Mathematical Models of Deforestation in the Brazilian Rainforest

UCLA Applied Math REU 2019

Bohan Chen, Raymond Chu, Yixuan (Sheryl) He,
Joseph McGuire, Kaiyan Peng

Mentor: Christian Parkinson
PI: Andrea Bertozzi, Stanley Osher

University of California, Los Angeles
Department of Mathematics

Abstract

Identifying two types of deforestation, land clearance for farming and illegal logging, we develop mathematical models to predict the modes of deforestation. Working from first principles, we build individual agents known as farmers that then illegally clear forest in the area of concern. To account for illegal logging we develop a continuous model in which loggers take optimal paths to the location of their crimes, who then must return to "sell" their goods. We discuss some of the optimal control results used in the illegal logging model, and describe the numerical implementation of the illegal logging and farming models. We detail the numerical methods for the optimal path and Hamilton-Jacobi equations involved in the model. Finally, we test the effects of different patrol strategies on the occurrences of illegal logging and measure to what extent they improve the amount of pristine area.

Contents

1 Introduction

Deforestation, and in particular, illegal logging and land clearance have some of the most damaging effects on the world's forest. Modeling and quantifying deforestation has become a recent area of study for ecologists, political scientists, and applied mathematicians. The efficient and effective deployment of law enforcement to the threatened area is the best deterrent for these crimes [1]. Building a model to predict the interactions between the criminals and police is a difficult problem. A crucial first step in this problem is identifying significant parameters to consider.

Rigorous studies have validated the correlation between certain parameters and deforestation in tropical regions such as Brazil [21]. The three dominant categories of parameters are identified by Pfaff and other authors, as accessibility, population, climate, and demand [3, 13]. Accessibility accounts for distance to roads, rivers, and major highways, elevation, the presence of trees or foliage, as well as recent deforestation events in the area. Population density accounts for distance to cities and markets, as well as the presence of farms and other rural settlements. Climate and demand factors refer to the seasonal effect, such as precipitation and the global demand of soy, beef, lumber and other commodities. With these significant parameters identified, the goal is to inform law enforcement agencies as to the best strategies for combating deforestation.

Further analysis of enforcement strategies has been carried out in the state of Roraima, Brazil, using satellite imaging built to detect deforestation events [7, 23]. The strategy on the part of the federal government of Brazil was to monitor deforestation events in each municipality and dispatch federal patrols to the municipalities with the most events. The satellite data employed (DETER), was updated daily and was effective in detecting any deforestation events larger than 25 hectares in size. This particular macro-strategy proved more effective than typical enforcement strategies, however, the issue remains of how to identify effective microscopic patrol strategies for law enforcement.

1.1 Previous Work

A similar question was asked and answered with a game-theoretic model, Protection Assistant for Wildlife Security (PAWS), where an iterative Stackelberg security game was used to describe the interactions between wildlife poachers and rangers [9]. Previously repeated Stackelberg security games

were utilized to model the protection of vital infrastructure in the case of attack. PAWS employed a discrete approach to determine effective patrol strategies in an area of interest. Machine learning techniques were used to determine animal density for each cell in the grid, as well as assign an accessibility score to each cell. Cells that were topographically significant, i.e., cells necessary to enter or exit a region, or cells with high accessibility score, were designated key access points. These were then deemed as nodes and edges were drawn to connect the nodes, so that cells with a higher accessibility score were traversed when possible. Other considerations such as time of patrol and the inclusion of base camp from where to start and end were included for realism, as well as an uncertainty of animal density based on the time of last patrol in that cell. Similar algorithms such as INTERcept and SHARP have been developed and deployed around the world with varying success [11, 12].

In a paper by Albers [1], the problem of deforestation was modeled in a spatially continuous setting. The author of this paper considered a circular area of interest, and modeling how deforestation and patrolling against this, might be described. The patrollers are allocated a budget E, meant to represent the resources available. Budget is used to determine a patrol strategy for the defenders. This is done by giving each radius r a probability of detection $\phi(r)$, then the chance of the attackers getting caught as they are moving from distance d_c to d is given by:

$$\Phi(d) = \int_{d_c}^{d} \phi(r)dr$$

The attackers are assumed to move in radially from their starting radius, d_c, and any ground touched by the attackers is now considered unpristine, while the remaining land is termed pristine as seen in figure 1b. Additionally, the profit that the attackers received by moving a distance d into the protected area is given by:

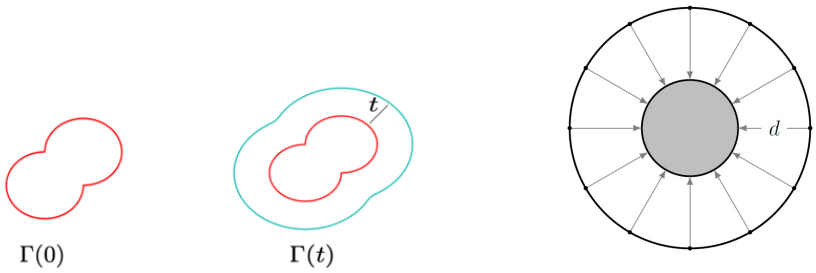
$$P(d) = (1 - \Phi(d))B(d) - C(d)$$

where B is the benefit to the attacker, C is the cost of traveling into depth d and $(1 - \Phi(d))$ represents the probability of not being captured.

A rational attacker will find the d that solves the following optimization problem:

$$\max_{d} (P(d))$$

The defenders then want to minimize the d that solves this problem.



(a) Level sets evolving to a time t (b) Albers' radial evolution

Figure 1: Level set method and Albers' curve evolution

Some issues with this model are the assumption that the area involved is circular, the lack of terrain information, and the uniform behavior of the attackers and defenders. These problems are addressed in a paper from Arnold et al. [2], where a similar modeling problem is generalized to any closed, simple curve in $\mathbb{R}^2$. The primary tool employed in this model is the level set method [19]. Starting with a closed, simple curve Γ, a Lipschitz continuous function is defined, such that $\phi_0 : \mathbb{R}^2 \to \mathbb{R}$, where $\phi_0(x, y)$ is positive inside Γ, negative inside of Γ, and zero at the boundary. Note that these requirements on ϕ_0 are satisfied by the signed distance function to the curve Γ, where distance outside of the curve is negative and positive outside. The function $\phi : \mathbb{R}^2 \times [0, \infty) \to \mathbb{R}$ is then defined by the initial value problem:

$$\begin{cases} \phi_t + v(x, y)|\nabla\phi| = 0 \\ \phi(x, y, 0) = \phi_0(x, y) \end{cases}$$

where $v(x, y)$ is some non-negative velocity function. Next we define the zero level set $\Gamma(t) = \{(x, y) \mid \phi(x, y, t) = 0\}$ and this contour represents the set of points that can be reached from the original contour Γ after traveling time t. In this model cost represents the effort expended by extracting at any point in the protected area, and the velocity is allowed to depend on capture probability and terrain data. The validity of this model hasn't been tested against real-world data, but has been modified and improved by Cartee and Vladimirsky [5].

The purpose of this project is to use the data gathered and categorized by Slough et al. [23] to build a predictive continuous model to describe deforestation events in the state of Roraima. Further, we attempt to determine effective patrol strategies for law enforcement on microscopic scale, and improve upon the model of Arnold et al. [2]. First, we will explain some of

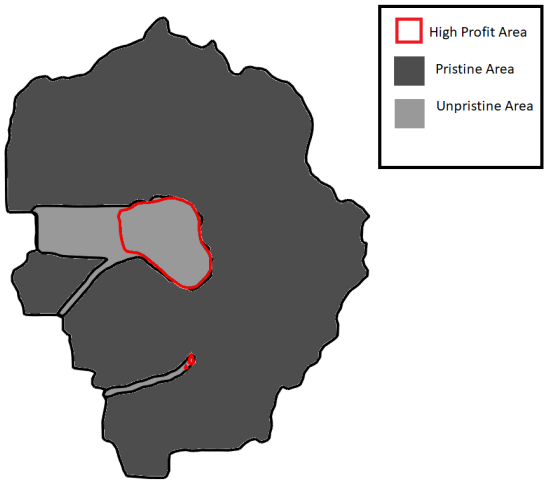


Figure 2: Yosemite national park, attackers moving in from the boundary

the analysis and data organization that went into building a foundation for our model. Using this data analysis as justification we then build the theory and implementation of a farming model which describes deforestation for agricultural purposes. Considering the other type of deforestation event that Slough et al. [23] observed, we construct a logging model which describes deforestation for the purpose of acquiring and selling timber. We discuss the control-theoretic framework of this latter model and conclude with the results of both models, and suggested directions for future work.

2 Preparation and Analysis

In the beginning of the project, we were given a data set named DETER. The purpose of DETER is to monitor and prevent deforestation in Brazil, specifically Mucajaí, a municipality within the state of Roraima, Brazil [23]. In the DETER data set, the space is discretized with polygons and each polygon is assigned a Boolean value to determine whether or not a deforestation event occurred in that polygon in each year and month. Information such as distance to roads and distance to river for each polygon is also included. This set contained data from 2006 to 2015.

We then found another data set, PRODES [15]. This data set is the official data set the Brazilian Government uses to make annual statistics relating to deforestation. The major difference between DETER and PRODES is that the purpose of PRODES is to observe deforestation to make annual statistics. PRODES only includes yearly data from 2001 to 2015.

2.1 Data Analysis

Our goal was to use both of these data sets to generate an accurate model of deforestation in Brazil. Our first step was visualizing the data set to have a better intuition for deforestation in Brazil. In figure 3 we plot the location of deforestation events, a portion of the roads and all the rivers near Mucajaí. One of the first observations we made about the data set is that most deforestation events happen near the roads and the events that are farther away tend to be very close to the rivers. In fact, 90 percent of the deforestation events from the PRODES data set lie within 5 km of the roads. This observation makes sense as proximity to the roads or rivers reduces the time it takes to travel to the extraction site and the city, which reduces the cost an extractor associates with going to the extraction site.

After this observation, we wanted to generate a probability density function (PDF) from the data set. Suppose we are given a set of points $\{\boldsymbol{x}_i\}_{i=1}^N$ where N is very large (for the PRODES data set $N \approx 36,000$) sampled from a PDF ϕ. Then we would like to reconstruct ϕ from these samples. This is an ill-posed problem since there are infinitely many PDFs that could give rise to the data. For instance one popular method known as kernal density estimates (KDE) fixes a PDF f with its mass centered at 0, then fixes a parameter $h > 0$ called the bandwidth. The predicted PDF is

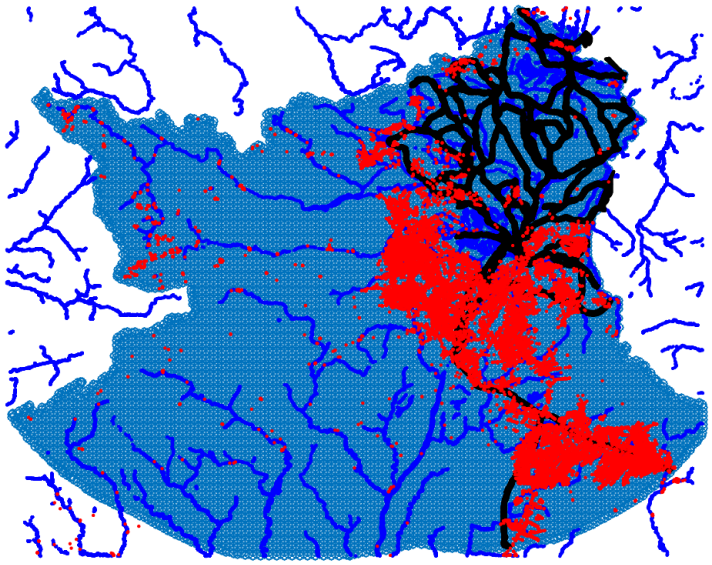


Figure 3: Dark blue is rivers, black is a subset of roads, light blue is area close to Mucajaí, and red dots correspond to deforestation events

defined by

$$\Phi(x) := \frac{1}{Nh^2} \sum_{i=1}^{N} f\left(\frac{\boldsymbol{x} - \boldsymbol{x}_i}{h}\right)$$

Thus Φ is a PDF where the points $\{\boldsymbol{x}_i\}_{i=1}^N$ were very likely to be chosen since f mass is centered at the origin. A popular choice of f is the standard Gaussian. We implemented this method with f as the standard Gaussian and $h = 3$ km. In figure 4 we plot a heat map of the PRODES data set with the color being dependent on Φ. The more yellow spots correspond to larger Φ.

Another popular method is maximizing a regularized logarithmic likelihood. We first motivate the maximization of log likelihood. Assume $\{\boldsymbol{x}_i\}_{i=1}^N$ was drawn independently and identically distributed with density ϕ. Then our goal is to find a PDF Φ such that it maximizes the chances of $\{\boldsymbol{x}_i\}_{i=1}^N$ being chosen and use Φ as an approximation for ϕ. Observe that as $\{\boldsymbol{x}_i\}$ are assumed to be independent

$$\mathbb{P}(\{\boldsymbol{x}_i\}_{i=1}^N) = \Pi_{i=1}^N \mathbb{P}(\boldsymbol{x}_i)$$

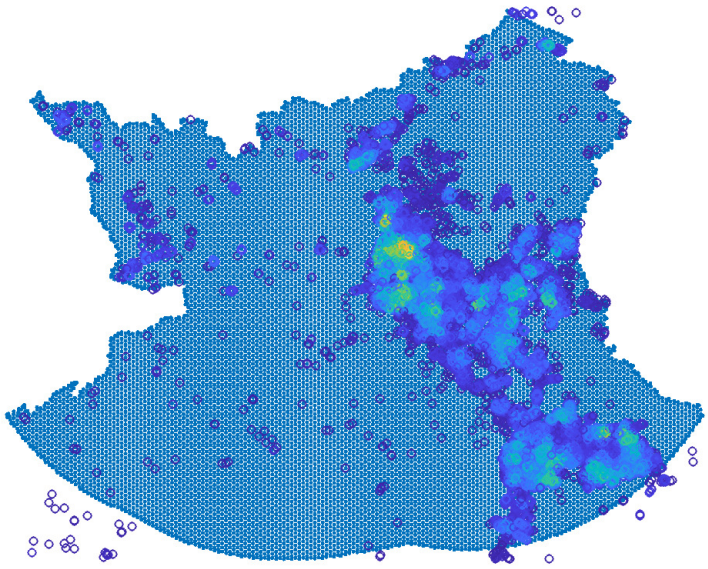


Figure 4: KDE with standard Gaussian and $h = 3$ km on PRODES data set

i.e., the probability of choosing the set $\{\boldsymbol{x}_i\}_{i=1}^N$ is equal to the product of choosing each event. We now take logarithms on both sides to convert the product into a sum:

$$\log(\mathbb{P}(\{\boldsymbol{x}_i\}_{i=1}^N)) = \sum_{i=1}^N \log(\mathbb{P}(\boldsymbol{x}_i))$$

Since $\log(x)$ is an increasing function, maximizing the probability of the set being chosen is equivalent to maximizing the log likelihood $\sum_{i=1}^N \log(\mathbb{P}(\boldsymbol{x}_i))$. The maximum log likelihood method approximates ϕ by

$$\phi \approx \underset{v, v\geq 0, \int v = 1}{\arg\max} \sum_{i=1}^N \log(v(\boldsymbol{x}_i))$$

However, this maximization problem tends to be lead to non-smooth PDFs. Accordingly, the method of maximized penalty likelihood estimate (MPLE) actually defines

$$\phi \approx \underset{v, v\geq 0, \int v = 1}{\arg\max} \sum_{i=1}^N \log(v(\boldsymbol{x}_i)) - f(v, \boldsymbol{x}) \tag{1}$$

where f is a regularization term added to increase smoothness of the approximation PDF. It has been observed that the PDFs generated by MPLE with sharp gradient changes lead to the PDF being not smooth, so a popular f is

$$f := -\alpha \int_{\Omega} |\nabla v|$$

where $\alpha > 0$ and Ω is the spatial domain we are interested in. This regularization term helps reduce sharp gradient changes, which leads to a smoother PDF.

