Searching for a Lost Plane: A Probabilistic, Neighborhood-Based Model for Locating Transoceanic Flights

Abstract

This paper proposes a model which determines the most probable locations of a missing transoceanic flight traveling from point A to point B. Since transoceanic flights are not detected by radar when they are more than 200 miles from a coast, radio communication and the occasional satellite imagery is essential to the construction of an effective search plan. Our search plan is built off the last distance in miles on the intended flight path (IFP) where Air Traffic Control heard from the pilot. We create a quadratic distance to casualty probability density function to predict the distance from the point of last contact to where the plane likely fell. Through trajectories and the possibility of veering off the IFP, a two-dimensional search region is determined and discretized into smaller areas. The trajectories account for any aircraft model, as parameters such as cruising altitude and Glide Ratio are assessed. Probabilities of containment (POC) are then assigned to each of these search areas, determining the probability the lost plane is within the given region.

We next consider the probabilities of detection (POD) during the search. We apply Koopman's equation for the probability of detecting a missing boat. Furthermore, Koopman's equation provides flexibility to input appropriate values pertaining to a given model of search plane and its instrumentation. Upon determining POD, we create an algorithm to effectively search the bounded area. Our algorithm applies the POD to the POC's and redistributes probabilities if the search plane is not found in a region while still considering the chance that the plane was actually in the region searched. If the plane is not found in a given cell, the algorithm outputs a neighboring region to search next. This customized neighborhood search plan maximizes resources while remaining time-efficient. An example of missing Transatlantic Flight ABCD is included as a demonstration of our methodology.

Our mathematically-derived search plan is built on a series of assumptions. These assumptions include that the IFP from point A to point B is linear and that the plane disassembled only upon impacting the water. Our model is sensitive to specific parameter changes but is therefore accommodating to different plane types. Strengths of the model include non-uniform probability distributions modeling distance to casualty and impact and a neighborhood-based, continuously-updating search mechanism.

Contents

1 Introduction

Since 1948, 83 aircrafts have vanished, some of which have still not been found (Bloomberg). Lost transoceanic flights are among the hardest flights to locate due to the lack of radar coverage. An average radius for radar coverage is approximately 200 miles in any direction (Hartmann). Because radar coverage is not an available method to track a transoceanic flight's activity, Air Traffic Control (ATC) relies on radio or messaging communication with a pilot on a regular, predetermined basis. If there are no location updates in the designated time frame, signals alert Air Traffic Control that there has been a loss of communication, and that there is a potentially missing flight.

In the past, Bayesian statistical methods have been used to track the whereabouts of missing flights. Air France Flight 447 went down in 2009 and was not found for over a year. Towards the end of the search, statisticians used Bayesian methods, narrowed the search plane, and picked up the blackbox's radio transmitter signal, locating the plane (Seidel). The available prior information, such as weather conditions on the flight path, floating bodies found soon-after, and the signal of the beacon eliminated search zones. On the other hand Malaysian Flight 370 contained little prior information, the search began later, no floating remnants were found, and individuals presumed the blackbox signal was turned off (Seidel). To this day, the location of flight MH370 remains unclear.

In order to confront the difficult situation in which a plane is lost and prior information is scarce, we set out to create a mathematical model and search plan. Our model accounts for an indefinite time until casualty and impact with the water. Potential deviations off the intended flight path, relevant to the Malaysian Flight, are also considered and time-efficiency is prioritized. We propose an effective search strategy in the case of a lost, signal-less transoceanic flight.

2 Model

Our model considers a lost flight, which starts at some location A and sets out to travel across the ocean linearly to its destination B. In order to develop an optimized search plan, we consider the probabilities of containment of the plane's whereabouts, the probabilities of detecting the plane while efficiently and conservatively using resources, and the continuously updating probabilities of the most likely location of the missing airplane.

2.1 Constructing Probabilities of Containment for a Missing Airplane and its Debris

The containment model maps the probability of the airplane's containment to specific regions surrounding the flight's last known position. Our first consideration in the containment model is estimating how far the plane most likely travels after a signal is lost. The variable y_0 is the last distance in miles from the starting location where air traffic controllers heard from the pilot. In addition to the mileage covered, it is important to take note of the initial latitude and longitude coordinates at y_0. It is assumed that the plane stayed on its Intended Flight Path (IFP) at this time. At this location, it is unclear if the plane continued traveling in the air or began falling to the ground. The Random Variable Y_c refers to the distance to casualty, or the distance on the IFP at which the plane started to malfunction. It is assumed that when a plane malfunctions, its descent for the water begins. The distance to casualty is measured from $y_0 = 0$. We have modeled the probability density function of Y_c as follows:

$$f_{y_c}(y_c) = \begin{cases} \frac{3}{(y_B - y_R)^3}(y_C - (y_B - y_R))^2 & : y_0 \leq y_C < y_B - y_R \\ 0 & : \text{elsewhere} \end{cases}$$

where y_B indicates the remaining distance to destination B from y_0 and y_R is the final mileage of the journey for which radar detection is available. Therefore, $y_B - y_R$ represents the flying distance left before Air-Traffic Control (ATC) radar detectors can spot the missing flight.

Our Random Variable Y_c is constructed under the assumption that it is more likely for the plane to reach a casualty-state immediately or soon after the last communication with ATC. This intuition is reasonable because a loss of communication often occurs due to abnormal plane operation. The probability continues to decrease quadratically and becomes 0 in the radar-detective zone. The parameter y_R can be determined using the equation below:

$$y_R = \sqrt{(200)^2 - h^2} = \sqrt{40,000 - h^2}$$

where h is the cruising altitude in miles of the jet. Radar detectors can track planes within a 200 mile radius, on average (Hartmann), so y_R can be determined by manipulating the Pythagorean Theorem.

We determine the possible locations where the airplane could have fallen based on a distribution of trajectories. The distance of impact, or the distance from y_0 at which the airplane hit the water is Y_I. This variable takes into account the glide of the airplane and the distance at which it is completely submerged and is modeled as follows:

$$Y_I = Y_C + Gh$$

where G is the glide ratio of the plane (NASA). Each airplane model has an average glide ratio which is defined as

$$G = \frac{\text{units of distance forward}}{\text{unit of distance downward}}.$$

The glide ratio is multiplied by the cruising altitude to determine the horizontal distance the plane traveled during its downward fall

(NASA). The probability density function of Y_I can therefore be modeled using the following transformation of variables technique:

$$\begin{aligned}
P(Y_I < y_I) &= P(Y_C + Gh < y_I) \\
&= P(Y_C < y_I - Gh) \\
&= \frac{3}{(y_B - y_R)^3} \int_0^{y_I - Gh} (y_C - (y_B - y_R))^2 dy_C \\
&= \frac{3}{(y_B - y_R)^3} [(y_C - (y_B - y_R))^2 dy_C] \Big|_0^{y_I - Gh} \\
&= \frac{3}{(y_B - y_R)^3} ([(y_I - Gh - y_B + y_R)^2][(0 - y_B + y_R)^2] \\
\implies & f_{y_I}(y_I) = \frac{d}{d_{y_I}} \frac{3}{(y_B - y_R)^3} ([(y_I - Gh - y_B + y_R)^2][(0 - y_B + y_R)^2]
\end{aligned}$$

After taking the derivative with respect to Y_I we can conclude that the probability density function for Y_I is:

$$f_{y_I}(y_I) = \begin{cases} \frac{3}{(y_B - y_R)^3} (y_I - Gh - (y_B - y_R))^2 & : Gh \leq y_I < y_B - y_R \\ 0 & : \text{elsewhere} \end{cases}$$

The probable distances from the point of lost contact to the point of impact are measured in line with the IFP. While this model has accounted for a single dimension, we can easily extend it to the left and to the right, indicating a plane's maneuver off course. The next portion of our model accounts for the possibility of veering off the intended linear path.

Veering away from the IFP is considered by assuming the segment of water representing the locations of impact are the average, or 0, and that the possible locations of the plane are normally distributed to the sides of y_I.

