Team Control Number

41806

For office use only
T1 ________________
T2 ________________
T3 ________________
T4 ________________

Problem Chosen

B

For office use only
F1 ________________
F2 ________________
F3 ________________
F4 ________________

2015 Mathematical Contest in Modeling (MCM) Summary Sheet
(Attach a copy of this page to your solution paper.)

Type a summary of your results on this page. Do not include
the name of your school, advisor, or team members on this page.

When an airplane is declared lost in the ocean, it is possible to give an upper-bound on the area in which the airplane could be found. However, this upper bound can be exceptionally large and may grow larger over time, making search time on the magnitude of months or years. In this way, the problem of finding a lost airplane in the ocean lends itself naturally to a probabilistic model, one which cannot eliminate possible locations, but can suggest a location to begin searching.

In our model, we separate the process of crashing into five sequential events:

1. Loss of signal: The airplane fails to report its location and we consider the possibility that it is lost.
2. Contact with ocean. The plane has touched the surface of the ocean either by crashing or landing in the water.
3. Ocean surface currents. Debris moves through ocean currents, which obscures the initial point of contact with the ocean.
4. Sinking. Large pieces of debris (possible the entire plane) move to the bottom of the ocean
5. Contact with ocean floor. Sunken debris settles on the ocean floor.

Each of these events in the crashing process affect possible area in which a lost airplane can be found, which is why the possible search space can be very large. However, in our model, the probability distribution is transformed through each event, giving a final probability distribution informed by each step.

With the final probability distribution, searchers can make informed decisions about where they want to search. The benefit of working with a probabilistic model is that the distribution can be refined with the addition of any kind of information; both the presence and the lack of debris contribute to the probability distribution. This allows us to create a model that adjusts according to real-time data.

Commercial Airplanes Lost at Sea

Team #41806

abstract

This paper will examine the problem of finding an airplane lost in the ocean given only the location at which the plane loses contact. Using a probabilistic model which changes over time, this paper considers the surface ocean currents, deep ocean currents, and likely behaviors of the pilot to suggest probable locations of airplane debris in the ocean. The model suggests an algorithm and a computer program that can be used by searchers to aid the process of finding the lost airplane. So alongside the probabilistic model, the paper will given a recommended protocol for the use of the model in aiding search planes.

Contents

1 Summary

When an airplane is declared lost in the ocean, it is possible to give an upper-bound on the area in which the airplane could be found. However, this upper bound can be exceptionally large and may grow larger over time, making search time on the magnitude of months or years. In this way, the problem of finding a lost airplane in the ocean lends itself naturally to a probabilistic model, one which cannot eliminate possible locations, but can suggest a location to begin searching.

In our model, we separate the process of crashing into five sequential events:

1. *Loss of signal.* The airplane fails to report its location and we consider the possibility that it is lost.

2. *Contact with ocean.* The plane has touched the surface of the ocean either by crashing or landing in the water.

3. *Ocean surface currents.* Debris moves through ocean currents, which obscures the initial point of contact with the ocean.

4. *Sinking.* Large pieces of debris (possible the entire plane) move to the bottom of the ocean

5. *Contact with ocean floor.* Sunken debris settles on the ocean floor.

Each of these events in the crashing process affect possible area in which a lost airplane can be found, which is why the possible search space can be very large. However, in our model, the probability distribution is transformed through each event, giving a final probability distribution informed by each step.

With the final probability distribution, searchers can make informed decisions about where they want to search. The benefit of working with a probabilistic model is that the distribution can be refined with the addition of any kind of information; both the presence and the lack of debris contribute to the probability distribution. This allows us to create a model that adjusts according to real-time data..

2 Introduction

2.1 Terminology and Abbreviations

Throughout this paper, we will use terminology specific to airplane protocol and engineering. We will also use terminology specific to our model, which we will define here.

ditching: The process of landing an airplane in water due to some forced condition [16].

equilibrium glide: A glide in which the components of gravitational force are balanced by the drag force and lift force. This kind of glide maximizes the horizontal distance travelled by a glider [13].

PDF: This stands for Probability Distribution Function, a function which describes the distribution of a probabilistic process. See section 3.

LPD: This is the landing/crashing Location Probability Distribution. See section 4.

TDPD: This is the Time Dependent Probability Distribution of the location of floating debris. See section 4 and 6.

2.2 Preliminary Assumptions

These are assumptions made based on the statement of the problem. Note that these are only the assumptions necessary to begin modeling the process. Our model has other assumptions, but they are supported by research and experimentation and are expressed throughout the paper.

Assumption 1: We exclude the possibility of terrorism. This drastically changes the ability to predict locations of the plane, especially due to human intervention.

Assumption 2: The airplane is no longer flyable after it has lost contact. In the other case, the plane may be late in its arrival or may have to make a precautionary landing in another location, but it will not be necessary to build a search model for finding it.

Assumption 3: The only information that we are given is the location and altitude of the plane at its last point before losing contact. Though this assumption restricts the model, the assumption of minimal information also provides a way to refine the model with more information.

2.3 Paper Structure

We order our sections by a sequence of considerations. In section 3, we discuss the probable area that the airplane crashes, using the last point of contact and an estimation of how far an airplane can glide. In section 4, we demonstrate how the probable area of the crash evolves over time due to the ocean currents, giving us the probable locations of survivors and/or debris on the ocean surface. We also illustrate how debris can be traced back in time to the initial impact.

We address the search for the flight data recorder in section 5, where we investigate how any sunken debris moves away from the initial crash site. Finally, we discuss how to optimally search for surface debris and/or survivors rapidly with search aircraft using our model for probable debris field area. We also describe how the model can be dynamically modified for better accuracy with real-time search data.