To approximate a solution to (1), we followed [14] and considered the following discrete version of (1)

$$\arg\min_{v \geq 0} \left\{ \sum_{i,j} |\nabla v_{i,j}| - \mu \sum_{i,j} w_{i,j} \log(v_{i,j}) \right\} \quad \text{subject to } \sum_{i,j} v = 1 \tag{2}$$

where w is a weight matrix and $\nabla v := \left(\frac{v_{i+1,j} - v_{i,j}}{\Delta x}, \frac{v_{i,j+1} - v_{i,j}}{\Delta y} \right)$ and $\mu > 0$. We solve this constrained optimization problem by adding two penalty functions for each constraint

$$\arg\min_{v \geq 0} \sum_{i,j} |d_{i,j}| - \mu \sum_{i,j} w_{i,j} + \frac{\lambda}{2} \sum_{i,j} |d_{i,j} - \nabla v_{i,j}| + \gamma (1 - \sum_{i,j} v_{i,j})^2$$

where $\lambda, \gamma > 0$ are fixed. These are punishment terms for not satisfying the constraints. To more strongly enforce this constraint, the authors of [14] solved

$$(u^k, d^k) = \arg\min_{v \geq 0, \boldsymbol{d}} \left\{ \sum_{i,j} |d_{i,j}| - \mu \sum_{i,j} w_{i,j} + \frac{\lambda}{2} \sum_{i,j} |d_{i,j} - \nabla v_{i,j} - \boldsymbol{b}^{k-1}| \right. \\ \left. + \gamma (1 - \sum_{i,j} v_{i,j} - b_1^{k-1})^2 \right\}$$

where

$$\boldsymbol{b}^k = \boldsymbol{b}^{k-1} + \nabla u^k - \boldsymbol{d}^k$$

$$\boldsymbol{b}_1^k = \boldsymbol{b}_1^{k-1} + \sum_{i,j} u_{i,j}^k - 1$$

These vectors $\boldsymbol{b}^k$ and $\boldsymbol{b}_1^k$—the so-called 'Bregman' vectors—are initialized so $\boldsymbol{b}^1 = \boldsymbol{b}_1^1 = \mathbf{0}$. These vectors introduce a harsher penalty to the minimization problem for straying away from the constraints. Then we used the authors' code to solve this minimization problem which converges with speed $O(n^2)$. The authors recommended the usage of $\mu = 10^{-4}$, $\lambda = 2\mu n^4$, and $\gamma = 2\mu n^2$. We tested this choice of μ with a 10-fold cross validation for $\mu = 10^{-i}$ for $i = 0,\ldots,10$ and found that $\mu = 10^{-4}$ was optimal.

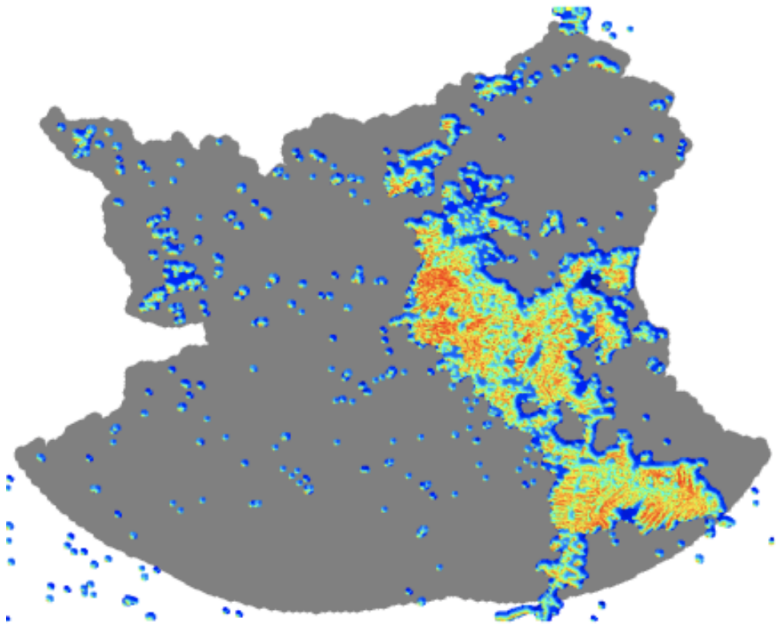
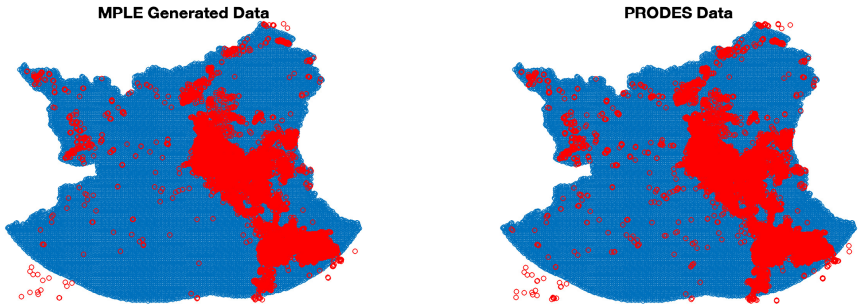


Figure 5: MPLE with $\mu = 10^{-4}$ on PRODES Data Set

In figure 5 we plot a heat map of MPLE. The red locations correspond to larger values of the density function. To check the accuracy of MPLE, we generated random events with the MPLE PDF and compared it to real data. We plot the results of this random sampling in figure 6. This figure shows that the predicted event locations from MPLE are almost identical to the actual data, which leads us to believe MPLE is a good PDF approximation for our purposes.

We also used the DETER data set to see if there is a seasonal component to deforestation. In figure 7, we plot the average percentage of deforestation that occurs in each month. This bar graph shows that in May to July there is a strong drop in deforestation compared to other months. Those months correspond to the rainy season in Mucajaí, confirming our suspicions that the data expresses seasonality.



(a) Deforestation location predicted by MPLE

(b) Deforestation location from PRODES

Figure 6: A comparison of PRODES deforestation data and synthetic data generated from the MPLE density function

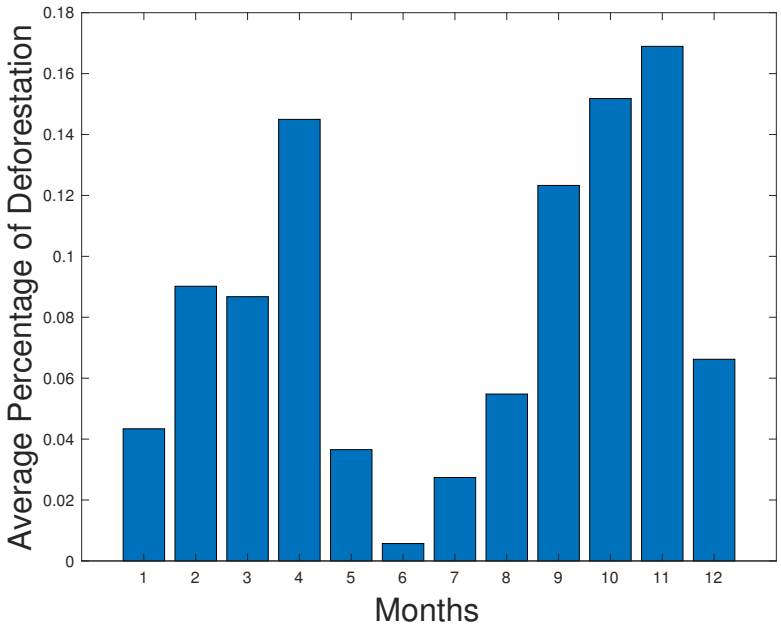


Figure 7: Seasonality of Deforestation in Mucajaí

2.2 Indicator Function of Trees

Since we are trying to model deforestation events, we need to know exactly where the trees are. Therefore, we need to define the indicator function of trees for all grid points in our region of interest. The PRODES data set provides some information about tree coverage locations, so we use the

PRODES data with some modifications to obtain our indicator function of trees over the years from 2001 to 2015. The specific algorithm we use to construct the indicator function of trees for each year is as follows:

1. Use the previous year's forest coverage location data for the next year's initial indicator function of trees. For example, we use year 2004's forest coverage location data for the year 2005's initial indicator function of trees. For $n = 2009, \ldots, 2015$, we denote the initial indicator function of trees in year n to be $\chi^0_{\text{trees},n}$. We only use such n because PRODES data set only provides forest coverage location data for the years 2008 to 2014 in our region of interest.

2. Get the modified indicator function of trees for the year 2009 by setting the locations where deforestation happened in year 2009 to be 1. That is, the modified indicator function of trees for the year 2009 is

$$\chi^1_{\text{trees},2009} = \min(1, \chi^0_{\text{trees},2009} + \chi_{\text{events},2009}),$$

   where $\chi_{\text{events},2009}$ is the indicator matrix of events in the year 2009.

3. For years from 2008 to 2001, we trace backwards. That is, for $n = 2008, 2007, \ldots, 2001$, we first let $\chi^0_{\text{trees},n} = \chi^1_{\text{trees},n+1}$. Then we get the modified indicator function of trees for the year n by setting the locations where deforestation happened this year to be 1:

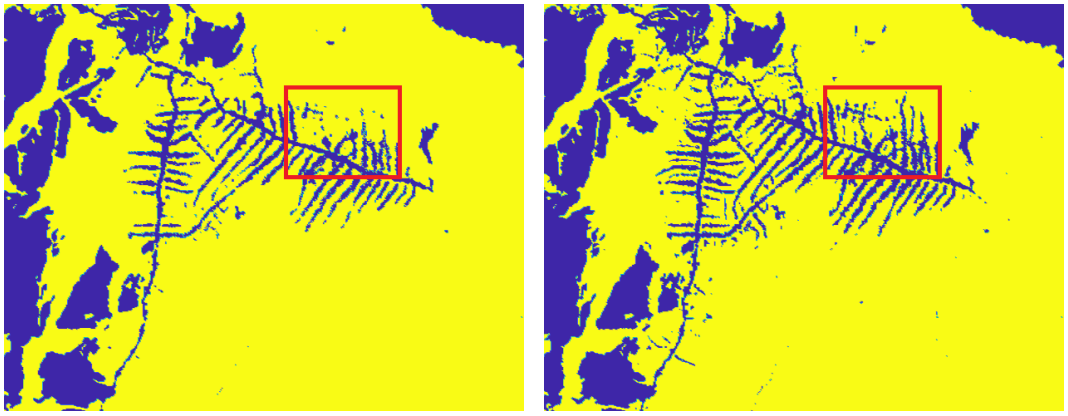
$$\chi^1_{\text{trees},n} = \min(1, \chi^0_{\text{trees},n} + \chi_{\text{events},n}).$$

4. For years from 2010 to 2015, we get the modified indicator function of trees for the year n by setting the locations where deforestation happened this year to be 1:

$$\chi^1_{\text{trees},n} = \min(1, \chi^0_{\text{trees},n} + \chi_{\text{events},n}).$$

5. Obtain the final indicator function for trees in the year 2015 as $\chi_{\text{trees},2015} = \chi^1_{\text{trees},2015}$.

6. Make sure that the indicator function will not increase over the years, by taking the maximum of two consecutive years to modify indicator functions of trees backwards. That is, for $n = 2014, 2013, \ldots, 2001$, we trace back to get the final indicator function for trees to be $\chi_{\text{trees},n} = \max(\chi^1_{\text{trees},n}, \chi_{\text{trees},n+1})$.



(a) Indicator function of trees for 2001 (b) Indicator function of trees for 2015

Figure 8: Indicator function at the beginning and end of the time period

Figure 8a and 8b show the indicator function of trees for the year 2001 and 2015. The yellow color denotes value 1 and blue denotes value 0 for the function. There is significant difference inside the red box as trees have been added as we trace back to 2001.

3 Farming Model

Slough et al. [23] concluded that there are two main types of deforestation that occur in Roraima. The first type is agricultural deforestation, wherein farmers illegally expand their farms to increase their crop output. In this section, we describe how we model this phenomenon. The second type of deforestation is illegal acquisition and sale of timber. This is addressed in section 4.1.

As mentioned in the previous section, we observe that most deforestation events occur near the roads or cities. Further, in the southeast portion of Roraima, there is a region with many roads and small cities. We focus on modelling illegal farming in this region, pictured in figure 9.

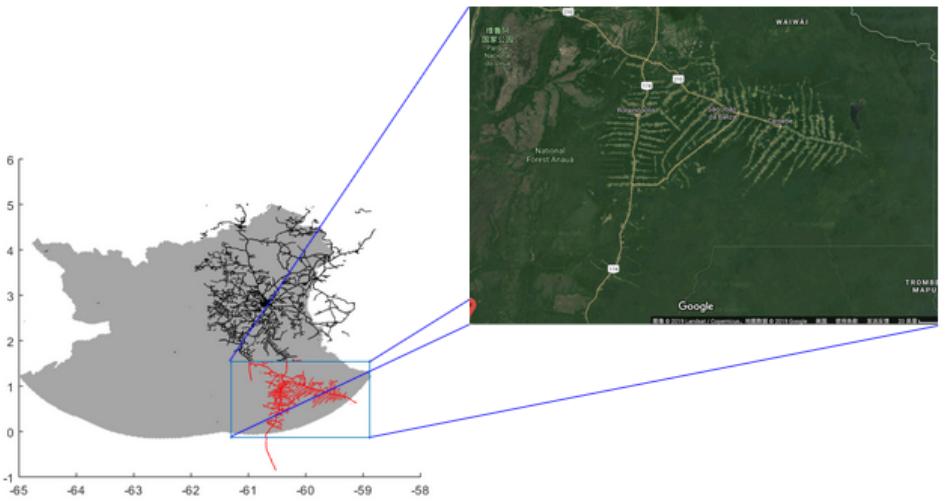


Figure 9: The southeast region of Roraima where we test our farming model.

3.1 Overview of the Farming Model

We first assume that the farmers have perfect information regarding patrol. Also, farmers choose locations for farms probabilistically based on expected profit. To define the expected profit, we first have to define the cost and expected benefit.

Because most events happen near cities and along the roads the cost has dependence on the distance to cities and distance to roads. In addition, we observed that the deforestation events tend to cluster around one another. Therefore, we defined the cost at each point per month in the year n as

$$C(x,y) = \frac{c_1 \cdot d_r(x,y) + c_2 \cdot d_c(x,y) + c_4}{1 + c_3 \cdot A^n(x,y)}, \tag{3}$$

where $d_r(x,y)$ denotes the distance to roads at location (x,y), and $d_c(x,y)$ denotes the distance to cities at location (x,y). The distances are calculated from the center of a farm to the nearest road or nearest city. The measurement of fixed costs associated to running a farm is denoted as c_4. Note that $c_1, c_2, c_3, c_4 \geq 0$ can be adjusted to fit the deforestation data. The final calculation of expected profit should be the sum of all costs in related months. This is because there exist ongoing costs when farmers are travelling back and forth. The accessibility at location (x,y) in the year n is denoted as $A^n(x,y)$. For Accessibility, we first define $\{(x_i^{(n)}, y_i^{(n)})\}_{i=1}^{M(n)}$ as the deforestation events which occurred in the year n. Then we fixed a parameter $h > 0$ called the bandwidth and define the accessibility contributed by the year n as

$$A_h^n(x,y) := \sum_{i=1}^{M(n)} \frac{1}{2\pi h} \exp\left(-\frac{(x - x_i^{(n)})^2 + (y - y_i^{(n)})^2}{h^2} \right) \tag{4}$$

Finally we obtain the accessibility for location (x,y) in the year n as

$$A^n(x,y) := \sum_{k=1}^{3} \frac{1}{2^k} A_h^{n-k}(x,y) \tag{5}$$

So accessibility is a weighted sum of Gaussian with expected values at the deforestation locations and standard deviation h where the influence of each Gaussian decays over time by a factor of $\frac{1}{2}$ for each year that has passed and we consider only the previous 3 years events. This assumes that farmers develop infrastructure as they plant farms, reducing costs for nearby farms.