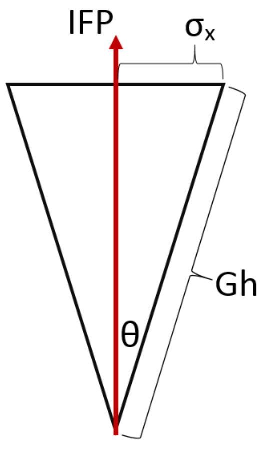


Figure 1: One standard deviation is calculated to be the distance from the IFP where a plane gliding $\frac{\pi}{8}$ off course would land.

Standard deviation of the normal probability distribution is determined by

$$\sigma_x = Gh \sin\left(\frac{\pi}{8}\right)$$

Since it is unlikely that a plane veers beyond $\frac{\pi}{8}$ from the IFP, $\frac{\pi}{8}$ will be used to find one standard deviation. Thus the normal probability distribution perpendicular to the IFP is:

$$f(x) = \frac{e^{\frac{-x^2}{2(Gh\sin(\frac{\pi}{8}))^2}}}{Gh\sin(\frac{\pi}{8})\sqrt{2\pi}} : -\infty < x < \infty$$

Because $f(x)$ and $f(y_I)$ are independent distributions, we can create a joint probability density function by multiplying $f(x)$ and $f(y_I)$ as follows: $f(x, y_I) = f(x)f(y_I)$.

$$f(x, y_I) = \begin{cases} \frac{3(y_I - Gh - (y_B - y_R))^2}{Gh\sin\frac{\pi}{8}\sqrt{2\pi}(y_B - y_R)^3} e^{\frac{-x^2}{2(Ghsin(\frac{\pi}{8}))^2}} & : -\infty < x < \infty, Gh \leq y_I < y_B - y_R \\ 0 & : \text{elsewhere} \end{cases}$$

2.1.1 Partitioning Search Region Through Model Discretization

Given $f(x, y_I)$, the search field can be divided into distinct regions. Each region is assigned a probability of containment (POC), indicating the probability that the region contains the missing plane and debris.

In order to discretize our model, we first consider X, the normal random variable. Since 99.7% of all observations are contained within 3 standard deviations of the mean, our search region is bounded on both sides by three standard deviations. We segment y_I over its entire range. Therefore, we consider $N-1$ partitions and N vertical regions through the following equation:

$$N = \left\lceil \frac{y_B - y_R - Gh}{t_{0.5} v_s} \right\rceil$$

where $t_{0.5}$ represents the quantity 0.5 hours and v_s is the velocity of the search plane.This guarantees that each search plane spends up to half an hour sweeping a length of each search region. Assuming more than one plane searches a given region at a time, these partitions lead to a highly efficient search duration.

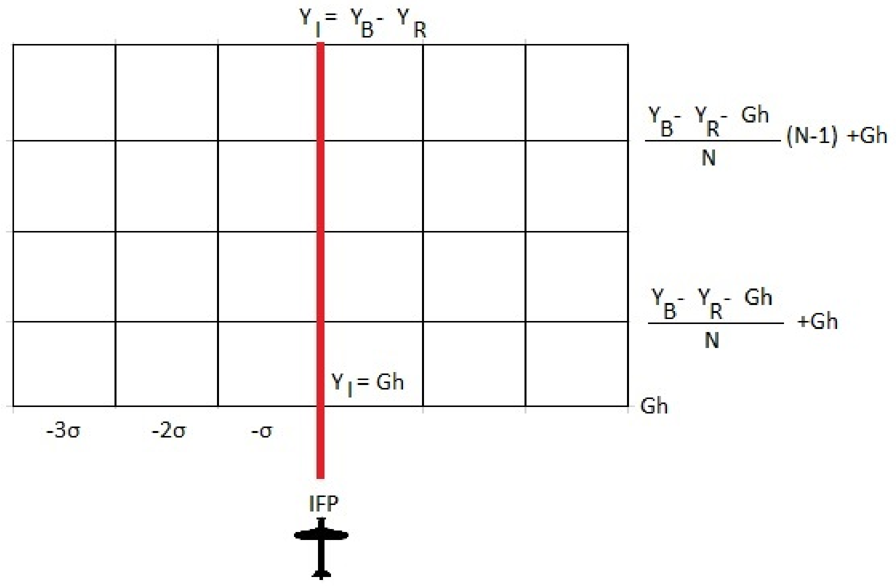


Figure 2: The discretization method is modeled by the above $SA_{N\times 6}$ search region matrix. Note that in this case $N = 4$.

To find the probabilities of containment for each of the above rectangular regions, the joint probability density function $f(x, y_I)$ must be integrated over each region's boundaries. In order to find $N \times 6$ probability values, a loop was created in Mathematica. The loop considers only the regions to the right of the IFP, since the normal distribution has the property of symmetry. The loop iterates through all N partitions in Y_I given each of the 3 set X regions. Below is the algorithm for the probability generation of each region within 1 standard deviation of the IFP:

$$\int_0^{Gh\sin(\frac{\pi}{8})}\int_{0(\frac{Y_B-Y_R-Gh}{N})+Gh}^{1(\frac{Y_B-Y_R-Gh}{N})+Gh} f(x,y_I)dxdy_I$$

$$\int_0^{Gh\sin(\frac{\pi}{8})}\int_{1(\frac{Y_B-Y_R-Gh}{N})+Gh}^{2(\frac{Y_B-Y_R-Gh}{N})+Gh} f(x,y_I)dxdy_I$$

$$\vdots$$

$$\int_0^{Gh\sin(\frac{\pi}{8})}\int_{(N-1)(\frac{Y_B-Y_R-Gh}{N})+Gh}^{N(\frac{Y_B-Y_R-Gh}{N})+Gh} f(x,y_I)dxdy_I$$

The output of this loop is a list of probability values that can be matched to each of the search regions. Refer to Appendix B for the Mathematica Code. Now that we have determined probabilities of containment for each of these ocean search regions, we must obtain the boundaries of each region in latitude and longitude coordinates to facilitate the search and rescue team's efforts.

2.1.2 Finding Latitude and Longitude Coordinates for Search Regions

Initially, the bounds of a given uniform probability region $r_{u\times l}$ can be represented as $0 \leq u \leq N$ and $-3 \leq l \leq 3$ where l, u are integer multiples of $n =$ region number along the IFP, σ_x respectively.

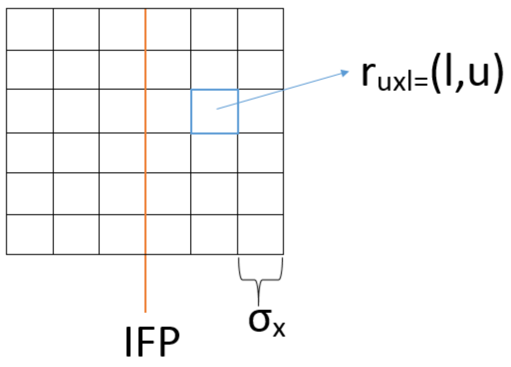


Figure 3: The location of a given $r_{u\times l} = (l, u)$ in reference to an IFP and σ_x

