Estimating Density of Humpback Whales in Hawaii from Shore-Based Observations

Megan Strong
090000961
MMath Project in Mathematics and Statistics
University of St. Andrews

Supervised by Dr Len Thomas

April 2013

Declaration

I certify that this project report has been written by me, is a record of work carried out by me, and is essentially different from work undertaken for any other purpose or assessment.

Signed:

Date:

Abstract

Distance sampling methods are used widely in estimating the density and abundance of biological populations. A critical assumption of these methods is that the species to be surveyed are distributed uniformly and independently about the transect from which detection distances are measured. This assumption is equivalent to the randomised placement of transects within the survey region. Within this project, we discuss how this assumption is violated where humpback whales are observed from a shore-based transit station and analysis methods to accommodate this non uniform distribution are discussed and applied.

This dissertation is intended for a statistical audience with an interest in distance sampling, specifically for those interested in exploring methods to accommodate the violation of assumptions in point transect sampling.

Contents

1 Introduction

Reliable methods for estimating the abundance and density of animals are fundamental to the study and conservation of many species. Distance sampling refers to a group of such methods, in which a population is observed from either a line or point transect, and the respective perpendicular or radial distances to each detected animal are measured and recorded. These distances are then used to draw inference on the density of the species within the vicinity of the transects. Where a marine species is observed from a shore-based observation point, we cannot assume that the distribution of the species is uniform and independent about the point. Since this is a critical assumption of point transect sampling, conventional distance sampling (CDS) methods cannot be utilised. Within this paper, we implement methods introduced by Arranz *et al.* (submitted) which assume that the density of whales is a function of some observable environmental variable. These methods build upon those first introduced by Marques *et al.* (2010) and Cox *et al.* (2011), which assume that animal density contours are parallel to some linear feature on which points are placed. We begin with a brief overview of point transect sampling and its assumptions, and a discussion of the previous works on point transect sampling where there is non uniform distribution of animals about the point.

1.1 An Introduction to Point Transect Sampling

Point transect sampling is a method of distance sampling that is used mostly, but not exclusively, for avian surveys (Buckland *et al.* 2001, 146). This method involves the observer visiting a number of points positioned randomly within the survey region. At any given point, the radial distance to each detected animal is measured and recorded. For the purpose of illustration, assume that the observer visits k randomised points and in doing so observes a total of n animals. Assume also that animals at a distance greater than w from the point are not recorded. The region covered by the survey is therefore $a = k\pi w^2$. We define P_a to be the proportion of animals within this covered region a that are detected, and we obtain some estimate $\hat{P}_a$ of this. We can then estimate population density $\hat{D}$ by

$$\hat{D} = \frac{n}{k\pi w^2 \hat{P}_a} \tag{1}$$

The detection function $g(r)$ is defined to be the probability of detecting an animal, given that it is at radial distance r from the random point (Buckland *et al.* 2001, 10). Clearly, $0 \leq g(r) \leq 1$ for all r, and we later discuss that $g(0) = 1$ is a critical assumption of point transect sampling. It can be shown that the probability of detecting an animal, given that it is within the covered region, is equal to the expected value of the detection function with respect to the distribution of radial distances available for detection, denoted by $\pi(r)$. An estimate of P_a is therefore given by

$$\hat{P}_a = \int_0^w \hat{g}(r)\pi(r)dr \tag{2}$$

A critical assumption of point transect sampling is the random placement of points within the survey region; this ensures the uniform and independent distribution of animals about the point and hence allows us to treat $\pi(r)$ as known. At a distance r from the point, an annulus of width dr will have an approximate area of $2\pi r dr$. The total proportion of the circle of radius w that falls within this annulus is therefore $2\pi r dr/\pi w^2$. Assuming uniform distribution of animals, this proportion also constitutes the expected proportion of animals in the circle that fall within the annulus. Multiplying by $g(r)$ gives the expected proportion of animals in the circle that are both within the annulus and detected. An estimate of P_a is therefore given by

$$\hat{P}_a = \int_0^w \frac{2\pi r\hat{g}(r)dr}{\pi w^2} \tag{3}$$

By (2), it follows that the distribution of radial distances, otherwise known as the density function, is equal to $\pi(r) = 2r/w^2$ when animals are uniformly and independently distributed about the points. Conventional distance sampling (CDS) methods can then be employed to estimate animal density by modelling the detection function $g(r)$ using the recorded distances from points to animals within the survey region; we outline these methods below.

Assume that an observer detects a total of n animals with respective radial detection distances of $r_1, r_2, ..., r_n$. Buckland *et al.* (2001, 61) define the likelihood of ϕ as follows, where ϕ is the unknown parameter vector of $g(r)$.

$$L(\phi) = \prod \frac{g(r_i)\pi(r_i)}{\int_0^w g(r)\pi(r)dr} \tag{4}$$

Where $\pi(r)$ is known, maximum likelihood parameter estimates can be obtained, in turn allowing estimation of the detection function $g(r)$. An estimate of P_a can then be obtained by (2), and thus density can be estimated by (1).

In cases where the uniformity assumption is violated, we cannot assume than $\pi(r)$ is known. Since $g(r)$ and $\pi(r)$ enter the above likelihood as a product, they cannot be estimated separately (Marques *et al.* 2010, 1249). Section 1.3 discusses an extension of standard methods introduced by Marques *et al.* (2010), which permit separate estimation of $\pi(r)$ and $g(r)$.

1.2 Assumptions of Point Transect Sampling

There are five main assumptions of point transect sampling:

- Probability of detection is equal to 1 at zero distance.
  - This assumption is critical and violation leads to negatively biased density estimates. Buckland *et al.* (2001, 30) define the bias to be a function of $g(0)$: $E(\hat{D}) - D = -[1 - g(0)]D$. When probability of detection is certain at zero distance, $g(0) = 1$ and so $E(\hat{D}) - D = 0$. In this case, the standard density estimate $E(\hat{D})$ is unbiased. Where $g(0) < 1$, it follows that the density estimate is negatively biased since $E(\hat{D}) < D$.
  - Borchers *et al.* (2002, 160) discuss how, when this assumption is violated, the density estimate is negatively biased by a factor equal to the probability that an object at zero distance is detected. For example, if $g(0) = 0.25$, then density estimates will be 0.25 times the true density on average.
  - Violation of this assumption may occur, for example, where bird surveys are carried out in areas with dense canopy cover. In such a situation, birds may be directly above the observer (i.e. at zero distance from the point) but may go undetected. In marine surveys, animals may be directly beneath the observation platform and hence may not be seen by the observer.
  - Methods may be implemented to avoid violation of this assumption. Cook and Jacobson (1979) developed a double-observer approach to estimate the proportion of undetected animals, thus allowing estimates to be adjusted accordingly. With regards to marine surveys, underwater cameras directly beneath the observation platform may be used to ensure detection on the point.

- Animals are distributed uniformly and independently of the points.
  - This is equivalent to the assumption that points are placed randomly within the survey region, or placed systematically but with a random start point. Random point placement ensures a valid representation of the whole survey region and also guarantees a representative sample of relevant detection distances, thus allowing a a valid estimate of density to be obtained.
  - Failure to account for non uniformity can lead to biased density estimates. The type of bias (i.e. positive or negative) is dependent on the actual distribution of animals. In general, where animals avoid the points, detection probability will be overestimated and thus density will be underestimated. The opposite effect is expected where animals are attracted to the points (Marques *et al* 2013, 77).
  - Violation of this assumption occurs when there is non random placement of points. For example, Marques *et al.* (2010) discuss the potential bias introduced by placing points along

  a road when conducting a survey of Irish hares. Given the tendency of this species to avoid field edges, coupled with the disturbance along roads, it is unreasonable to assume that the animals are distributed uniformly and independently of the points.

  - The focus of this project will be on this assumption and we will see how random point placement is violated within the case study. Analysis methods accounting for non uniformity will be used to obtain valid density estimates.

- Animals are detected at their initial location.

  - There are two types of animal movement to consider, namely random movement and responsive movement. Positive bias in density estimates results from random movement since the encounter rate is increased, leading to an overestimation of animal density. Responsive movement towards the observer causes positive bias, whereas movement from the observer induces negative bias (Buckland *et al.* 2001, 12).
  - The underlying principle of point transect sampling is to obtain a 'snapshot', estimating the location of an animal at an instant in time. Where the location of an animal at the snapshot moment cannot be estimated with reasonable certainty, its distance from the point is not recorded (Buckland *et al.* 2001, 173). Kyhn *et al.* (2012, 552) discuss the use of linear interpolation to estimate animal location at a snapshot moment. With this method, the movement of an animal between two consecutive sightings is assumed to have constant speed and direction. The case study of this project will utilise the method of linear interpolation.

- Distances to animals are measured without error.

  - Where there is constant underestimation or overestimation of detection distances, there can be a significant effect on density estimates. If distances are consistently overestimated by 10%, density estimates are underestimated by 17%. Similarly, a consistent 10% underestimation of distances would give rise to density estimates being overestimated by 23% (Buckland *et al.* 2001, 176).
  - Where distance estimation is on average unbiased, small measurement errors are not problematic. To reduce the occurrence of measurement errors, permanent markers at a known distance from the point are useful (Buckland et al 2001, 176).

- Detections of animals are independent.

  - Although estimates of density are robust to this assumption of independent detections (Thomas *et al.* 2006, 547), robust variance estimates require independence between detections in different sampling units (Buckland *et al.* 2001, 130). Where there are several points within the survey region, we may take point to be the sampling unit. Within this project, the survey region comprises a single point, and so we take the sampling unit to be day.

1.3 Previous Works

Marques *et al.* (2010) discuss how violation of the uniformity assumption occurs where Irish hares are observed from point transects situated along a road. Given the tendency of this species to avoid field edges, coupled with the disturbance along the roads, it is unreasonable to assume that hare densities are representative along roads. To account for this, $\pi(r)$ is modelled as function of radial detection distance r and detection angle θ which allows animal density to increase with distance from the road. This clearly differs from the approach taken in conventional distance sampling, where it is assumed that the distribution of animals is independent of the transect; that is, the density of animals about the transect does not depend on whereabouts the transect is placed within the survey region.

Cox *et al.* (2011) estimate the density of Antarctic krill from multi-beam echo sounder surveys. This is formulated as a point transect sampling problem, where the points are situated on the sea surface. It is recognised that krill are not uniformly distributed with respect to depth, and so again the density function $\pi(r)$ is modelled as a function of both radial distance r and angle θ to each detected krill swarm. This again accounts for density increasing with distance from the linear feature.

Both Marques *et al.* (2010) and Cox *et al.* (2011) consider the case where density gradients are parallel to some linear feature, and model the distribution of detection distances as a function of distance from this linear feature. Arranz *et al.* (submitted) provide an extension of this method, in assuming that the density of animals at a given point is dependent on some observable environmental variable, namely ocean depth, where beaked whales are surveyed from a shore-based observation point in the Canary Islands. This project will use the same approach in an attempt to obtain valid density estimates of humpback whales off the western coast of Maui, Hawaii. The underlying concept of this approach is discussed within the Methods section.

2 Case Study

Figure 1: Humpback whale http://www.whalewatchmaui.com/

The humpback whale (Megaptera novaeanglia) is a species of baleen whale known for its intricate songs and elaborate displays. Found in oceans across the world, humpback whales tend to situate themselves close to coastlines and feed on krill, plankton and small fish. During the winter months, the whales migrate to tropical or subtropical breeding grounds, such as the Hawaiian Islands in the Pacific Ocean. The humpback whale was once hunted to the brink of extinction with the global population depleted by 95%, before a ban was enforced by the International Whaling Commission in 1966 to outlaw the commercial whaling of humpbacks (Edrén and Teilmann 2005, 899). Since then, the population of humpback whales has seen a steady increase, although the species are still listed as endangered by the U.S. Endangered Species Act of 1973 (NOAA Fisheries). Reliable inference about the population of this species is fundamental to its management and conservation.

2.1 Background and Methods

During the winter months, the western coast of Maui in Hawaii is a spot favoured by North Pacific humpbacks, who migrate from their summer feeding grounds in Alaska. The warm, shallow waters off the coast of Maui make this area an ideal breeding ground. The data for this project were provided by Dan Smith of Woods Hole Oceanographic Institution, and consist of whale observations made from an 80 metre hill named Kilea near the town of Olowalu (Tyack 1981, 106). Observations were made on a total of 121 days in the months of January to April from 1979 to 1981.

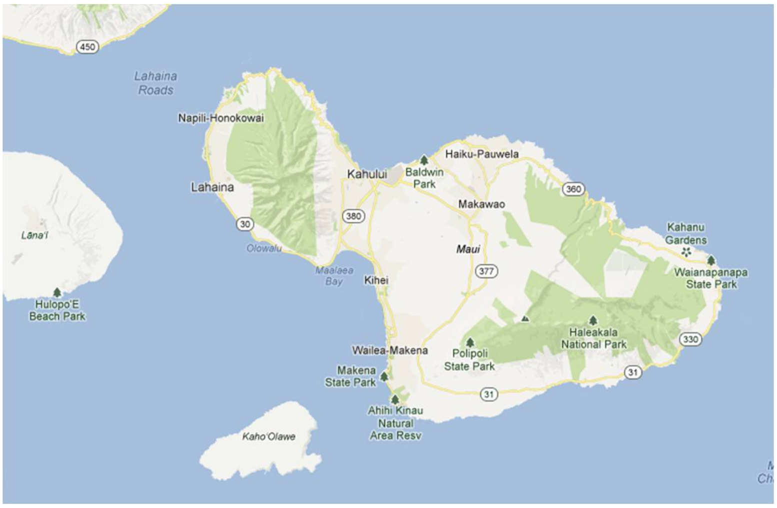


Figure 2: Map of Maui https://maps.google.com/

Shore-based observers were able to detect whales either when they blew or when part of their bodies emerged above the water. On sighting of a whale, a theodolite was used to measure both the vertical and horizontal bearings to the whale. The vertical zero was taken to be perpendicular to sea level (i.e. directly above the observer) and the horizontal zero was taken to be 36.7° west of north (Dan Smith, pers. comm.). Each whale or group of whales was assigned an identification letter when sighted. Due to the relatively slow movement and low density of whales within the survey area, observers were usually able to monitor the movement of the whales between surfacings. Figure 3 below (Tyack 1981, 106) shows the transit station positioned on the coast of Maui.

2.2 Formulation into a Point Transect Sampling Problem

Where a marine species is observed from a shore-based observation point, we typically treat this as a point transect sampling problem. The total area of the survey region will be less than $k\pi w^2$ (where k is the number of points visited, and w is the radius of each point), since part of the circle about each point transect will lie on land. Calculating the area of the covered region (i.e. the total area of the circles that lies on the sea), we can obtain density estimates of the surveyed species.

The data provided for this project contain information regarding the size of observed whale groups. However, a significant amount of this data is missing and in many cases, the exact group size could not be determined at the time of data collection. We will therefore proceed with this project by estimating the density of whale *groups* and defining 'observation' to be the sighting of a whale group, where the group constitutes a minimum of one whale. Should information on the average size of whale groups become readily available in future, this information can be incorporated easily to obtain an estimate of whale density (rather than whale *group* density) by multiplying the estimated density of whale groups by by $E(s)$, the expected group size.