Now we move on to define expected benefit. First we defined the benefit per square meter per month by:

$$B(x,y) := \xi \chi_{\text{trees}}(x,y) \frac{1}{1 + ||\nabla E(x,y)||^\gamma}$$

where $\chi_{\text{trees}}(x,y)$ is the indicator function for the presence of a tree at location (x,y), $E(x,y)$ is the elevation, and ξ represents the conversion of the value of a tree to currency. Typically, we choose $\xi = 5000$. We scale this factor by $\frac{1}{1+||\nabla E(x,y)||^\gamma}$ because if the area has sharp changes in

elevation, then it will be harder to farm, while if it is almost flat then $||\nabla E(x,y)|| \approx 0 \Rightarrow B(x,y) \approx \xi\chi_{\text{trees}}(x,y)$.

Benefit and cost are not the only factors that farmers consider when choosing a farm location. They also account for the probability of getting caught illegally making and expanding their farms in their expected benefit calculation. In any given month, we model the probability of not being captured by

$$1 - \psi(x,y) \cdot S$$

where $\psi(x,y)$ is capture probability at location (x,y) (dependent on patrol) and S is the area of the farm.

Assume a farm starts with area S_0, and expands for i months at a rate of dS, then the size of the farm at the end of the i^{th} month will be $S_i = S_0 + i \cdot dS$. As the minimum area of deforestation in PRODES is about 10 square meters and the average is about 10^5 square meters per year, we choose $S_0 = 10, dS = \frac{10^5}{12}$. Thus, the probability of not being captured by the end of the i^{th} month at location (x,y), if we assume that the capture probability in different months are independent, is

$$\beta_i(x,y) = \prod_{j=1}^{i}(1 - \psi(x,y) \cdot S_j).$$

The expected benefit farmer will receive in month i is

$$\mathbb{E}(\text{Benefit}_i(x,y)) = B(x,y) \cdot S_i \cdot \beta_i(x,y) - \alpha \cdot B(x,y) \cdot S_i \cdot (1 - \beta_i(x,y))$$

Here $\alpha > 0$ is a parameter that measures the strength of punishment. For example, $\alpha = 2.5$, means that if a farmer is caught, then he will be fined with the amount of 2.5 times the benefit he will gain from that farm in month i. And the expected benefit which a farmer will receive by the end of month i is

$$\sum_{j=1}^{i} \mathbb{E}(\text{Benefit}_j(x,y))$$

As a farm cannot last forever partly due to lack of nutrition in the land, we assumed that a farm can only operate for at most $K_{\max}$ months. For our experiments, we choose $K_{\max} = 36$. Therefore, a farmer will chose a month between $1, .., K_{\max}$ such that they maximize their expected profit.

Thus, the expected profit function for a farmer is

$$\mathbb{E}(P(x,y)) := \left(\max_{i=1,\ldots,K_{max}} \sum_{j=1}^{i} \left(\mathbb{E}(\text{Benefit}_j(x,y)) - C(x,y) \right) \right)_+$$

where $(f)_+ := \max(f, 0)$. So if farmers choose a spot (x, y), then they will stay in that spot until they gain the expected highest profit or until get caught.

Then we normalize $\mathbb{E}(P(x, y))$ to make it into a probability density function. Farmers will choose where to plant their farms randomly with this density.

3.2 Fitting Parameters

To fit the model to our data, we try to optimize our set of parameters by maximizing the average F1 score, as defined later in this section. Note that we are fitting the parameters using the predicted model and PRODES data from 2005 to 2015, on the bottom right part of the region described in PRODES, as in Figure 9. The range of area is [-61.3, -58.8868] in longitude and [-0.1293, 1.54] in latitude. We fit 500×500 grid points in this region to generate our results and fit the parameters.

Due to sparsity of the events, the way we define true positive instances should have some tolerance of position offset. Specifically, given a predicted probability density function for illegal farming, the true data points where events happen in the region from 2005 to 2015, and a threshold, we claim that an event is truly predicted if for a square centered at the event point, there is at least one point within the square with predicted probability density function value larger than or equal to the threshold.

First we denote the grid points to be (x_i, y_j), for $i, j = 1, \ldots, 500$. Then we define the matrix that counts the number of events in each grid in the year n by NE^n, where $NE^n(x_i, y_j)$ is the number of real events in year n whose location has (x_i, y_j) as the nearest grid point. Denote the probability density function generated by the expected profit for the year n to be f^n. Then the event prediction function can be defined as

$$p^n(x_i, y_j) := \chi_{f^n(x_i,y_j) > \epsilon \cdot NE^n(x_i,y_j)} \cdot \chi_{NE^n(x_i,y_j)>0},$$

where ϵ is the threshold for prediction, and χ is the indicator function. For a grid point to have events, we require $NE^n(x_i, y_j) > 0$. To account for multiple events, we require $f^n(x_i, y_j) > \epsilon \cdot NE^n(x_i, y_j)$, which means that if there are, for example, two events at a grid in the year n, then to claim that a grid point is predicted, we require the probability density function by the expected profit to be at least twice that of the threshold.

Now we define the notion of true positive and false positive. To deal with sparsity of the events, we define the true positive function for the year

n by

$$tp^n(x_i, y_j) := \max\{p^n(x, y) : (x, y) \in \mathbb{U}(x_i, y_j)\},$$

where $\mathbb{U}(x_i, y_j)$ is a neighborhood of the point (x_i, y_j). For our experiment, we take a square centered at (x_i, y_j). That is, we define $\mathbb{U}(x_i, y_j) := \left\{(x_{i+di}, y_{j+dj}) : di, dj = -4, \ldots, 4,\ i + di, j + dj \in [1, 500]\right\}$.

To define the false positive function, we omit the tolerance region. That is, we define the false positive function for the year n by

$$fp^n(x_i, y_j) := \chi_{f^n(x_i,y_j)>\epsilon} \cdot \chi_{NE^n(x_i,y_j)=0}.$$

Therefore, the number of true positive points and false positive points in the year n can be calculated as $TP^n := \sum_{i=1}^{500} \sum_{j=1}^{500} tp^n(x_i, y_j)$ and $FP^n := \sum_{i=1}^{500} \sum_{j=1}^{500} fp^n(x_i, y_j)$ respectively.

The recall (true positive rate) for the year n can then be calculated as

$$R^n := \frac{TP^n}{\text{number of events in year } n} = \frac{TP^n}{\sum_{i=1}^{500} \sum_{j=1}^{500} NE^n(x_i, y_j)},$$

and precision for the year n can then be calculated as

$$P^n := \frac{TP^n}{\text{number of 'predicted' events in year } n} = \frac{TP^n}{TP^n + FP^n}.$$

The 'predicted' here is not quite well-defined, as we consider a tolerance neighborhood only for true positive events but not for false positive events. However, by summing over the number of true positive events and false positive events, we can approximate the notion of the number of 'predicted' events.

Finally, we have the F1 score for each year:

$$F1^n := \frac{2 \cdot (P^n \cdot R^n)}{P^n + R^n}. \tag{6}$$

We then average the F1 scores to obtain the average F1 score for the predicted model, and optimize parameters $c_1, c_2, c_3, c_4, c_p, \gamma, \epsilon$ and α to achieve the highest average F1 score, where ϵ is the threshold we use to define the true positives. Here, c_p is a parameter to fit in our initial model, where we use $\psi^n = c_p \cdot mf^{n-1}$, and mf^{n-1} is the probability density function for deforestation events generated using the MPLE method from PRODES for the region in the previous year, year $n - 1$. The optimized set of parameters we find is $c_1 = \frac{1}{1000}, c_2 = \frac{1}{10000}, c_3 = 10, c_4 = 10, c_p = 1, \gamma = 1, \epsilon = 10^{-5}$;

and $\alpha = 2.5$. These yield an average F1 score of 0.59. This score is not perfect because of the sparsity of real deforestation events, but the average recall (true positive rate), which measures the ability of our model to predict where events will happen, is 0.72, which is better.

3.3 Time Series Model

In the farmer's model so far, the Profit density has been static. Now we consider a simulation wherein multiple farmers plant their own farms. Then the benefit near their selected farm should decrease as the next farmers cannot farm there. Hence, the Profit density should be updated based on each farmer. Accordingly, we want to design a time series model with the goal of accurately predicting deforestation events caused by farmers.

We assume that for a fixed year N that we have data about the deforestation events for year $N-1, N-2$, and $N-3$ and a patrol density $\psi(\boldsymbol{x})$. We first generate an initial Profit PDF by following the procedures in section (3.1). Now we randomly generate a point $\boldsymbol{x}_1 \in \Omega$ according to the profit PDF where Ω is our spatial domain. This node $\boldsymbol{x}_1$ represents the location of where the first farmer of this year decided to plant a farm. In particular, we assume that the farmer creates a square farm centered at $\boldsymbol{x}_1$ with length $\sqrt{S_0}$ (and hence area S_0). Now we expect the benefit to decrease near this farm, so if we fix $K \in [0, 1)$, and let $\mathcal{S}_1$ denote square farm belonging to the first farmer, then we do the following update to benefit:

$$B(\boldsymbol{x}) = B(\boldsymbol{x}) \cdot \min\{1, K + \text{dist}(\boldsymbol{x}, \mathcal{S}_1)\}$$

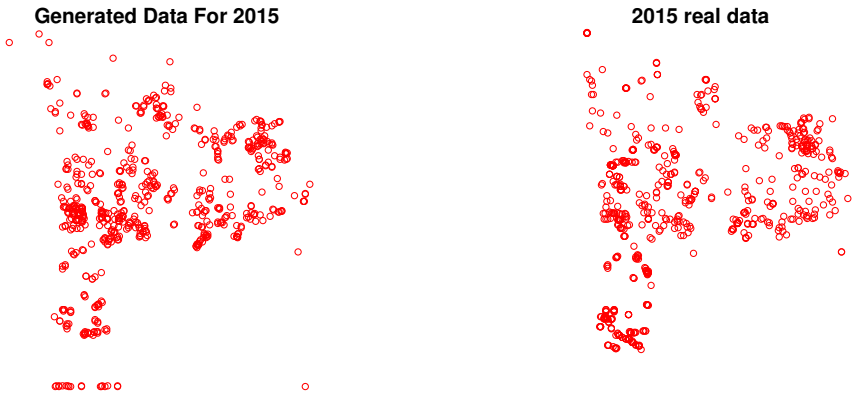
This update decreases the benefit linearly based on distance to the farm with points in the farm benefit being decreased by the factor K. In practice, we chose $K = \frac{1}{2}$ and we choose to keep $K > 0$ because in our data set, multiple events happened on the same node which is due to the coarseness of our grid. In addition, as the farmer constructs a farm at $\boldsymbol{x}_1$ we expect this farmer to create infrastructure which makes it more accessible for other farmers to get near $\boldsymbol{x}_1$. Therefore, we update accessibility by the following formula

$$A(\boldsymbol{x}) = A(\boldsymbol{x}) + \frac{1}{2\pi h} \exp\left(-\frac{||\boldsymbol{x}_1 - \boldsymbol{x}||^2}{h^2}\right)$$

Then we normalized A so that it retains its initial mass.

Now we also assume that farmer 1 wants to expand his farm's area by dS by the end of this month, but as this activity is illegal, there is a chance

he can get caught while expanding their farm. To simulate this we do a Bernoulli trial with probability $p = \psi(\boldsymbol{x}_1) \cdot S_0$. If the trial is successful, the farmer is caught, so he does not expand his farm and he also receives a fine of $\alpha B(\boldsymbol{x}_1)S_0$. Otherwise, if the trial is a failure, the farmer is not caught and expands his farm to by a constant factor dS. Now we use the updated benefit and accessibility score to update the Profit PDF. We repeat this procedure for all the farmers we want to simulate in the first month.



(a) Time Series Model for 2015 (b) Data from PRODES data set for 2015

Figure 10: Time Series Generated Data compared to Real Data

Now assume we are in month i where $1 < i \leq 12$. We update ψ to account for changes in patrol strategy. Next we define $N(i)$ to be the number of farmers released in the simulation from month 1 to the end of month i, $\mathcal{S}_j$ denote the farm of the jth farmer, and $\boldsymbol{x}_j$ as the center of the square farm of farmer j. Now for $1 \leq j \leq N(i-1)$ we simulate a Bernoulli trial with $p = |S_j|\psi(\boldsymbol{x}_j)$ to determine whether or not they get caught expanding their farm this month. Then we update the Benefit generated by each farmer j for each new node inside S_j and update the Profit. We repeat the initial procedure with this updated Profit to generate new farmers location and repeat this process for all desired months. In figure 10 we plot the Time Series prediction for centers of the farms created in 2015 compared to the deforestation events that occurred in 2015 according to PRODES data. In figure 11 and 12 we plot an exaggerated square farms corresponding to the farms the farmer chose in the months June and December. The red cross indicates the farmer got caught expanding their farm, and we had 34 farmers getting caught in the simulation for 2015 out of 626 extractors.

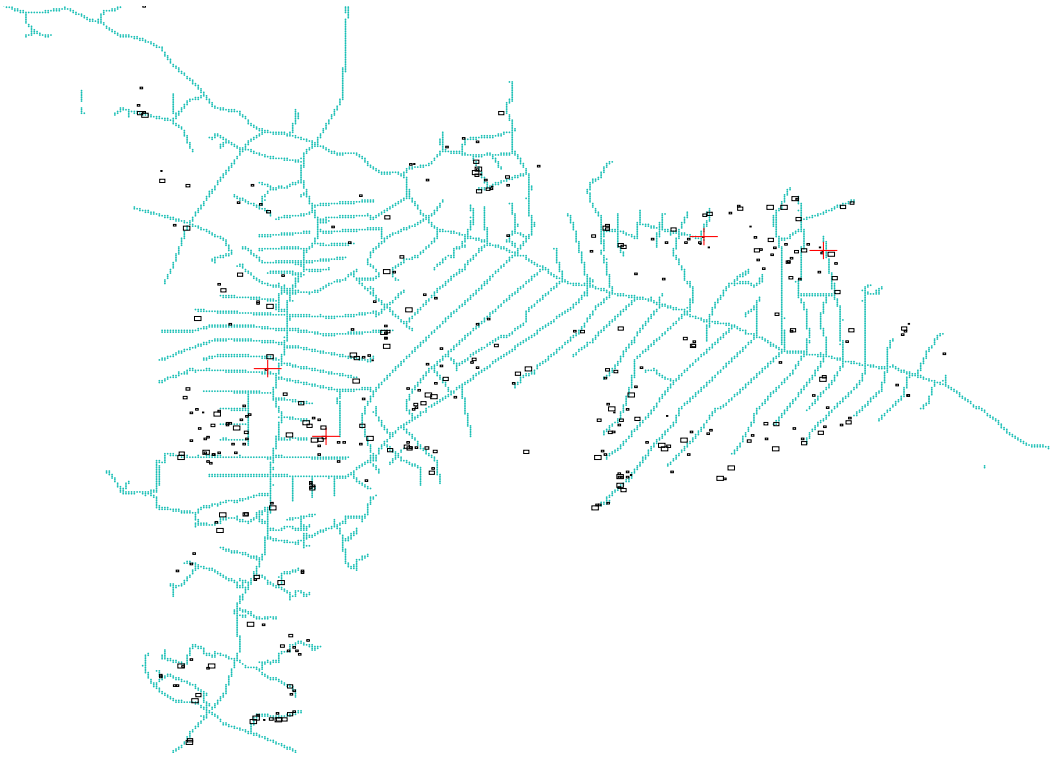


Figure 11: Result of time series simulation in June of 2015. Black boxes correspond to farms and red crosses means the farmer was caught. The sizes of the farm has been exaggerated to make farms more visible.