3 Loss of Signal

The moment the airplane losses contact, there is a possibility that the plane is lost and viable to crash. Given the possible new policy that airplanes need to report their location every 15 minutes, the loss of contact indicates an accurate point of reference as the earliest possible location at which the plane is in a distress state [10]. In this section, we will give an upper bound on the distance that the airplane can travel, and then use a probability distribution to model the probable locations that the airplane makes contact with the ocean.

3.1 Ditching Protocol Assumptions

We assume in our model that the pilot will attempt to reach the closest land mass. By this assumption, we attempt to maximize the possible distance

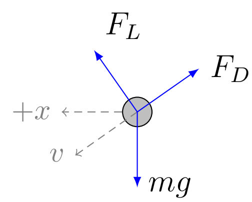

Figure 1: Free body diagram for an Airplane in an Equilibrium-Glide

that the airplane can glide given no tail-winds. This is probably the strongest assumption in our model since it weights a primary direction based on human behavior. However, we justify this assumption by the fact that search-and-rescue is easier to perform near the coast. The FAA is unclear about how far a plane will travel in order to ditch in its ditching procedures [16].

3.2 Upper Bound on Glide Range

We assume that the airplane travels in an equilibrium glide. This will maximize the horizontal distance that the airplane travels without any thrust force. Let m be the mass of the airplane, H the altitude, and R the variable for the range of the glide. Let θ be the angle between the positive x axis and the velocity vector in Figure 1. Then we have the following equations:

$$
\begin{aligned}
F_D &= mg\sin\theta \\
F_L &= mg\cos\theta \\
\tan\theta &= \frac{F_D}{F_L}.
\end{aligned}
$$

Now since $\tan\theta = \frac{H}{R}$, we have that $R = \frac{F_L H}{F_D}$. So we can use the lift-to-drag (L/D) ratio for estimating R.

The L/D ratio of most commercial airplanes is between 15-20, and the average cruising altitude is between 10-12 kilometers, which yields R between 150 km and 240 km [11, 12]. Since we wish to apply our model to different airplane models, we leave R as a variable, understanding that $R = H(L/D)$, where H is the last reported altitude. The L/D ratio is a well-documented number for different airplane models [11].

3.2.1 Descent Time

Later parts of our model require the time that the airplane is in the air before ditching. Since the airplane travels in an equilibrium glide, θ is small and the following approximations are possible:

$$\cos\theta \approx 1$$
$$\sin\theta \approx \tan\theta. \text{ [13]}$$

Let v_d be the downward velocity and v be the airspeed of the airplane in the glide. The downward velocity $v_d = v\sin\theta \approx v\frac{F_L}{F_D}$ by the above approximation. Then $F_L = mg\cos\theta \approx mg$. We use the high velocity lift equation $\frac{1}{2}\rho v^2 C_L A$ where ρ is the density of air, C_L is the lift constant, and A is the cross-sectional area of the airplane to approximate the downard velocity as

$$v_d = v\frac{F_L}{F_D} = \sqrt{\frac{2mg}{\rho C_L A}}\frac{F_L}{F_D}.$$

Using the specifications of the Boeing 777, this is about 25 minutes [12].

3.3 Probabilistic Model for Ocean Contact

Initially, we assume that weather is clear on the day of the accident. If the landmass closest to the last location of the airplane is further than R, the pilot will be forced to ditch. By our above ditching protocol assumptions, there are few cases in which a pilot will choose not to maximize the glide time of the airplane so this variation is somewhat low. Likewise, since we assume that the pilot will always approach the closest landmass, the angle by which this path will differ is relatively small.

Fore these reasons we choose to represent the distribution of the values of R as a chi-squared distribution with a mean value $k = 10$. We represent the deviation from the straight path by a normal distribution with standard deviation $\sigma = \frac{\pi}{6}$. See Figure 2. This gives a sector of probable ditching areas around the point of lost contact.

3.3.1 Considerations of weather

In the above calculations, we have assumed that there is no intervention of weather. However, weather plays a major role in the distribution, especially

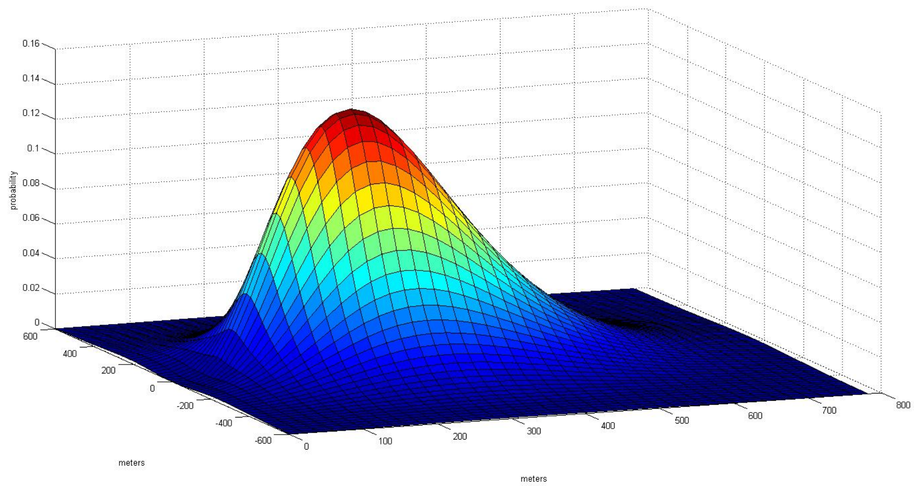

Figure 2: Probability Distribution Function for Possible Crash Areas

since airplane crashes are most likely to happen during storms or high wind situations. Though it is possible to build a model that accurately depicts the affects of wind on the distribution, we suggest a simple approximation.