In order to reference a discretized probability region to the IFP and y_0, we must define a coordinate axis with a y-axis along the IFP, an x-axis in the direction of σ_x and an origin at $(y_I, 0\sigma_x)$. Now the bounds of a region $r_{u\times l}$ in miles are:

$$-l(Gh\sin(\frac{\pi}{8})) \leq x \leq l(Gh\sin(\frac{\pi}{8}))$$

and

$$0 \leq y \leq \frac{u(||\mathbf{y_B} - \mathbf{y_0}|| - Gh)}{N}.$$

To translate any given number m of zone-bounding coordinates in (x,y) to latitude and longitude, build a $2 \times m$ matrix

$$C = [c_1 c_2 c_3 ... c_m]$$

where each bounding coordinate $\mathbf{c_i}$ where $0 < i \leq m$ is of the form

$$\begin{bmatrix} c_x \\ c_y \end{bmatrix}.$$

Next, using a rotation matrix, vector addition and vector scaling will effectively shift an "IFP-referenced" coordinate to latitude and longitude. $\mathbf{y_0}$ and $\mathbf{y_B}$ are known in the "IFP-referenced" system and in traditional coordinates. The components of $\mathbf{y_B} - \mathbf{y_0}$ in latitude and longitude are designated r_e, r_n. To build a rotation matrix from the "IFP-referenced" system to latitude and longitude, we first must find the counter-clockwise angle ϕ between the two coordinate systems as follows:

$$\phi = \arctan(\frac{r_e}{r_n}).$$

Using ϕ, the rotation matrix R is either:

$$R = \begin{bmatrix} \cos\phi & \sin\phi \\ -\sin\phi & \cos\phi \end{bmatrix} : \frac{\pi}{2} \leq \phi < 0$$

or

$$R = \begin{bmatrix} \cos\phi & -\sin\phi \\ \sin\phi & \cos\phi \end{bmatrix} : 0 \leq \phi \leq \frac{\pi}{2}.$$

One way to test whether the rotation matrix is calculated correctly is to verify that $\det(R) = 1$. After finding a suitable R, multiply C by R to express every $\mathbf{c}$ in C in miles of latitude and longitude with reference to $\mathbf{y_0}\rho$ where ρ is the average distance in miles of one degree North and one degree East from $\mathbf{y_0}$. Finally, add the resulting vector to $\mathbf{y_0}$ to have coordinates in degrees North and East. In summary, changing C in the "IFP-referenced" system to the corresponding coordinate matrix C_L in latitude and longitude amounts to:

$$C_L = \rho RC + \mathbf{y_0}.$$

2.2 Probability of Detection

A standard model for finding the Probability of Detection (POD) is the exponential function

$$POD = 1 - e^{-\text{coverage}}$$

where coverage is the area effectively swept divided by the entire search area A_S (Frost 1999). To determine our region's POD, we use a slight variation of B.O. Koopman's Exponential Model. The search area is defined as

$$A_S = Gh\sin(\frac{\pi}{8})(\frac{y_B - y_R - Gh}{N}).$$

where the A_S represents the area of a single partitioned region of the overall search region.The area effectively swept is dependent on the effective sweep width S and the effort. The effective sweep width depends on the effective detection distance $\kappa\epsilon$ of the instrument (eg. human eyes, radar, etc.) used for the search, and the elevation of the cruising search plane is h_s. We modify Koopman's model to account for variations in weather, conditions, search abilities and more. This variation is denoted by the scalar κ, which takes values between 0 and 1. The distance from the search plane's path projected on the ocean to the furthest possible detected object is $\sqrt{(\kappa\epsilon)^2 - h_s^2}$. Assuming the search plane flies in parallel swaths, the effective sweep width would be

$$S = 2\sqrt{(\kappa\epsilon)^2 - h_s^2}$$

in order to detect all of A_S. The effort is said to be $d_s n$ where the distance searched by an individual aircraft per swath is d_s and the number of search planes included in the search is n. Assuming the search plane flies parallel to the IFP, the number of swaths the plane must fly is the number of subregions N so d_s is

$$d_s = (\frac{y_b - y_r - Gh}{N}).$$

In order to search simultaneously and most efficiently, the planes are assumed to be flying at the same velocity. Thus d_s is the same for each plane since they begin searching simultaneously. Incorporating each of these components into a single model yields the following POD equation:

$$POD = 1 - e^{-2\frac{d_s n\sqrt{(\kappa\epsilon)^2 - h_s^2}}{Gh\sin\frac{\pi}{8}\frac{(y_B - y_R - Gh)}{N}}}$$

Unless search effort is changed based on the search region, it is assumed that the POD is uniform among all regions of probable containment.

2.3 The POC-Updating Algorithm

If there were no such thing as instrumentation error or human error, POD would equal 1 and thus not adjust the POC's of discretized zones. Since search conditions are nowhere near perfect, we assign $1 - POD_q$ to be the probability that the lost plane in region q went undetected. Given that we do not find the plane after searching the zone q, we update the POC in q to be

$$POC_{q'} = (POC_q)(1 - POD_q).$$

This is the probability that the plane is actually in the searched region, even though it was not detected. Assuming that net probability of containment of all discretized zones is constant, the probability lost when POC_q becomes $POC_{q'}$ must be accounted for in the other regions. To do this, we add

$$\frac{POC_q - POC_{q'}}{6N - 1}$$

to each of the POC values except $POC_{q'}$. This process can be iterated during a search to update all the POC values per searched discrete probability zone.

2.3.1 Using the POC-Updating Algorithm to Determine Search Pattern

To determine the best order to search the $6 \times N$ different regions, we use the POC-updating algorithm described above using gradients to choose the next search region. By the "best" order we mean the most efficient in terms of resources and time. We want to spend the least amount of time and fuel covering the entire map. Thus we want to minimize flight between nonadjacent regions because that increases less-useful or redundant coverage and increases time needed to cover the entire area. The search plan algorithm involves updating the POC map after completing the search of one region to find the new probability of containment map, given the downed aircraft was not detected in the recently-searched region. Using a matrix in Mathematica we represent the map of POC's. A loop was created in Mathematica to evaluate, for each iteration, the differences between the updated POC's of the just-searched region k and its adjacent neighbors. We take the minimum of the differences, and then locate the row n_{new} and column σ_{new} of that minimum element, to find the next region $r_{n_{new},\sigma_{new}}$ to be searched. Essentially this method consists of updating the POC matrix, calculating a gradient/difference to identify the highest adjacent POC, and taking that region as the next to be searched.

2.4 Assumptions in the Methodology

There are several assumptions upon which our model and the resulting search plan are built. Below is a list of assumptions we consider:

- The lost plane's pre-established course is a linear path from point A to point B.
- The lost plane's glide affects its trajectory into the ocean. The plane does not explode in the sky.
- Any debris disassembles upon impacting the water.
- The plane will not veer more than $3Ghsin(\frac{\pi}{8})$ off the IFP.
- There are no forces (eg. currents) to move the debris or plane from where they first landed.
- All planes (lost plane and search plane(s)) in the model are flying at a constant velocity.
- The lost plane is more likely to be found near the last point of communication.
- The lost plane remains afloat (detectable) during the entire time the search is in action.
- Every searching plane has the same instrumentation, searches at the same speed and flies at the same altitude.

2.5 Sensitivity Analysis

In order to test how changing parameters effect our model, we include a discussion of the relationships between specific parameters and their effects on the components of our model and search plan.

We assess the impacts of changing parameters on $f(x, y_I)$ by considering $f(G)$ and $f(\theta)$. From experimentation with fixing the random variables x and y_I, the shape of the curve of $f(G)$, and $f(\theta)$ are strikingly similar.

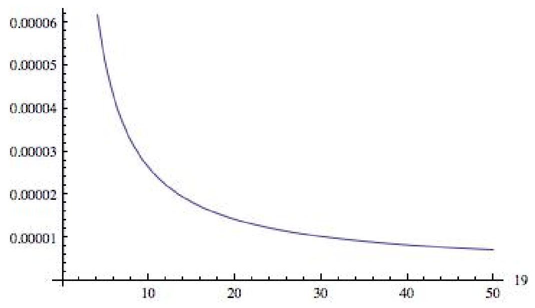


Figure 4: $f(G)$ where $(x, y_I) = (0.5, 0.5)$ exemplifies the curvature of $f(G)$ and $f(\theta)$ regardless of the x and y_I values

As G and θ increase, the respective $f(G)$ and $f(\theta)$ exhibit a rapid decrease. Since the bounded integral of $f(x, y)$ produces a POC, G and θ have the same effect on POC as they do on $f(x, y)$. In short, as $G + \theta$ decreases, the probability of finding the lost plane increases.

Increasing θ will directly increase the search area because the X distance is measured in mile-multiples of $\sigma_x = Gh \sin(\theta)$ which is proportional to a chosen $0 \leq \theta \leq \frac{\pi}{2}$. The following graphs are generated using the constant parameter values found in the Working Example (Section 3).