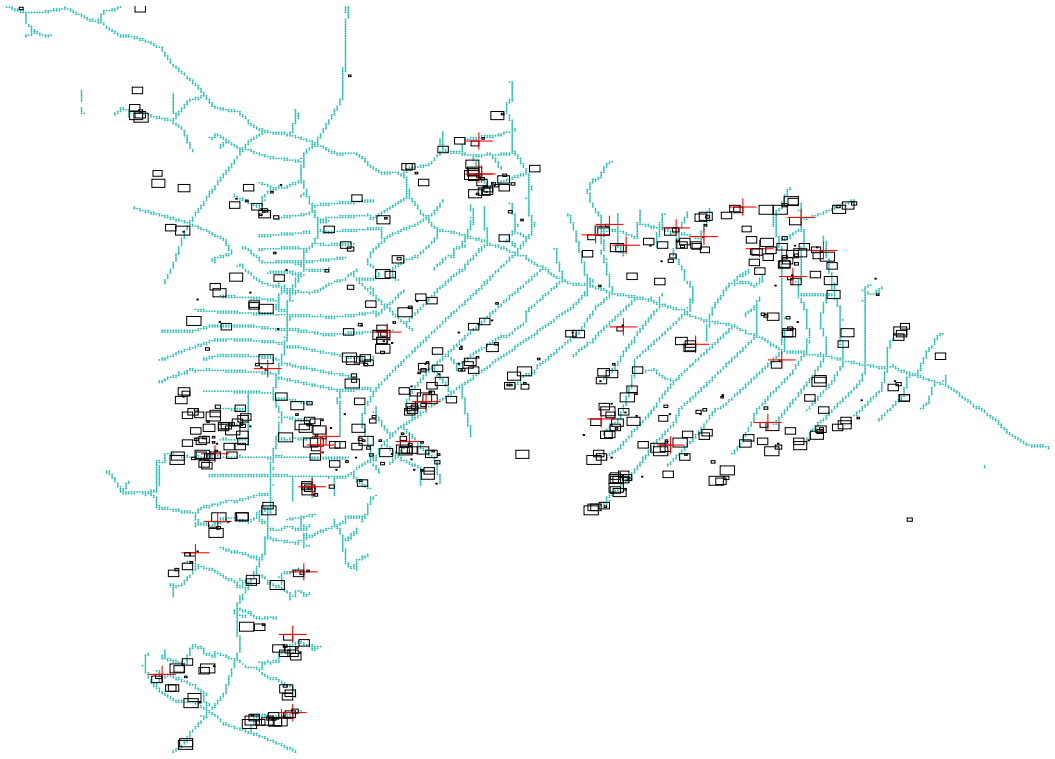


Figure 12: Result of time series simulation in December of 2015. Black boxes correspond to farms and red crosses means the farmer was caught. The sizes of the farm has been exaggerated to make farms more visible.

4 Logging model

Illegal logging is another contributing factor to deforestation, which exhibits quite different behavior from farming. Loggers obtain profit from timber rather than land. They may go deeper into the forest where trees are more valuable. Judicious path planning that balances both travel time and capture risk then plays a major role in loggers' decision making. In our model, we assume that deforestation events at least 10 km from the highways are logging events. We build up a more realistic model to describe loggers' decision making process which can be useful to evaluate, compare and design patrol strategies.

Our logging model is based on Albers [1] where loggers want to maximize their expected profit P, defined as

$$P = \Phi B - C.$$

Here B is the pre-determined initial benefit that depends on the category and quality of timber and is assumed to be static. C represents the cost and is measured by the traveling time of both going in and out of the forest. Φ describes the probability of not being captured, which depends on patrollers' detection ability and loggers' trajectories. We follow the Stakelberg game model and assume loggers have perfect information about patrol.

Diverging from previous models where extracting happens instantaneously, we introduce the notion of logging time. Actual logging time will influence the risk of being detected and the amount of trees loggers can obtain. The latter will further influence the travel velocity when loggers return from the forest.

The loggers then need to choose the optimal logging spot, along with logging time and traveling path. To address the resulting optimization problem in random domain geometries, we transform the optimal path planning problem into a time-dependent Hamilton Jacobi equation and appeal to the level set method as in Arnold et al. [2]. We then perform a parameter sweep to find the optimal logging location and time.

In section 4.1 we will fully formulate the problem. The resulting optimization problem will be further addressed in section 5.

4.1 Model Construction

We assume each location x in the domain $\Omega \subset \mathbb{R}^2$ has a fixed amount of timber, but with different total value $B(x)$. Different from previous mod-

els where logging happens instantaneously, we use t to denote the actual logging time. Under the constant production rate assumption, $\frac{t}{T}B(x)$ gives the actual benefit if there were no patrollers, where T is a global constant representing the time to clear all the trees in one location.

Loggers can be detected when they are logging or on their path back while returning with their illegal goods. We assume that the capturing event is a Poisson process, with (capture) intensity $\psi : \Omega \to \mathbb{R}$ known to loggers. The probability of not being captured while logging at x after time t is $e^{-\psi(x)t}$. Longer logging time means larger benefits, but also larger risk of being detected. The probability of not being captured when they are walking back following path $X(s)$ is then given by $e^{-\int_0^\tau \psi(X(s))ds}$ where τ is the traveling time. Here we assume the loggers will only get captured when they actually have timber with them and they will lose all of the benefit after being detected. The expected benefit one can obtain by logging at location x for time t is then $B(x)\frac{t}{T}e^{-\psi(x)t}e^{-\int_0^\tau \psi(X(s))ds}$.

The cost, represented by the travel time is easy to calculate given a path and the associated velocity. In our model, we firstly define the move-in velocity field $v : \Omega \to \mathbb{R}^2$ following the transportation system, so that patrollers travel with highest velocity when they are on major highways and a little bit slower when they are on waters and secondary highways. When patrollers are off highways or waters, their velocity is scaled according to terrain slope following Arnold et al. [2]. When they travel out, we assume their velocity will decrease because of the loaded cars or boats in the form of $v(x)/(1 + c(t/T)^\gamma)$, where t is the logging time mentioned before and c, γ are two parameters that model the impact of traveling with trees. The increased traveling cost may be another reason that loggers decide to spend less than the maximal logging time.

The previous analysis leads to a more realistic way of calculating profit for logging at position x for time t and following path X_{in}, X_{out}. The profit function is

$$P(x, t, X_{in}, X_{out}) = B(x)\frac{t}{T}e^{-\psi(x)t}e^{-\int_0^{\tau_{out}} \psi(X_{out}(s))ds} - \alpha(\tau_{in} + \tau_{out}),$$

where α is the coefficient to convert time to monetary value, while τ_{in} and τ_{out} are the traveling time. Rational loggers will then try to solve the optimization problem

$$P_{opt}(x) = \max_{t, X_{in}, X_{out}} P(x, t, X_{in}, X_{out}) \tag{7}$$

and may go to the spots with positive profit. It's worth mentioning that the optimal path going in and out of the forest can be different because of the patrollers.

4.2 Simplification and Approximation

We need to simplify and rephrase the optimization problem (7) before we move on to solve it.

First of all, since X_{in} is independent of the other two terms, we will be able to solve for X_{in} as in the classical minimal time path planning problem. As for the coupled t and X_{out}, we focus on the path planning part in our work. For each given t, we will try to find the optimal X_{out}. In practice, we discretize $[0, T]$ into n levels and solve the model for $t = iT/n$, at each x, where $i = 1, \ldots, n$. We then take the maximum profit among these n candidates as an approximation for the optimal expected profit over the entire period.

Second, the exponential term $e^{-\int_0^{\tau_{out}} \psi(X_{out}(s))ds}$ causes some trouble. Since the capture intensity, i.e., the path integral on the power takes the form of the standard running cost, we use the linearization $1 - \beta \int_0^{\tau_{out}} \psi(X_{out}(s))ds$ as an approximation for some positive constant β. Note that the expected benefit goes to 0 under estimation when $\int_0^{\tau_{out}} \psi(X_{out}(s))ds$ reaches $1/\beta$, while the actual probability of being caught on the way back is $1 - e^{-1/\beta}$. Different β may be used to describe extractors' degree of risk-aversion. When β is set to be 1, the linearization gives the first order Taylor expansion of the exponential term at 0 but the resulting expression is always an under-estimation. Loggers will refuse to go to one spot if the probability of being captured on their way back is greater than $1 - e^{-1}$, which is approximately 0.63. For less risk-averse loggers, we may use a smaller β and the resulting expression can be an over-estimation when the capture intensity is within certain range.

Based on these two modifications, the problem can be rewritten as

$$\begin{aligned} P_{opt}(x,t) \approx & \max_{X_{in}, X_{out}} B(x)\frac{t}{T}e^{-\psi(x)t}\left(1 - \beta\int_0^{\tau_{out}} \psi(X_{out}(s))ds\right) - \alpha(\tau_{in} + \tau_{out}) \\ = & B(x)\frac{t}{T}e^{-\psi(x)t} - \min_{X_{in}} \int_0^{\tau_{in}} \alpha ds \\ & - \min_{X_{out}} \int_0^{\tau_{out}} B(x)\frac{t}{T}e^{-\psi(x)t}\beta\psi(X_{out}(s)) + \alpha ds, \end{aligned} \tag{8}$$

where both minimization problems fall into the category of free-time-fixed-point path planning problem, with the minimal time problem being a typical example. We will move on to solve this optimization problem in the following two sections.

5 Path Planning with Optimal Control Theory

In this section, we will transform the optimal control problem in our model into a Hamilton-Jacobi equation, then describe a method to find the optimal path based on the solution to the Hamilton-Jacobi equation.

When we mention the solution to a Hamilton-Jacobi(-Bellman) equation or eikonal equation below, it always denotes the viscosity solution in the sense of Crandall and Lions [6].

5.1 Static Hamilton-Jacobi-Bellman Equation

A general formulation of the optimal control we need to solve in our model can be written as

$$\begin{aligned} \min_{\alpha \in \mathcal{A}} \quad & \int_0^{\tau} r(x(s))ds \\ s.t. \quad & \dot{x} = v(x)\alpha(x) \\ & x(0) = x_0 \\ & \tau = \inf\{t : x(t) = x_{end}\} \end{aligned} \tag{9}$$

Here $\alpha(\cdot)$ is the control, taken from set of valid control functions $\mathcal{A}$. The function r is always positive and represents some running cost incurred along the trajectory determined by α. Now we transform this problem into a static eikonal equation. From Bardi and Capuzzo-Dolcetta [4], there are two theorems regarding this transformation.

Theorem 1.1: Dynamic Programming Principle (DPP). Suppose $y_x(t, \alpha)$ is the trajectory following the differential equation in (9), starting from x and using the control function α. Define the time of reaching the end point as

$$t_x(\alpha) = \begin{cases} \inf\{t : y_x(t, \alpha) = x_{end}\}, \text{ if } \{t : y_x(t, \alpha) = x_{end}\} \neq \emptyset, \\ +\infty, \text{ otherwise.} \end{cases}$$

Define the minimal cost by

$$u(x) := \inf_{\alpha \in \mathcal{A}} \left\{ \int_0^{t_x(\alpha)} r(y_x(s, \alpha))ds \right\}. \tag{10}$$

Then $u(x)$ satisfies the Dynamic Programming Principle (DPP). That is, for $t \leq t_x(\alpha)$,

$$u(x) = \inf_{\alpha \in \mathcal{A}} \left\{ \int_0^t rds + u(y_x(t, \alpha)) \right\}, \ \forall t \geq 0, \tag{11}$$

Theorem 1.2: Static Hamilton–Jacobi–Bellman equation. Suppose $u(x)$ is defined as in **Theorem 1.1**, then $u(x)$ is the solution to the static Hamilton–Jacobi–Bellman equation

$$\begin{aligned} &\sup_{\alpha \in \mathcal{A}} \{-v(x)\alpha(x) \cdot \nabla u(x)\} = r(x), \\ &u(x_{\text{end}}) = 0, u(x) \geq 0, \ \forall x \in \mathbb{R}^n \end{aligned} \tag{12}$$

Formally, to go from (11) to (12), we move $u(x)$ to the right hand side of (11), divide by t and let $t \to 0$. Using the chain rule and the ODE solved by $y_x(t, \alpha)$, we arrive at (12).

5.2 The Eikonal and Hamilton-Jacobi Equations

In this part we will show the connection between eikonal equation and Hamilton-Jacobi equation. For nonnegative function $v(x)$, we consider a Hamilton-Jacobi equation

$$\begin{aligned} &u_t + v(x)|\nabla u| = 0, \ (x,t) \in \mathbb{R}^n \times (0, +\infty), \\ &u(x,0) = g(x), \ x \in \mathbb{R}^n. \end{aligned} \tag{13}$$

From Dolcetta [8], we have these two theorems:

Theorem 2.1 For any $y \in \mathbb{R}^n$, the equation

$$\begin{aligned} &v(x)|\nabla d(x;y)| = 1, \ x \in \mathbb{R}^n \setminus \{y\}, \\ &d(y;y) = 0, \ d(x;y) \geq 0, \ \forall x \in \mathbb{R}^n, \end{aligned} \tag{14}$$

has a unique solution $d(\cdot\ ;\ y)$.

Theorem 2.2 Assume that $g(x) \geq -C(1 + |x|)$ for some $C > 0$. Define the function

$$\Phi(x) = \begin{cases} 0 & \text{if } x = 1 \\ +\infty & \text{otherwise} \end{cases}$$

Then the function

$$u(x,t) = \inf_{y \in \mathbb{R}^n} \left\{ g(y) + t\Phi\left(\frac{d(x;y)}{t}\right) \right\}$$

is the unique solution of (13) which is bounded below by a function of linear growth and such that

$$\liminf_{(y,t)\to(x,0^+)} u(y,t) = g(x).$$

Consider the initial function $g(x) = |x - z|$, the Euclidean distance to a point z. By **Theorem 2.2**, we have that

$$\begin{aligned} u(x,t) &= \inf_{y\in\mathbb{R}^n} \left\{ |y-z| + t\Phi\left(\frac{d(x;y)}{t}\right)\right\} \\ &= \inf_{\substack{y\in\mathbb{R}^n \\ d(x;y)\,=\,t}} |y-z| \end{aligned} \tag{15}$$

is a solution to the equation (13), where $d(x;y)$ is the solution to the eikonal equation (14).

According to (15), $u(x,t) = 0$ if and only if $d(x;z) = t$. Thus we deduce the connection between the solution $w(x) = d(x;z)$ to the equation (14) where $y = z$ and the solution $u(x,t)$ to the equation (13):

$$w(x) = t \text{ if and only if } u(x,t) = 0.$$

In this way, any eikonal equation can be transformed into a time-dependent Hamilton-Jacobi equation, and vice versa.

Note that the equation (13) is a level set equation as considered by Osher and Sethian [19]. If $g(x)$ is the signed distance function to a contour $\Gamma(0)$, we can evolve $\Gamma(0)$ by level set motion when we solve the PDE. If we define $\Gamma(t) = \{x : u(x,t) = 0\}$, then $\Gamma(t)$ represents the set of points which can be reached from the original contour with traveling time t in the velocity field $v(x)$. This is displayed in figure 13. The interpretation of the level sets $\Gamma(t)$ as the reachable sets after traveling for time t from an initial state gives a connection between the level set equation and the optimal control problem for path planning. In view of the equivalence of (13) and (14), we can obtain the level sets from the solution to either equation; that is, $\Gamma(t) = \{x : u(x,t) = 0\} = \{x : w(x) = t\}$.