We assume that we are able to get simple wind data for this model, which includes an average wind velocity vector in the area around the loss of contact. In order to approximate the affect of wind we consider the components of the average wind velocity vector. Let v_x be the component in the direction of the the airplanes most likely path. The direction of v_x determines the amount of *skew* in the distribution of R while $|v_x|$ determines the *spread* of the distribution. The approximation is identical for v_y, the component perpendicular to v_x, with reference to how it affects the skew and spread of the θ distribution. The terms skew and spread are intentionally vague since they may depend on different features of the weather.

The benefits of choosing this simpler model are it avoids *overfitting* data to make the distribution too tight and it is computationally more efficient. There may be optimizations in this aspect of the model but our time constraints did not allow us to explore them further.

4 Surface Currents

Note that this model so far is not particularly interesting. It describes the natural intuition that the pilot will attempt to reach as far as possible in a particular direction with some amount of randomness. However, it is necessary to codify this intuition into a probability distribution that will change with respect to ocean surface currents. In this section, we will outline the general problem of modelling this idea as well as a computation method for solving this it.

4.1 Generalized Problem

Once an airplane lands or crashes on the ocean surface, its floating parts move with respect to the ocean surface currents. This implies that the location with the highest probability of finding debris may not be the same an hour later. We would like a model which accounts for this change.

4.1.1 Mathematical Representation

The surface currents of the ocean can be modeled as a two-dimensional vector field $f : \mathbb{R}^2 \to \mathbb{R}^2$ We assume that this vector field is continuous and solenoidal, which means that the divergence is zero at all points in the field. This assumption just gives the vector field convenient properties that are realistic in terms of ocean currents since water is essentially an incompressible fluid [15]. Next we may consider the PDF from above as a scalar field $P : \mathbb{R}^2 \to [0, 1]$.

The problem then generalizes to producing a function $P(x, y, t)$ which represents the probability at $(x, y) \in \mathbb{R}^2$, after a time t. This naturally implies that the initial condition of this function is $P(x, y, 0) = P(x, y)$. Informally, the vector field *transforms* the probability distribution to produce a new probability distribution which corresponds to the movement of the currents.

This problem becomes more interesting with the addition of more conditions, but first consider the simplest case: a point travels through the vector field by travelling through the field lines. This case leads to the following proposition. Though this proposition is intuitive, it gives insight to the complexity of the model in this simple case.

Proposition 1. The field lines of a continuous solenoidal vector field $f : \mathbb{R}^2 \to \mathbb{R}^2$ cannot intersect.

Proof. Assume for contradiction that there exists two field lines which intersect. Let a and b from $[0, t] \to \mathbb{R}^2$ be parametrizations of portions of these field lines. Then there exists a t' such that $a(t') = b(t')$. In order for $f(a(t'))$ to be well defined, two vectors must point to $a(t')$ and one must leave. However, this can only maintain that the divergence at $a(t')$ is zero if the vector leaving is great than the two entering, but this contradicts the continuity of the vector field. $\square$

The following corollary is immediate.

Corollary 1. If two points in a continuous solenoidal vector field from $\mathbb{R}^2 \to \mathbb{R}^2$ are connected by a field line, then the path is unique.

This corollary implies that in this simple case, the probabilities will never *intersect*, but rather will be shifted by the vector field. Then $P(x, y, t) = P(x', y')$ where (x', y') is the final point of the t length reversed path on the field line. In other words, this is an uninteresting case since the vector field will shift the entire distribution equally.

4.1.2 Variations

This system become interesting when we consider different conditions. There are three cases in particular that we enumerate here.

1. *The addition of mass to each point in the plane.* This will give each point momentum, which causes more interesting paths through the vector field.

2. *Considering multiple pieces of different masses.* With multiple pieces of debris, the probability distribution models the likelihood of finding any arbitrary piece of debris, which may increase the chance of finding positive information.

3. *Noise or time dependence in the vector field.* Proposition 1 no longer holds in this case.

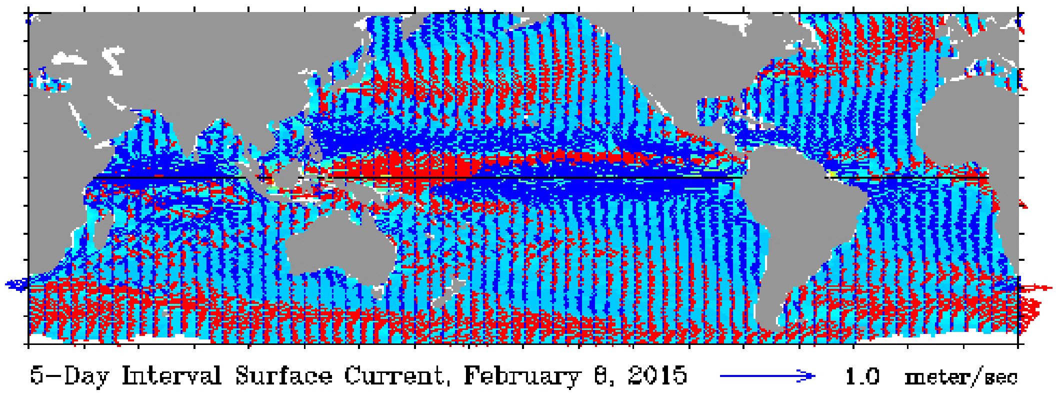

Figure 3: NOAA near-realtime data on global ocean current velocities

In each of these cases, values of the probability distribution can non-trivially converge at single points. Intuitively this means that searchers are likely to find wreckage coming from two different directions or sources. In the following section, we examine the first case to demonstrate how these values may be computed.

remark. The theoretical examination of this problem yields some rather interesting questions. Despite our research, we could not find a way to analytically *transform* a scalar field through a continuous vector field in this way, though the ability to do so seems rather powerful. Likewise, we were unable produce a reason why this would not be possible.