Using the method of point transect sampling where the single point transect is the shore-based transit station, we will use the data provided by Dan Smith to estimate the density of groups of humpback whales within the covered region. Conventional distance sampling methods can be employed only when all assumptions are met (see Section 1.2). Where one or more assumptions are violated, conventional methods must be adapted to ensure that valid density estimates are obtained. Within the next section, we revisit the main assumptions of point transect sampling and discuss which assumptions are violated within the case study.

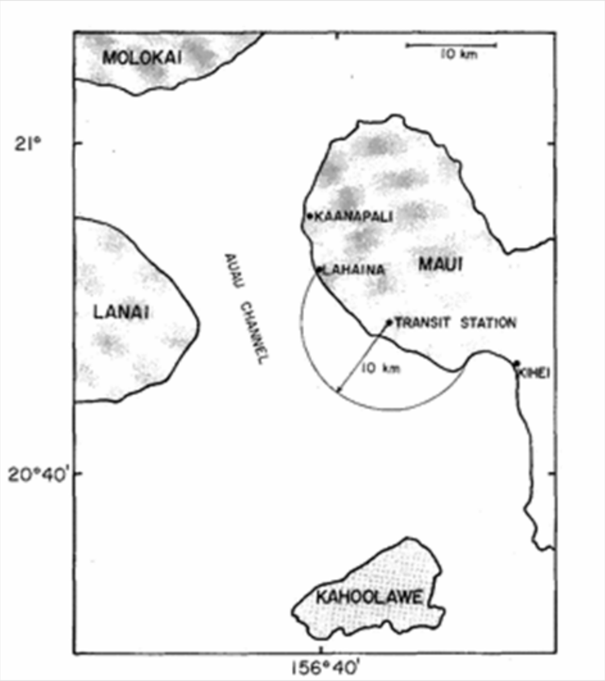


Figure 3: Transit station on coast of Maui

2.3 Revisiting Assumptions

- Probability of detection is equal to 1 at zero distance.
  - The single observation point within the case study is on dry land. Given this location, there will be no whales at zero distance from the point, and so no whales at zero distance will be missed. This assumption is therefore not violated.
- Animals are distributed uniformly and independently of the points.
  - Equivalent to the assumption of random point placement within the survey region, there is clearly violation within the case study. Since the observation point is on the shore, we cannot assume that the whales distribute themselves randomly about the point, and so we cannot

  assume that the distribution of whales about the point is known. This project will focus on this assumption, and we discuss methods that can be implemented to account for the non uniform distribution of humpback whales.

- Animals are detected at their initial location.
  - It was earlier discussed that animals can exhibit two types of movement, namely random or responsive, either of which would cause this assumption to be violated. In this case study, the random movement of whales is inevitable and so we cannot assume that the whales are detected at their initial location. During data collection, whales were observed not only from the shore but also from boats traversing the survey region. We assume in this case study that the whales did not exhibit any evasive movement in response to these boats, although this might provide grounds for further research. Due to the random movement of whales during the survey period, we use linear interpolation to estimate the location of whales at snapshot moments, namely five minute intervals from 8:05am to 6pm on each day of observation. This method assumes that the movement of whales between sightings has constant speed and direction.
- Distances to animals are measured without error.
  - A buoy at fixed distance 2.25km from the observation point was used as a permanent marker within the case study. Repeated readings of this buoy showed the Lietz TM-1A theodolite used in the data collection to be accurate to 10" of an arc. Throughout each day of observation, repeated checks were carried out and the horizontal zero of the transit was reset if it was off by more than 3" of an arc (Tyack 1981, 106).
- Detections of animals are independent.
  - On any given day of observations, it is unreasonable to assume that detections are independent; whether we detect a whale will not be independent of whether we have already seen it. In order to obtain robust variance estimates of density, we require a choice of sampling unit between which detections are independent; for this case study, we will take day to be the sampling unit.

3 Preliminary Work

3.1 Data Preparation

Within this project, considerable effort was expended in preparing the data provided by Dan Smith, in order to ensure that it was in an appropriate format for analysis in R. This data processing is documented in detail within Appendix A. In total, twenty files of data were provided, each comprising information regarding the data collection across 121 days; these files included details of whale observations, whale behaviour, locations of observers in boats and weather conditions. Data relating to whale observations were extracted and processed; where the sighting location was not recorded, or where the whale ID was unknown, the observation was discarded.

Vertical and horizontal bearings of each whale sighting were provided in the data files. Using the method of Lerzcak and Hobbs (1998, 598), which takes into account curvature of the earth, the sighting distance from the transit station to each whale observation was calculated. The formula used to calculate sighting distance d is given below, where r_e is the radius of the earth (taken to be 6371km), h is the height above sea level of the transit station (82m; 80m plus 2m height of theodolite) and θ is the angle in radians from the horizon to the whale observation.

$$d = (sin^{-1}[sin(\frac{\pi}{2} - \theta) * \frac{(r_e + h)}{r_e}] - \frac{\pi}{2})) * r_e \tag{5}$$

Using the sighting distance and horizontal bearing of each whale observation, the Cartesian coordinates of each observation were calculated, where the transit station is at the origin, and the x-axis is in line with the horizontal zero. Whale detections with a sighting distance of greater than 15km were truncated to allow for more robust estimation of the detection function; this removed 5% of all whale observations.

In order to assess the effect of changes in tide and theodolite height (due to the height of the observer) on sighting distances, the height above sea level was set at both 81m and 83m, and the sighting distances re-calculated. Where the height of the transit station was taken to be 81m, sighting distances were at most 1.95% lower; where the height was 83m, this increased sighting distances by a maximum of 2%. It was therefore not deemed necessary to include a correction for variations in tidal height or height of the observer; however, including such a correction may provide beneficial in future work with this data.

3.2 Linear Interpolation

As discussed, an assumption of point transect sampling is that the animals to be surveyed are detected at their initial location; that is, the animals exhibit neither random nor responsive movement. To account for random movement of whales, the method of linear interpolation was used to estimate the location of whales at five minute intervals from 8:05am to 6pm on each day of observation. Where a whale was seen at least twice on any given day, it was assumed that movement between sightings had constant speed and direction. Assuming this, the estimated track of the whale between sightings was split into one second intervals. If the time during which a whale was tracked coincided with a 'snapshot' moment, the location of the whale at that moment could be estimated by its position along the track. Figure 4 shows a plot of the estimated track of Whale K on 7^{th} April 1981. The start and end position of this track are marked, along with the actual observations and interpolated observations of the whale.

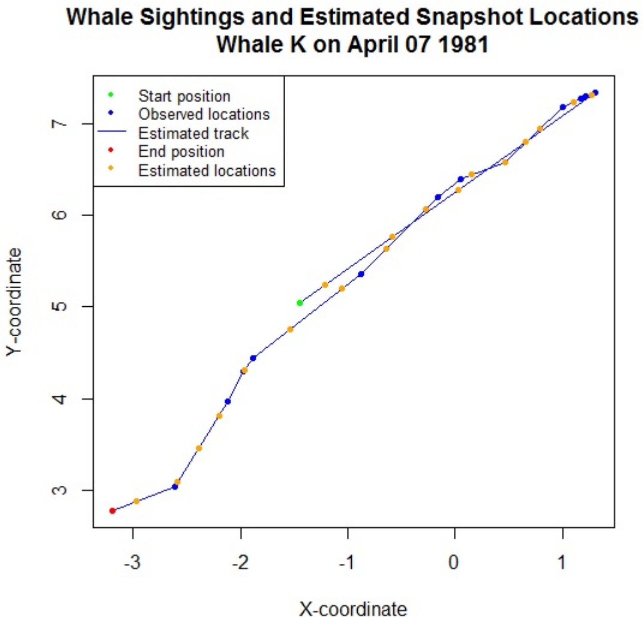


Figure 4: Estimated whale track

3.3 Exploratory Data Analysis

The data to be used for analysis within this project consist of the interpolated whale observations obtained after the relevant data processing (see Appendix A). Figure 5 shows these observations in relation to the Maui coastline, shown in blue. The coordinates of the coastline relative to the transit station were provided by Dan Smith (pers. comm.). The highest density of whale detections appears to be close to the transit station (shown in red) on the shore; this is intuitively what we would expect, since we assume that the detection function 'drops off' as radial distance from the observer increases.

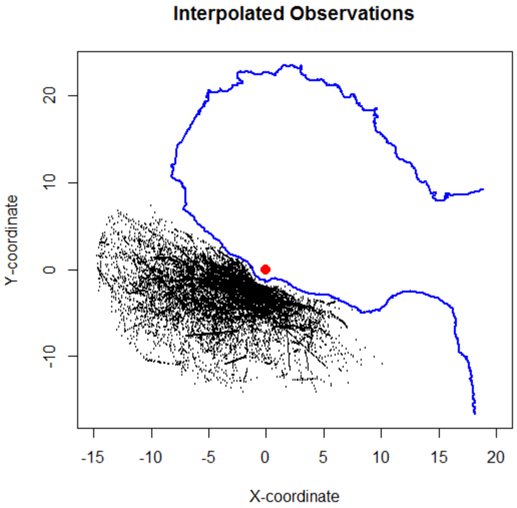


Figure 5: All interpolated whale observations

After the data processing, 118 unique days remained within the data set. In order to facilitate the process of nonparametric bootstrap resampling that will be carried out later in the analysis, a random sample of 30 days was taken. It is the data for these days that will be used for analysis within this project.

4 Methods

Within this project, the methods of Arranz *et al.* (submitted) are implemented. Such methods model the density function $\pi(r)$ as a function of some observable environmental variable, namely ocean depth. This section provides a description of these methods, and a discussion of how we adapt these methods to our case study.

A Cartesian coordinate system is defined across the survey region, where the y axis runs perpendicularly offshore. It is assumed that the distribution of animals in the x direction is uniform (Marques *et al.* 2010, Cox *et al.* 2011) and independent of the environmental variable to be considered. Figure 6 shows the interpolated observations in relation to this coordinate system, where the x and y axes are marked in red. We make the following definitions:

- (x,y) - Cartesian coordinates of a point on the grid
- $z(x,y)$ - Value of observable environmental variable at (x,y)
- r - Radial distance from observer of any point
- ϕ - Parameter vector of density function
- θ - Parameter vector of detection function

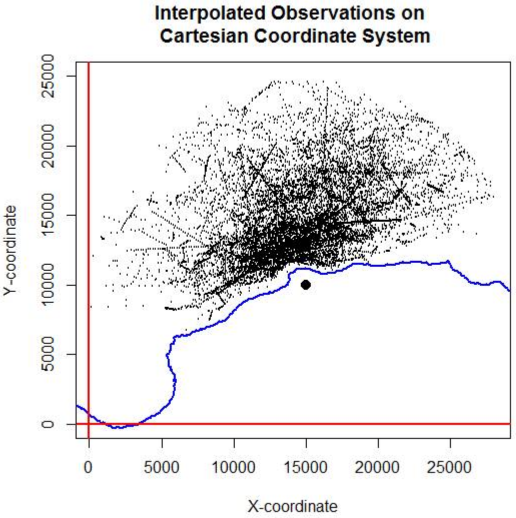


Figure 6: Cartesian coordinate system

For the density and detection functions respectively, z and r are taken to be the explanatory variables. It is assumed that animals are uniformly distributed along the x axis *independent* of the value of z, and so we can define the joint probability density function of z and x as $\pi(z, x; \phi) = \pi(z; \phi)\pi(x)$. Since we assume that animals are uniformly distributed about the x axis, we make the following definition, where x_{lim} is the range of x values considered within the analysis.

$$\pi(x) = \frac{1}{x_{lim}} \tag{6}$$

As discussed, Marques *et al.* (2010) and Cox *et al.* (2011) estimate the density function by assuming that density in the y direction is non uniform, that is, the density function has distance from this linear feature as an explanatory variable. Arranz *et al.* (submitted) extend this idea by assuming that z is not a function of distance from the point, but rather a function of some observable environmental variable z, which is a non simple function of Cartesian coordinates x and y. The methods of Marques *et al.* (2010) and Cox *et al.* (2011) are adapted by changing variables (x, y) to the (x, z), and expressing z as a function of both x and y. This change of variables requires the calculation of the derivative of z with respect to y.

Where animal density is assumed to be a function of distance from some linear feature, we can write the density function as $\pi(x, y)$. Since the distribution along the x axis is assumed to be uniform and independent of y, this can be written as $\pi(x)\pi(y)$. Following the methods just described, we adapt the methods of Marques *et al.* (2010) and Cox *et al.* (2011) to model density as a function of z, as described by the following equation.

$$\pi(x, y) = \pi(x)\pi(y) = \pi(x)\pi(z(x, y))\frac{\delta z}{\delta y} = \frac{1}{x_{lim}}\pi(z(x, y))\frac{\delta z}{\delta y} = \frac{\pi(z(x, y))}{x_{lim}}\frac{\delta z}{\delta y} \tag{7}$$

We now suppose that we make n observations, and for observation i at point (x_i, y_i) , we denote the corresponding radial distance to this observation by r_i. Modelling the detection function as a function of radial distance alone, we obtain the following likelihood for the parameter vectors ϕ and θ of the density and detection functions, respectively. Note that this likelihood is of the form given in (3), where we have here substituted in the relevant density and detection functions.

$$L(\phi, \theta) = \prod_{i=1}^{n} \frac{g(r; \theta)\frac{\pi(z(x_i, y_i; \phi))}{x_{lim}}\frac{\delta z}{\delta y}}{\int_0^{y_{lim}}\int_{-x_{lim}}^{x_{lim}} g(r; \theta)\frac{\pi(z(x, y; \phi))}{x_{lim}}\frac{\delta z}{\delta y}\delta x \delta y} \tag{8}$$

In order to calculate the value of $\delta z/\delta y$ at each observation, we place a grid across the survey region. At each grid point, the value of z and $\delta z/\delta y$ is known. Using bi-linear interpolation, this allows us to calculate $\delta z/\delta y$ at each sighting location (Arranz *et al.* (submitted)). The value of z at each observation can also be calculated using this method of bi-linear interpolation. Having obtained values of z and $\delta z/\delta y$ for each sighting location, the above likelihood is then maximised in order to obtain parameter estimates for both the density and detection functions. The `R` package `nupoint` facilitates this, and will be used within this project to fit a number of candidate models, allowing selection of the most parsimonious on the basis of the AIC.

Within this project, two environmental variables were initially considered as explanatory variables for the density function, namely depth and shortest distance to the coast. Depth data for longitude and latitude positions across Maui were obtained from the NOAA website. The ocean depth at each observation point was taken to be equal to the closest point for which depth data was readily available from this source. For each observation point, shortest distance to the coast was calculated using the `sp` package in `R`. Appendix B documents the relevant `R` code used in the calculation of these environmental variables.

For the methods of Arranz *et al.* (submitted) to be implemented, the derivative of the environmental variable with respect to offshore distance needs to be calculated. Derivatives were calculated for both of the considered environmental variables. On inspection of plots of these derivatives, it was decided that smooth surfaces should be fitted to both depth and shortest distance across the survey region, from which predictions of both z and its derivatives could be made. For depth, the fitted surface predicted negative depths within the survey region, that is, it predicted areas of the survey region to be on land. Since more accurate depth data was not readily available for this project, it was decided that depth should not be considered as an explanatory variable for the density function within this project. The following analyses therefore consider shortest distance to the coastline as an explanatory variable for the density function.