5.3 Finding the Optimal Path

In our model, the optimal control problems have the form (9) where $\mathcal{A} = \{\alpha : |\alpha(\cdot)| = 1\}$. If we define the optimal cost $u(x)$ as (10), then $u(x)$ satisfies

$$\sup_{\alpha\in\mathcal{A}}\{-v(x)\alpha(x)\cdot\nabla u(x)\} = r(x).$$

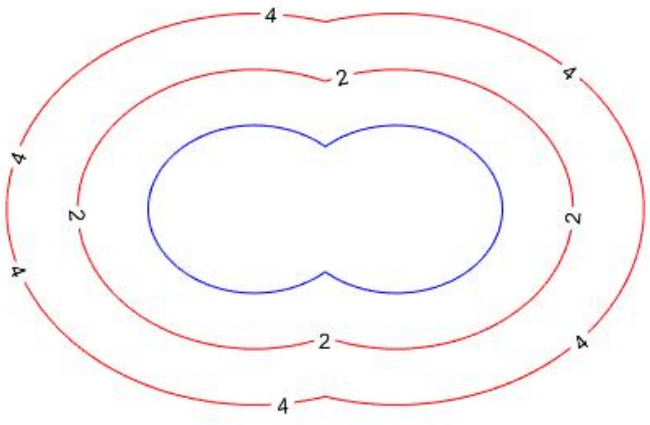


Figure 13: Level set evolution with velocity $v(x) = 1$, from an initial contour $\Gamma(0)$ (blue). The contours $\Gamma(2)$ and $\Gamma(4)$ (red) represent the contours which could be reached after traveling for $t = 2$ or $t = 4$ respectively. These are zero level contours of the solution $u(x,t)$ to equation (13).

Since $v(x) \geq 0,\ \forall x \in \mathbb{R}^n$, we can solve for the optimal control $\alpha(x)$ explicitly using the Cauchy-Schwarz inequality:

$$\alpha(x) = -\frac{\nabla u(x)}{|\nabla u(x)|}.$$

Thus $u(x)$ solves the eikonal equation

$$\begin{aligned} &v(x)|\nabla u(x)| = r(x), \\ &u(x_{\text{end}}) = 0, u(x) \geq 0,\ \forall x \in \mathbb{R}^n. \end{aligned}$$

Since the running cost function $r(\cdot)$ is always positive, we can transform this into the corresponding Hamilton-Jacobi equation

$$\begin{aligned} &\phi_t + \frac{v(x)}{r(x)}|\nabla \phi| = 0,\ (x,t) \in \mathbb{R}^n \times (0, +\infty), \\ &\phi(x,0) = |x - x_{\text{end}}|,\ x \in \mathbb{R}^n. \end{aligned} \tag{16}$$

In order to find the best trajectory for the optimal control problem (9), we can evolve level sets $\Gamma(t)$ outward from x_{end} according to (16). This level sets represent the reachable sets after time t with the velocity field

$v(x)/r(x)$. This time t is no longer traveling time, but rather cost associated with the optimal control problem. Hence the level sets $\Gamma(t)$ represent contours of equal traveling cost. To find the optimal path from x_0 to a desired end point x_{end}. We evolve level sets outward from x_{end} until they reach x_0. That is, we evolve (16) until the time t^* such that $\phi(x_0, t^*) = 0$. This t^* represents the optimal cost associated with traveling from x_0 to x_{end}.

As we solve (16), at each (x, t) we can resolve the optimal control value

$$\alpha(x,t) = -\frac{\nabla\phi(x,t)}{|\nabla\phi(x,t)|}.$$

We can use this to resolve the optimal trajectory, by integrating the ODE in (9) backwards from time $t = t^*$ to $t = 0$, so that the optimal trajectory is given by

$$\begin{aligned} \dot{x} &= -v(x)\frac{\nabla\phi(x,t)}{|\nabla\phi(x,t)|}, \\ x(t^*) &= x_0. \end{aligned} \tag{17}$$

We should solve the ODE backwards because the function $\phi(x, t)$ may not be differentiable at $(x_{\text{end}}, 0)$, and thus we cannot determine the value of the optimal control at that point. Note, the optimal trajectory will always travel parallel to $\nabla\phi$ which is normal to the level sets $\Gamma(t)$.

In summary, we find the optimal trajectory in two steps:

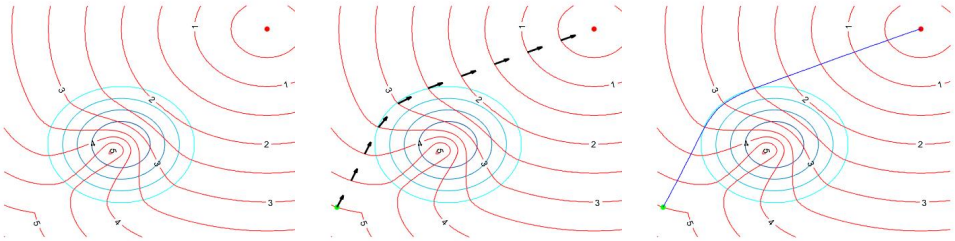
1. **Evolve the level sets**. Solve the equation (16) until the time t^* such that $\phi(x_0, t^*) = 0$. Here t^* is the optimal cost in the problem (9).
2. **Track back**. Solve the dynamic ODE (17) backwards in time, starting from $x(t^*) = x_0$. This ODE gives the optimal path we need.

The process of finding an optimal path is displayed in figure 14.

5.4 Optimal Control Problem in Our Model

As described in the previous section, the loggers in our model attempt to solve the optimization problem

$$\begin{aligned} P_{opt}(x,t) \approx & \max_{X_{in}, X_{out}} B(x)\frac{t}{T}e^{-\psi(x)t}\left(1 - \beta\int_0^{\tau_{out}} \psi(X_{out}(s))ds\right) - \alpha(\tau_{in} + \tau_{out}) \\ =& B(x)\frac{t}{T}e^{-\psi(x)t} - \min_{X_{in}}\int_0^{\tau_{in}} \alpha ds \\ & - \min_{X_{out}}\int_0^{\tau_{out}} B(x)\frac{t}{T}e^{-\psi(x)t}\beta\psi(X_{out}(s)) + \alpha ds. \end{aligned}$$



(a) Level set contours (b) Normal directions (c) Optimal path

Figure 14: To find the optimal path from the start point (green dot) to the end point (red dot), we first evolve the level sets from the end point until they reach the start point, and then let the point travel along the normal direction of the level sets to find the optimal path.

This formula contains two optimal path planning problems: one is the path moving into the forest and the other is the path moving out. Given a start point x_0 and an end point x_{end}, we need to solve these two optimal control problems.
The optimal control problem regarding travel into the forest is:

$$\begin{aligned} \min_{a\in\mathcal{A}} \quad & \int_0^{\tau_{\text{in}}} \alpha ds \\ s.t. \quad & \dot{X}_{in} = v(X_{in})a(X_{in}) \\ & X_{in}(0) = x_0 \\ & \tau_{\text{in}} = \inf\{t : X_{in}(t) = x_{end}\}. \end{aligned} \tag{18}$$

The optimal control problem regarding the path moving out of the forest is:

$$\begin{aligned} \min_{a\in\mathcal{A}} \quad & \int_0^{\tau_{\text{out}}} B(x_{\text{end}})\frac{t}{T}e^{-\psi(x_{\text{end}})t}\beta\psi(X_{out}(s)) + \alpha ds \\ s.t. \quad & \dot{X}_{out} = \frac{v(X_{out})}{1 + c(t/T)^{\gamma}}a(X_{out}) \\ & X_{out}(0) = x_{\text{end}} \\ & \tau_{\text{out}} = \inf\{t : X_{out}(t) = x_0\}. \end{aligned} \tag{19}$$

The corresponding level set equation for (18) is

$$\begin{aligned} & \phi_t + \frac{v(x)}{\alpha}|\nabla\phi| = 0, \ (x,t) \in \mathbb{R}^n \times (0,+\infty), \\ & \phi(x,0) = |x - x_{\text{end}}|, \ x \in \mathbb{R}^n, \end{aligned} \tag{20}$$

and the level set equation for (19) is

$$\begin{aligned} &\phi_t + \tilde{v}_{\text{out}}(x)|\nabla\phi| = 0,\ (x,t) \in \mathbb{R}^n \times (0,+\infty), \\ &\phi(x,0) = |x - x_0|,\ x \in \mathbb{R}^n, \end{aligned} \tag{21}$$

where

$$\tilde{v}_{\text{out}}(x) = \frac{v(x)}{(1 + c(t/T)^\gamma)(\alpha + B(x_{\text{end}})\frac{t}{T}e^{-\psi(x_{\text{end}})t}\beta\psi(x))}. \tag{22}$$

We describe the numerical methods for solving these equations in the ensuing section.

Note that the denominator of (22) depends on the end point x_{end}. In our model, for a fixed logging time t and a fixed starting point, we need to evaluate $P_{opt}(x_{\text{end}}, t)$ for every possible end point. Theoretically, we need to solve all these problems over different end points independently and evolve different level set equations for each end point. In order to reduce the computational burden, in practice, we discretize $B(x)\frac{t}{T}e^{-\psi(x)t}$ into several levels. Let $b_{\min} = \min_x \left\{B(x)\frac{t}{T}e^{-\psi(x)t}\right\}$ and $b_{\max} = \max_x \left\{B(x)\frac{t}{T}e^{-\psi(x)t}\right\}$. We let $b_0, b_1, b_2, \ldots, b_n$ be a uniform partition of the interval $[b_{\min}, b_{\max}]$. Then grouping the possible ending points into $n+1$ collections based on these values, we only need to evolve $n+1$ level set equations with velocity modified for each of $b_0, b_1, \ldots, b_n$. A similar discretization is employed by Arnold et al. [2].

6 Numerical Methods and Implementation

In this section, we describe the numerical implementation of the optimal control problem from Section 5. To begin, we discuss the general concerns related to numerically solving Hamilton-Jacobi equations, and address some specific issues related to level set equations.

6.1 Numerical Schemes for Hamilton-Jacobi Equations

Recall, the general Hamilton-Jacobi equation in two spatial dimensions is given by

$$\begin{aligned} &\phi_t + H(\phi_x, \phi_y) = 0, \\ &\phi(x, 0) = \phi_0(x). \end{aligned} \tag{23}$$

In the case of the basic optimal path planning problem, the Hamiltonian will take the form $H(x, y, \phi_x, \phi_y) = v(x, y)\sqrt{\phi_x^2 + \phi_y^2}$; however, we suppress the dependence of H on x and y to simplify the notation.

Because Hamilton-Jacobi equations give rise to solutions with discontinuities in their derivatives, ordinary difference methods, which are perfectly suitable for advection equations, will often fail to capture the complex dynamics at play [6]. Instead, we trade the Hamiltonian $H(\phi_x, \phi_y)$ for the numerical Hamiltonian $\hat{H}(\phi_x^+, \phi_x^-; \phi_y^+, \phi_y^-)$ which somehow combines the forward difference approximations ϕ_x^+, ϕ_y^+ and backward difference approximations ϕ_x^-, ϕ_y^- to the derivatives, in order to more accurately simulate the system.

Osher and Shu [18] suggest many choices for the numerical Hamiltonian, each having its own advantages and disadvantages. Perhaps the easiest to implement is the Lax-Friedrichs Hamiltonian:

$$\begin{aligned} \hat{H}_{LF}(\phi_x^+, \phi_x^-; \phi_y^+, \phi_y^-) = H\left(\frac{\phi_x^+ + \phi_x^-}{2}, \frac{\phi_y^+ + \phi_y^-}{2}\right) \\ - \frac{\alpha_x}{2}(\phi_x^+ - \phi_x^-) - \frac{\alpha_y}{2}(\phi_y^+ - \phi_y^-). \end{aligned} \tag{24}$$

Here we are adding diffusion to the equation which is on the order of the grid parameters $\Delta x, \Delta y$. The coefficients α_x, α_y depend on the derivatives of H and determine the amount of diffusion that is added. This numerical Hamiltonian has two advantages: first, it is easy to implement, and second, the added diffusion results in smooth solutions. However, in cases

of optimal path planning, where the motion is driven by a given velocity field $v(x,y)$, the excessive diffusion can qualitatively change the results, especially near regions of zero velocity [2, 20].

Accordingly, we opt for the minimally diffusive Godunov Hamiltonian:

$$\hat{H}_G(\phi_x^+, \phi_x^-, \phi_y^+, \phi_y^-) = \operatorname*{ext}_{u \in I(\phi_x^-, \phi_x^+)} \operatorname*{ext}_{v \in I(\phi_y^-, \phi_y^+)} H(u,v) \tag{25}$$

where

$$I(a,b) = [\min(a,b), \max(a,b)] \tag{26}$$

and

$$\operatorname*{ext}_{x \in I(a,b)} = \begin{cases} \min_{a \le x \le b} & \text{if } a \le b, \\ \max_{b \le x \le a} & \text{if } a > b. \end{cases} \tag{27}$$

These extrema are designed to capture a formal similarity between Hamilton-Jacobi equations and conservation laws, wherein the most important numerical consideration is tracking the directions of the characteristics.

The Godunov Hamiltonian can be more difficult to implement, but because it is non-diffusive, it is far preferable for optimal path planning problems. In certain cases, the extrema in (25) can be resolved explicitly. For example, when $H(\phi_x, \phi_y) = v(x,y)\sqrt{\phi_x^2 + \phi_y^2}$ and $v \ge 0$, we have

$$\begin{aligned} &H_G(\phi_x^+, \phi_x^-, \phi_y^+, \phi_y^-) = \\ &\qquad v(x,y)\sqrt{\max\{(\phi_x^-)_+, (\phi_x^+)_-\}^2 + \max\{(\phi_y^-)_+, (\phi_y^+)_-\}^2} \end{aligned} \tag{28}$$

where $A_+ = \max\{A, 0\}$ and $A_- = -\min\{A, 0\}$.

Having decided on the Godunov Hamiltonian, we simply need to approximate the derivatives ϕ_x, ϕ_y and integrate (23) in time. Following Osher and Shu [18], we use the essentially non-oscillatory second order approximations to the spatial derivatives and explicit second order total variation diminishing Runge-Kutta time stepping to evolve the equation. Specifically, if our discrete domain is given by $\{x_i\}, \{y_j\}, \{t_k\}$ and $\phi_{i,j}^k$ is our approximation to $\phi(x_i, y_j, t_k)$, then our scheme reads

$$\begin{aligned} \phi_{i,j}^* &= \phi_{i,j}^k - \hat{H}_{i,j}^k \Delta t, \\ \phi_{i,j}^{k+1} &= \frac{1}{2}\phi_{i,j}^k + \frac{1}{2}\phi_{i,j}^* - \frac{1}{2}\hat{H}_{i,j}^* \Delta t. \end{aligned}$$

where $\hat{H}^k$ is the Hamiltonian evaluated at ϕ^k and $\hat{H}^*$ is the Hamiltonian evaluated at ϕ^*.

Considering our specific case once again, this scheme will be stable and approximate the solution at second order so long as our grid spacing parameters $\Delta x, \Delta y, \Delta t$ satisfy the CFL condition

$$V\Delta t\left(\frac{1}{\Delta x}+\frac{1}{\Delta y}\right)<1$$

where $V=\max_{x,y} v(x,y)$ [17].

6.2 The Redistancing Problem for Level Set Equations

As discussed in Section 5, the level set equation

$$\begin{aligned}&\phi_t + v(x)|\nabla\phi| = 0,\\ &\phi(x,0)=\phi_0(x),\end{aligned} \tag{29}$$

is of particular interest to us. Here the initial function is the signed distance function to the initial contour $\Gamma(0)$ which we desire to evolve via level set motion. However, when we solve the equation numerically, since the velocity function v may be non-smooth, after some steps, there will be some regions where the distortion accumulates and instabilities arise, causing the level sets $\Gamma(t)=\{x : \phi(x,t)=0\}$ to become an unreliable measure of the points which can be reached by time t. However, because ϕ is a mathematical abstraction, and the only quantity we are concerned with is $\Gamma(t)$, we can circumvent this issue by periodically halting the time integration of the equation and replacing ϕ with the signed distance to the current level set $\Gamma(t)$. This is knows as *redistancing*. It will prevent ϕ from developing instabilities, and thus ensure that the level sets $\Gamma(t)$ are reliable.