4.2 Computational Method

The focus of designing this algorithm is to create a method to compute the outcome of applying a discrete vector field to a probability distribution. The python code for the simulation is in the Appendix.

4.2.1 Data

Computing the probability distribution of a floating object requires data on the ocean currents. Since ocean current data is readily available in the form of near-realtime discrete velocity vector fields, we based our simulation on discrete ocean current vector fields [14]. The data available, as can be seen in Figure 3, is both extensive and regularly updated via satellite altimeter

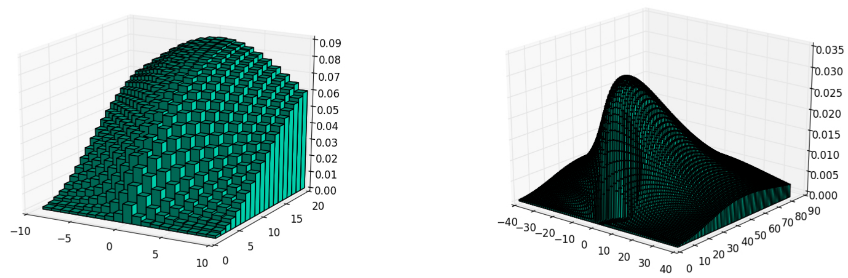

Figure 4: Low density discretization on the left and high density discretization on the right

and scatterometer readings [14].

Because we are given a discrete vector field to use as ocean currents in our computation, we also discretized our landing probability distribution (LPD) so it is in a more convenient form for computation. Inevitably, discretizing data leads to discretization error, but this error is unavoidable to begin with since the ocean current data itself is discrete. To minimize the discretization error, we can increase the number of points we use to discretize the continuous LPD (Figure 4). For example, assuming water currents are not volatile on a small scale, if the vector field data only contains one velocity vector for every 5 m^2, then we can discretize the LPD by assigning a scalar to every 5 cm^2, almost making any discretization error negligible.

4.2.2 Method

To set up the LPD before transforming it by the ocean currents, we first defined a data structure called a ”floater” to represent the LPD discretely. The structure ”Floaters” is essentially a list of floaters. Every floater has four fields:

1. Location: A (x, y) point that corresponds to where in the ocean the floater is currently.
2. Mass: The mass of the object being tracked
3. Momentum: The current momentum vector of the floater

4. Probability: The probability of the floater is the object being tracked

The initiation of the floaters is simply stepping through the domain of our LPD and creating a floater with the corresponding (x, y) and assigning the scalar LPD(x, y) as the floater's probability. In addition, each floater's mass and momentum are set to the mass of the object being tracked and 0, respectively. Setting the momentum to 0 initially is an assumption that may lead to error, but since we are approaching this search statistically, we include the variation of momentum in the variation of the LPD.

After the initiation, we the start the simulation. Stepping through time in one second time steps, we update the position and momentum of each floater systematically. Different examples of the simulation updating are shown in Figures 5-7 on the previous page. The current version of the code does this by calculating the momentum change caused by a force vector whose magnitude is decided by the drag force on a sphere caused by a current. The physics is overly simplified in the current simulation. However, the physics could easily be updated to handle more accurate geometries and different masses.

5 Movement of Sinking Debris

An important goal of airplane crash investigations is the recovery of the flight data recorder (FDR). As we have previously established, a ditched aircraft will inevitably sink into the ocean. To locate the FDR, we must find the probable area of the ocean floor in which the sunken debris lies.

5.1 Sunken Debris Movement

Once the FDR lands on the ocean floor, we assume that only ocean currents at the sea floor, known as "bottom currents", have the potential of shifting the debris. Bottom currents move normally at a rate of less than 10 centimeters per second ($\frac{cm}{s}$), and rarely up to 1 meter per second ($\frac{m}{s}$) depending on the terrain [1]. The FDR is usually located in the rear of the plane- we make this assumption for the rest of the analysis [2]. In the case of the airplane ditching and remaining intact, it is clear that the FDR would be attached to a large piece of sunken wreckage. In the case of a catastrophic ditching, we

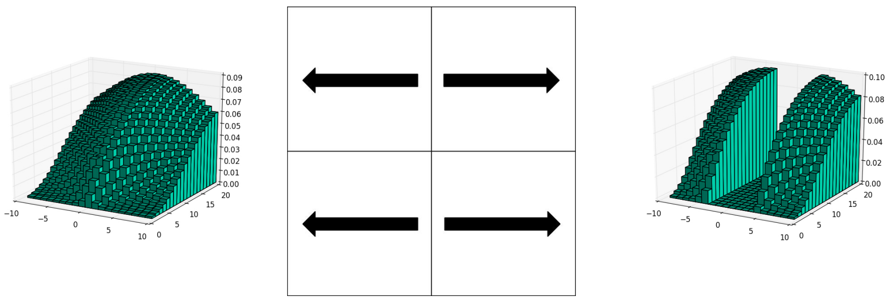

Figure 5: Here the left side vectors of the ocean current field are $[-1, 0]$ vectors and the right side vectors are $[1, 0]$. So, with time steps, the LPD splits in half and spreads towards left and right.