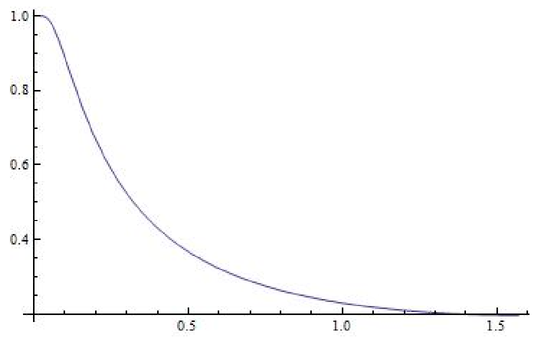


Figure 5: $POD(\theta)$ where θ is in radians

As θ increases, POD rapidly decreases from one to zero. The most intriguing part of $POD(\theta)$ is that the graph suggests $POD(0) \approx 1$. Also, the POD is extremely small for any plane that has veered more than ninety degrees off the IFP.

Increasing the glide ratio effects the POD similarly to increasing θ.

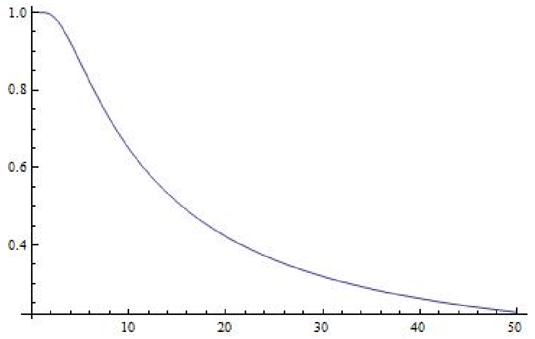


Figure 6: $POD(G)$

Increasing the exponent's denominator will have an inverse effect on the POD. Thus, a search party following the methodology explained in the Working Example will be more likely to find a plane with a small glide ratio.

Another variable that clearly effects the search plan is the cruising altitude, h_S, at which the search plane flies because the POD depends on $\sqrt{\kappa\epsilon^2 - h_s^2}$. Varying h_s within $0 \leq h_s < \epsilon$ produces an elliptic graph of POD centered around the origin:

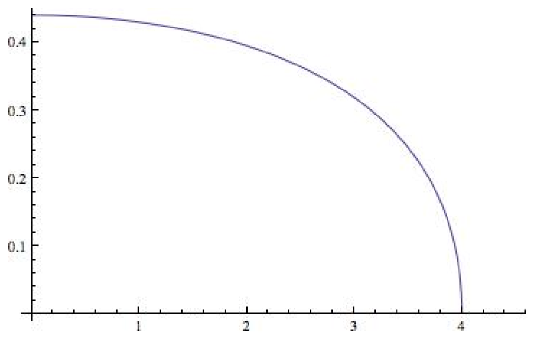


Figure 7: $POD(h_s)$ where $0 \leq h_s < \epsilon$

Figure 4 illustrates that when $\frac{h_s}{\epsilon}$ increases, the POD increases. This is a logical relationship because each swath will cover more ground which will in turn decrease the search effort. Thus, it is best to run the search at the lowest possible altitude in order to improve the chance of finding the lost plane.

The number of search planes n also effect the POD because the search effort n is in the numerator of the search effort fraction. Illustrated graphically:

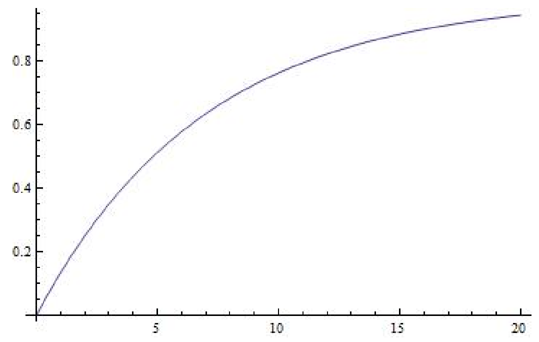


Figure 8: $POD(h_s)$ where $n \in \mathbb{N}$

In a practical sense, incorporating more search planes will increase the likelihood of detection.

Since the POD equation contains the element $\sqrt{(\kappa\epsilon)^2 - h_s^2}$ there may be a square root of a negative number if $h_s^2 > (\kappa\epsilon)^2$. A plane must fly low enough for the effective range of the instrumentation $(\kappa\epsilon)$ to reach the ground. Otherwise there is no chance the plane will detect anything below it and thus the POD would be effectively zero. Thus when $h_s^2 > (\kappa\epsilon)^2$ there is no point in searching until conditions improve.

Clearly, our model is sensitive to changes in several of the parameters and therefore may not be considered a robust probabilistic model. However, the sensitivity to changes in the input parameters is necessary to have a model that is accommodating to the specifications of different types of planes and search instruments.

2.6 Strengths and Weaknesses of Our Model

2.6.1 Strengths

Our outlined model has particular strengths, unavailable in other Search and Rescue methods. First, we use a quadratic probability density function rather than the commonly-used uniform distribution. In many Search and Rescue Methodologies, it is assumed that the time of casualty is uniformly distributed over all possible times in a given range (Lawrence 1977). Here, we carefully construct a decreasing, concave down distribution to account for the large likelihood that complications arise early after y_0. Our model also accounts for a flight trajectory to map the flight's motion at the time of casualty to a region of the ocean surface. The change of bases allows the searchers to locate the search regions using actual latitude, longitude coordinates rather than tracking angles and distances from the point of lost contact. The POD incorporates several factors, including an added constant k.

When $k = 1$, the search occurs in perfect conditions, while $k = 0.5$ may indicate some fog, choppy water, or other inhibiting factors. This constant helps to better explain the rate of coverage while still using Koopman's commonly-used Exponential Model.

Our search plan is a constantly updating algorithm keeping time efficiency in mind. If an item is not detected in a highly probable region, the POC's of the other search regions are updated. The cell with the next highest probability in the searched region's "neighborhood" is investigated. Implementing a neighborhood restriction speeds up the search process, wasting no travel time while still searching regions with high POC's. Our search plan can easily be generalized to search planes of different speeds and larger or smaller regions of the ocean.

2.6.2 Weaknesses

It is important to consider the weaknesses of our model that may be worth addressing in the future. First, the POC of the regions closest to Gh but more than 1 standard deviation away in the X direction are slightly over-allocated. The standard deviation was calculated using trigonometry, indicating that our rectangular probability region is based on a triangular calculation. Refer to Figure 1. Because of this over-allocation, there is a slight under-calculation of probabilities 1 or more standard deviations away from the IFP towards the latter partitions of Y_I.

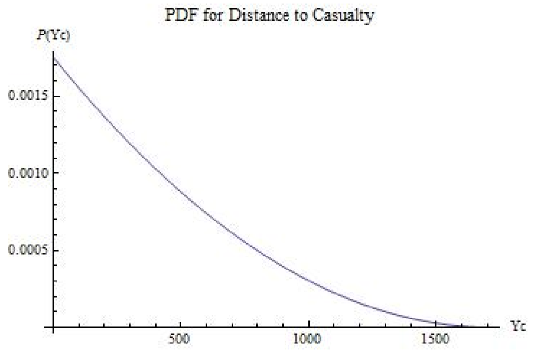
Another weakness to consider is the lack of consideration of currents in each of our partitioned search regions. With more time, we hope to extend the impacts of surface currents to locating debris and other remnants of the aircraft.