There are many ways to accomplish this redistancing. In our algorithm, we use a "crossing time" method. We need to solve the eikonal equation

$$\begin{aligned}&|\nabla u| = 1,\ x\in\mathbb{R}^2\\ &u(x)>0,\ x\in\mathbb{R}^2\setminus\Omega,\\ &u(x)<0,\ x\in\Omega\setminus\partial\Omega,\end{aligned}$$

where Ω is a closed set in $\mathbb{R}^2$. In our case, Ω will represent the interior of the current level set $\Gamma(t)$. We consider the Hamilton-Jacobi form of this eikonal equation,

$$\begin{aligned}&w_t + |\nabla w| = 0,\ x\in\mathbb{R}^2,\ t>0\\ &w(x,0)=w_0(x),\ x\in\mathbb{R}^2,\end{aligned} \tag{30}$$

where the initial function $w_0(x)$ has zero level set $\partial\Omega$ but is not the signed distance function to $\partial\Omega$ as desired. The connection between the Hamilton-Jacobi equation and the eikonal equation is that $w(x,t) = 0$ if and only if $u(x) = t$ [8].

We use the first order Godunov scheme and first order time step to solve the equation (30). First, the zero level-set contour of $w(x,t)$ will be evolved outward from Ω. For each point (x_i, y_j) outside Ω, when we detect that the signs of $w_{i,j}^k$ and $w_{i,j}^{k+1}$ are different, we set the signed distance function $u(x_i, y_j) = (k+1)\Delta t$. Next, we use the initial function $-w_0$, so that the zero level-set contour will be evolved inward from Ω. Again for each (x_i, y_j) inside Ω, when we detect that $w_{i,j}^k$ and $w_{i,j}^{k+1}$ have different signs, we set different $u(x_i, y_j) = -(k+1)\Delta t$.

We also implemented several other methods to do the redistancing. One method based on a Hopf-Lax formulation is suggested by Royston et al. [22]. Given $x \in \mathbb{R}^n$ and w satisfying equation (30), we need to solve $w(x,t) = 0$. This equation can be solved numerically by the secant method,

$$t^{k+1} = t^k - w(x,t^k)\frac{t^k - t^{k-1}}{w(x,t^k) - w(x,t^{k-1})},$$

where the initial t^0 and t^1 can be chosen from the nearby nodes. Equation (30) can be solved by Hopf-Lax formula,

$$w(x,t) = \min_{y\in\mathbb{R}^n}\left\{w(y,0) + tH^*\left(\frac{x-y}{t}\right)\right\},$$

where H^* is the convex conjugate of H. Since $H(p) = \|p\|_2$, this formula can be written

$$w(x,t) = \min_{y\in B(x,t)} w(y,0)$$

and this optimization problem can be solved by projected gradient descent. This method has the advantage of being easily parallelizable so that the computational complexity will not be influenced too much by the dimension. The disadvantage of this method is that it contains two iterative parts: the secant method to solve the equation $\phi(x,t) = 0$ and projected gradient method to find the minimum of $\phi_0(y)$ in $y \in B(x,t)$ and these pieces are embedded in the sense that we must re-solve (29) via the Hopf-Lax formula for each iteration t^k of the secant method. Since our model is in $\mathbb{R}^2$, the calculation burden of those solvers is much greater than that of simply

solving the time-dependent eikonal equation, which is why we opt for the previous method.

Another method for redistancing, as employed in Arnold et al. [2], Parkinson et al. [20], is to explicitly calculate the distance from each grid point to a polygon which approximates the current level set. For each point, the distance to the given polygon is the minimum distance between the point and each edge of the polygon. Thus we must calculate the distance from the point to each edge individually. This computation can be parallelized in both the grid points, and the sides of the polygon. However, in our 2-dimensional model problem, if the size of grid is $n \times n$, the number of edges of the polygon will be $O(n^2)$ after some time steps, and calculating the distance between each point and this polygon will have the computational complexity $O(n^4)$. Empirically, we found this to be much slower than the crossing time method.

Finally Zhao [24] proposed a fast sweeping method for redistancing. This method is a type of Gauss-Seidel iteration. The strategy is to 'sweep' through the grid from different directions and update the entries in a specific order accordingly. Again, our implementation of this scheme is slower than the crossing time method. We also tried to replace the Gauss-Seidel iteration here with the Jacobi iteration. Since for an $n \times n$ matrix, the Gauss-Seidel iteration needs n steps to update all the entries in this matrix while Jacobi iteration only needs one step. However, the convergence of the Jacobi iteration is suspect. One advantage of this method is that it does not require the initial function to be a good approximation of the signed distance function; it only requires knowledge of the boundary of Ω. However, in our model problem, when we pause the time integration in order to perform the redistancing, we assume that the function $\phi(x, t)$ is a good approximation to the distance function to the current level set $\Gamma(t)$, and since this is true, the fast sweeping method has no advantage over the crossing time method.

6.3 Implementation Tricks

In order to give a complete account of our numerical methods, we also describe three small implementation tricks that are specific to the problem at hand.

First, when we solve the level set equation numerically, we perform redistancing very frequently—roughly every 10 time steps. In order to reduce the computational burden, we only perform the redistancing near

the zero level set contour, then use very simple linear approximation of the distance on the points which are far away from the zero level set. Since the zero level set is the only quantity of interest, and since the level set evolution depends only on the local values of $\phi(x,t)$, this will not affect the quality of the solution. More explicitly, suppose the current level set is $\Gamma(t)$ and we want to find the signed distance function $u(x)$ to this level set. Further, assume that $\Gamma(t)$ is contained in the box $[a,b]\times[c,d]$. If we perform the redistancing every n time steps, then we only need to use the crossing time method to calculate the distance function for points in the rectangle

$$R = [a-2n\Delta x, b+2n\Delta x]\times[c-2n\Delta y, d+2n\Delta y].$$

This is because the crossing time level set is propagating with speed 1, so if it is originally contained in $[a,b]\times[c,d]$, then it will remain inside R after n steps, and further, it will remain unaffected by information propagating inward from the exterior of R, where we are not perfectly resolving the distance function. For any point x in the grid but not in R, we take the value of the signed distance function $u(x)$ as the boundary value of R plus the L_1 distance between this point and R, namely

$$u(x) = u(y) + d_1(x,y),\ y = \arg\min_{y\in\partial R} d_1(x,y).$$

Thus, for example, if (x_i, y_j) is the top right corner of R, then $u(x_{i+k}, y_{j+\ell}) = u(x_i,y_j) + k\Delta x + \ell\Delta y$.

Second, in implementing crossing time method, we need to evolve equation (30) two times, once outward and once inward. In some cases the inward evolution does not need many steps since the region inside the zero level-set contour will be much smaller than the region outside, due to the increased velocity along the road and river network in Roraima. So in practice, if there are some successive steps wherein we don't detect any sign changes from $w_{i,j}^k$ to $w_{i,j}^{k+1}$, we assume that the inward evolution is complete and halt the iteration.

Third and finally, as described above, when we perform the redistancing, the value of $u(x_i,y_j) = (k+1)\Delta t$ when $w_{i,j}^k$ and $w_{i,j}^{k+1}$ have a different sign. However, in reality, the point (x_i,y_j) could be much closer to $k\Delta t$ than it is to $(k+1)\Delta t$. This is especially true of points along the initial contour. These points lie between distances $-\Delta t$ and Δt and will be moved from their original location to the midpoint of the contours corresponding to distance $-\Delta t$ and Δt, causing the initial contour to be distorted. Accordingly,

rather than explicitly setting $u(x_i, y_j) = (k+1)\Delta t$, we construct a linear interpolant $\tilde{w}_{i,j}(t)$ between the values $w_{i,j}^k$ and $w_{i,j}^{k+1}$ and set $u(x_i, y_j) = t_0$ such that $\tilde{w}_{i,j}(t_0) = 0$.

7 Logging Experiments

In this section we will apply the level set solver to our model as described in section 6. We give a detailed description of the experimental setup in section 7.1 and analyze the numerical results in section 7.2.

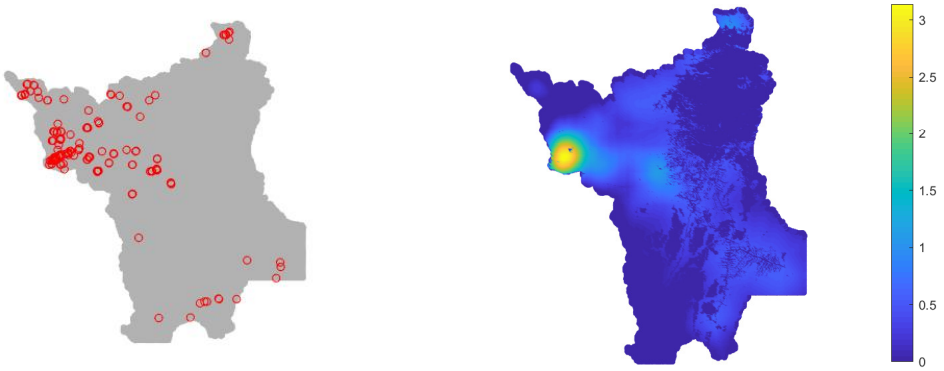
7.1 Experimental Setup

In order to test our model, we first need to prepare data, choose coefficients and design patrol evaluation metrics.

First, we discuss the calculation of the logging benefit function. we weren't able to find references on tree category data in the state of Roraima. Moreover, neither PRODES nor DETER is a good indicator of logging events due to low spatial resolution and other technical issues. For example, logging generally will not lead to land clearance, which would be detected by PRODES or DETER. Nevertheless, We decided to construct the benefit function for logging based on PRODES data. We make the assumption that deforestation for farming purpose only takes place within 10km of the major highways and treat all the other deforestation events as the result of logging, which are plotted in figure 15a. We then design the benefit based on the post-hoc reasoning that high benefit gives rise to high event frequency within the region. Specifically, we use the same technique as in KDE by stamping a 2-D Gaussian to each event. We then generate the logging benefit by linearly combining the generated density function and the binary benefit function used in the farming model as is shown in figure 15b.

Next, the logging model is much more sensitive to the transportation system than the farming model because of the long travel distance to high benefit regions. We try to design a realistic velocity field, as shown in figure 16, to accurately capture the movement of loggers throughout the region. We first get the highway and water map from OpenStreetMap contributors [16]. We assign velocity 1, 0.8, 0.7 to major highways, waterways and secondary highways respectively. For off highway and off water areas, the velocity is the average of the vertical and horizontal gradients of each cell, based on a USGS digital elevation model [10].

After this, we need to model the capture intensity. Patrollers' capture intensity is a complicated concept related with patrol resources like available patrollers and equipment, patrol strategies like patrol path and frequency, and patrollers' capture ability. For now, we just deal with the capture in-



(a) Events at least 10km away from highways

(b) Benefit constructed for logging

Figure 15: Logging benefit

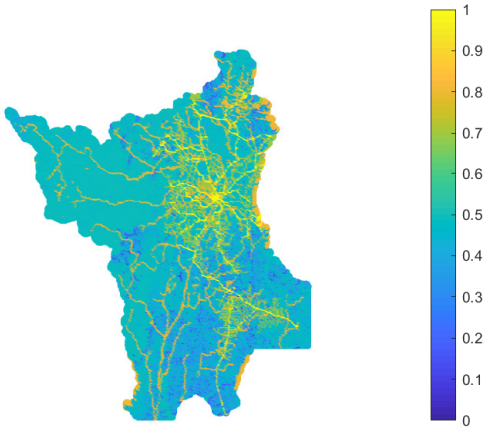


Figure 16: Velocity field in the state of Roraima

tensity in a simplified and ideal setting, where $\psi(x)$ can be any function that fulfills the budget constraint:

$$\int_{\Omega} \psi(x)(1+\mu d(x))^2 dx \leq E$$

where E is the total budget, d is the Euclidean distance to the major highways and μ is a constant. The penalty term $(1+\mu d(x))^2$ reflects the fact that it is more expensive to patrol regions far away from highways. In the experiment, we always set capture intensity so that the constraint reaches

equality, which means the patrollers are always using the entirety of their budget.

Among ψ satisfying the budget constraint, we would like to decide which patrol strategies are more effective than others. We've designed three different metrics to evaluate patrol efficiency.

- Pristine area ratio PR: we define the regions with non-positive profit as pristine area. PR calculates the ratio of the area of pristine region over the area of the state as $\frac{\int_\Omega \mathbb{1}_{\{P(x)\leq 0\}} dx}{\int_\Omega 1 dx}$.
- Pristine benefit ratio PB: this metric weighs pristine area by benefit as $PB = \frac{\int_\Omega B(x)\mathbb{1}_{\{P(x)\leq 0\}} dx}{\int_\Omega B(x) dx}$ and represents the ratio of benefit within the pristine area over the total benefit.
- Weighted profit WP: as in the farming model, we also interpret the positive part of the profit as the probability density for loggers to choose the logging location. We then define WP as the expected profit by $WP = \frac{\int_\Omega P_+(x)^2 dx}{\int_\Omega P_+(x) dx}$, where $P_+(x) = P(x)\mathbb{1}_{\{P(x)\geq 0\}}$.

We run the model on a 600×600 grid and set time step size to be $\min\{\Delta x, \Delta y\}/(2\max \tilde{V})$ to satisfy the CFL condition, where Δx and Δy are the spatial grid size in x and y directions and $\tilde{V}$ is the modified velocity in the time-dependent Hamilton-Jacobi equation. We will do a reinitialization every 10 time steps. We set $\alpha = \mu = \frac{2}{(5\max_{x\in\Omega} d(x))} \approx 7.33 \times 10^7, \beta = \frac{1}{ln(10)} \approx 4.34 \times 10^{-1}, T = 2000000$.

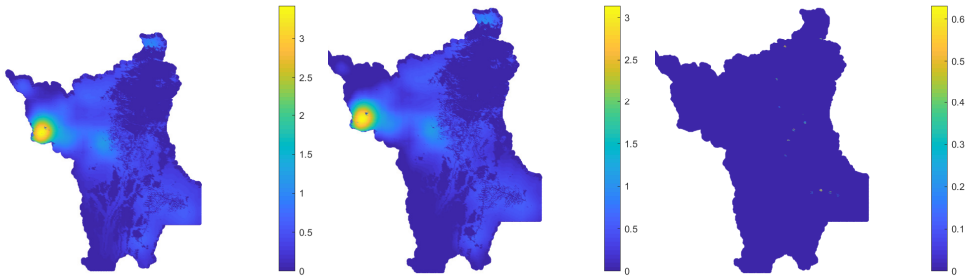
7.2 Results

We tested our model with different patrol budgets and patrol strategies. We also explored the influence of logging time and changing the velocity when traveling with goods. In the end, we will look into optimal logging paths and discuss interesting findings.

7.2.1 Example 1: No Patrol

We first impose no patrol. Recall that the returning velocity is modified following $v(x)/(1 + c(t/T)^\gamma)$ to take into account the influence of carrying timber. When the amount of timber has no influence, i.e., $c = 0$, the optimal paths traveling in and traveling out are the minimal time path, and the

loggers will always use the maximal logging time T. The resulting profit is shown in figure 17a. We then tested $c = 1$, $\gamma = 0.5$ and $\gamma = 2$. Since $0 \leq t/T \leq 1$, $\gamma = 0.5$ gives a harsher penalty to velocity than $\gamma = 2$, and thus a higher cost. As we can see from figure 17, expected profit is 0 in most of the region when $\gamma = 0.5$. Profit for $\gamma = 2$ is only a little bit smaller than the $c = 0$ case. In all of the following experiments, we will set $c = 0$ and only look into the influence of patrol.