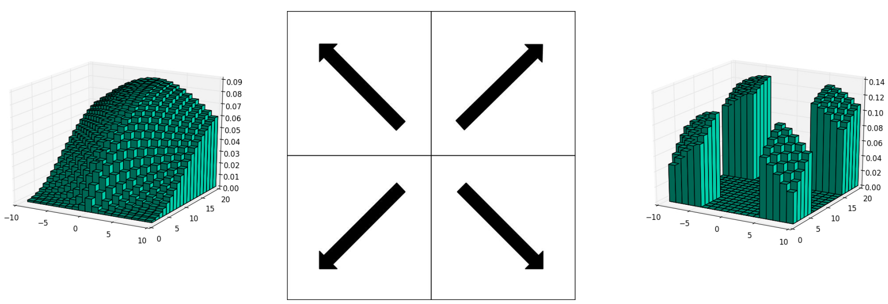

Figure 6: In this instance, the ocean current field vectors are all pointing along the diagonal outwards, resulting in the LPD spreading outwards in fourths.

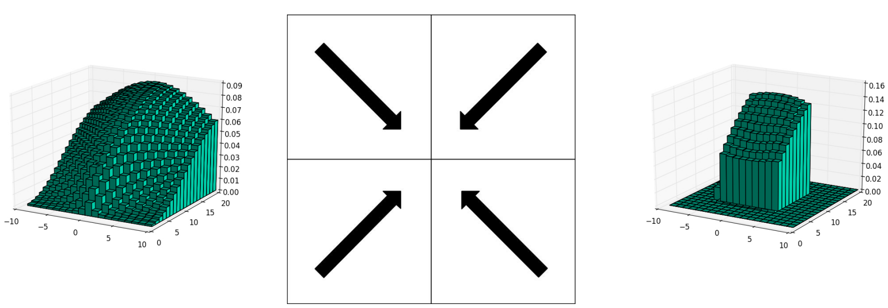

Figure 7: This is the inverse of the previous example. It demonstrates the increasing probability of certain locations as floaters overlap.

assume that the tail of the aircraft will break off, but in such a way so that it is a very large piece of debris.

With our assumptions in place, we see that any movement of the FDR and the large piece of wreckage it is attached to is negligible during the time frame of our search. Assuming that the FDR moves at the usual speed of $1cm/s$ due to bottom current, it travels .933 nautical miles (nm) in a day. Active sonar systems used for finding ship wrecks can detect objects up to 15 nm away, with a lower bound of $5nm$ [3], so it would be trivial for any nautical search vessel equipped with active sonar to locate the FDR, given its initial touchdown location on the ocean floor, even after 2 weeks of drifting in a single direction.

5.2 Drifting while Sinking Model

Supposing that we know exactly where the initial impact location, we can better locate the FDR by correcting for any drifting during its journey to the ocean floor. We develop a model to calculate an airplane's position sinking behavior starting from a submerged state. We assume that the wings of the airplane will break off, either due to force of impact, or water pressure. Hence, we model the aircraft as a long cylinder, traveling such that the water flows perpendicular to its faces. The forces on the aircraft are gravity, buoyancy, and drag. Our equation of motion is:

$$mv' = \frac{1}{2}\rho C_d A + (\rho V - m)g \tag{1}$$

where, m is the mass of the object, v is the velocity, ρ is the density of displaced fluid, C_d is the drag coefficient, A is the cross-sectional area, and V is the volume of displaced fluid. In the following calculations, we used the data for an Airbus 340-500 for each parameter[5]. We took into consideration the pycnocline- the varying density of ocean water at different depths. We approximated the pycnocline curve as linearly increasing from 1025 kilgrams per cubic meter (kg/m^3), to $1028kg/m^3$ from depth $0m$ to $1000m$, then a constant $1028kg/m^3$ beyond[6]. By our model of the airplane as a long cylinder, we use a drag coefficient of .81 [4].

We rephrase the 2nd order equation of motion as system of 1st order ODE's for use in Matlab's numerical integrator (relevant code in the Appendix).

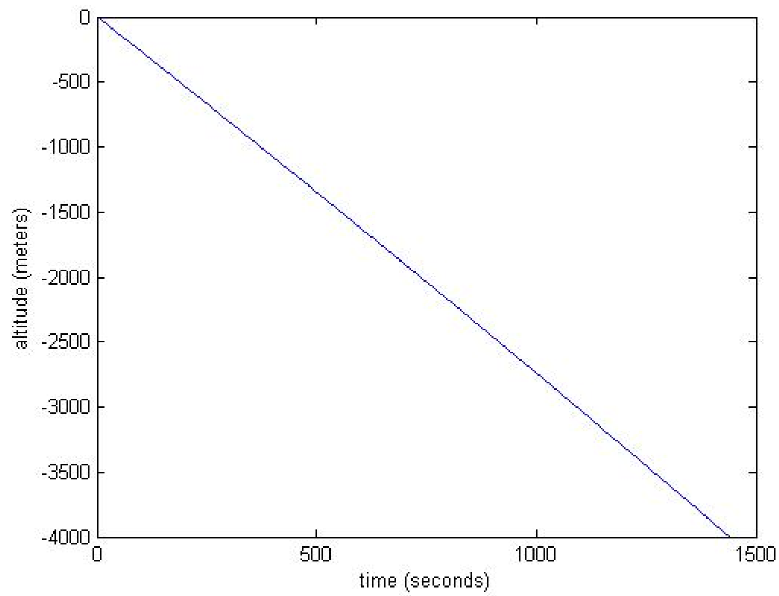

Figure 8: A half-plane's journey to the ocean floor

We chose to have our object be half of an Airbus 340-500; we model the aircraft as broken into two halves upon ditching. Our initial conditions were that the object was initially at rest. The average depth of oceans is around $4000m$ (excluding Arctic ocean) [7]. Running our simulation, we see that the half-plane sinks to a depth of $4000m$ in 24 minutes. For reference, the Titanic sunk to the ocean floor after submerging in 15 minutes [8].