3 Working Example: Flight ABCD to Lisbon, Portugal Reported Missing

8 h 15 min
3,186 miles
Boston, MA
Lisbon

Imagine that on February 9, 2015 Flight ABCD from Boston, MA to Lisbon Portugal disappeared on a Boeing 777 aircraft. The point of last communication occurred at coordinates 40.9275° N, 46.2575° W. Let $y_B = 1,912$ miles. The plane was cruising at a speed of 560 miles per hour at an altitude of $40,000$ feet (7.58 miles), so $y_R = 199.86$. Assume that the flight is entirely over open ocean and the Lisbon airport is the nearest location to y_0 that can detect an airplane. So, y_R is measured from the Lisbon airport along the IFP. Below is the probability distribution of Y_C, the random variable representing the probability that at a given distance in the direction of the IFP the plane experiences a casualty that causes it to begin a downward trajectory.

$$f_{y_c}(y_c) = \begin{cases} \frac{3}{(1912-199.86)^3}(y_C - (1912-199.86))^2 & : 0 \leq y_C < 1712.14 \\ 0 & : \text{elsewhere} \end{cases}$$



This airplane has a glide ratio of approximately 19 : 1, similar to other large aircrafts. With this information, y_I can be modeled as follows:

$$f_{y_I}(y_I) = \begin{cases} \frac{3}{(1712.14)^3}(y_I - 1568.12)^2 & : 144.02 \leq y_I < 1712.14 \\ 0 & : \text{elsewhere} \end{cases}$$

After considering the distribution of distances of impact, we must now account for the possibility of veering off the IFP. The following X distribution, distributed in the direction perpendicular to the intended flight path, accounts for Flight ABCD's potential off-course location of impact:

$$f(x) = \frac{e^{\frac{-x^2}{2(144.02\sin(\frac{\pi}{8}))^2}}}{144.02\sin(\frac{\pi}{8})\sqrt{2\pi}} : -\infty < x < \infty$$

By multiplying the independent equations $f(x) * f(y_I)$, a joint probability density function is obtained.

$$f(x,y_I) = \begin{cases} \frac{3(y_I-1568.12)^2}{144.02\sin(\frac{\pi}{8})\sqrt{2\pi}(1712.14)^3} e^{\frac{-x^2}{2(144.02\sin(\frac{\pi}{8}))^2}} & : -\infty < x < \infty, 144.02 \leq y_I < 1712.14 \\ 0 & : \text{elsewhere} \end{cases}$$

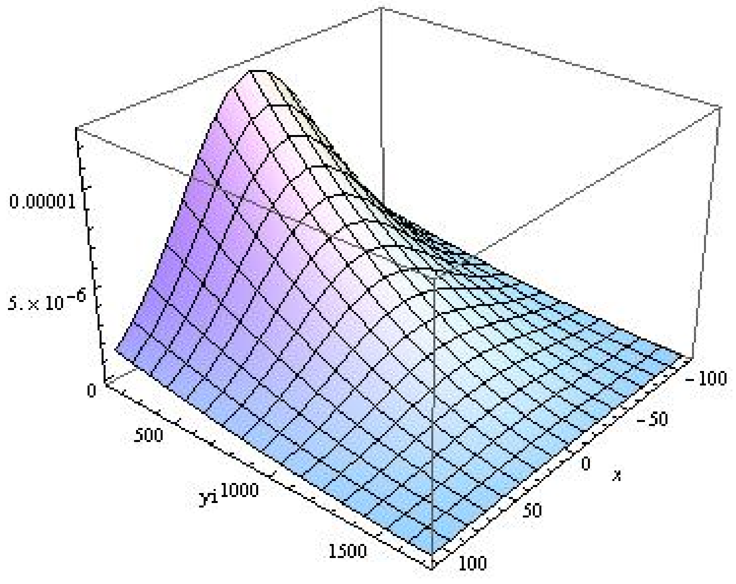


Figure 9: The joint density function is shown above, indicating the locations in which the POC is largest, and locations where the POC is smallest

Now, the multi-dimensional probability distribution, POC, can be discretized into 36 subregion each calculated by a double integral, as described in section 2.1.1 Partitioning Search Regions Through Model Discretization.

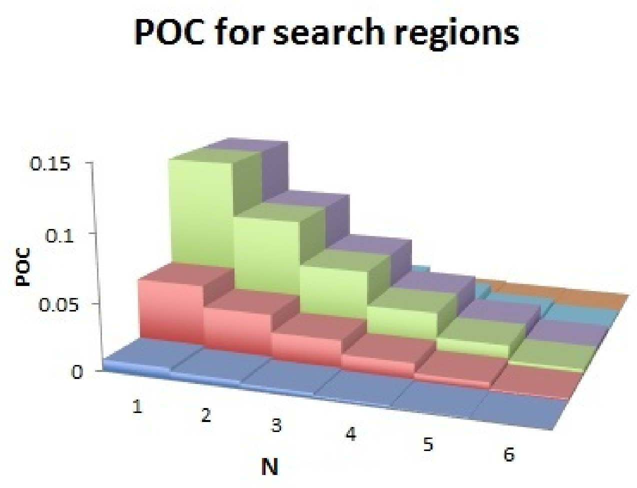
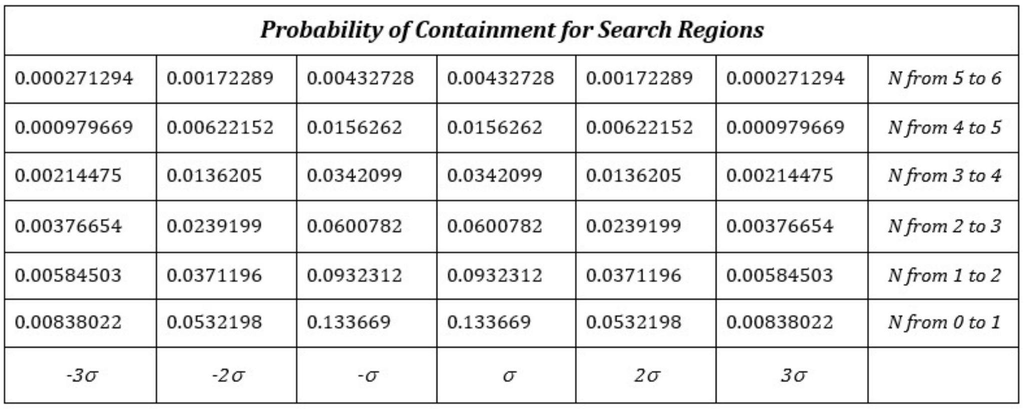


Figure 10: The graph above represents the discretized POC of Flight ABCD



Probability of Containment for Search Regions						
0.000271294	0.00172289	0.00432728	0.00432728	0.00172289	0.000271294	*N from 5 to 6*
0.000979669	0.00622152	0.0156262	0.0156262	0.00622152	0.000979669	*N from 4 to 5*
0.00214475	0.0136205	0.0342099	0.0342099	0.0136205	0.00214475	*N from 3 to 4*
0.00376654	0.0239199	0.0600782	0.0600782	0.0239199	0.00376654	*N from 2 to 3*
0.00584503	0.0371196	0.0932312	0.0932312	0.0371196	0.00584503	*N from 1 to 2*
0.00838022	0.0532198	0.133669	0.133669	0.0532198	0.00838022	*N from 0 to 1*
-3σ	*-2σ*	*-σ*	*σ*	*2σ*	*3σ*	

Figure 11: Map representation of POC allocated to each of the search regions

Next, the Probability of Detection is found by the formula as described in the section 2.2:

$$POD = 1 - e^{-\frac{2*d_s n\sqrt{(k\epsilon)^2 - h_s^2}}{Gh\sin\frac{\pi}{8}(y_B - y_R - Gh)/N}}$$

The list of parameters for this model, either estimated or researched values, is as follows:

- Remaining distance of flight from y_0 to y_R, $y_B = 1912$
- Glide constant for a large plane such as a a Boeing 777-200, $G = 19$

- Average altitude at which a commercial airliner cruises, $h = 7.58$
- $y_R = \sqrt{200^2 - h^2}$
- Velocity of the search planes (all of which are the same model in order to facilitate easier coordination and timing in the search), $v_s = 550$
- $t_{.05} = .5$ hours
- The number of partitions in the y direction, $N = \left\lceil \frac{y_B - y_R - Gh}{t_{.05} v_s} \right\rceil$
- Distance traveled by search planes in searching a single region, $d_s = \frac{(y_B - y_R - Gh)}{np}$
- Detection distance of instrument, $\epsilon = 4$
- Scales detection distance based on search conditions such as weather, $\kappa = 0.98$, note $< 0 \kappa < 1$. In this case, κ is near one, indicating clear calm search conditions.
- Number of search planes n, chosen in this example to be 8.
- Then, $POD = 0.677831$ for our example, when we plug in all the variables

Taking the values in Figure 11, we input them to Mathematica to form the initial POC matrix. We then utilize the method described in section 2.3 and subsection 2.3.1, and the Mathematica code in Appendix C, to determine the order and path of the most efficient search pattern. We also utilize the model to update probabilities that certain regions contain the lost plane given the plane was not detected when the an area was most recently searched.