(a) $c = 0$ (b) $c = 1, \gamma = 2$ (c) $c = 1, \gamma = 0.5$

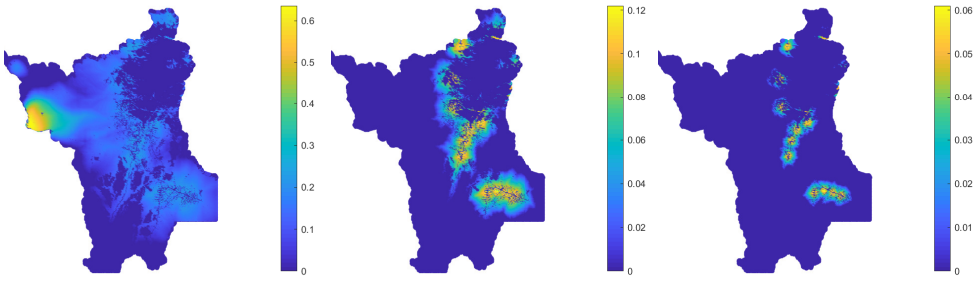
Figure 17: Returning velocity depends on trees obtained, no patrol imposed

7.2.2 Example 2: Comparison of Different Budget

In this example, we set $\psi(x) = \frac{B(x)E}{(1+\alpha d(x))^{11} \int_\Omega (1+\alpha d(x))^{-9} B(x) dx}$, which is shown to be a good patrol strategy in section 7.2.4. We then set E to be 0.001, 0.002 and 0.004. The resulting profit is plotted in figure 18. It's clear that higher budget will give lower profit. In all of the following experiments, we fix E to be 0.002.

7.2.3 Example 3: Influence of Patrol on Logging Time

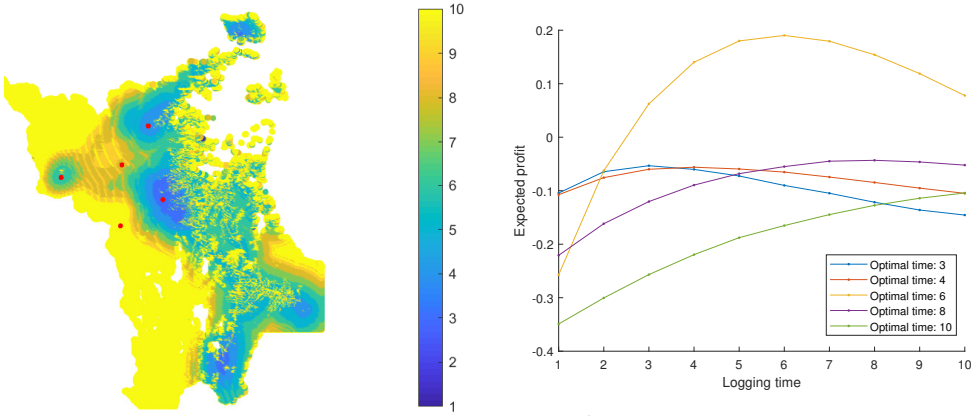
We use the same experimental set up as in the previous example, and we fix E as 0.002. Recall that we discretize the logging time and search for the optimal time by parameter sweep. In all of the experiments, we discretize logging time into 10 different levels. We plot the optimal logging time in figure 19a. We then sample 5 points in this region which have optimal logging time $3, 4, 6, 8$ and 10 respectively and are marked as red points in figure 19a. Figure 19b shows the profit as a function of logging time at each point and we see that there are different optimal logging times for



(a) $E = 0.001$ (b) $E = 0.002$ (c) $E = 0.004$

Figure 18: Expected profit with different budget

each point. This is because the risk of capture is higher at some points, making logging for extended periods of time very dangerous.



(a) Optimal time

(b) Profit changes with logging time at different spots

Figure 19: Influence of patrol time

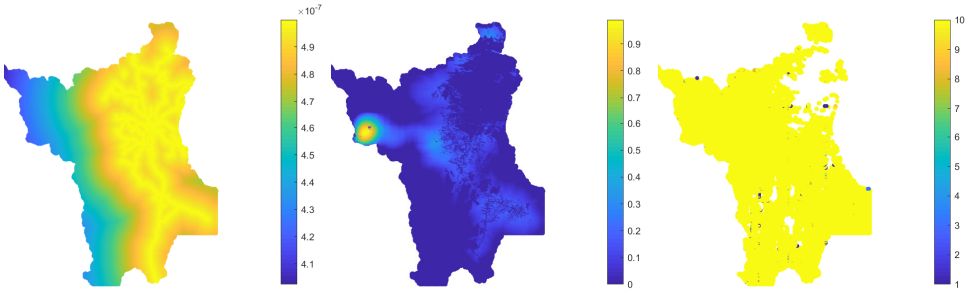
7.2.4 Example 4: Comparison of Different Patrol Strategies

In this example, we compare the patrolling efficiency for different capture intensity functions $\psi(x)$. We plot the corresponding capture intensity function, profit and optimal time for each experiment and summarize the evaluation based on aforementioned metrics in table 1.

First, we consider a patrol only based on distance to roads by setting

$$\psi(x) = \frac{E}{(1+\alpha d(x))^r \int_\Omega (1+\alpha d(x))^{2-r} dx},$$

where r is chosen to be $1, 3, 7, 11$. We may want to give more attention to regions which are close to roads as logging and patrol costs are low in these regions. Larger r means the patrol is more concentrated near the highways, while smaller r leads to more uniformly distributed patrol. Figures 20 - 23 exhibit the corresponding capture intensity, profit and optimal time. When $r = 1$, loggers will fully exploit logging time and clear the forest at almost all of the locations, which means capture risk is not a huge threat. When r increases, the optimal logging time along with the profit near highways starts to decrease, while profit far away into the forest increases. These results give us the clue that we shouldn't concentrate too much along highways. Moreover, all four profit plots have the pattern that high benefit regions are high profit regions, which inspire us to take benefit into consideration.



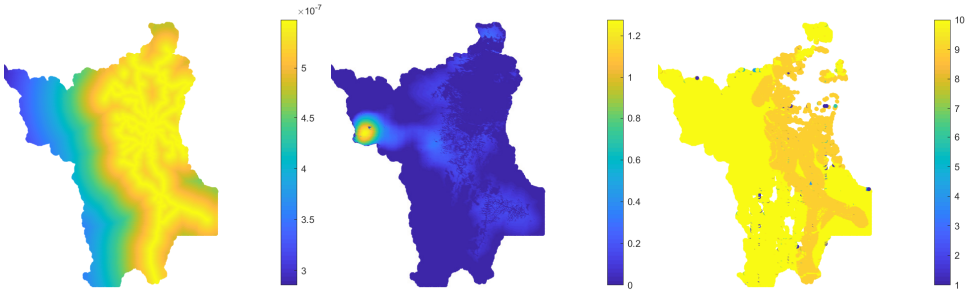
(a) Capture intensity (b) Profit (c) Optimal time

Figure 20: Patrol based on distance only, $r = 1$

Next, we set

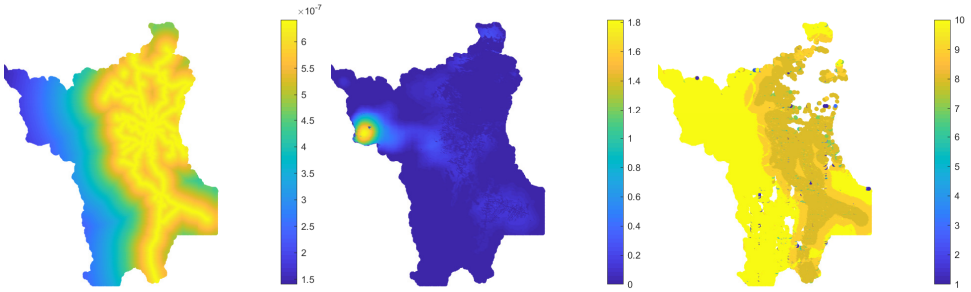
$$\psi(x) = \frac{B(x)E}{\int_\Omega (1+\alpha d(x))^2 B(x) dx}$$

so that the capture intensity is proportional to the benefit. The result is shown in figure 24. Clearly, the intense patrol in high profit regions make those regions less vulnerable. Figure 24b shows that profitable regions now cluster around highways where both the initial benefit and the travel cost



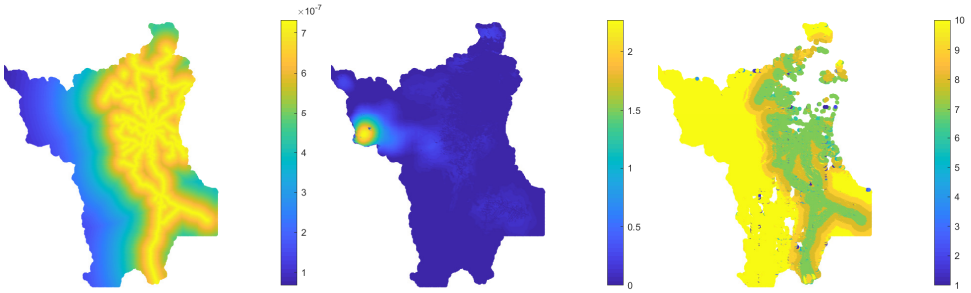
(a) Capture intensity (b) Profit (c) Optimal time

Figure 21: Patrol based on distance only, $r = 3$



(a) Capture intensity (b) Profit (c) Optimal time

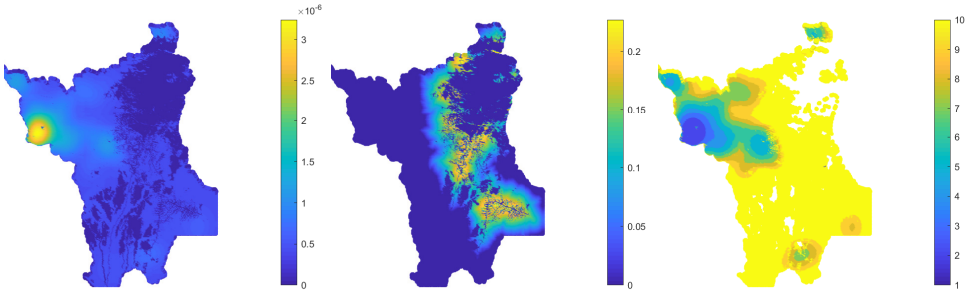
Figure 22: Patrol based on distance only, $r = 7$



(a) Capture intensity (b) Profit (c) Optimal time

Figure 23: Patrol based on distance only, $r = 11$

is relatively low. The optimal logging time at the northwest corner is much shorter than in other regions.



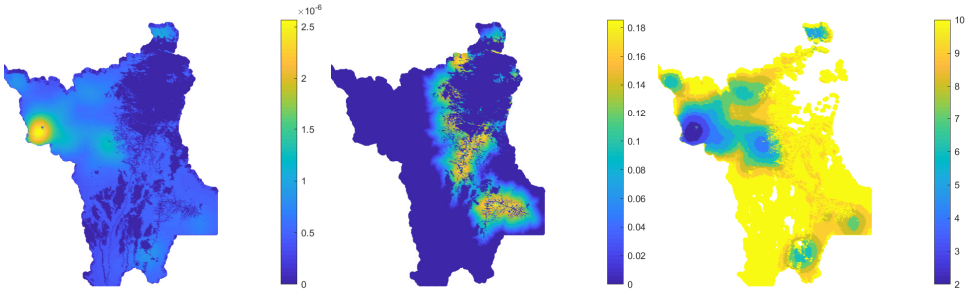
(a) Capture intensity (b) Profit (c) Optimal time

Figure 24: Patrol based on benefit only

Previous experiments inform us that we need to balance benefit and distance when designing a patrol strategy. Here, we set the patrol

$$\psi(x) = \frac{B(x)^w E}{(1+\alpha d(x))^r \int_\Omega (1+\alpha d(x))^{2-r} B(x)^w dx}$$

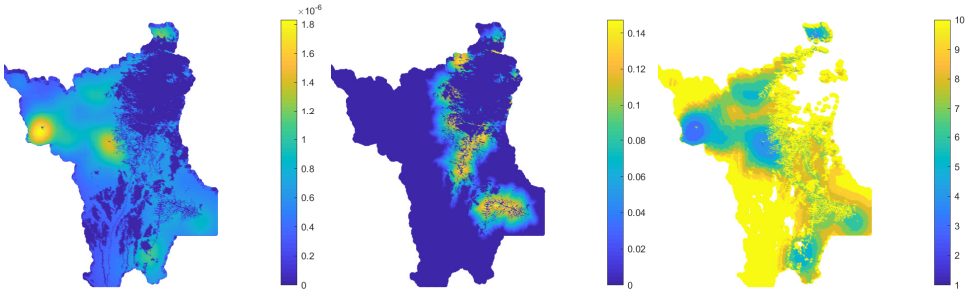
We tested this patrol with $w = 0.5$ and 1, $r = 3, 7, 11, 15$. Results are plotted in figures 25 - 32. Both the figures and table 1 confirm that $w = 1$, $r = 11$ is the best choice.



(a) Capture intensity (b) Profit (c) Optimal time

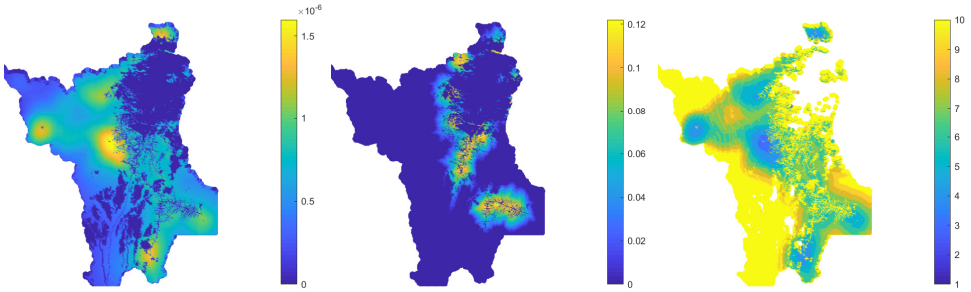
Figure 25: Patrol based on benefit and distance, $w = 1$, $r = 3$

All previous experiments show that both distance to roads and benefit are important factors for patrol allocation. For now we do not have a method to find optimal patrol strategies, but our model can be applied to evaluate and compare different strategies.