From our graph, we see that a half-plane sinks to $500m$ in 3 minutes. We assume that surface currents affect its movement up until $500m$ depth, and bottom currents take over beyond. We also assume fair weather conditions, so surface currents have an upper bound of $1m/s$. The maximum expected drifting amounts to $306m$ ($1/3nm$) during sinking. Combined with bottom current drift after settling, this is still a trivial search area for a nautical vessel equipped with active sonar, if it is given the initial impact site- it could just scan at the impact site, and it would locate the sunken FDR, even if it has been 2 weeks of time since the crash.

5.3 Drifting During Submerging

The last factor to consider is how far the aircraft drifts while it is floating and while it is in the process of submerging. We expect an aircraft that ditches successfully to take the longest for this process, since it would take on water much slower than an aircraft that's been torn into 3 pieces. Aircraft are designed to float for as long as necessary for the passengers to evacuate, which is 90 seconds [9]. If we upper-bound our floating and submerging time by 5 minutes, we see that in fair weather conditions, surface currents would shift the planes position by $300m$.

So, the total amount of distance from the sunken debris to the impact site is negligible for any vessel that is capable of searching for the FDR. Using our debris tracing method, a nautical vessel is guaranteed to pass over the local area of the impact site, at which point it should discover the debris. This is supported by the recent events with AirAsia Flight 8501, in which the sunken wreck was discovered soon after surface debris was located.

6 Search Strategy

Now that we have identified the area to be searched, we must establish a search strategy that minimizes search time and use of resources while maximizing information gain during a search.

6.1 Recommended Protocol

Before an airplane is considered lost, it is possible to run our model with real-time ocean current data in order to estimate its location given its last point of contact. When the airplane is officially declared lost, the simulation has already been running, which gives searchers up-to-date locations with high probability of finding debris. The next section will outline how this model can change dynamically in the search process.

6.1.1 Dynamic Search Model

The beauty of using a probabilistic model is that all information, both positive and negative, refines the model. Suppose that we only have one search plane. We assume that the search plane perfectly searches the ocean surface in a circle around itself or radius r, meaning that if there is debris within its search radius, the plane will find it. If the search plane finds debris, then the simulation is over and we reverse the computations to find the possible impact location of the debris. However, if the search plane finds no debris, then in the circle of radius r, the probability distribution for the initial impact positions of that debris can be drastically decreased. This will raise all other probabilities of other initial impact positions and the model can be rerun with the refined data.

6.1.2 Consequent Graph Problem

The following proposition is given without proof.

Proposition 2. The integral of the TDPD over a circle of radius r is maximized if the center of the cicle is a locally maximum point.

This proposition implies that the problem of finding wreckage through our model generalizes to an optimization path problem in a graph; the best places to look are the those with high probability. However, due to the dynamic nature of the model, a path is dependent on current information and may change which new information. For example, even though there may be a high probability spot a distance away, it may be more telling to check several closer spots of lower probability first to see how the dynamic model changes.

We define a cost function for optimal path planning by d/p, where d is distance, and p is the probability of locating debris at the destination, according to our surface debris shift model. This helps optimize the time aspect of the search by prioritizing *interesting* areas near a searcher's current location. This gives the following algorithm: In order from greatest to least, follow path between local maximum points which minimizes the cost function. This way, the seacher chooses points which are close, but also consider which are most likely to give the most data.

6.1.3 Computational Considerations

In practice, it is difficult to test the complexity or correctness of this algorithm. Since the graph changes dynamically, the algorithm is difficult to analyze using graph algorithm techniques. This algorithm may also take more computing power than an average laptop. For these reasons we have left this algorithm open for consideration.

7 Conclusions

In this section, we outline the advantages and disadvantages of our model. This will follow with our final remarks and possible avenues of future research.

7.1 Advantages

1. Our model uses easily obtainable parameters.
2. Our model evolves with time, so it is relevant regardless of when the search begins.
3. Our model is dynamic, so any real-time information which may be pertinent can be integrated into it.
4. The process is naturally algorithmic and parallel, which means fast computation and the possibility of a usable computer program.
5. Our search strategy is able to accommodate for any number of search planes working simultaneously.
6. When more search planes are added, the increased complexity of the search strategy is hidden in the computer algorithm- the search plane operators do not have to be mathematicians to understand what to do.
7. Our search algorithm covers all three objectives of a search-and-rescue operation.

7.2 Disadvantages

1. We make a strong initial assumption to limit the behavior of the airplane.
2. Bad weather is a factor in many aircraft crashes, but we assume fair weather throughout our modeling process. This also affects visibility for search planes.

3. Although ocean currents are fairly stable, the data comes in the form of a 5-day average, so our model is always using 'outdated' information.

4. Our model for surface debris movement assumes the size/mass of surface debris.

5. Our model only restricts search area in the event that there are no survivors and floating debris.

6. Our model assumes that search planes are perfect at searching.

7.3 Final Remarks

We note that our model makes a lot of assumptions about the conditions surrounding a crash, notably that once contact is lost with the airplane, it enters a glide and crashes. There are many considerations that make this case seem unlikely, such as in large wide-body jets. Those aircraft typically have 4 engines, and can operate with only 2, albeit fly much slower. It is far more likely that 3 engines fail and the craft is forced into a powered descent, than for all 4 engines to fail and the craft is forced into an unpowered (but controlled) glide.

However, the greatest advantage of our model is the abstraction between each step of analysis. In the event of an actual crash, we would know the type of aircraft, so we can update the probability distribution used to model the range of the aircraft since point-of-last-contact. With that updated, we can still apply our computer algorithm and search strategy to the new input with no further modifications.