4.1 Model Fitting

A half-normal form was considered for the detection function, and a total of six forms for the density function; normal, log-normal, beta, uniform, 2-mixture normal and 3-mixture normal. The `nupoint` library in `R` was used to fit each of these six distribution models. With regards to the beta form for the density function, the parameters attained values at the upper bound of the parameter space, and an error message regarding this was returned by `R` (see Appendix B). The beta density function was therefore excluded from subsequent analyses, leaving five distribution models to consider. The most parsimonious model was chosen on the basis of Akaike's Information Criterion (AIC). This criterion includes a penalty for model complexity, which increases as the number of model parameters increases (Garthwaite et al. 2002, 92). The model with the lowest AIC value was preferred.

4.2 Density Estimates

Having arrived at a distribution model deemed appropriate, the following steps were taken to obtain an estimate of whale group density per km^2 at an instant in time during the survey period.

- At each grid point within the covered region, calculate the value of the density function $\pi(z)$ using the value of the environmental variable z at that grid point.
- Take an average of these values to obtain an estimate for the density of whale groups at each grid point, i.e. the density of whale groups per m^2.
- Where n is the total number of snapshots for which we have data, divide the above estimate by n to obtain an estimate of whale group density per m^2 per snapshot.
- Multiply the above estimate by 1,000,000 to obtain a final estimate of average whale density per km^2 at any given snapshot moment during the survey region.

In order to obtain robust estimates of density variance, we require a sampling unit between which detections are independent (Buckland *et al* 2001, 130). In this case study, the sampling unit is taken to be day. For each nonparametric bootstrap resample, we sample 30 days with replacement from the initial data set. We pool the detections from these 30 sampled days, and refit the model as before, obtaining new estimates for the model parameters. This new model is then used to estimate the average density of whale groups per km^2 at an instant in time during the survey period (this is achieved following the steps outlined above). Repeating this process yields several estimates of density, allowing us to calculate the variance of the estimator, along with confidence intervals for density. The number of bootstrap resamples B should be a minimum of 200, although Buckland *et al.* (2001, 82) recommend B to be in the region of 400-1000. Within this project, we obtain 2000 bootstrap resamples and thus 2000 density estimates.

5 Results

5.1 Model Fitting

Table 1 shows the parameter estimates of the fitted density and detection functions, along with the AIC values of each distribution model, the difference in AIC from that of the most parsimonious model, and the relative AIC weight. The vector of parameter estimates for the density function is denoted by $\hat{\phi}$ and the parameter estimate for the half normal detection function is denoted by $\hat{\theta}$. With regards to the 2 and 3-normal mixture models, $\hat{\mu}$ denotes the vector of means of the normal components, $\hat{\sigma}$ the vector of standard deviations, and $\hat{w}$ the vector of weights of each normal component.

Using the value of the AIC as the criteria for model selection, the 2-normal mixture form would be chosen for the density function. However, when nonparametric bootstrapping was carried out, and this model was refitted to each sample of data, one or more parameter estimates tended to be at the upper bound of the parameter space. This causes concern regarding the fit of the model, and so the distribution model with the second best AIC score was selected. The density function of this model has a normal form with parameter estimates $\hat{\phi} = \{9159.997, 11065.62\}$.

Density Function	$\hat{\phi}$	$\hat{\theta}$	AIC	ΔAIC	AIC Weight
2-Normal Mixture	$\hat{\mu}$={256.43,17858.51} $\hat{\sigma}$={16037.81,7583.035} $\hat{w}$={0.284,0.716}	$\hat{\theta} = 4066.85$	67102.17	0	0.71
Normal	$\hat{\phi} = \{9159.997, 11065.62\}$	$\hat{\theta} = 4167.013$	67104.3	2.13	0.25
3-Normal Mixture	$\hat{\mu}$={258.898,14941.49,22716.35} $\hat{\sigma}$={41496.26,9821.211} $\hat{w}$={0.018,0.018,0.964}	$\hat{\theta} = 4065.486$	67108.1	5.93	0.04
Uniform	-	$\hat{\theta} = 4301.421$	67113.68	10.51	0
Log-normal	$\hat{\phi} = \{25, 3.92\}$	$\hat{\theta} = 4208.348$	67125.4	23.23	0

Table 1: Parameter Estimates and AIC Values

Model diagnostics for the selected model were carried out to assess its goodness-of-fit. Within Figure 7 below, there are four plots. The upper left shows the value of the environmental variable z over the covered region, the upper right shows the intervals over which the expected number of whale observations is compared with the observed number, and the lower left shows the expected percentage of detections across the covered region. In the bottom right plot, the expected number of whale observations within each interval (shown in black) is plotted along with the observed number of whale observations within each interval (shown in red). From this plot, it appears that the model is a good fit to the data. Further diagnostics were carried out using the χ^2 goodness-of-fit test. The null hypothesis of this test is that the model is a good fit to the data; that is, within each interval, the observed number of observations does not differ significantly from the number expected under the model. The results of this goodness-of-fit test are shown in Appendix 3; the p-value of this test provides significant evidence against the null hypothesis at the 5% level. However, the data set for this project is large, and so it is inevitable that there will be discrepancies between the observed and expected number of observations within each interval. We do not therefore rely on the results of this test to indicate whether the model is a good fit; instead, we use the bottom right plot in Figure 7 to conclude that the model appears to be a good fit to the data.

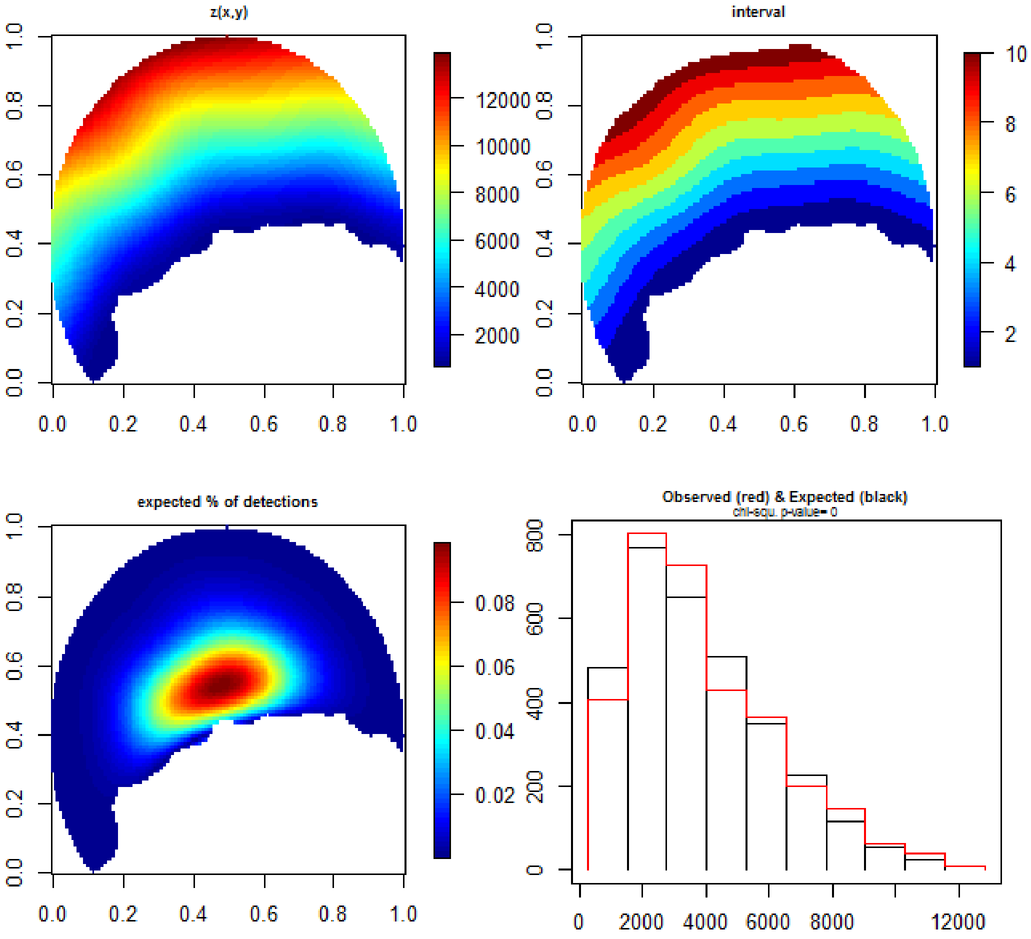


Figure 7: Model Diagnostics

The detection function chosen for the distribution model has a half-normal form with parameter estimate $\hat{\theta} = 4167.013$. Figures 8 and 9 below provide a graphical representation of this detection function $g(r)$, which clearly satisfies the critical assumption of $g(0) = 1$. Probability of detection appears to drop off to 0.5 at around the 5000m mark. That is, where a whale group is at radial distance 5000m from the observer, the probability of it being detected is expected to be (approximately) 0.5.

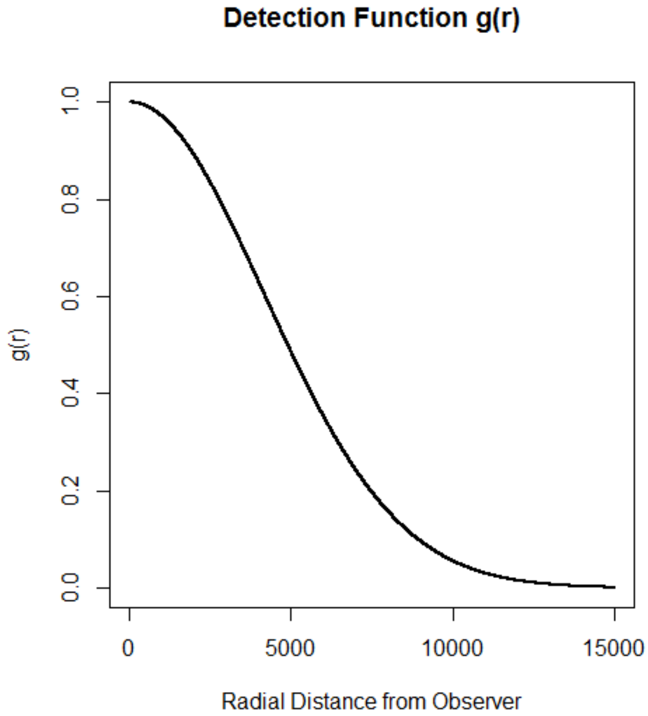


Figure 8: Detection function $g(r)$

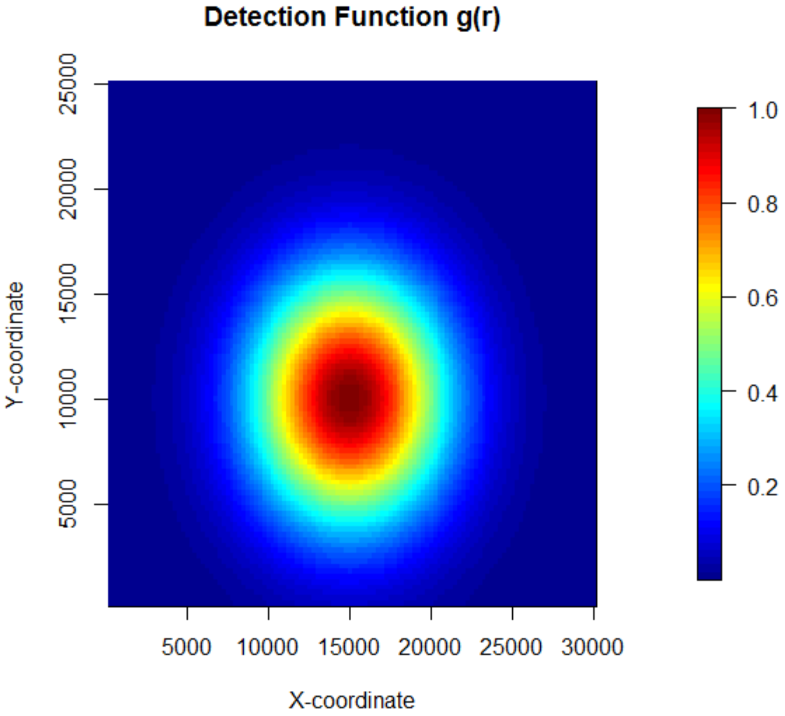


Figure 9: Detection function $g(r)$

5.2 Density Estimates

The average density of whale groups per km^2 was estimated to be 0.0208. Multiplying this by 398.26 (the total area of the covered region), we estimate the abundance of whale groups within the covered region to be 8.27 at any instant within the survey period. These statistics were calculated using the entire data set comprising 30 individual days. In the next section, we compare this estimate of density with that obtained by nonparametric bootstrap resampling.

5.3 Nonparametric Bootstrap Resampling

In order to obtain an estimate for the variance of density, along with a 95% confidence interval, 2000 bootstrap resamples were generated as described in the Methods section. The results of this nonparametric bootstrapping are shown below in Table 2. Figure 10 shows the distribution of density estimates, with the upper and lower limits of the 95% confidence interval (obtained by the percentile method) marked in red. Given that the bootstrap distribution is asymmetric, obtaining confidence intervals by the percentile method is often considered inappropriate. This suggests that a parametric bootstrap might be more suitable for calculating the limits of the confidence intervals.

Number of Bootstrap Resamples B	Lower 95% CI Limit	Upper 95% CI Limit	Variance	Bootstrap Mean
2000	0.0104	0.0348	0.0000531	0.0171

Table 2: Results of nonparametric bootstrapping

The bootstrap sample mean is 0.0171 whale groups per km^2, which is lower than the estimate obtained from the original data set. This estimate corresponds with an estimate of 6.8 whale groups within the survey region at any given point within the survey period.

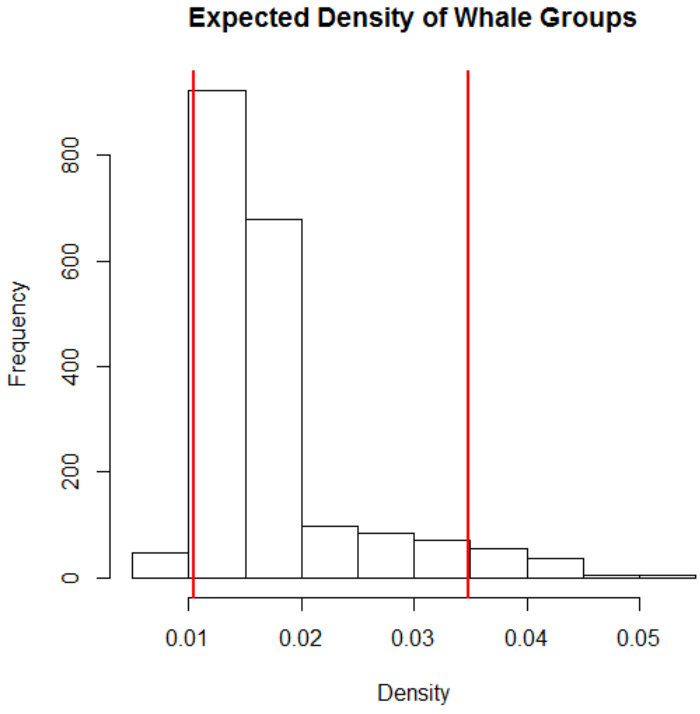


Figure 10: 95% confidence intervals for density

Two hundred days were sampled with replacement from the original data set, and for each day the estimated whale density at each grid point was calculated. The average density of whale groups across these 200 days was then calculated for each grid point. Figure 11 shows a plot of these average densities. Under the chosen model, the lowest average density of whale groups is expected directly next to the coast, which we might intuitively expect. Density of whale groups initially increases with an increase in shortest distance to the coast, and then drops off again at the edge of the covered region. For each grid point, 95% confidence limits of expected density were calculated, and these are shown in Figure 12. Again, these plots appear to show the lowest density of whale groups closest to the coast. Density initially increases as distance from the coastline increases, and then drops off again at the edge of the covered region. This perhaps reflects the tendency of humpback whales to situate themselves closer to the coast in shallow waters.