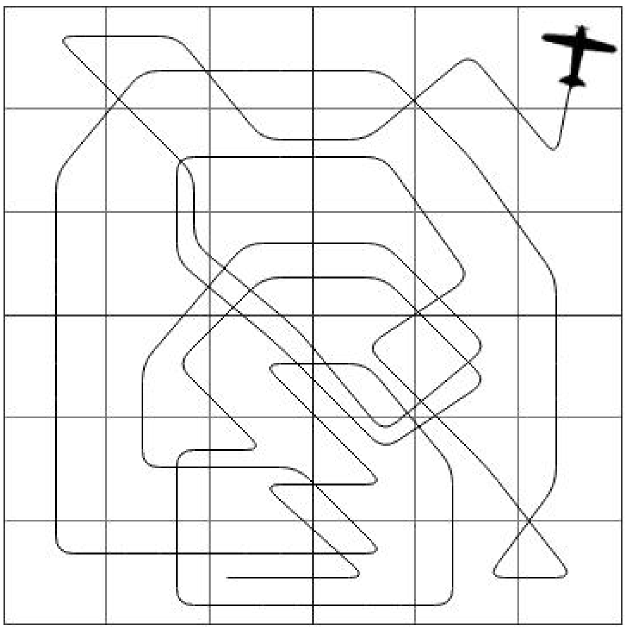


Figure 12: The diagram above shows the search path produced by our algorithm and described by the output of the Mathematica code in Appendix C.

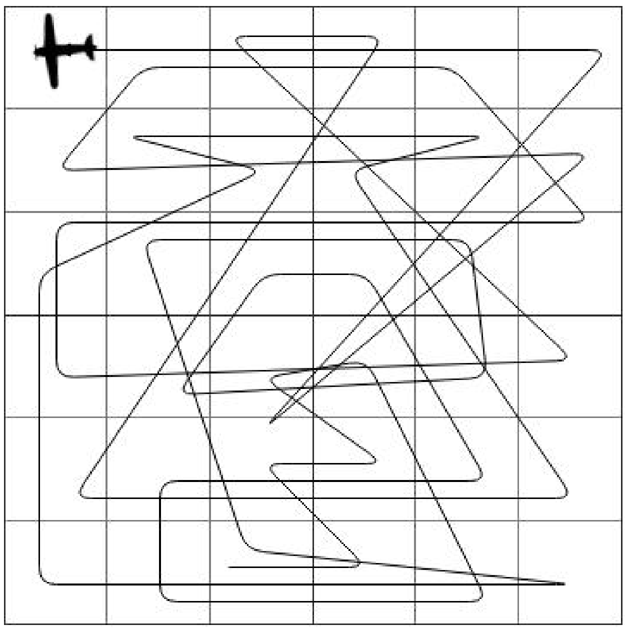


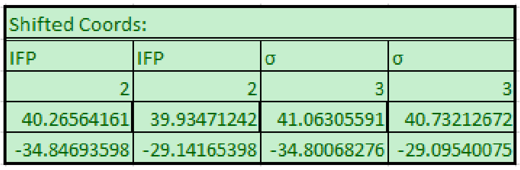
Figure 13: The diagram above shows the search path produced by an algorithm similar to ours but without requiring subsequent search regions to be adjacent. The code for this algorithm is contained in Appendix D.

By comparing the two figures above, we can observe some key differences. When adjacency is not a requirement in our algorithm, frequently the subsequent search region is the reflection of the previous due to the symmetry of the joint probability distribution across the IFP. This leads to much more back-and-forth movement, and as a result, more distance traveled, thus wasting time and fuel.

To search a certain region, that region must be described in Latitude and Longitude. Continuing with the example, say the region in in need of searching is the discrete probability zone σ, N from 2 to 3. The algorithm from Section 2.3 in action is as follows: First, the rotation matrix:

$$\begin{bmatrix} \cos\phi & \sin\phi \\ -\sin\phi & \cos\phi \end{bmatrix} = \begin{bmatrix} 0.9983 & -0.0579 \\ 0.0579 & 0.9983 \end{bmatrix}$$

is defined using the rotation angle $\phi = -0.057939$. It calculates the four latitude and longitude bounding coordinates of the rectangular search region σ, N from 2 to 3 as:



Shifted Coords:			
IFP	IFP	σ	σ
2	2	3	3
40.26564161	39.93471242	41.06305591	40.73212672
-34.84693598	-29.14165398	-34.80068276	-29.09540075

See Appendix B for the Excel tables used for the calculation.

4 Conclusion

The final results of our model present a valid, time-efficient search plan for locating missing flight ABCD from Boston to Portugal. Our methodology accounts for all sizes and types of airplanes as well as all types of search planes. Our sensitivity analysis stresses the importance of careful consideration of resources and the number of search planes. A greater number of search planes drastically increases the probability of detecting the missing plane. While some parameters are more robust than others, carefully choosing the number of search planes per square will optimize time-efficiency and overall detection. Our neighborhood-based search plan makes this model a strong choice when considering especially large search regions. While we did not have time, we hoped to incorporate the impacts of currents in the movement of airplane debris. We plan to further explore this topic to eventually extend our model to account for debris search plans too.

5 Summary for an Airline's Press Conference

There has been a new model proposed for locating missing transoceanic flights, such as Malaysian Flight 370. Since 1948, 83 aircrafts have vanished, some of which have still not been found (Bloomberg). In the case of Malaysian flight 370, not enough information was available on the plane's modified path, to effectively use the Search and Rescue models currently in place(Seidel). This new mathematical model narrows the search region of missing planes and predicts their location of impact with the water, considering deviations from their intended paths. The model incorporates a time-efficient search method, allocating time and effort to the most likely locations of a missing transoceanic flight traveling from point A to point B.

Since transoceanic flights are not detected by radar when they are more than 200 miles from a coast, radio communication and the occasional satellite imagery is important in tracking the off-the-radar plane. These mathematicians created a search plan which is based on the last distance in miles on the intended flight path (IFP) where Air Traffic Control heard from the pilot. The mathematicians assign probabilities to different search regions. The model accounts for the likelihood that the plane fell closer to the distance of last contact rather than farther away, since loss of contact often indicates a mechanical issue or takeover of the plane. They use physics, primarily modeling the gliding of planes from the sky, to determine a search area. The trajectories of the planes falling from the sky can be calculated for any aircraft model, as they determine these distances using measurements of the cruising altitude and the Glide Ratio of a given plane. Basically, after performing several calculations, Probabilities of containment (POC) are assigned to each of these search areas. The probabilities will tell searchers how likely it is that the missing plane (or its parts) lie in that region.

After mapping the likelihoods that the plane is in each of the regions, the mathematicians account for the probabilities that the search equipment will actually detect the plane. The simple equation for the probability of detection makes it easy to quickly input values like the speed of a search plane, a search plane's radius, the number of search planes and more. If this probability is not as high as the team would like, it can easily be increased by adding more search parties to the team.

Once the searchers know their probability of detecting the lost plane, the mathematicians use a new algorithm to search the bounded area. This algorithm applies the probability of detecting the plane to each rectangular region which has a certain chance of containing the plane. If a plane is not found in a given region, the algorithm determines an appropriate neighboring region to search, making the search plan very time efficient, and the process continues until the plane is located. There is still a chance that the plane was missed in a given region, so the search plan includes the possibility of revisiting certain areas later on. This customized neighborhood search plan maximizes resources while remaining time-efficient.