An example to demonstrate the power of our abstractions: Suppose there's a seismic shift in the time frame of the crash. Then we would expect the sunken debris to shift, and for the ocean surface currents to change. We can update our sunken debris model using the seismic data, and use ocean-seismic activity simulators (those used to see if an earthquake would generate a tsunami) to update the ocean current data. If our other assumptions still hold, then we can progress with using the rest of our model.

7.4 Further Research

One big issue that would greatly optimize the search process is the development of the full Time-Dependent-Probability-Distribution (TDPD) algorithm. It is important to find a way to parallelize the algorithm as much as possible, so the computation does not require a super computer. The optimal solution would be to parallelize over CUDA cores in computer GPUs; then, any search plane equipped with a thousand dollar computer can be running the TDPD in real-time.

Another general point of research is on how to refine our various models for typical aircraft operating transcontinental flights. For example, even experienced pilots may be unable to maintain our assumed equilibrium unpowered glide, so research can be done to see how most pilots attempt to ditch the aircraft. In addition, it is unlikely that pilots would control a Boeing 747 in the same way they would try to control a Airbus 340, so the distinction between aircraft should also be explored.

Finally, more research can be done on the nature of accidents- what percentage of them result from engine failure, instrument failure, pilot error, etc. The initial restriction on search area can be greatly improved if there is information on what kind of accidents occur the most often. We would expect a plane suffering from instrument failure to fly much farther than a plane with engine failure after loss of contact. Unfortunately, this research is very difficult, as it requires more data, i.e. more plane accidents over the ocean.

8 Press Conference Paper

When an airplane is lost in the ocean, it is a time sensitive process to find the wreckage. Survivors who are stranded are at risk of hypothermia and the search area increases with time.

Though no location in the ocean can be ruled out as a possible location for the airplane, there are places that are obvious to look first. For example, it is clear that the airplane is unlikely to be near anywhere where there is a dense population of ships and boats, as the boats would have seen the airplane crash. Our model attempt to quantify these intuitions and model the complex system which can shift around the possible locations of the wreckage.

First we consider the locations that the plane could land if it is forced to ditch. This is the hardest part to model since it makes a lot of assumptions about human behavior. Unless we can talk to the pilot, we don't really know what he or she is going to do in the high pressure situation. However, we know that without thrust a commercial airplane can glide around 240 km. This gives us an upper bound on the distances that the airplane *can* travel, so we just build some randomness into this assumption to make it more realistic. In the following graph, the peak of the graph is the place which has the highest probability of containing the wreckage.

We then consider the way that the ocean currents change the location of the wreckage. Though this is a complex system, we have built a computational model that can simulate the system with some approximation. If we run this simulation using ocean current data, we can essentially follow the debris without actually seeing it. This is the most power part of the model, since it allows use to essentially *predict* where the debris might be given our predictions about where it started.

Finally, we consider the sinking of the airplane since the important flight data usually sinks with the large parts of airplane. In our analysis, we found that airplanes sink rapidly, around 24 minutes for 4000 km. By these estimates we claim that the amount of horizontal movement caused by deep ocean currents in negligible considering the sink time. The result of this is that once we find the initial crash point of the plane, it will not be hard with modern sonar, to find the rest of the airplane that has sunken nearby.

Though we hope to never implement this system, we believe this will prepare us with the ability to save time and resources in our search for lost airplanes.

References

[1] "Bottom Currents." National Oceanography Centre, 01 Jan. 2015. Web. 09 Feb. 2015. http://noc.ac.uk/science-technology/earth-ocean-system/seafloor/bottom-currents.

[2] "Air Crash Investigation." National Geographic Channel, 1 Jan. 2015. Web. 09 Feb. 2015. http://natgeotv.com/uk/air-crash-investigation/black-box.

[3] Nelson, Erik J. "PASSIVE AND ACTIVE SONAR PROSECUTION OF DIESEL SUBMARINES BY NUCLEAR SUBMARINES." Post Naval Graduate Thesis (2008): n. pag. Print.

[4] Hoerner, Sighard F. Fluid-dynamic Drag: Practical Information on Aerodynamic Drag and Hydrodynamic Resistance. Alburqueque, NM: Db Hoerner Fluid Dynamics, 1965. Print.

[5] "Dimensions Key Data." Airbus, 01 Jan. 2015. Web. 09 Feb. 2015. http://www.airbus.com/aircraftfamilies/passengeraircraft/a340family/a340-500/specifications/.

[6] Bergman, Jennifer. "Density of Ocean Water." Windows to the Universe. Windows to the Universe, 31 Aug. 2001. Web. 09 Feb. 2015. http://www.windows2universe.org/earth/Water/density.html.

[7] "Ocean Data." Minnesota Botanical Gardens, 01 Jan. 2002. Web. 09 Feb. 2015. http://www.mbgnet.net/salt/oceans/data.htm.

[8] Fowler, Dvaid. "Sinking of the Titanic." Titanic Facts, 01 Jan. 2013. Web. 08 Feb. 2015. http://www.titanicfacts.net/titanic-sinking.html.

[9] Romero, Frances. "Learning from Flight 1549: How to Land on Water." Time. Time Inc., 17 Jan. 2009. Web. 09 Feb. 2015. http://content.time.com/time/nation/article/0,8599,1872195,00.html

[10] Schaper, David. "U.N. Agency Sets New Standards For Tracking Aircraft In Flight." The Two-way. NPR. United States, 05 Feb. 2015. Radio. Transcript.