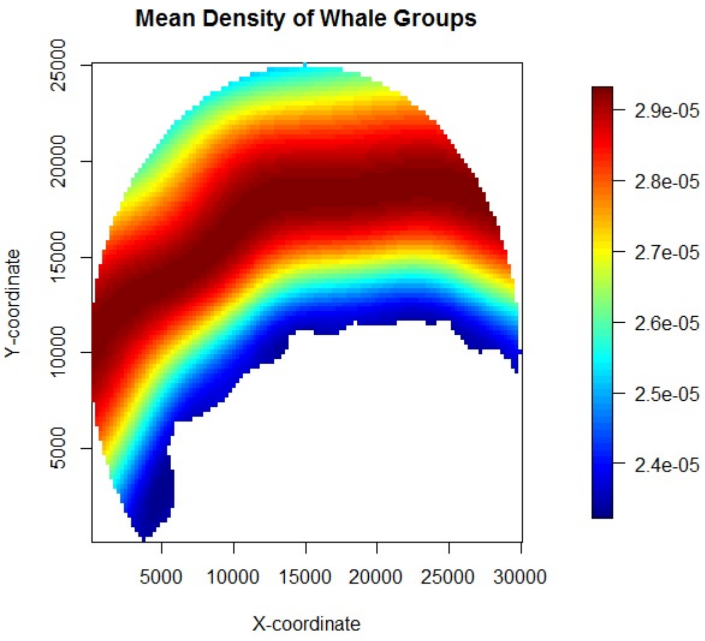


Figure 11: Mean density of whale groups

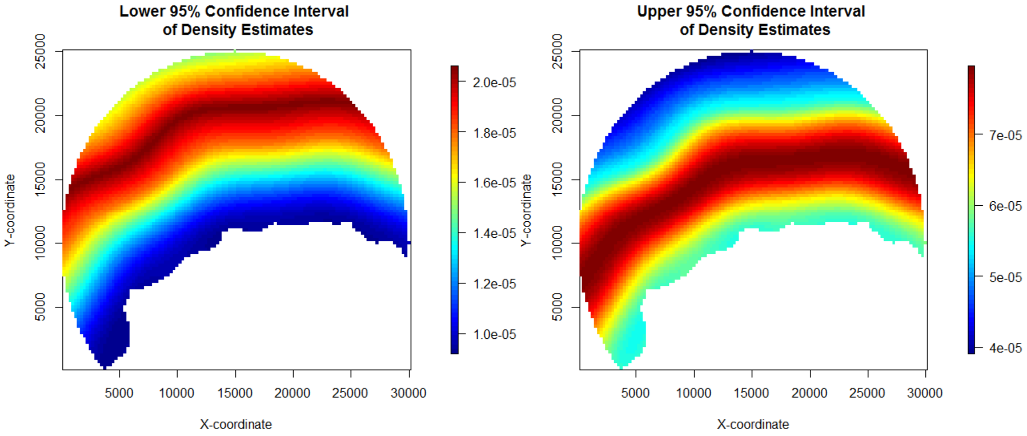


Figure 12: Lower and upper 95% confidence intervals of density

5.4 Variance of Parameter Estimates

Using day as a sampling unit, nonparametric bootstrapping was carried out to estimate the variance of the model parameter estimates. The results are shown below in Table 3. The CV (coefficient of variation) measures the variability in a model parameter relative to its expected value, and it is expressed below as a percentage. The coefficients of variation for both the mean and standard deviation of the density function are quite high at 60.62 and 33.48, respectively. This suggests high variance in these parameter estimates, which in turn suggests that our chosen model may not be the most appropriate. Furthermore, within the bootstrap resamples, the parameter estimate for the density mean attained values at the upper and lower bounds of the parameter space. This further suggests that the fit of our chosen model may be suspect.

	Point Estimate	Coefficient of Variation	Minimum	Maximum
Density Mean	9159.997	60.62	-2000	20000
Density Sigma	11065.62	33.48	2312.624	15000
Detection Sigma	4167.013	6.61	3473.661	4814.774

Table 3: Parameter Estimates and AIC Values

6 Discussion

The aim of this paper has been to obtain valid density estimates of humpback whales in Maui using data collected from a shore-based observation point. Conventional distance sampling (CDS) methods could not be used, since these methods assume that the animals to be surveyed are distributed uniformly and independently about the point of observation. This assumption is clearly violated where the point transect is positioned on the shore. Using the approach of Arranz *et al.* (submitted), the density of animals was considered to be dependent on some observable environmental variable. We initially considered ocean depth and shortest distance to coast as two environmental variables. However, the depth data to which we had access was not deemed accurate enough to perform the required analyses, and so shortest distance to the coast was considered as the explanatory variable of the density function.

Figure 13 shows plots of depth and shortest distance to coast against radial distance, which is taken to be the explanatory variable for the detection function. Since we wish to estimate the density and detection functions separately, we require there to be enough of a difference between the environmental variable (explanatory variable for density function) and radial distance (explanatory variable for detection function). Figure 13 indicates the presence of a stronger relationship between shortest distance and radial distance than there is between depth and radial distance. Therefore, depth may well prove to be a more suitable explanatory variable for the density function. Should more accurate depth data for the survey region become available in due course, this would provide grounds for further research.

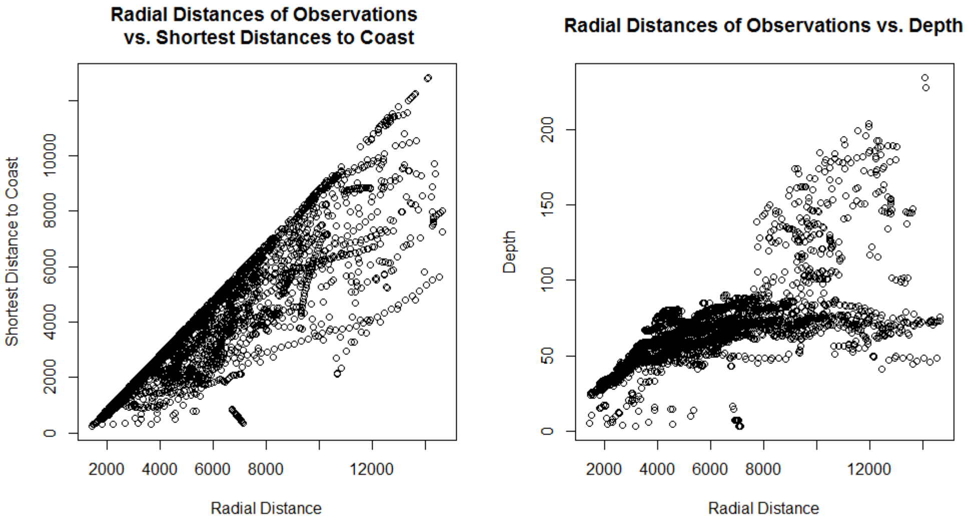


Figure 13: Plot of environmental variables against radial distance

On the basis of the AIC value alone, the 2-normal mixture would have been chosen as the most appropriate density function for the distribution model. However, when nonparametric bootstrap resamples were obtained, the parameter estimates of the density function often attained values at the upper bound of the parameter space. A normal form was therefore chosen as most appropriate for the density function. When nonparametric bootstrapping was carried out to obtain replicate estimates of density, the parameter estimates did not attain values at the bounds of the parameter space a significant number of times. However, when nonparametric bootstrapping was used to estimate the variance of the parameter estimates, the bounds of the parameter space were reached by the density mean, and the upper bound of the parameter space was reached by the density sigma. Given the relatively high coefficients of variation for the parameter estimates of the density function, this may lead us to infer that our chosen model is not necessarily the most appropriate.

For the detection function, only a half normal form was considered. The resulting detection function had a parameter estimate of $\hat{\theta} = 4167.013$, with a coefficient of variation of 6.61. This suggests that there is little variance in the parameter estimate of the detection function, and so we have no reason to doubt that this form is appropriate. In future work, it might be useful to consider a hazard rate detection function, for example.

Under the chosen model, the average density of whale groups per km^2 at any given moment in time during the survey period was estimated to be 0.0208. The nonparametric bootstrap mean for this quantity was 0.0171. Multiplying these density estimates by the area of the covered region, we obtain whale group abundance estimates of 8.27 and 6.81, respectively. Using nonparametric bootstrapping, a 95% confidence interval for the density of whale groups per km^2 was given as (0.0104, 0.0348). This corresponds to a 95% confidence interval of (4.14, 13.86) for the abundance of whale groups within the covered region at any given moment during the survey period.

Although these results certainly seem plausible, there is some doubt as to whether the chosen model was suitable. The coefficients of variation for the parameter estimates of the density function were high, indicating high variance in these estimates. Furthermore, the parameters attained values at the bounds of the parameter space when nonparametric bootstrapping was carried out to assess the variance of the parameter estimates.

6.1 Recommendations

In future work, I would recommend the consideration of ocean depth as an explanatory variable for the density function. This was not possible within this project since accurate depth data for the survey region was not readily available. I would also recommend the consideration of a hazard rate detection function within the distribution model. This function is more flexible than the half normal form in that it includes both a shape and scale parameter. It was earlier discussed that data regarding the size of whale groups was missing for many observations, and so we were unable to quantify the average size of whale groups. This project has therefore focused on estimating the density of whale groups, rather than the density of individual whales. Should information regarding average whale group size become readily available, I would suggest that this is incorporated with the results of this paper in order to obtain estimates of the density and abundance of individual whales. Finally, we assume in this paper that the whales exhibit no evasive movement in response to boats traversing the covered region during the survey period. This assumption is perhaps not realistic, and so methods to accommodate any responsive movement might be employed to gain more robust estimates of the density of humpback whales.

References

Arranz. P., D. L. Borchers, N. Aguilar Soto, M. P. Johnson and M. J. Cox. “A New Method To Study Inshore Whale Cue Distribution From Land-Based Observations.” (2013)

Borchers, D. L., S. T. Buckland and W. Zucchini. *Estimating Animal Abundance.* London: Springer-Verlag, 2002.

Buckland, S. T., D. R. Anderson, K. P. Burnham, J. L. Laake, D. L. Borchers and L. Thomas. *Introduction to Distance Sampling.* Oxford: Oxford University Press, 2001.

Cook, R. D. and J. D. Jacobson. “A design for estimating visibility bias in aerial surveys”. *Biometrics* 35 (1979): 735-742.

Cox, M. J., D. L. Borchers, D. A. Demer, G. R. Cutter and A. S. Brierley. “Estimating the density of Antarctic krill (*Euphausia superba*) from multi-beam echo-sounder observations using distance sampling methods.” *Journal of the Royal Statistical Society: Series C (Applied Statistics)* 60, no 2 (2011): 301-316.

Edrén, S. and Teilmann, J. “Humpback Whale.” *Encyclopedia of the Arctic: A-F*, edited by Mark Nuttall, 898-899. New York: Routledge, 2005.

Garthwaite. P. H., I. T. Jolliffe and B. Jones. *Statistical Inference.* New York: Oxford University Press, 2002.

Kyhn L. A., J. Tougaard, L. Thomas, L. R. Duve, J. Stenback, M. Amundin, G. Desportes and J. Teilmann. “From echolocation clicks to animal density - Acoustic sampling of harbor porpoises with static dataloggers.” *Journal of the Acoustical Society of America* 131, no 1 (2012): 550-560.

Marques, T. A., S. T. Buckland, D. L. Borchers, D. Tosh and R. A. Mcdonald. “Point transect sampling along linear features.” *Biometrics* 66, no 4 (2010): 1247-1255.

Marques, T. A., S. T. Buckland, R Bispo and B Howland. “Accounting for animal density gradients using independent information in distance sampling surveys.” *Statistical Methods and Applications* 22 (2013): 67-80.

NOAA Fisheries (National Oceanic and Atmospheric Administration). “Endangered Species Act (ESA).” http://www.nmfs.noaa.gov/pr/laws/esa/ (accessed April 19, 2013).

Thomas, L., S. T. Buckland, K. P. Burnham, D. R. Anderson, J. L. Laake, D. L. Borchers and S. Strindberg. “Distance Sampling.” *Encyclopedia of Environmetrics*, edited by Abdel H. El-Shaarawi and

Walter W. Piegorsch, 544-552. Chichester: John Wiley & Sons, 2006.

Tyack, P. “Interactions Between Singing Hawaiian Humpback Whales and Conspecifics Nearby.” *Behavioural Ecology and Sociobiology* 8, no 2 (1981): 105-116.

Acknowledgements

Many thanks to Len for his guidance and supervision, to Dan Smith of Woods Hole Oceanographic Institution for providing us with the data and for his knowledge on the collection and interpretation of data records, and for Peter Tyack of the University of St. Andrews for his help and information on the methods of data collection.

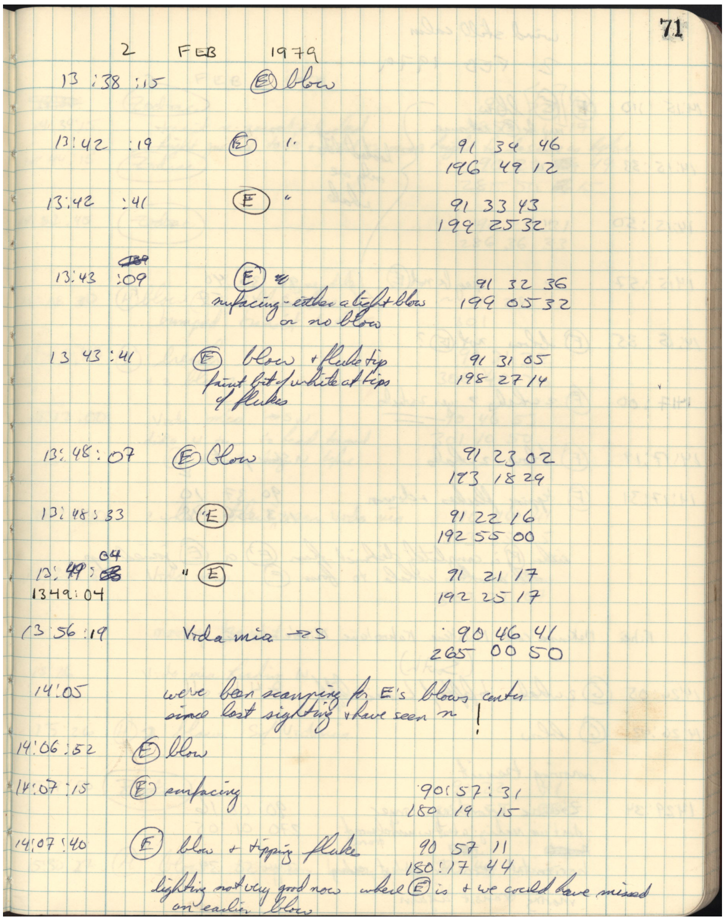
A Data Preparation

Considerable effort was expended by Dan Smith in transferring the data from its original format to the format in which the data was provided for this project (see Figures 5 and 6, respectively). In order to carry out the analyses within this project, further manipulation of the data was required. Documented within this section are the steps taken to prepare the data, along with the relevant `R` code used. For each whale observation, the data provided were as follows. These variables will be referred to throughout this section.