Of course we must use this search plan with a few restrictions in mind. If the path from destination A to destination B is not a straight line, this method becomes more difficult to use. The mathematicians note that while statistics are an incredibly powerful tool, there are always assumptions to consider, such as up to how far the plane could have veered from its intended path. This model is a breakthrough in Search and Rescue Theory and will be implemented when needed in the future.

References

[1] Benson, Tom. “Glide Angle.” Glide Angle. NASA, n.d. Web. 09 Feb. 2015.

[2] Frost, J. R. ”Principles of Search Theory.” *Sarinfo.bc.ca.* Soza and Company, Ltd., 1999. Web. 6 Feb. 2015.

[3] Hartmann, Christoph, and Sonya Angelica Diehn. “The Technologies behind the Search for Malaysia Airlines Flight 370.” *DW.de.* Deutsche Welle, 12 Mar. 2014. Web. 7 Feb. 2015.

[4] Stone, Lawrence D. ”Search Theory: A Mathematical Theory for Finding Lost Objects.” *Mathematics Magazine* 50.5 (1977): 248-56. Web. 9 Feb. 2015.

[5] Merrill, Dave, Chloe Whitaker, and Alex Tribou. ”Dozens of Planes Have Vanished in Post-WWII Era.” *Bloomberg.com.* Bloomberg, 20 Mar. 2014. Web. 9 Feb. 2015.

[6] Seidel, Jamie. ”Missing Malaysia Airlines Flight MH370: Search Goes Back to Basics Turning to the Power of Maths to Solve the Mystery.” *News.com.au.* News Limited, 4 June 2014. Web. 6 Feb. 2015.

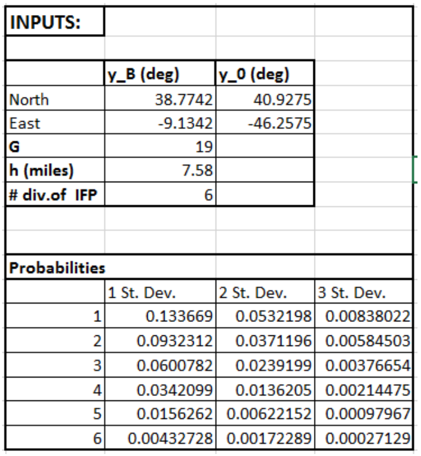
6 Appendix A

Mathematica Code for Probability Generation:

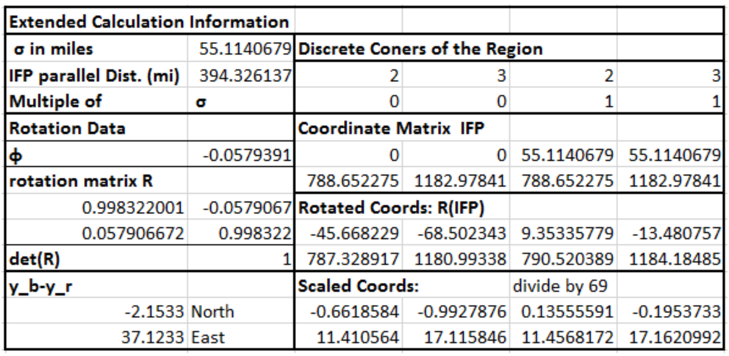
```
np = Ceiling[(yb - yr - G*h)/(t05*vs)]

For[n = 1, n < 7, n++,
 discretefxyi =
  Integrate[
   Integrate[
    f[x, yi], {yi, (n - 1)*(yb - yr - G*h)/np,
     n*(yb - yr - G*h)/np}], {x, 0, G*h*Sin[Pi/8]}];
 Print[discretefxyi // N]]

For[n = 1, n < 7, n++,
 discretefxyi =
  Integrate[
   Integrate[
    f[x, yi], {yi, (n - 1)*(yb - yr - G*h)/np,
     n*(yb - yr - G*h)/np}], {x, G*h*Sin[Pi/8], 2*G*h*Sin[Pi/8]}];
 Print[discretefxyi // N]]

For[n = 1, n < 7, n++,
 discretefxyi =
  Integrate[
   Integrate[
    f[x, yi], {yi, (n - 1)*(yb - yr - G*h)/np,
     n*(yb - yr - G*h)/np}], {x, 2*G*h*Sin[Pi/8], 3*G*h*Sin[Pi/8]}];
 Print[discretefxyi // N]]
```

Excel tables used to shift from IFP referenced search regions to Latitude and Longitude.

7 Appendix B



INPUTS:			
	y_B (deg)	y_0 (deg)	
North	38.7742	40.9275	
East	-9.1342	-46.2575	
G	19		
h (miles)	7.58		
# div.of IFP	6		
Probabilities			
	1 St. Dev.	2 St. Dev.	3 St. Dev.
1	0.133669	0.0532198	0.00838022
2	0.0932312	0.0371196	0.00584503
3	0.0600782	0.0239199	0.00376654
4	0.0342099	0.0136205	0.00214475
5	0.0156262	0.00622152	0.00097967
6	0.00432728	0.00172289	0.00027129



Extended Calculation Information					
σ in miles	55.1140679	Discrete Coners of the Region			
IFP parallel Dist. (mi)	394.326137	2	3	2	3
Multiple of	σ	0	0	1	1
Rotation Data		Coordinate Matrix IFP			
ϕ	-0.0579391	0	0	55.1140679	55.1140679
rotation matrix R		788.652275	1182.97841	788.652275	1182.97841
0.998322001	-0.0579067	Rotated Coords: R(IFP)			
0.057906672	0.998322	-45.668229	-68.502343	9.35335779	-13.480757
det(R)	1	787.328917	1180.99338	790.520389	1184.18485
y_b-y_r		Scaled Coords:		divide by 69	
-2.1533	North	-0.6618584	-0.9927876	0.13555591	-0.1953733
37.1233	East	11.410564	17.115846	11.4568172	17.1620992

8 Appendix C

Set up parameters and define POC matrix:

```
yb = 1912;
G = 19;
h = 7.58;
yr = Sqrt[200^2 - h^2];
t05 = .5;
vs = 550;
ds = (yb - yr - G*h)/np;
epsilon = 4;
kappa = .98;
hs = 2000/5280;
np = Ceiling[(yb - yr - G*h)/(t05*vs)];
sp = 8;

POCn1s1 = 0.13367; POCn2s1 = 0.09323; POCn3s1 = 0.06008; POCn4s1 = 0.03421; POCn5

POCn1s2 = 0.05322; POCn2s2 = 0.03712; POCn3s2 = 0.02392; POCn4s2 = 0.01362; POCn5

POCn1s3 = 0.00838; POCn2s3 = 0.00585; POCn3s3 = 0.00377; POCn4s3 = 0.00214; POCn5

POCn1negs1 = 0.13367; POCn2negs1 = 0.09323; POCn3negs1 = 0.06008; POCn4negs1 = 0.

POCn1negs2 = 0.05322; POCn2negs2 = 0.03712; POCn3negs2 = 0.02392; POCn4negs2 = 0.

POCn1negs3 = 0.00838; POCn2negs3 = 0.00585; POCn3negs3 = 0.00377; POCn4negs3 = 0.

POC = {{0, 0, 0, 0, 0, 0, 0, 0}, {0, POCn1negs3, POCn1negs2,
    POCn1negs1, POCn1s1, POCn1s2, POCn1s3, 0}, {0, POCn2negs3,
    POCn2negs2, POCn2negs1, POCn2s1, POCn2s2, POCn2s3, 0}, {0,
    POCn3negs3, POCn3negs2, POCn3negs1, POCn3s1, POCn3s2, POCn3s3,
    0}, {0, POCn4negs3, POCn4negs2, POCn4negs1, POCn4s1, POCn4s2,
    POCn4s3, 0}, {0, POCn5negs3, POCn5negs2, POCn5negs1, POCn5s1,
    POCn5s2, POCn5s3, 0}, {0, POCn6negs3, POCn6negs2, POCn6negs1,
    POCn6s1, POCn6s2, POCn6s3, 0}, {0, 0, 0, 0, 0, 0, 0, 0}};

POD = 1 - Exp[-((2*sp*ds*Sqrt[(k*epsilon)^2 - hs^2])/(
     G*h*Sin[Pi/8] (yb - yr - G*h)/np))]
```