[11] "Air Crash Investigation." National Geographic Channel. National Geographic Channel, 1 Jan. 2015. Web. 09 Feb. 2015. http://natgeotv.com/uk/air-crash-investigation/black-box

[12] "Boeing 777-200 Technical Specifications." Boeing. Boeing Corp., 01 Jan. 2015. Web. 02 Feb. 2015. http%3A%2F%2Fwww.boeing.com%2Fboeing%2Fcommercial%2F777family %2Fpf%2Fpf%5C_200product.page.

[13] Performance. London: Routledge, 2003. University of Virginia Tech. Web. 2 Feb. 2015. http://www.dept.aoe.vt.edu/l̃utze/AOE3104/glidingflight.pdf.

[14] "World Ocean Surface Currents (OSCAR - Ocean Surface Current Analyses - Real Time) - Methodology." NOAA - National Oceanic and Atmospheric Administration. United States Government, n.d. Web. 09 Feb. 2015. http://www.oscar.noaa.gov/methodology.html.

[15] "Water Compressibility." USGS Water Science School. U.S Geological Survey, n.d. Web. 09 Feb. 2015. http://water.usgs.gov/edu/compressibility.html.

[16] "Distress and Urgency Procedure." FAA. Federal Aviation Administration, 8 Jan. 2015. Web. 09 Feb. 2015. https://www.faa.gov/air_traffic/publications/atpubs/aim/aim0603.html.

9 Appendix

9.1 Code

For original simulation code: https://github.com/fjiang6o2/MCM2015

9.1.1 currents.py

Note, this code snippet is formatted to fit the page

```
def simulate(currents, ocean_scale, dist_x, dist_y, dist_z,
             time):
    #initialization
    floaters = []
    offset = -1*dist_x[0]
    for i in range(len(dist_x)):
        x_entry, y_entry, prob = dist_x[i]+offset, dist_y[i]-1,
                                 dist_z[i]
        floaters += [Floater(40, x_entry, y_entry, prob)]
    #updates - moving floaters according to corresponding force
             vectors
    for t in range(time):
        for floater_num in range(len(floaters)):
            floater = floaters[floater_num]
            plot_x = floater.x//ocean_scale
            plot_y = floater.y//ocean_scale
            if (plot_x >= len(currents)) or (plot_x < 0) or
               (plot_y >= len(currents[0])) or (plot_y < 0):
                floater.floater_gone() #floater floated out of
                                        search space
            else:
                floater.mom_update(currents[plot_x, plot_y])
                floater.pos_update()
                floaters[floater_num] = floater
    #format output
    output_distribution = np.zeros((sqrt(len(dist_x)),
                                    sqrt(len(dist_y))))
    for floater in floaters:
        x_coord, y_coord = floater.x, floater.y
```

```
        if not ((x_coord >= len(output_distribution)) or
                (y_coord >= len(output_distribution[0])) or
                (x_coord < 0) or (y_coord < 0)) :
            output_distribution[x_coord, y_coord]+=floater.prob
    return output_distribution

class Floater():
    def __init__(self, m, x, y, prob, name = "floater"):
        self.name = name
        self.x = x
        self.y = y
        self.m = m #mass
        self.prob = prob
        self.mom = [0, 0]
    def mom_update(self, velocity): #updates both designed
                                     around being called every
                                     once every second
        f = force(self.m, velocity) #momentum = force * time
        self.mom[0] += f[0]
        self.mom[1] += f[1]
    def pos_update(self):
        velocity = [self.mom[0]/self.m, self.mom[1]/self.m]
        self.x, self.y = self.x+velocity[0], self.y+velocity[1]
        #distance = velocity * time
    def floater_gone(self):
        self.prob = 0

def force(m, velocity, drag_co = .47, cross_area = .5):
    #calculate drag force vector from velocity vector and dummy
     values
    v_magnitude = sqrt(pow(velocity[0], 2) + pow(velocity[1], 2))
    magnitude = .5 * 1025 * pow(v_magnitude, 2) * drag_co
                *cross_area
    v_unit_vec = [velocity[0]/v_magnitude, velocity[1]/v_magnitude]
    return [v_unit_vec[0] * magnitude, v_unit_vec[1] * magnitude]
```

9.1.2 MatLab sinking code

```
function rho = p(d)

if d < 0
    rho = 1025 + 3*(d/1000);
else
    rho = 1028;
end
```

```
function [xprime] = depthEq(t,x)

V = 104.8689;
m = 230000/2;
A = pi*(5.64/2)^2;
C_d = .81;
g = 9.81;

xprime = [0;0];
xprime(1) = x(2);

xprime(2) = (1/2 * p(x(1)) * x(2)^2 * C_d * A + (p(x(1))*V-m)*g)/m;
```

```
v0 = 0;
d0 = 0;

[t,x] = ode45(@depthEq,[0,24*60],[d0,v0]);
plot(t,x(:,1))
xlabel('time (seconds)')
ylabel('altitude (meters)')
```

9.1.3 MatLab Probability Density Function Graph code

```
function F = chiSquare(X)

F = chi2pdf(X,10);
```

```
function F = normDist(X)

F = normpdf(X,0,pi/6);
```

```
function F = Totes(X,Y)

F = normDist(X) .* chiSquare(Y);
```

```
function P = TotesPDF(X,Y)

normConst = 300;

angle = atan(Y./X);
range = 10*sqrt(X.^2 + Y.^2)/normConst;

Tote = integral2(@Totes,-200,200, 0, 100);

P = normDist(angle) .* chiSquare(range)/Tote;
```

```
[X,Y] = meshgrid(1:20:800,-600:20:600);

Z = TotesPDF(X,Y);

surf(X,Y,Z)
xlabel('meters')
ylabel('meters')
zlabel('probability')
```