- `DATE` - Date of whale observation, in MM/DD/YY format
- `TIME` - Time of whale observation, in HH:MM:SS format
- `ID` - Identification letter(s) assigned to observed whale(s)
- `NO` - Size of whale group, if known
- `PLUS` - Where the group size was known to be greater than a certain number, but the exact group size was unknown, the number was followed by a +
- `VERT1, VERT2, VERT3` - Degrees, minutes and seconds, respectively, of vertical bearing
- `HORIZ1, HORIZ2, HORIZ3` - Degrees, minutes and seconds, respectively, of horizontal bearing

During the course of the data processing, new variables were created. These are defined below.

- `VERT` - Vertical bearing in decimal form
- `HORIZ` - Horizontal bearing in decimal form
- `DISTANCE` - Sighting distance of each whale observation from transit station, calculated using the formula of Lerczak and Hobbs (1998)
- `INDEX` - Numbers 1 to 121 inclusive, used as a reference to the observation days



71

2 FEB 1979

13:38:15 Ⓔ blow

13:42:19 Ⓔ " 91 34 46
196 49 12

13:42:41 Ⓔ " 91 33 43
199 25 32

13:43:~~10~~:09 Ⓔ " 91 32 36
199 05 32
surfacing - either a light blow or no blow

13 43:41 Ⓔ blow + fluke tip 91 31 05
198 27 14
faint bit of white at tips of flukes

13:48:07 Ⓔ blow 91 23 02
193 18 29

13:48:33 Ⓔ 91 22 16
192 55 00

13:~~49~~:~~03~~ 04 " Ⓔ 91 21 17
192 25 17
1349:04

13 56:19 Vida mia → S 90 46 41
265 00 50

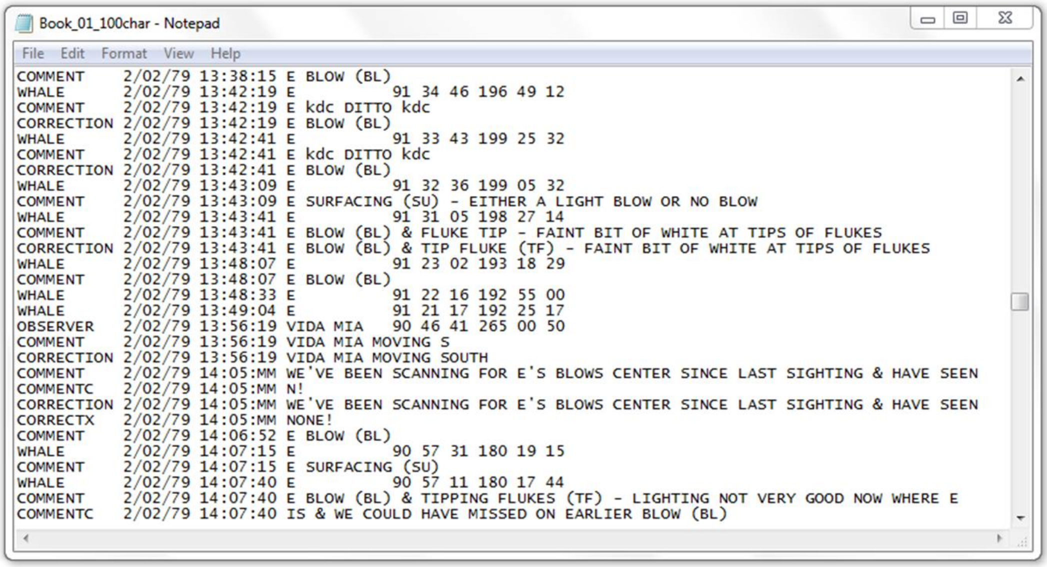
14:05 we've been scanning for E's blows center since lost sighting & have seen n !

14:06:52 Ⓔ blow

14:07:15 Ⓔ surfacing 90:57:31
180 19 15

14:07:40 Ⓔ blow + tipping flukes 90 57 11
180:17 44
lighting not very good now where Ⓔ is + we could have missed on earlier blow

Figure 5: Sample page showing data in original format



```
COMMENT     2/02/79 13:38:15 E BLOW (BL)
WHALE       2/02/79 13:42:19 E          91 34 46 196 49 12
COMMENT     2/02/79 13:42:19 E kdc DITTO kdc
CORRECTION  2/02/79 13:42:19 E BLOW (BL)
WHALE       2/02/79 13:42:41 E          91 33 43 199 25 32
COMMENT     2/02/79 13:42:41 E kdc DITTO kdc
CORRECTION  2/02/79 13:42:41 E BLOW (BL)
WHALE       2/02/79 13:43:09 E          91 32 36 199 05 32
COMMENT     2/02/79 13:43:09 E SURFACING (SU) - EITHER A LIGHT BLOW OR NO BLOW
WHALE       2/02/79 13:43:41 E          91 31 05 198 27 14
COMMENT     2/02/79 13:43:41 E BLOW (BL) & FLUKE TIP - FAINT BIT OF WHITE AT TIPS OF FLUKES
CORRECTION  2/02/79 13:43:41 E BLOW (BL) & TIP FLUKE (TF) - FAINT BIT OF WHITE AT TIPS OF FLUKES
WHALE       2/02/79 13:48:07 E          91 23 02 193 18 29
COMMENT     2/02/79 13:48:07 E BLOW (BL)
WHALE       2/02/79 13:48:33 E          91 22 16 192 55 00
WHALE       2/02/79 13:49:04 E          91 21 17 192 25 17
OBSERVER    2/02/79 13:56:19 VIDA MIA   90 46 41 265 00 50
COMMENT     2/02/79 13:56:19 VIDA MIA MOVING S
CORRECTION  2/02/79 13:56:19 VIDA MIA MOVING SOUTH
COMMENT     2/02/79 14:05:MM WE'VE BEEN SCANNING FOR E'S BLOWS CENTER SINCE LAST SIGHTING & HAVE SEEN
COMMENTC    2/02/79 14:05:MM N!
CORRECTION  2/02/79 14:05:MM WE'VE BEEN SCANNING FOR E'S BLOWS CENTER SINCE LAST SIGHTING & HAVE SEEN
CORRECTX    2/02/79 14:05:MM NONE!
COMMENT     2/02/79 14:06:52 E BLOW (BL)
WHALE       2/02/79 14:07:15 E          90 57 31 180 19 15
COMMENT     2/02/79 14:07:15 E SURFACING (SU)
WHALE       2/02/79 14:07:40 E          90 57 11 180 17 44
COMMENT     2/02/79 14:07:40 E BLOW (BL) & TIPPING FLUKES (TF) - LIGHTING NOT VERY GOOD NOW WHERE E
COMMENTC    2/02/79 14:07:40 IS & WE COULD HAVE MISSED ON EARLIER BLOW (BL)
```

Figure 6: Screenshot of provided data

A.1 Reading In An Individual File of Data

In total, twenty files of data were provided, each in the format as shown in Figure 6. The following code reads in an individual file, extracts the data relating to whale observations, and formats this data into a data frame.

```
# reads in data file from chosen directory
table<-as.data.frame(read.table("Book_01_100char_corrected.txt",fill=TRUE,header=FALSE,
       stringsAsFactors=F,col.names=paste(rep(1:50)),flush=T),stringsAsFactors=F)

# extracts rows of data relating to whale observations
attach(table)
table1<-table[X1=="WHALE",1:13]

# extracts WHALE, DATE and TIME columns
col1to3<-table1[,1:3]

# reverses order of remaining columns; this will facilitate the pattern matching to be
# carried out; these columns are then re-attached to col1to3
revcol4to13<-rev(table1[,4:13])
table1rev<-cbind(col1to3,revcol4to13)
n<-nrow(table1)
x<-numeric(n)
detach(table)

# creates a character vector of length equal to number of whale observations;
# for each row, the columns are collapsed to place all data into a single list element;
# the elements are then placed into character vector x
for (i in 1:n) {x[i]<-paste(matrix(unlist(table1rev[i,]),1,13,byrow=T),collapse=" ")}

# defines a pattern to be matched by the data for each whale sighting;
# where the pattern is matched, the list element containing sighting data is split
# accordingly into 12 columns; these split list elements are compiled into an
# nrow x 12 matrix
pattern<-"([[:upper:]]{5})[[:space:]]+([0-9]+/[0-9]+/[0-9]+)[[:space:]]+([0-9]+:[0-9]+:
         [0-9[:upper:]]+)[[:space:]]+([0-9[:upper:]]{0,3})[[:space:]]+([0-9[:upper:]]
         {0,2})[[:space:]]+([0-9[:upper:]]{0,3})[[:space:]]+([0-9[:upper:]]{0,2})
         [[:space:]]+([0-9[:upper:]]{0,2})[[:space:]]+([0-9[:upper:]]{0,2})
         ([0-9[:space:]\\+]{0,3})([0-9[:upper:][:punct:][:space:]]{0,})"
y<-regexec(pattern,x)
match<-regmatches(x,y)
match1<-match[match!="character(0)"]
res<-matrix(unlist(match1),length(match1),12,byrow=T)

# matrix containing WHALE, DATE and TIME columns
final1<-res[,2:4]

# missing seconds are taken to be equal to 30
final1[,3]<-gsub("MM",30,final1[,3])

# defines a pattern to be matched by the column containing whale ID data;
# where this pattern is matched, the whale ID column is split accordingly into 4 columns;
# these split list elements are compiled into a length(zmatch) x 4 matrix
pattern1<-"[[:space:]]*([[:upper:][:punct:]]*)[[:space:]]*([[:punct:][:upper:]]*)
          [[:space:]]*([[:upper:]]*)[[:space:]]*"
z<-regexec(pattern1,res[,12])
zmatch<-regmatches(res[,12],z)
res1<-matrix(unlist(zmatch),length(zmatch),4,byrow=T)
res1rev<-rev(as.data.frame(res1[,2:4]))
m<-length(match1)
res1rev1<-numeric(m)
```

```
# columns containing whale ID data are collapsed into a single column
for (i in 1:m) {res1rev1[i]<-paste(matrix(unlist(res1rev[i,]),1,3,byrow=T),collapse=" ")}

# removes spaces in whale ID column
res1rev1<-gsub(" ","",res1rev1)
final2<-res1rev1
final3rev<-res[,5:11]

# replaces missing data with NA
final3rev[final3rev=="MM"]<-NA
final3rev[final3rev=="MMR"]<-NA
final3rev[final3rev=="MMM"]<-NA

# where an element of the first column of final3rev is followed by an M or R, this letter
# is deleted
for(i in 1:dim(final3rev)[1]){
   if(length(final3rev[i]==3)){
     final3rev[i]<-substr(final3rev[i],1,2)
   }
}

# deletes any values of M within the matrix
final3rev<-gsub("M","",final3rev)
grp<-final3rev[,7]

# defines a pattern to be matched by data relating to group size;
# where this pattern is matched, the group size column is split accordingly into 3
# columns; these split list elements are compiled into a length(grpmatch) x 3 matrix
pattern2<-"[[:space:]]*([0-9]{0,1})[[:space:]]*([\\+]{0,1})"
grp1<-regexec(pattern2,grp)
grpmatch<-regmatches(grp,grp1)
res1<-matrix(unlist(grpmatch),length(grpmatch),3,byrow=T)
res1grp<-res1[,2:3]
final3rev1<-cbind(final3rev[,1:6],res1grp)
final3rev2<-final3rev1[,1:7]
plus<-final3rev1[,8]
final3<-rev(as.data.frame(final3rev2))

# combines all columns of data to create final data frame
book<-cbind(final1,final2,final3,plus)

# removes spaces from final column of book (containing +)
book[,12]<-gsub(" ","",book[,12])
book01<-book
```