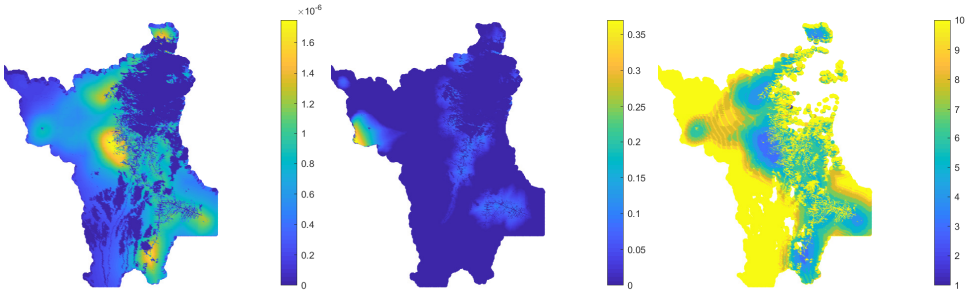
(a) Capture intensity (b) Profit (c) Optimal time

Figure 26: Patrol based on benefit and distance, $w = 1$, $r = 7$



(a) Capture intensity (b) Profit (c) Optimal time

Figure 27: Patrol based on benefit and distance, $w = 1$, $r = 11$

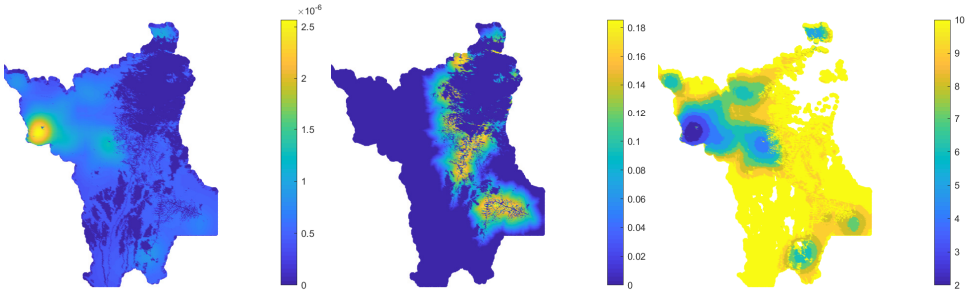


(a) Capture intensity (b) Profit (c) Optimal time

Figure 28: Patrol based on benefit and distance, $w = 1$, $r = 15$

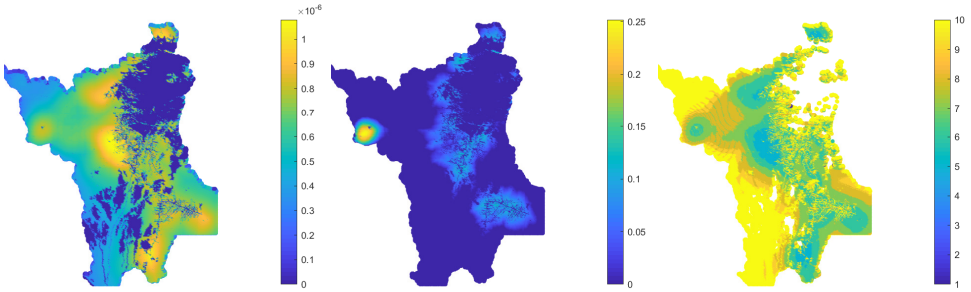
7.2.5 Example 5: Optimal Paths

Finally, we calculate the optimal paths that loggers take to and from their logging sites. We use the same patrol strategy as shown in figure 30a. We



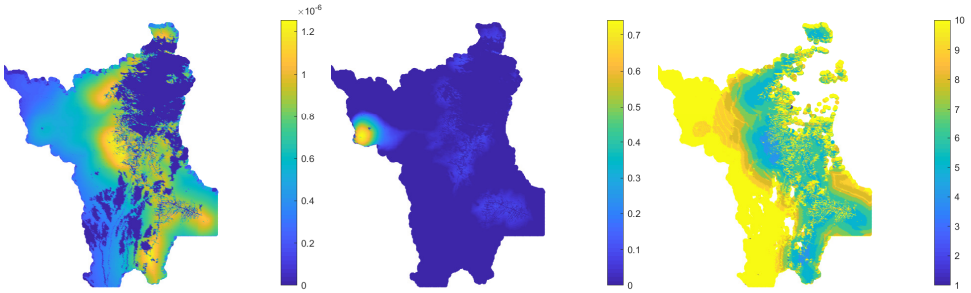
(a) Capture intensity (b) Profit (c) Optimal time

Figure 29: Patrol based on benefit and distance, $w = 0.5$, $r = 3$



(a) Capture intensity (b) Profit (c) Optimal time

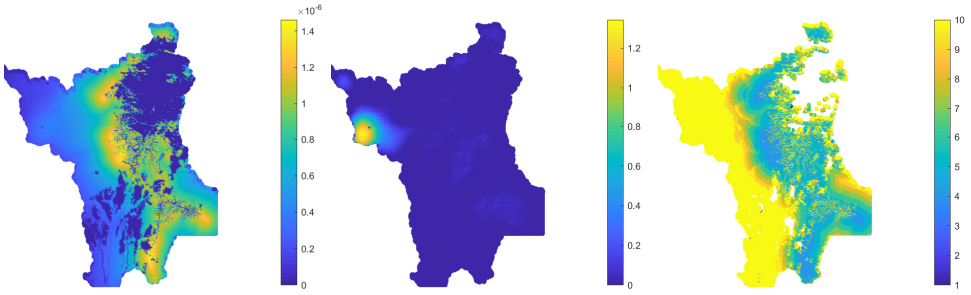
Figure 30: Patrol based on benefit and distance, $w = 0.5$, $r = 7$



(a) Capture intensity (b) Profit (c) Optimal time

Figure 31: Patrol based on benefit and distance, $w = 0.5$, $r = 11$

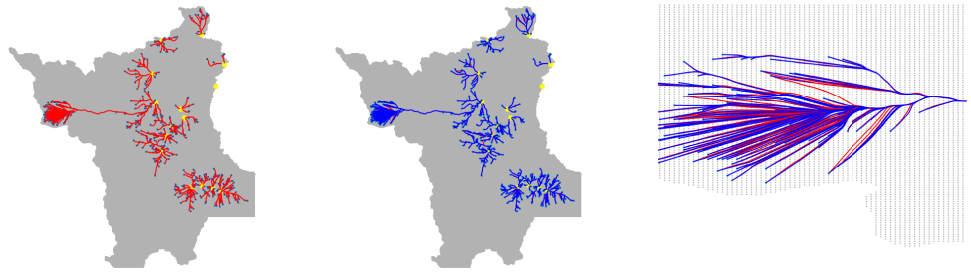
sampled 500 points in Roraima based on expected profit and plotted the optimal path going into the forest (figure 33a) and going out of the forest (figure 33b). We find that the optimal paths that go deeper into the forest



(a) Capture intensity (b) Profit (c) Optimal time

Figure 32: Patrol based on benefit and distance, $w = 0.5$, $r = 15$

in the northwest corner cluster into one trajectory and the optimal paths going in and going out are quite similar. One reason why this happens is that the capture intensity is much smoother than the velocity field due to the rivers running across this region. The fast travel speed along the river outweighs the risk of being captured. Moreover, if we zoom into the northwest corner, we can see that the optimal paths for different directions are not exactly the same because of the influence of patrol.



(a) Optimal path of going into the forest (b) Optimal path of going out of the forest (c) Optimal path zoom in

Figure 33: Optimal path

Patrol based on	Parameters	WP	PA	PB
Distance only	$r = 1$	0.2792	0.5674	0.3485
	$r = 3$	0.3813	0.5630	0.3302
	$r = 7$	0.6192	0.5459	0.2864
	$r = 11$	0.8215	0.5237	0.2459
Benefit only		0.1696	0.6820	0.5105
Benefit and distance	$w = 1, r = 3$	0.1044	0.6828	0.6805
	$w = 1, r = 7$	0.0811	0.7318	0.7394
	$w = 1, r = 11$	0.0661	0.7737	0.7883
	$w = 1, r = 15$	0.1091	0.7508	0.6115
	$w = 0.5, r = 3$	0.1044	0.6828	0.6804
	$w = 0.5, r = 7$	0.0856	0.7374	0.6121
	$w = 0.5, r = 11$	0.3038	0.7370	0.5569
	$w = 0.5, r = 15$	0.6211	0.7232	0.5018

Table 1: Comparison of different patrol strategies

8 Conclusion

Identifying two modes of deforestation, clearance of land for farming and illegal logging, we build up two mathematical models to describe the observed events. To describe land clearance we developed an agent based model for individual farmers who acted as static attackers, clearing more land with each time step and thus causing more deforestation. For illegal logging, we built a control theoretic model to predict the paths of the loggers and the locations they may choose to target. Additionally, we detailed the numerical solvers used for the Hamilton-Jacobi and control theoretic equations. Finally, we tested various patrol strategies with the logging model and the determined which patrol strategies best protected the forest.

8.1 Farming Model: Main Achievements and Future Work

In our farming model, we constructed the expected profit formula for farmers to decide where to clear land for illegal farming. We also performed time series analysis to simulate the behaviors of farmers and growth of farms, with capture probability built in. We compared our model with real deforestation data and evaluated the results with average F1 score. The strength of our model is that it takes into account many realistic factors such as distance to roads, distance to cities, clustering effect, tree coverage, punishment, capture possibility and maximum operation time for farms. Besides this, our model is easy to explain and understand.

In future, we can improve our model by doing some of the following:

- Further optimize our set of parameters.
- Consider more realistic factors.
- Try different patrol strategies and possibly find the optimal one.
- Design a better algorithm to model farm expansion.

8.2 Logging Model: Main Achievements and Future Work

In all previous models, the illegal logging events happen instantaneously when the criminals arrive at the logging location. In our model, we consider the logging events occurring over a period of time so that the criminals have a chance of being caught while they are cutting down trees. The

criminals may not take all the benefit from a spot (i.e. cut down all the trees in a spot) and they will balance between the chance of getting caught and the benefit they can gain in order to find a optimal logging time.

We believed this is closer to modeling real life, since we can see on the satellite map that there are some places where logging events occurred but trees still remain. With our model, we can evaluate the intensity of a logging event rather than merely decide whether an event will happen or not.

With different logging times, criminals carry different amount of timber with them as they travel out of the forest, and the amount of timber will influence the velocity. We also calculate the capture probability in a more reasonable way than [2].

Besides this, our model and experiments are based on real data set. The elevation, river, different types of roads and cities are all included in our model, and criminals embark from different cities rather than from the boundary of the region.

In the future, we can do the following to improve our model:

- **Remove the linear approximation of capture probability**. In our current logging model, we use a linear approximation of the exponential term to make the formula in the optimization problem easier to deal with. This approximation is not ideal since the exponential term will never become negative but the linear approximation will. We would like to directly deal with the exponential term in the formula to make our model more accurate.

- **Optimal patrol strategy**. Currently we have some metrics to evaluate whether a patrol strategy is good or not. We use these metrics to compare different patrol strategies, but we would like to try to analytically solve the optimization problem over the patrol strategy to maximize or minimize these metrics. Failing this, we would like to design a method to perturb a given patrol strategy in order to improve it.

- **Realistic patrol strategy**. Not all the patrol strategies are realistic. For example, a patrol strategy that only patrols some spots far inside the forest is not available since most of the policemen live in the cities and they need to travel along some path from where they live to go deep into the forest. We'd like to solve the following problem: given a desired patrol density, how do we arrange policemen to get a realistic patrol strategy to approximate the desired one. Here the arrangement

of policemen includes the traveling path, the velocity, the number of policeman (or the ratio of the number of policemen in each policeman group to the total number of policemen).

- **Improve the model with more real data** We suspect that there might be some other data which could be included to improve our model. For example, the price of timber, the distribution of different kinds of trees and the population density all likely have significant effects on illegal logging. We don't have these data now but would like to include them in the future.

Bibliography

[1] H. J. Albers. Spatial modeling of extraction and enforcement in developing country protected areas. *Resource and Energy Economics*, 2010.

[2] D. J. Arnold, D. Fernandez, R. Jia, C. Parkinson, D. Tonne, Y. Yaniv, A. L. Bertozzi, and S. J. Osher. Modeling environmental crime in protected areas using the level set method. *SIAM Journal on Applied Mathematics*, 79(3):802–821, 2019. ISSN 0036-1399. doi: 10.1137/18M1205339.

[3] J. J. Assunção, C. Gandour, and R. Rocha. Deforestation slowdown in the Brazilian Amazon: prices or policies? 2012.

[4] Martino Bardi and Italo Capuzzo-Dolcetta. *Optimal control and viscosity solutions of Hamilton-Jacobi-Bellman equations*. Springer Science & Business Media, 2008.

[5] Elliot Cartee and Alexander Vladimirsky. Control-theoretic models of environmental crime - preprint, 06 2019. URL `https://arxiv.org/abs/1906.09289`.

[6] M. G. Crandall and P.-L. Lions. Two approximations of solutions of Hamilton-Jacobi equations. *Mathematics of Computation*, 43(167):1–19, 1984.

[7] C. G. Diniz, A. A. d. A. Souza, D. C. Santos, M. C. Dias, N. C. d. Luz, D. R. V. d. Moraes, J. S. Maia, A. R. Gomes, I. d. S. Narvaes, D. M. Valeriano, L. E. P. Maurano, and M. Adami. DETER-B: The new Amazon near real-time deforestation detection system. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 8(7):3619–3628, July 2015. ISSN 1939-1404. doi: 10.1109/JSTARS.2015.2437075.

[8] I. C. Dolcetta. The Hopf-Lax solution for state dependent Hamilton-Jacobi equations (viscosity solutions of differential equations and related topics). 2002.

[9] F. Fang, T. H. Nguyen, R. Pickles, W. Y. Lam, G. R. Clements, B. An, A. Singh, B. C. Schwedock, M. Tambe, and A. Lemieux. PAWS — a deployed game-theoretic application to combat poaching. *AI Magazine*, 38(1):23–36, Mar. 2017. doi: 10.1609/aimag.v38i1.2710. URL `https://www.aaai.org/ojs/index.php/aimagazine/article/view/2710`.

[10] T. G. Farr, P. A. Rosen, E. Caro, R. Crippen, R. Duren, S. Hensley, M. Kobrick, M. Paller, E. Rodriguez, L. Roth, D. Seal, S. Shaffer, J. Shimada, J. Umland, M. Werner, M. Oskin, D. Burbank, and D. Alsdorf. The shuttle radar topography mission. *Reviews of Geophysics*, 45(2), 2007. doi: 10.1029/2005RG000183. URL `https://agupubs.onlinelibrary.wiley.com/doi/abs/10.1029/2005RG000183`.

[11] D. Kar, F. Fang, F. Delle Fave, N. Sintov, and M. Tambe. A game of thrones: When human behavior models compete in repeated Stackelberg security games. In *Proceedings of the 2015 International Conference on Autonomous Agents and Multiagent Systems*, pages 1381–1390. International Foundation for Autonomous Agents and Multiagent Systems, 2015.

[12] D. Kar, B. J. Ford, S. Gholami, F. Fang, A. J. Plumptre, M. Tambe, M. Driciru, F. Wanyama, A. Rwetsiba, M. Nsubaga, and J. Mabonga. Cloudy with a chance of poaching: Adversary behavior modeling and forecasting with real-world poaching data. In *AAMAS*, 2017.

[13] W. F. Laurance, A. K. M. Albernaz, G. Schroth, P. M. Fearnside, S. Bergen, E. M. Venticinque, and C. Da Costa. Predictors of deforestation in the Brazilian Amazon. *Journal of Biogeography*, 29(5-6):737–748, 2002. doi: 10.1046/j.1365-2699.2002.00721.x. URL `https://onlinelibrary.wiley.com/doi/abs/10.1046/j.1365-2699.2002.00721.x`.

[14] G. O. Mohler, A. L. Bertozzi, T. A. Goldstein, and S. J. Osher. Fast TV regularization for 2D maximum penalized likelihood estimation. *Journal of Computational and Graphical Statistics*, 20(2):479–491, 2011.

[15] National Institute of Space Research (INPE). PRODES deforestation. Accessed through Global Forest Watch in 07/2019. www.globalforestwatch.org.

[16] OpenStreetMap contributors. Planet dump retrieved from https://planet.osm.org . `https://www.openstreetmap.org`, 2017.

[17] S. J. Osher and R. P. Fedkiw. *Level Set Methods and Dynamic Implicit Surfaces*. Springer Verlag, 2003.

[18] S. J. Osher and C.-W. Shu. High order essentially non-oscillatory schemes for Hamilton-Jacobi equations. *SIAM Journal of Numerical Analysis*, 28(4):907–922, 1991.

[19] Stanley Osher and James Sethian. Fronts propagating with curvature dependent speed: Algorithms based on Hamilton-Jacobi formulations. *Journal of Computational Physics*, 79:12–49, 1988.

[20] C. Parkinson, D. J. Arnold, A. L. Bertozzi, Y. T. Chow, and S. J. Osher. Optimal human navigation in steep terrain: a Hamilton-Jacobi-Bellman approach. *Communications in Mathematical Sciences*, 17:227–242, 01 2019. doi: 10.4310/CMS.2019.v17.n1.a9.

[21] A. S. P. Pfaff. What drives deforestation in the Brazilian Amazon?: Evidence from satellite and socioeconomic data. *Journal of Environmental Economics and Management*, 37(1):26 – 43, 1999. ISSN 0095-0696. doi: https://doi.org/10.1006/jeem.1998.1056. URL `http://www.sciencedirect.com/science/article/pii/S0095069698910567`.

[22] M. Royston, A. Pradhana, B. Lee, Y. T. Chow, W. Yin, J. Teran, and S. J. Osher. Parallel redistancing using the Hopf–Lax formula. *Journal of Computational Physics*, 365:7–17, 2018.

[23] T. Slough, J. Urpelainen, and Johns Hopkins SAIS. Public policy under limited state capacity: Evidence from deforestation control in the Brazilian Amazon. Technical report, mimeo, 2018.

[24] H. Zhao. A fast sweeping method for eikonal equations. *Mathematics of computation*, 74(250):603–627, 2005.