading_time);
leading_time_range= [min_of_leading_time,max_of_leading_time];

min_of_following_time= min(following_time);
max_of_following_time= max(following_time);
following_time_range= [min_of_following_time,max_of_following_time];

%find the time instances where the two flights--leading and following overlap
common_starting_time = max(min_of_leading_time,min_of_following_time);
%common starting time is the maximum of the two minimum times
common_ending_time  = min(max_of_leading_time,max_of_following_time);
%common ending time is the minimum of the two maximum times
%subtract the max_of_both_min_times from all times to normalize the time values
%to start from 0

all_normalized_times_leading   = leading_time - common_starting_time;
all_normalized_times_following = following_time - common_starting_time;
normalized_ending_time= floor(common_ending_time- common_starting_time);%Round down


range_of_common_times= 0:TimeStep:normalized_ending_time;

%interpolate to find PDARS info at common times for leading and following
common_leading_recordnum   =
interp1(all_normalized_times_leading,leading_recordnum,range_of_common_times);
common_leading_latitudes   = interp1(all_normalized_times_leading,leading_latitude,range_of_common_times);
common_leading_longitudes  = interp1(all_normalized_times_leading,leading_longitude,range_of_common_times);
common_leading_altitudes   = interp1(all_normalized_times_leading,leading_altitude,range_of_common_times);
common_leading_groundspeeds=interp1(all_normalized_times_leading,leading_groundspeed,range_of_common_ti
mes);
common_leading_headings     =interp1(all_normalized_times_leading,leading_heading,range_of_common_times);

common_following_recordnum   =
interp1(all_normalized_times_following,following_recordnum,range_of_common_times);
common_following_latitudes   =
interp1(all_normalized_times_following,following_latitude,range_of_common_times);
common_following_longitudes
=interp1(all_normalized_times_following,following_longitude,range_of_common_times);
common_following_altitudes
=interp1(all_normalized_times_following,following_altitude,range_of_common_times);
common_following_groundspeeds=interp1(all_normalized_times_following,following_groundspeed,range_of_com
mon_times);
common_following_headings
=interp1(all_normalized_times_following,following_heading,range_of_common_times);

%compute the distance between wake pairs at each time step in nm
% distance_deg=
distance(common_leading_latitudes',common_leading_longitudes',common_following_latitudes',common_followin
g_longitudes');
% distance_nm=deg2nm(distance_deg);
% distance_string=num2str(distance_nm);
```

```
%Find OD of each aicraft
leading_index_to_use_for_OD= find(unique_leading_aircraft(i)==flightrecs);
following_index_to_use_for_OD= find(unique_following_aircraft(i)==flightrecs);
Origin_leader=deptarpt (leading_index_to_use_for_OD);
Destination_leader= arrvarpt (leading_index_to_use_for_OD);
Origin_following=deptarpt (following_index_to_use_for_OD);
Destination_following= arrvarpt (following_index_to_use_for_OD);


%Store results in a struct array
wake_pair(i).leader_info=
[common_leading_recordnum',range_of_common_times',common_leading_latitudes',common_leading_longitudes',
common_leading_altitudes',common_leading_groundspeeds',common_leading_headings'];
wake_pair(i).follower_info=
[common_following_recordnum',range_of_common_times',common_following_latitudes',common_following_longi
tudes',common_following_altitudes',common_following_groundspeeds',common_following_headings'];
% wake_pair(i).distance= deg2nm(distance_deg);
wake_pair(i).acTypes= [leading_acType_rev, following_acType_rev];
wake_pair(i).leader_info.OD= [Origin_leader,Destination_leader];
wake_pair(i).follower_info.OD= [Origin_following,Destination_following];

%%
%PLOTS
%# 1 plot time versus distance and 2.5, 4,5 and 6 nm separations LINES
vector_size=length(range_of_common_times);
one_matrix= ones(vector_size,1);