A.2 Compiling All Data Files

The following code combines all data frames of whale observations into a single data frame. Where there is missing data with regards to the vertical/horizontal bearings, or where the whale ID is ambiguous, the observation is removed. Where two whales are seen together, the observation is split into two individual observations.

```
# loads each data frame of sightings from the specified directory
load("G:/StrongMeganHumpback/Books/book01.Rdata")
load("G:/StrongMeganHumpback/Books/book02.Rdata")
load("G:/StrongMeganHumpback/Books/book03.Rdata")
load("G:/StrongMeganHumpback/Books/book04.Rdata")
load("G:/StrongMeganHumpback/Books/book05.Rdata")
load("G:/StrongMeganHumpback/Books/book06.Rdata")
load("G:/StrongMeganHumpback/Books/book07.Rdata")
load("G:/StrongMeganHumpback/Books/book08.Rdata")
load("G:/StrongMeganHumpback/Books/book09.Rdata")
load("G:/StrongMeganHumpback/Books/book10.Rdata")
load("G:/StrongMeganHumpback/Books/book11.Rdata")
load("G:/StrongMeganHumpback/Books/book12.Rdata")
load("G:/StrongMeganHumpback/Books/book14.Rdata")
load("G:/StrongMeganHumpback/Books/book15.Rdata")
load("G:/StrongMeganHumpback/Books/bookS1.Rdata")
load("G:/StrongMeganHumpback/Books/bookS2.Rdata")
load("G:/StrongMeganHumpback/Books/bookS3.Rdata")
load("G:/StrongMeganHumpback/Books/bookS4.Rdata")
load("G:/StrongMeganHumpback/Books/bookS5.Rdata")
load("G:/StrongMeganHumpback/Books/bookS6.Rdata")#

# loads required library; lubridate changes date and time into numeric form
library(lubridate)

# combines individual data frames, retaining only unique rows
book<-unique(rbind(book01,book02,book03,book04,book05,book06,book07,book08,book09,book10,
book11,book12,book14,book15,bookS1,bookS2,bookS3,bookS4,bookS5,bookS6))

# changes date and time into numeric form; start date is 01/01/70 and start time is 00:00:00
book[,2]<-as.numeric(mdy(book[,2]),origin=mdy("1/01/70"))
book[,3]<-as.numeric(period_to_seconds(hms(book[,3])-hms("00:00:00")))

# assigns column names to data frame
colnames(book)<-c("WHALE","DATE","TIME","ID","NO","VERT1","VERT2","VERT3","HORIZ1","HORIZ2",
"HORIZ3","PLUS")

# observation removed where there are missing coordinates - omits 1868 of 16671 rows
book<-na.omit(book)
# number of rows in updated data frame
dim1<-dim(book)[1]

# the following commands reduce the number of unique whale IDs from 209 to 189

# where the whale ID is a single letter preceding a +, the + is assigned to the PLUS column
# and only the letter for the whale ID is retained
for (i in 1:dim1) {if ((nchar(as.character(book$ID[i]))==2)&(substr(book$ID[i],2,2)=="+"))
                  {book$PLUS[i]<-"+"}}
for (i in 1:dim1) {if ((nchar(as.character(book$ID[i]))==2)&(substr(book$ID[i],2,2)=="+"))
                  {book$ID[i]<-substr(book$ID[i],1,1)}}
# where the whale ID is 2 or 3 of the same letter preceding a +, the + is assigned to the
# PLUS column and only the letters for the whale ID are retained
for (i in 1:dim1) {if ((nchar(as.character(book$ID[i]))==3)&(substr(book$ID[i],3,3)=="+")&
                  (substr(book$ID[i],1,1)==substr(book$ID[i],2,2)))
                  {book$PLUS[i]<-"+"}}
for (i in 1:dim1) {if ((nchar(as.character(book$ID[i]))==4)&(substr(book$ID[i],4,4)=="+")&
                  (substr(book$ID[i],1,1)==substr(book$ID[i],2,2))&(substr(book$ID[i],1,1)
                  == substr(book$ID[i],3,3)))
                  {book$PLUS[i]<-"+"}}
for (i in 1:dim1) {if ((nchar(as.character(book$ID[i]))==3)&(substr(book$ID[i],3,3)=="+")&
                  (substr(book$ID[i],1,1)==substr(book$ID[i],2,2)))
                  {book$ID[i]<-substr(book$ID[i],1,2)}}
for (i in 1:dim1) {if ((nchar(as.character(book$ID[i]))==4)&(substr(book$ID[i],4,4)=="+")&
                  (substr(book$ID[i],1,1)==substr(book$ID[i],2,2))&(substr(book$ID[i],1,1)
                  ==substr(book$ID[i],3,3)))
                  {book$ID[i]<-substr(book$ID[i],1,3)}}
# where the whale ID is followed by ++, a + is assigned to the PLUS column and only the
# letter for the whale ID is retained
for (i in 1:dim1) {if ((nchar(as.character(book$ID[i]))==3)&(substr(book$ID[i],2,2)=="+")&
                  (substr(book$ID[i],3,3)=="+"))
                  {book$PLUS[i]<-"+"}}
for (i in 1:dim1) {if ((nchar(as.character(book$ID[i]))==3)&(substr(book$ID[i],2,2)=="+")&
                  (substr(book$ID[i],3,3)=="+"))
                  {book$ID[i]<-substr(book$ID[i],1,1)}}

# the following commands reduce the number of unique whale IDs from 189 to 179

# these commands identify IDs relating to boat names/no data/calf etc
# NOTE: these IDs are replaced with empty space - NAs will later be assigned to all empty
# spaces in the ID column
for (i in 1:dim1) {if (nchar(as.character(book$ID[i]))>=6) {book$ID[i]<-""}}
for (i in 1:dim1) {if (book$ID[i]=="NONE") {book$ID[i]<-""}}
for (i in 1:dim1) {if (book$ID[i]=="BOAT") {book$ID[i]<-""}}
for (i in 1:dim1) {if (book$ID[i]=="CALF") {book$ID[i]<-""}}

# the following commands remove "-" values that are directly after the whale IDs - the
# number of unique whale IDs is reduced from 179 to 176
for (i in 1:dim1) {if ((nchar(as.character(book$ID[i]))==2)&(substr(book$ID[i],2,2)=="-"))
                  {book$ID[i]<-substr(book$ID[i],1,1)}}

# where the whale ID is W followed by a letter EXCLUDING D,S or W, W is removed - WD may
# relate to WHALE DIFFERENT and WS may relate to WHALE SAME
# the number of unique whale IDs is reduced from 176 to 168
for (i in 1:dim1) {if ((nchar(as.character(book$ID[i]))==2)&(substr(book$ID[i],1,1)=="W")&
                  (substr(book$ID[i],2,2)!="D")&(substr(book$ID[i],2,2)!="S")&
                  (substr(book$ID[i],2,2)!="W")&(substr(book$ID[i],2,2)!="?"))
                  {book$ID[i]<-substr(book$ID[i],2,2)}}

# the following commands reduce the number of unique whale IDs from 168 to 102

# where the whale ID is ambiguous, the observation is removed
for (i in 1:dim1) {if (as.character(book$ID[i])=="?") {book$ID[i]<-""}}
for (i in 1:dim1) {if (substr(as.character(book$ID[i]),2,3)=="OR") {book$ID[i]<-""}}
for (i in 1:dim1) {if (substr(as.character(book$ID[i]),2,2)=="/") {book$ID[i]<-""}}
for (i in 1:dim1) {if (substr(as.character(book$ID[i]),2,2)=="?") {book$ID[i]<-""}}
for (i in 1:dim1) {if (substr(as.character(book$ID[i]),3,3)=="?") {book$ID[i]<-""}}
for (i in 1:dim1) {if (substr(as.character(book$ID[i]),4,4)=="?") {book$ID[i]<-""}}
for (i in 1:dim1) {if (as.character(book$ID[i])=="WD") {book$ID[i]<-""}}
for (i in 1:dim1) {if ((nchar(as.character(book$ID[i]))==2)&
                  (substr(as.character(book$ID[i]),1,1)!=substr(as.character(book$ID[i]),2,2))
                  &(substr(as.character(book$ID[i]),1,1)!="W"))
                  {book$ID[i]<-""}}
for (i in 1:dim1) {if ((nchar(as.character(book$ID[i]))==3)&
                  (substr(as.character(book$ID[i]),2,2)!="&")&(substr(as.character(book$ID[i])
                  ,2,2)!="+")&(substr(as.character(book$ID[i]),1,1)!=substr(as.character
                  (book$ID[i]),2,2))&(substr(as.character(book$ID[i]),2,2)!=substr
                  (as.character(book$ID[i]),3,3)))
                  {book$ID[i]<-""}}
for (i in 2:dim1) {if (as.character(book$ID[i])=="WS") {book$ID[i]<-book$ID[i-1]}}

# creates copy of data frame up to this point
book.update1<-book

# the following commands reduce the number of unique whale IDs from 102 to 70

# where two whales are seen together, the observation is split into two observations
split1<-book[nchar(as.character(book$ID))==5,]
split1$ID<-substr(split1$ID,1,2)
split1copy<-book[nchar(as.character(book$ID))==5,]
split1copy$ID<-substr(split1copy$ID,4,5)
split2<-book[substr(book$ID,2,2)=="&",]
```

```
split2$ID<-substr(split2$ID,1,1)
split2copy<-book[substr(book$ID,2,2)=="&",]
split2copy$ID<-substr(split2copy$ID,3,3)
split3<-book[substr(book$ID,2,2)=="+",]
split3$ID<-substr(split3$ID,1,1)
split3copy<-book[substr(book$ID,2,2)=="+",]
split3copy$ID<-substr(split3copy$ID,3,3)
for (i in 1:dim1) {if (substr(as.character(book$ID[i]),2,2)=="&") {book$ID[i]<-""}}
for (i in 1:dim1) {if (substr(as.character(book$ID[i]),3,3)=="&") {book$ID[i]<-""}}
for (i in 1:dim1) {if (substr(as.character(book$ID[i]),2,2)=="+") {book$ID[i]<-""}}
for (i in 1:dim1) {if (book$ID[i]=="") {book$ID[i]<-NA}}
book<-na.omit(book)

# creates new data frame of all observations
book.update2<-rbind(book,split1,split1copy,split2,split2copy,split3,split3copy)

# transforms vertical and horizontal bearings into decimal form
coords<-book.update2[,6:11]
coords.matrix<-array(as.numeric(as.matrix(coords)),dim=dim(as.matrix(coords)))
c<-dim(coords.matrix)[1]
vertical<-coords.matrix[,1:3]
horiz<-coords.matrix[,4:6]
v.decimal<-numeric(c)
h.decimal<-numeric(c)
for (i in 1:c) {v.decimal[i]<-vertical[i,1]+vertical[i,2]/60+vertical[i,3]/3600}
for (i in 1:c) {h.decimal[i]<-horiz[i,1]+horiz[i,2]/60+horiz[i,3]/3600}

# creates data frame containing bearings in decimal form
book.update3<-cbind(book.update2[,1:5],v.decimal,h.decimal,book.update2[,12])
colnames(book.update3)<-c("WHALE","DATE","TIME","ID","NO","VERT","HORIZ","PLUS")

# creates copy of data frame up to this point
book<-book.update3

# orders data frame by DATE and then by TIME
book<-book[order(book$DATE,book$TIME),]

# deletes rows where vertical angle is less than or equal to 90 (these will be errors) -
# a total of 4 observations are removed
book<-book[book$VERT>=90,]

# creates a vector of observation distances, using Lerczak and Hobbs method (1998)
# this method works for values of theta greater than or equal to 0.005075; if theta value
# is less than this, we round up to 0.005075
thetas<-(book$VERT-90)*pi/180
for (i in 1:dim(book)[1]) {if (thetas[i]<0.005075) {thetas[i]<-0.005075}}
re<-6371
h<-0.082
distances<-numeric(dim(book)[1])
for (i in 1:dim(book)[1]) {
    distances[i]<-re*(asin(sin(pi/2-thetas[i])*(re+h)/re)-(pi/2-thetas[i]))
}

# creates new data frame with sighting distances included
book.update4<-cbind(book[,1:5],distances,book[,7:8])
colnames(book.update4)<-c("WHALE","DATE","TIME","ID","NO","DISTANCE","HORIZ","PLUS")

book<-book.update4

# omits all observations where distance is greater than 15km - omits 648 of 12958 obsns
# (a reduction of 5%)
book<-book[book$DISTANCE<=15,]
```

```
# assigns column names to data frame
colnames(book)<-c("WHALE","DATE","TIME","ID","NO","DISTANCE","HORIZ","PLUS")
```

A.3 Calculating Cartesian Coordinates of Observations

The following `R` code creates two new data frames, `bookxy` and `bookxy2`. The former contains the Cartesian coordinates of the whale observations from the data frame `book`, where the x-axis is taken to be in line with the horizontal zero (36.7° degrees west of true north). The latter contains the Cartesian coordinates of the whale observations, where the x-axis is taken to be perpendicular to true north, with the positive x-axis in the easterly direction. In either case, the transit station is taken to be at the origin (i.e. at $(0, 0)$).

```
# loads book from specified directory
load("G:/StrongMeganHumpback/Books/book.Rdata")
attach(book)

# ensures that horizontal angles increase as we move anticlockwise
# sets x axis to be in line with horizontal zero (36.7 degrees west of true north)
horiz.new<-360-book$HORIZ
horizrad<-horiz.new*(pi/180)
# calculates x and y coordinates of observations
x<-book$DISTANCE*cos(horizrad)
y<-book$DISTANCE*sin(horizrad)

# sets x axis to be perpendicular to true north
horiz.new2<-book$HORIZ-126.7
for (i in 1:(length(horiz.new2))) {if (horiz.new2[i]<0) {horiz.new2[i]<-360+horiz.new2[i]}}
# ensures that horizontal angles increase as we move anticlockwise
horiz.new2<-360-horiz.new2
horizrad2<-horiz.new2*(pi/180)
# calculates x and y coordinates of observations
x2<-book$DISTANCE*cos(horizrad2)
y2<-book$DISTANCE*sin(horizrad2)

bookxy<-cbind(book[,1:5],x,y,book[,8])
bookxy2<-cbind(book[,1:5],x2,y2,book[,8])

# loads file containing coordinates of Maui coast where x axis is in line with horizontal zero
load("G:/StrongMeganHumpback/Books/maui.Rdata")
# loads file containing coordinates of Maui coast where x axis is perpendicular to true north
load("G:/StrongMeganHumpback/Books/maui.north.Rdata")

# loads required library; SDMTools allows us to identify which observations lie on land
library(SDMTools)

# for both bookxy and bookxy2, finds which points lie on coast and deletes these points
# these observations will either be errors, or will have been obtained using a different
# horizontal zero
points<-cbind(x,y)
points2<-cbind(x2,y2)
coast<-maui
coast2<-maui.north

# identifies the points in bookxy that lie on land
pip<-pnt.in.poly(points,coast)
pip1<-pip[pip$pip<1,]

# identifies the points in bookxy2 that lie on land
pip2<-pnt.in.poly(points2,coast2)
```

```
pip2.1<-pip2[pip2$pip<1,]

# identifies the observations that lie in the sea
insea<-which(pip$pip==0)
bookxy<-bookxy[insea,]
bookxy2<-bookxy2[insea,]

# adds extra column to data frames, containing index relating to date
uniquedays<-unique(bookxy$DATE)
numunique<-length(uniquedays)
obsperday<-numeric(length(uniquedays))
for (i in 1:numunique) {obsperday[i]<-nrow(bookxy[bookxy$DATE==uniquedays[i],])}
index<-rep(1:length(uniquedays),obsperday)
bookxy<-cbind(bookxy,index)
bookxy2<-cbind(bookxy2,index)

# assigns column names
colnames(bookxy)<-c("WHALE","DATE","TIME","ID","NO","X","Y","PLUS","INDEX")
colnames(bookxy2)<-c("WHALE","DATE","TIME","ID","NO","X","Y","PLUS","INDEX")
```

A.4 Obtaining Interpolated Observations at Snapshot Moments

As previously discussed, an assumption of point transect sampling is that animals are detected at their initial location. In practice, we obtain a ‘snapshot’ to estimate the location of animals at a single moment in time. Using the method of linear interpolation, the function documented below obtains snapshots at five minute intervals from 8:05am to 6pm on each day of observation, to estimate the location of whales seen at least twice on that day.

```
# Purpose: Uses linear interpolation to obtain estimates of whale locations at snapshot
#          moments. The snapshot moments are defined by the input values when the function
#          is called. It is assumed that whales move at a constant speed and in a constant
#          direction between sightings.
#
# Inputs: data.frame - data frame of whale observations including (at a minimum) INDEX, TIME,
#         ID, X and Y columns
#         split - no. of seconds between each snapshot moment;
#                 default = 300 (for 5 min intervals)
#         stime - time of first snapshot moment in seconds;
#                 default = 29100 (08:05:00)
#         ftime - time of last snapshot moment in seconds;
#                 default = 64800 (18:00:00)
#
# Output: Data frame of whale observations with locations obtained by linear interpolation.
#         The output data frame contains TIME, X, Y, ID and INDEX columns


interpolate<-function(data.frame, split=300, stime=29100, ftime=64800) {

    # total number of unique days for which we have whale sightings
    numdays<-length(unique(data.frame$INDEX))

    # creates empty data frame to be filled with interpolated observations across all days
    interp.all.days<-data.frame(TIME=numeric(),X=numeric(),Y=numeric(),IDs=numeric(),
    INDEX=numeric())

    # returns vector containing time of each snapshot (in seconds with start time of 00:00:00)
    totalsec<-ftime-(stime-split)
    numsnap<-totalsec/split
    snaptime<-numeric(numsnap)
```

```
for (s in 1:numsnap) {snaptime[s]<-(((stime-split)/split)+s)*split}

for (i in 1:numdays) {

    # splits data frame according to day
    day<-data.frame[data.frame$INDEX==i,]
    # returns vector of unique whales seen on the day
    unique.whales<-unique(day$ID)
    lunique<-length(unique.whales)
    # no. of sightings of each whale per day
    count<-numeric(lunique)
    for (c in 1:lunique) {count[c]<-sum(day$ID==unique.whales[c])}
    # returns whale IDs for which there is more than one obsn per day (a minimum of
    # two sightings is needed for linear interpolation
    gthan1<-which(count>1)
    whales.new<-unique.whales[gthan1]
    lunique1<-length(whales.new)

    # creates empty data frame to be filled with interpolated locations of whales on
    # this day
    empty<-data.frame(TIME=numeric(),X=numeric(),Y=numeric(),ID=character())

    for (j in 1:lunique1) {

        # splits data frame according to whale ID
        id<-day[day$ID==whales.new[j],]
        n<-dim(id)[1]-1

        for (k in 1:n) {

            perint<-id[k:(k+1),]

            # Purpose: To obtain interpolated observations of whale between 2 sightings
            #
            # Input: data.frame - data frame consisting of 2 consecutive whale sightings
            #
            # Output: Data frame containing whale location at snapshot times, where
            #         locations are obtained by linear interpolation

            interpolate.id<-function(data.frame) {

                # returns start and end x coordinates
                xstart<-id$X[k]
                xfinish<-id$X[k+1]
                # change in x between consecutive obsns
                xdiff<-xfinish-xstart

                # returns start and end y coordinates
                ystart<-id$Y[k]
                yfinish<-id$Y[k+1]
                # change in y between consecutive obsns
                ydiff<-yfinish-ystart

                # time difference (seconds) between consecutive obsns
                timediff<-id$TIME[k+1]-id$TIME[k]
                # time list is a vector of length equal to time diff
                timelist<-c(id$TIME[k]:(id$TIME[k+1]-1))

                # change in x and y coordinates per second
                xsplit<-xdiff/timediff
                ysplit<-ydiff/timediff
```

```
                    # returns vectors containing x and y coordinates of whale at each second
                    # between sightings
                    x<-numeric(length(timelist))
                    y<-numeric(length(timelist))
                    for (m in 1:length(timelist)) {
                        x[m]<-xstart+xsplit*(m-1)
                        y[m]<-ystart+ysplit*(m-1)
                    }

                    # creates data frame of snapshot times, coordinates and whale ID
                    xy.seconds<-data.frame(cbind(timelist,x,y))
                    # identifies entries of xy.seconds where time is equal to one of snapshot
                    # moments
                    matches<-match(snaptime,xy.seconds$timelist)
                    snapshots<-xy.seconds[matches,]
                    whale.id<-rep(id$ID[1],dim(snapshots)[1])
                    snapshots.with.id<-cbind(snapshots,whale.id)
                    snapshots.with.id<-na.omit(snapshots.with.id)

                }

                # for each pair of consecutive whale obsns, interpolate.id function is applied
                # and resulting data frame added to empty data frame
                add<-interpolate.id(perint)
                empty<-rbind(empty,add)

            }

            # adds column containing index
            index<-rep(id$INDEX[1],nrow(empty))
            empty.with.index<-cbind(empty,index)
            empty.with.index<-na.omit(empty.with.index)

        }

        # adds data frame of obsns at snapshot
        interp.all.days<-rbind(interp.all.days,empty.with.index)

    }

    colnames(interp.all.days)<-c("TIME","X","Y","IDs","INDEX")
    return(interp.all.days)

}
```