Loop for determining the order in which we will search regions without the requirement of adjacency; build loop with initial conditions as the highest or one of the highest initial POC regions:

```
n = 2;
s = 4;
i = 1;
POC = {{0, 0, 0, 0, 0, 0, 0, 0}, {0, POCn1negs3, POCn1negs2,
    POCn1negs1, POCn1s1, POCn1s2, POCn1s3, 0}, {0, POCn2negs3,
    POCn2negs2, POCn2negs1, POCn2s1, POCn2s2, POCn2s3, 0}, {0,
    POCn3negs3, POCn3negs2, POCn3negs1, POCn3s1, POCn3s2,
    POCn3s3,0}, {0, POCn4negs3, POCn4negs2, POCn4negs1,
    POCn4s1,POCn4s2, POCn4s3, 0}, {0, POCn5negs3,
    POCn5negs2,POCn5negs1, POCn5s1,POCn5s2, POCn5s3, 0},
    {0,POCn6negs3, POCn6negs2, POCn6negs1,
    POCn6s1,POCn6s2,POCn6s3, 0}, {0, 0, 0, 0, 0, 0, 0, 0}};



While[1 < n < 7; 1 < s < 7; 0 < i < 60,

 POC = (POC - Normal[SparseArray[{{n, s} -> POC[[n, s]]}, {8, 8}]] +
     Normal[SparseArray[{{n, s} -> (POC[[n, s]]*(1 - POD))}, {8,
        8}]]) + (Ceiling[
      POC - Normal[SparseArray[{{n, s} -> POC[[n, s]]}, {8, 8}]]]*(
     POC[[n, s]] - (POC[[n, s]]) (1 - POD))/(6*np - 1));
 Min[Part[POC, n, s]*ConstantArray[1, {3, 3}] -
   Part[POC, n - 1 ;; n + 1, s - 1 ;; s + 1]];

 rnxt = Position[Part[POC, n, s]*ConstantArray[1, {8, 8}] - POC,
   Min[Part[POC, n, s]*ConstantArray[1, {3, 3}] -
     Part[POC, n - 1 ;; n + 1, s - 1 ;; s + 1]]];

 If[Length[rnxt] > 1,
  n = Part[Part[rnxt, If[Abs[n - Part[Part[rnxt, 1], 1]] > 1, 2, 1]],
    1], n = Part[Part[rnxt, 1], 1]];

 If[Length[rnxt] > 1,
  s = Part[Part[rnxt, If[Abs[s - Part[Part[rnxt, 1], 2]] > 1, 2, 1]],
    2], s = Part[Part[rnxt, 1], 2]];

 i++; POC = POC; Print["  n=", n - 1, ", s=" , s - 1]]
```

Output:

```
n=1, s=4
n=2, s=3
n=2, s=4
n=3, s=3
n=3, s=4
n=2, s=5
n=1, s=5
n=1, s=4
n=1, s=3
n=1, s=2
n=2, s=2
n=2, s=3
n=3, s=2
n=4, s=3
n=4, s=4
n=3, s=5
n=2, s=4
n=3, s=3
n=4, s=2
n=5, s=2
n=5, s=3
n=5, s=4
n=4, s=5
n=3, s=4
n=2, s=5
n=1, s=6
n=1, s=5
n=2, s=6
n=3, s=6
n=4, s=6
n=5, s=5
n=6, s=4
n=6, s=3
n=6, s=2
n=5, s=1
n=4, s=1
n=3, s=1
n=2, s=1
n=1, s=1
n=1, s=2
n=1, s=3
n=1, s=4
```

```
  n=2, s=3
  n=2, s=2
  n=3, s=2
  n=4, s=3
  n=4, s=4
  n=3, s=5
  n=2, s=4
  n=3, s=3
  n=4, s=2
  n=5, s=2
  n=6, s=1
  n=6, s=2
  n=5, s=3
  n=5, s=4
  n=6, s=5
  n=5, s=6
  n=6, s=6
```

9 Appendix D

Loop for determining the order in which we will search regions without the requirement of adjacency:

```
n = 2;
s = 4;
i = 1;
POC = {{0, 0, 0, 0, 0, 0, 0, 0}, {0, POCn1negs3, POCn1negs2,
    POCn1negs1, POCn1s1, POCn1s2, POCn1s3, 0}, {0, POCn2negs3,
    POCn2negs2, POCn2negs1, POCn2s1, POCn2s2, POCn2s3, 0}, {0,
    POCn3negs3, POCn3negs2, POCn3negs1, POCn3s1, POCn3s2,
    POCn3s3,0}, {0, POCn4negs3, POCn4negs2, POCn4negs1,
    POCn4s1,POCn4s2, POCn4s3, 0}, {0, POCn5negs3,
    POCn5negs2,POCn5negs1, POCn5s1,POCn5s2, POCn5s3, 0},
    {0,POCn6negs3, POCn6negs2, POCn6negs1,
    POCn6s1,POCn6s2,POCn6s3, 0}, {0, 0, 0, 0, 0, 0, 0, 0}};


While[1 < n < 7; 1 < s < 7; 0 < i < 60,

 POC = (POC - Normal[SparseArray[{{n, s} -> POC[[n, s]]}, {8, 8}]] +
     Normal[SparseArray[{{n, s} -> (POC[[n, s]]*(1 - POD))}, {8,
        8}]]) + (Ceiling[
      POC - Normal[SparseArray[{{n, s} -> POC[[n, s]]}, {8, 8}]]]*(
     POC[[n, s]] - (POC[[n, s]]) (1 - POD))/(6*np - 1));
 Min[Part[POC, n, s]*ConstantArray[1, {8, 8}] - POC];

 rnxt = Position[Part[POC, n, s]*ConstantArray[1, {8, 8}] - POC,
   Min[Part[POC, n, s]*ConstantArray[1, {8, 8}] - POC]];

 If[Length[rnxt] > 1,
  n = Part[Part[rnxt, If[Abs[n - Part[Part[rnxt, 1], 1]] > 1, 2, 1]],
    1], n = Part[Part[rnxt, 1], 1]];

 If[Length[rnxt] > 1,
  s = Part[Part[rnxt, If[Abs[s - Part[Part[rnxt, 1], 2]] > 1, 2, 1]],
    2], s = Part[Part[rnxt, 1], 2]];
 i++; POC = POC; Print["  n=", n - 1, ", s=" , s - 1]]

  n=1, s=4
  n=2, s=3
  n=2, s=4
  n=3, s=3
  n=3, s=4
  n=1, s=5
  n=1, s=2
  n=2, s=2
  n=2, s=5
  n=4, s=4
  n=4, s=3
  n=3, s=2
  n=3, s=5
  n=4, s=5
  n=4, s=2
  n=1, s=3
  n=1, s=6
  n=1, s=1
  n=1, s=4
  n=5, s=3
  n=5, s=2
  n=5, s=5
  n=5, s=4
  n=2, s=6
  n=2, s=1
  n=6, s=4
```

```
n=6, s=3
n=3, s=6
n=3, s=1
n=4, s=1
n=4, s=6
n=6, s=5
n=6, s=2
n=5, s=1
n=5, s=6
n=2, s=3
n=6, s=6
n=6, s=1
n=2, s=4
n=3, s=3
n=3, s=4
n=1, s=5
n=1, s=2
n=2, s=2
n=2, s=5
n=4, s=4
n=4, s=3
n=3, s=2
n=3, s=5
n=4, s=5
n=4, s=2
n=1, s=3
n=1, s=6
n=1, s=1
n=1, s=4
n=5, s=3
n=5, s=2
n=5, s=5
n=5, s=4
```