% figure
% hold on
% plot(range_of_common_times',2.5*one_matrix,'m--','LineWidth',2)
% plot(range_of_common_times',4*one_matrix,'r--','LineWidth',2)
% plot(range_of_common_times',5*one_matrix,'g--','LineWidth',2)
% plot(range_of_common_times',6*one_matrix,'k--','LineWidth',2)
%
% plot(range_of_common_times',distance_nm)
% legend('2.5 nm','4 nm','5 nm', '6 nm')
% xlabel('Time (sec)','FontSize',30)
% ylabel('Separation (nm)','FontSize',30)
Title1 = sprintf('%s - %s ',leading_acType_rev,following_acType_rev);
leaderOD= cell2mat(wake_pair(i).leader_info.OD);
 followerOD= cell2mat(wake_pair(i).follower_info.OD);
 Title2 = sprintf('%s - %s , %s  - %s' ,leaderOD,followerOD);
% title({Title1,Title2},'Fontsize',30)
% grid on

%
%% To see animation use script below


%Create NEW figure
figure1 = figure;
% Create axes
axes1 = axes('Parent',figure1);
view(axes1,[36 14]); %set view of animations
hold(axes1,'all');
Title = sprintf('%s - %s ',leading_acType_rev,following_acType_rev);
```

```
xlabel('Latitude (deg)','Fontsize',20);
ylabel('Longitude (deg)','Fontsize',20);
zlabel('Altitude (100*ft)','Fontsize',20);
% legend('Leader', 'Follower')
% title(Title,'Fontsize',20)
title({Title,Title2},'Fontsize',20)
grid on

%plot locations of airport
%MUST HAVE THE RIGHT AIRPORT NAME TO PLOT THE RIGHT AREA
plot3(JFK(1), JFK(2),JFK(3),'m^','MarkerFaceColor','m','MarkerSize',10)
text(JFK(1)+.01,JFK(2)+.01,0,'JFK','FontSize',16)
hold on
plot3(LGA(1), LGA(2),LGA(3),'g^','MarkerFaceColor','g','MarkerSize',10)
text(LGA(1)+.01,LGA(2)+.01,0,'LGA','FontSize',16)

plot3(TEB(1), TEB(2),TEB(3),'c^','MarkerFaceColor','c','MarkerSize',10)
text(TEB(1)+.01,TEB(2)+.01,0,'TEB','FontSize',16)

plot3(EWR(1), EWR(2),EWR(3),'k^','MarkerFaceColor','k','MarkerSize',10)
text(EWR(1)+.01,EWR(2)+.01,0,'EWR','FontSize',16)

plot3(HPN(1), HPN(2),HPN(3),'k^','MarkerFaceColor','k','MarkerSize',8)
text(HPN(1)+.01,HPN(2)+.01,0,'HPN','FontSize',14)
plot3(ISP(1), ISP(2),ISP(3),'k^','MarkerFaceColor','k','MarkerSize',8)
text(ISP(1)+.01,ISP(2)+.01,0,'ISP','FontSize',14)
plot3(SWF(1), SWF(2),SWF(3),'k^','MarkerFaceColor','k','MarkerSize',8)
text(SWF(1)+.01,SWF(2)+.01,0,'SWF','FontSize',14)

%Start the animation FROM TIME 0 TO THE LAST COMMON TIME
for j=1 :size(range_of_common_times,2)

plot3(common_leading_latitudes(j)',common_leading_longitudes(j)',common_leading_altitudes(j)','bo',common_foll
owing_latitudes(j)',common_following_longitudes(j)',common_following_altitudes(j)','r*')
%If you want to display the distance uncomment next 3 lines. set to display
%if distance is less than 5nm
%     if distance_nm(j)<5
%
text(common_leading_latitudes(j)+.01,common_leading_longitudes(j)+0.01,common_leading_altitudes(j)'+0.1,num
2str(distance_nm(j)))
%     end
pause (0.1)
legend('JFK','LGA', 'TEB', 'EWR','HPN','ISP','SWF','Leader', 'Follower')
% view (0,90)
saveas (gcf, [ Title], 'jpg');

end %for j=1: size(range_of_common_times,2)

 end %for i= 1:size(unique_leading_aircraft,1)
```