A.5 Plot of Whale Sightings and Interpolated Observations

Given a specific day and whale ID, the following function creates a plot of the estimated whale track (assuming movement of constant speed and direction between sightings), with start and end position clearly marked. The interpolated observations of the whale at the snapshot moments are also included in the plot.

```
# Purpose: To obtain a singe plot consisting of estimated whale track (assuming movement
#          has constant speed and direction between sightings), location of actual sightings
#          and the estimated locations of whales at snapshot moments (obtained by linear
#          interpolation)
#
# Inputs: data.frame1 - data frame containing actual whale obsns
#         data.frame2 - data frame containing estimated locations of whales at snapshot
#                       moments, obtained by linear interpolation
#         day - index value denoting day of obsns for which we want to compare actual
#               obsns with estimated locations obtained by linear interpolation;
#               default = 1
#         id - whale ID for which we want to compare actual obsns with estimated
#              locations obtained by linear interpolation;
#              default = A
#
# Output: Plot of estimated whale track, including points of actual observations,
#         along with points to show estimated location of whale at snapshot moments

plot.obs.est<-function(data.frame1=bookxy,data.frame2=int5,day=1,id="A") {

    # subsets input data frames to include only the obsns from specified day
    book.day<-data.frame1[data.frame1$INDEX==day,]
    interp.day<-data.frame2[data.frame2$INDEX==day,]
    # subsets above data frames to include only the obsns for specified whale ID
    book.day.id<-book.day[book.day$ID==id,]
    interp.day.id<-interp.day[interp.day$ID==id,]

    # returns observed x and y coordinates
    xknown<-book.day.id$X
    yknown<-book.day.id$Y
    # returns estimated x and y coordinates at snapshot moments
    xint<-round(interp.day.id$X,9)
    yint<-round(interp.day.id$Y,9)

    lx<-length(xknown)-1

    id<-id
    match<-which(date$INDEX==day)[1]
    date<-date$DATE[match]

    # Purpose: To include specified day and whale ID in plot title
    #
    # Inputs: day - day of obsns as specified in input of plotcompare
    #         id - whale ID as specified in input of plotcompare
    #
    # Output: Second row of plot title, containing date and whale ID

    title<-function(date,id) {

        subtitle<-paste("Whale", id, "on", date)
        return(subtitle)

    }

    secondrow<-title(date,id)

    # plot actual points at which whale was observed
    plot(xknown,yknown,col='blue',pch=20,xlab="X-coordinate",ylab="Y-coordinate",main=paste
    ("Whale Sightings and Interpolated Snapshot Locations",secondrow,sep='\n'))
    # plot track between each sighting, assuming movement has constant speed and direction
    for (i in 1:lx) {lines(c(xknown[i],xknown[i+1]),c(yknown[i],yknown[i+1]),col='blue')}
    # mark beginning and end of whale track
    points(xknown[1],yknown[1],col='green',pch=20)
    points(xknown[lx+1],yknown[lx+1],col='red',pch=20)
    # add points to plot to show estimated locations at snapshot moments
    points(xint,yint,col='orange',pch=20)

    # include legend on plot
    # note that the position of the legend may need to be changed depending on whale track
    labels<-c("Start position","Observed locations","Estimated track","End position",
    "Estimated locations")
    colours<-c("green","blue","blue","red","orange")
    legend('topright',labels,pch=c(20,20,NA,20,20),lty=c(0,0,1,0,0),col=colours,cex=0.8)

}
```

A.6 Longitude and Latitude of Each Interpolated Observation

The transit station has known longitude of 156.62°W and known latitude of 20.82°N. Given the x and y coordinates of the observations relative to the transit station, the longitude and latitude of each interpolated sighting is calculated.

```
# loads book of interpolated observations where x axis is in line with horizontal zero
load("G:/StrongMeganHumpback/Books/int30.Rdata")
# loads book of interpolated observations where x axis is in easterly direction
load("G:/StrongMeganHumpback/Books/interpolations30.Rdata")

# latitude and longitude of transit station (in radians)
startlat<-(20.8222222)*pi/180
startlong<-(-156.6208333)*pi/180
# difference in longitude between each observation and transit station
dlong<-interpolations30$X/(6371*cos(startlat))
# vector of longitude values for each observation (in degrees)
endlong<-(startlong+dlong)*180/pi
# difference in latitude between each observation and transit station
dlat<-interpolations30$Y/6371
# vector of latitude values for each observation (in degrees)
endlat<-(startlat+dlat)*180/pi

# creates data frame containing interpolated locations (x axis 36.7 degrees west
# of true north), along with longitude and latitude values
int30.longlat<-cbind(int30,endlong,endlat)
# assigns column names
colnames(int30.longlat)<-c("TIME","X","Y","IDs","INDEX","LONG","LAT")
```

A.7 Longitude and Latitude of Maui Coastline

Cartesian coordinates of the Maui coastline with respect to the transit station were provided by Dan Smith. The following code converts these Cartesian coordinates into longitude and latitude coordinates. We need the longitude and latitude coordinates of the coastline in order to calculate the shortest distance from each observation to the coast (using the `spDistsN1` function in `R`).

```
# loads data frame containing x and y coordinates of Maui coastline, where x axis
# is in easterly direction
load("G:/StrongMeganHumpback/Books/maui.north.Rdata")

# converts metre distances to km distances
maui.north$V1<-maui.north$V1/1000
maui.north$V2<-maui.north$V2/1000

# latitude and longitude of transit station (in radians)
startlat<-(20.8222222)*pi/180
startlong<-(-156.6208333)*pi/180
# difference in longitude between each coordinate and transit station
dlong<-maui.north$V1/(6371*cos(startlat))
```

```
# vector of longitude values for coordinates on coastline (in degrees)
endlong<-(startlong+dlong)*180/pi
# difference in latitude between each coordinate and transit station
dlat<-maui.north$V2/6371
# vector of latitude values for each coordinate on coastline (in degrees)
endlat<-(startlat+dlat)*180/pi

# creates data frame containing x and y coordinates of Maui coastline, along with
# longitude and latitude values
maui.north.longlat<-cbind(maui.north,endlong,endlat)
colnames(maui.north.longlat)<-c("X","Y","LONG","LAT")
```

A.8 Depth, Shortest Distance to Coast & Radial Distance from Observer for Each Interpolated Observation

For each interpolated observation, the following code identifies the closest point for which we have depth data. The depth at the location of observation is taken to be equal to the depth at this closest point. Using the spDistsN1 function within the sp library, the closest distance from each point to the Maui coastline is also calculated. Finally, the code calculates the radial distance from the observer to each sighting location.

```
# loads data frame of depths, data frame of interpolated observations and data frame of
# Maui coastline
load("G:/StrongMeganHumpback/Books/depth.Rdata")
load("G:/StrongMeganHumpback/Books/int5.longlat.Rdata")
load("G:/StrongMeganHumpback/Books/maui.longlat.Rdata")

# loads required library; sp allows us to calculate closest distance from a point to
# a set of points (in this case, the Maui coastline)
library(sp)

# converts x and y coordinates from km to m; also ensures that the entire survey region
# lies within the first quadrant (i.e. both x and y coordinates are positive); this
# moves position of transit station from (0,0) to (15000,10000)
int5.longlat$X<-(int5.longlat$X+15)*1000
int5.longlat$Y<-(int5.longlat$Y+10)*1000
# number of interpolated observations
dim<-dim(int5.longlat)[1]

# converts depth data frame to matrix
depth<-as.matrix(depth)

# for each interpolated observation, identifies closest point for which we have depth
# data (as included in depth data frame); the vector returned is the depths at these
# closest points
depth.col<-numeric(dim)
for (i in 1:dim) {
    point<-c(int5.longlat$LONG[i],int5.longlat$LAT[i])
    min<-which.min(spDistsN1(depth[,1:2],point,longlat=TRUE))
    depth.col[i]<-depth[min,3]
}

# for each interpolated observation, identifies shortest distance to coastline; this
# distance is returned in km by spDistsN1, and so we convert this distance to metres
maui.long<-maui.longlat[,3]
maui.lat<-maui.longlat[,4]
long.lat<-cbind(maui.long,maui.lat)
short.dist<-numeric(dim)
for (j in 1:dim) {
```

```
point<-c(int5.longlat$LONG[j],int5.longlat$LAT[j])
    short.dist[j]<-min(spDistsN1(long.lat,point,longlat=TRUE))
}
short.dist<-short.dist*1000

# for each interpolated observation, calculates radial distance from observation to
# transit station
radial<-numeric(dim)
for (j in 1:dim) {
    x<-int5.longlat$X[j]
    y<-int5.longlat$Y[j]
    obsn<-c(x,y)
    radial[j]<-sqrt((x-15000)^2+(y-10000)^2)
}

# creates new data frame containing depth at point closest to each interpolated
# observation, shortest distance from each observation to coastline, and radial distance
# to each interpolated observation from transit station
obsns<-cbind(int5.longlat, depth.col, short.dist, radial)
colnames(obsns)<-c("TIME","X","Y","IDs","INDEX","LONG","LAT","DEPTH","SHORTDIST","RDIST")

# changes depths to be positive
obsns$DEPTH<--obsns$DEPTH
# this removes one problematic observation, in which the closest point for which we
# have depth data is on land (and so depth is negative)
obsns<-obsns[obsns$DEPTH>=0,]

# for the analysis, we retain only a subset of days out of the 118 for which we still
# have whale observation data (following the data processing documented above); we
# generate a random number between 1 and 5 (to select the first day), and then
# systematically choose days at 4 day intervals; the random number generated was 1,
# and so we have 30 days within the subset (first day indexed by 1, last by 117);
# we subset the obsns data frame to include only data from days included in subset
list<-cumsum(c(sample(1:5,1),rep(4,35)))
list<-list[list<=118]
include<-numeric(dim(obsns)[1])
for (i in 1:dim(obsns)[1]) {
    for (j in 1:length(list)) {
        if (obsns$INDEX[i]==list[j]) {
            include[i]<-1
        }
    }
}
subset<-which(include==1)
obsns<-obsns[subset,]

# finds number of snapshots on each day included in subset
snaps<-numeric(length(unique(obsns$INDEX)))
for (k in 1:length(snaps)) {
    split<-obsns[obsns$INDEX==list[k],]
    snapshots<-length(unique(split$TIME))
    snaps[k]<-snapshots
}

snapshots<-cbind(list,snaps)
colnames(snapshots)<-c("INDEX","SNAPSHOTS")
```

A.9 Calculating Area of Covered Region

We require the area of the region covered by the survey in order to calculate estimated abundance of whales given density estimates. This area is equal to $398.2575 km^2$.

```
# loads data file containing Cartesian coordinates of Maui coastline
load("G:/StrongMeganHumpback/Books/maui.Rdata")

# loads required library; gpclib calculates area of polygons
library(gpclib)

# converts Maui coordinates from km to metres and creates polygon in required form
# from these coordinates
maui[,1]<-maui[,1]*1000
maui[,2]<-maui[,2]*1000
mauigpc<-as(maui,"gpc.poly")

# Purpose: To obtain a matrix containing the Cartesian coordinates of a circle with centre
#          and radius as specified
#
# Inputs: x - x coordinate of circle centre
#         y - y coordinate of circle centre
#         rad - radius of circle; default = 1
#         nvert - number of pairs of Cartesian coordinates to obtain; default = 500
#
# Output: nvert x 2 matrix containing Cartesian coordinates of circle

circle<-function(x,y,rad=1,nvert=500){
    rads<-seq(0,2*pi,length.out=nvert)
    xcoords<-cos(rads)*rad+x
    ycoords<-sin(rads)*rad+y
    coords<-cbind(xcoords,ycoords)
    return(coords)
}

# obtains coordinates of circle we require (taking transit station to be at (0,0))
# and truncating at 15km
circ<-circle(0,0,15000)

# creates polygon in required form from returned Cartesian coordinates
circgpc<-as(circ,"gpc.poly")

# calculates area of intersection between circle and polygon defined by coastline coords
intersect<-area.poly(mauigpc)+area.poly(circgpc)-area.poly(union(mauigpc,circgpc))

# subtracts this area of intersection from area of circle to obtain the total
# area covered within the survey; this is equal to 398.2575km^2
surveyarea<-area.poly(circgpc)-intersect

# the following code creates a shaded plot showing the survey region
# shades area of circle that does not intersection with Maui coastline; this shaded area
# represents the survey region
plot(setdiff(circgpc, mauigpc), poly.args = list(col = "dodgerblue2"), add = FALSE)
# shades Maui coastline
plot(union(setdiff(mauigpc,circgpc),intersect(mauigpc,circgpc)), poly.args =
list(col = "grey"), add = TRUE)
# adds point at location of transit station
points(0,0,pch=20,cex=2)
# adds title to plot
title(main="Plot of Survey Region")
```

B R code

Documented within this section is the R code used to carry out the analysis upon which this project focuses.

B.1 Placing Grid Across Survey Area

Within this project, we parameterise the density model $\pi(r)$ as a function of some observable environmental variable, namely shortest distance to coast. Since the relationship between this shortest distance and the Cartesian coordinates is not simple, we place a grid across the survey area, and calculate the shortest distance to the coast at each of the grid points that fall within the survey region. For grid points that fall on land or beyond the truncation distance of 15km, we replace the depth with NA.

```
# loads depth data frame, and data frame containing coordinates of Maui coastline
load("G:/StrongMeganHumpback/Books/maui.Rdata")
load("G:/StrongMeganHumpback/Books/depth.Rdata")
load("G:/StrongMeganHumpback/Books/maui.longlat.Rdata")

# converts Cartesian coordinates of Maui coastline to km, and changes them relative
# to location of observer
maui[,1]<-(maui[,1]+15)*1000
maui[,2]<-(maui[,2]+10)*1000

# loads required library; sp allows us to place a grid over survey region
library(sp)

# specifies grid to span survey region
x<-GridTopology(c(250,250), c(250,250), c(120,100))

# identifies coordinates of all grid points
coords<-coordinates(x)

# total number of grid points
gridpts<-dim(coords)[1]

# converts depth to matrix
depth<-as.matrix(depth)

# for each grid point, calculates radial distance to transit station
radial<-numeric(gridpts)
for(i in 1:gridpts) {
    radial[i]<-sqrt((coords[i,1]-15000)^2+(coords[i,2]-10000)^2)
}

# data frame of x and y-coordinates with corresponding radial distances
coords<-cbind(coords,radial)

# changes coordinates so that transit station is at origin; this will allow us to
# obtain longitude and latitude of each grid point
coords[,1]<-coords[,1]-15000
coords[,2]<-coords[,2]-10000

# splits grid according to positive/negative x/y values
xpos<-coords[coords[,1]>=0,]
xneg<-coords[coords[,1]<0,]
ypos<-coords[coords[,2]>=0,]
yneg<-coords[coords[,2]<0,]
```

```
# identifies the four quadrants
q1<-ypos[ypos[,1]>=0,]
q2<-ypos[ypos[,1]<0,]
q3<-yneg[yneg[,1]<0,]
q4<-yneg[yneg[,1]>=0,]

# calculates counterclockwise angle from x axis to each gridpoint
q1angle<-atan(q1[,2]/q1[,1])
q2angle<-atan(q2[,2]/q2[,1])+pi
q3angle<-atan(q3[,2]/q3[,1])+pi
q4angle<-atan(q4[,2]/q4[,1])+(2*pi)

# matrices of coordinates, radial distances and angles
q1all<-cbind(q1,q1angle)
colnames(q1all)<-c("X","Y","d","theta")
q2all<-cbind(q2,q2angle)
colnames(q2all)<-c("X","Y","d","theta")
q3all<-cbind(q3,q3angle)
colnames(q3all)<-c("X","Y","d","theta")
q4all<-cbind(q4,q4angle)
colnames(q4all)<-c("X","Y","d","theta")

# combines above matrices into a data frame
cart.polar<-as.data.frame(rbind(q1all,q2all,q3all,q4all))

# converts angles to degrees
cart.polar[,4]<-cart.polar[,4]*(180/pi)

# calculates angle bearing from true north
cart.polar[,4]<-360-(cart.polar[,4]+36.7)

# converts angles back to radians
cart.polar[,4]<-cart.polar[,4]*pi/180

# latitude and longitude of transit station (in radians)
startlat<-(20.8222222)*pi/180
startlong<-(-156.6208333)*pi/180

# defines radius of the earth
re<-6371000

# defines vectors consis
d<-cart.polar[,3]
theta<-cart.polar[,4]

# orders data frame by y and then x coordinates
cart.polar<-cart.polar[order(-cart.polar[,2],cart.polar[,1]),]

# creates vectors of latitude and longitude of each grid point
# we need to know the latitude and longitude in order than we can identify
# the closest point for which we have depth data
endlat<-numeric(gridpts)
endlong<-numeric(gridpts)
for (i in 1:gridpts) {
    endlat[i]<-asin((sin(startlat)*cos(cart.polar$d[i]/re))+(cos(startlat)*sin
    (cart.polar$d[i]/re)*cos(cart.polar$theta[i])))endlong[i]<-startlong+atan2
    (sin(cart.polar$theta[i])*sin(cart.polar$d[i]/re)*cos(startlat),cos
    (cart.polar$d[i]/re)-sin(startlat)*sin(endlat[i]))
}

# converts latitude and longitude to degrees
```

```
endlat<-endlat*180/pi
endlong<-endlong*180/pi

cart.polar[,1]<-cart.polar[,1]+15000
cart.polar[,2]<-cart.polar[,2]+10000

# combines cart.polar data frame with latitude and longitude values
grid<-cbind(cart.polar,endlong,endlat)
colnames(grid)<-c("X","Y","RDIST","THETA","LONG","LAT")

# for grid point at origin, amends grid entry
grid[7260,4]<-0
grid[7260,5]<-startlong*180/pi
grid[7260,6]<-startlat*180/pi

# for each grid point, identifies closest point for which we have depth data (as included
# in depth data frame); the vector returned is the depths at these closest points
dim<-dim(grid)[1]
depth.col<-numeric(dim)
for (i in 1:dim) {
    point<-c(grid$LONG[i],grid$LAT[i])
    min<-which.min(spDistsN1(depth[,1:2],point,longlat=TRUE))
    depth.col[i]<-depth[min,3]
}

# for each grid point, identifies shortest distance to coastline
maui.long<-maui.longlat[,3]
maui.lat<-maui.longlat[,4]
long.lat<-cbind(maui.long,maui.lat)
short.dist<-numeric(dim)
for (j in 1:dim) {
    point<-c(grid$LONG[j],grid$LAT[j])
    short.dist[j]<-min(spDistsN1(long.lat,point,longlat=TRUE))
}
short.dist<-short.dist*1000

# identifies grid points that lie within a 15km radial distance of the transit station
# we will retain grid points for which inrange is equal to 0
dim<-dim(grid)[1]
inrange<-numeric(dim)
for (j in 1:dim) {
    if (grid$RDIST[j]<=15000) {
        inrange[j]<-0} else {inrange[j]<-1
    }
}

# loads required library; SDMTools allows us to identify which grid points lie on land
library(SDMTools)

points<-as.matrix(grid[,1:2])
coast<-maui

pip<-pnt.in.poly(points,coast)

insea<-numeric(dim)
for (k in 1:dim) {
    if (pip$pip[k]<=0) {
        insea[k]<-0} else {insea[k]<-1
    }
}

# sums the entries of inrange and insea; where the entry is greater than zero, the
# corresponding grid point will either be beyond a 15km radial distance of the
# transit station, or the grid point will be on land
discard<-inrange+insea
retain<-which(discard==0)

# creates vector where each grid point is assigned a value;  a value of 1 means that
# grid point is within survey region, a value of 0 means that it is not
sregion<-numeric(dim)
for (m in 1:length(discard)) {
    if (discard[m]<=0) {
        sregion[m]<-1} else {sregion[m]<-0
    }
}

# changes sign of values in depth column
grid$DEPTH<--grid$DEPTH

# combines grid data frame with column indexing whether grid point lies in survey
# region
grid<-cbind(grid[,1:2],depth.col,short.dist,grid[,3],sregion)
colnames(grid)<-c("X","Y","DEPTH","SHORTDIST","RDIST","SREGION")
```

B.2 Calculating Derivatives of Depth and Shortest Distance to Coast WRT y for Each Interpolated Observation

Using the nupoint package within R to fit density and detection models, we require the derivative of the environmental variable z at each sighting with respect to y (the offshore distance). This code identifies the grid point to the top right of each sighting, and the grid point directly below this (i.e. with the same x-coordinate, but with a y-coordinate of 250m less than the first point. To calculate the derivative of shortest distance to coast with respect to y at the observation, the difference in shortest distance between these two grid points is calculated and divided by 250. An analogous method is used to calculate the derivative of depth with respect to y for each observation.

```
# loads data frame of interpolated observations and grid data
load("G:/StrongMeganHumpback/Books/obsns.Rdata")
load("G:/StrongMeganHumpback/Books/grid.Rdata")

# finds derivative of shortest distance to coast WRT y at each sighting
dzdy.obsns.dist<-numeric(dim(obsns)[1])

for (i in 1:dim(obsns)[1]) {

    point<-c(obsns$X[i],obsns$Y[i])
    xgrid<-unique(grid[,1])
    ygrid<-unique(grid[,2])

    right<-sort(xgrid[xgrid>=point[1]])
    above<-sort(ygrid[ygrid>=point[2]])

    x<-right[1]
    y<-above[1]
    y1<-y-250

    ptsx<-grid[grid[,1]==x,]
    ptsy<-ptsx[ptsx[,2]==y,]
    ptsy1<-ptsx[ptsx[,2]==y1,]
    dist.change<-ptsy$SHORTDIST-ptsy1$SHORTDIST
    diff<-dist.change/250
    dzdy.obsns.dist[i]<-diff

}

# finds derivative of depth WRT y at each sighting
dzdy.obsns.depth<-numeric(dim(obsns)[1])

for (i in 1:dim(obsns)[1]) {

    point<-c(obsns$X[i],obsns$Y[i])
    xgrid<-unique(grid[,1])
    ygrid<-unique(grid[,2])

    right<-sort(xgrid[xgrid>=point[1]])
    above<-sort(ygrid[ygrid>=point[2]])

    x<-right[1]
    y<-above[1]
    y1<-y-250

    ptsx<-grid[grid[,1]==x,]
    ptsy<-ptsx[ptsx[,2]==y,]
    ptsy1<-ptsx[ptsx[,2]==y1,]
    depth.change<-ptsy$DEPTH-ptsy1$DEPTH
    diff<-depth.change/250

    dzdy.obsns.depth[i]<-diff

}

# creates two new data frames; one including depth and derivatives of depth, the
# other including shortest distance to coast and derivatives
np.depth<-cbind(obsns[,2:3],obsns[,10],obsns[,8],dzdy.obsns.depth,obsns[,5])
colnames(np.depth)<-c("x","y","r","z","dzdy","obs.period")
np.dist<-cbind(obsns[,2:3],obsns[,10],obsns[,9],dzdy.obsns.dist,obsns[,5])
colnames(np.dist)<-c("x","y","r","z","dzdy","obs.period")
```

B.3 Calculating Derivative of Depth and Shortest Distance to Coast WRT y for each Grid Point

Model fitting in the nupoint package also requires the derivative of environmental variable z with respect to y at each grid point. The following code calculates these derivatives. For each grid point, the grid points directly above and below are identified (i.e. grid points have same x-coordinates). The difference between z at each of these grid points is calculated, and divided by 500. Where the grid point is at a maximum value of y, we calculate the change in depth between the grid point and the grid point directly below it. The derivative of depth with respect to y is taken to be this difference in depth divided by 250. The analogous holds for when the grid point is at the minimum value of y.

```
# loads grid data frame
load("G:/StrongMeganHumpback/Books/grid.Rdata")

# finds derivative of shortest distance to coast WRT y at each grid point
dzdy.gpts.dist<-numeric(dim(grid)[1])

for (i in 1:dim(grid)[1]) {

    x<-grid$X[i]
    y<-grid$Y[i]

    point<-c(x,y)
    xgrid<-unique(grid[,1])
    ygrid<-unique(grid[,2])

    if (y==max(ygrid)) {

        pt1<-c(x,max(ygrid))
        pt2<-c(x,max(ygrid)-250)
        ptsx<-grid[grid[,1]==x,]
        ptsy<-ptsx[ptsx[,2]==max(ygrid),]
        ptsy1<-ptsx[ptsx[,2]==(max(ygrid)-250),]
        dist.change<-ptsy$SHORTDIST-ptsy1$SHORTDIST
        diff<-dist.change/250
        dzdy.gpts.dist[i]<-diff

    }

    else {

        if (y==min(ygrid)) {

             pt1<-c(x,min(ygrid)+250)
             pt2<-c(x,min(ygrid))
             ptsx<-grid[grid[,1]==x,]
             ptsy<-ptsx[ptsx[,2]==min(ygrid)+250,]
             ptsy1<-ptsx[ptsx[,2]==min(ygrid),]
             dist.change<-ptsy$SHORTDIST-ptsy1$SHORTDIST
             diff<-dist.change/250
             dzdy.gpts.dist[i]<-diff

          }

          else {

             pt1<-c(x,y+250)
             pt2<-c(x,y-250)
             ptsx<-grid[grid[,1]==x,]
             ptsy<-ptsx[ptsx[,2]==y+250,]
             ptsy1<-ptsx[ptsx[,2]==y-250,]
             dist.change<-ptsy$SHORTDIST-ptsy1$SHORTDIST
             diff<-dist.change/500
             dzdy.gpts.dist[i]<-diff}

   }

}

# finds derivative of depth WRT y at each grid point
dzdy.gpts.depth<-numeric(dim(grid)[1])

for (i in 1:dim(grid)[1]) {

    x<-grid$X[i]
    y<-grid$Y[i]

    point<-c(x,y)
    xgrid<-unique(grid[,1])
    ygrid<-unique(grid[,2])

    if (y==max(ygrid)) {
        pt1<-c(x,max(ygrid))
        pt2<-c(x,max(ygrid)-250)
        ptsx<-grid[grid[,1]==x,]
        ptsy<-ptsx[ptsx[,2]==max(ygrid),]
        ptsy1<-ptsx[ptsx[,2]==(max(ygrid)-250),]
        depth.change<-ptsy$DEPTH-ptsy1$DEPTH
        diff<-depth.change/250
        dzdy.gpts.depth[i]<-diff

    }

    else {

        if (y==min(ygrid)) {

            pt1<-c(x,min(ygrid)+250)
            pt2<-c(x,min(ygrid))
            ptsx<-grid[grid[,1]==x,]
            ptsy<-ptsx[ptsx[,2]==min(ygrid)+250,]
            ptsy1<-ptsx[ptsx[,2]==min(ygrid),]
            depth.change<-ptsy$DEPTH-ptsy1$DEPTH
            diff<-depth.change/250
            dzdy.gpts.depth[i]<-diff

        }

        else {

            pt1<-c(x,y+250)
            pt2<-c(x,y-250)
            ptsx<-grid[grid[,1]==x,]
            ptsy<-ptsx[ptsx[,2]==y+250,]
            ptsy1<-ptsx[ptsx[,2]==y-250,]
            depth.change<-ptsy$DEPTH-ptsy1$DEPTH
            diff<-depth.change/500
            dzdy.gpts.depth[i]<-diff}

    }

}

# adds derivatives column to grid data frame and assigns new column names
grid.depth<-cbind(grid[,1:2],grid[,5],grid[,3],dzdy.gpts.depth,grid[,6])
colnames(grid.depth)<-c("X","Y","RDIST","DEPTH","DZDYD","SREGION")
grid.dist<-cbind(grid[,1:2],grid[,5],grid[,4],dzdy.gpts.dist,grid[,6])
colnames(grid.dist)<-c("X","Y","RDIST","SHORTDIST","DZDYS","SREGION")
```

Having calculated the derivatives of both shortest distance to coast and depth with respect to y using the above method, these derivatives were plotted; the resulting plots (shown below) indicated that it might be beneficial to fit a smooth surface to the grid points for each of these variables. From these surfaces, the values of z and the derivatives of z with respect to y can be predicted.

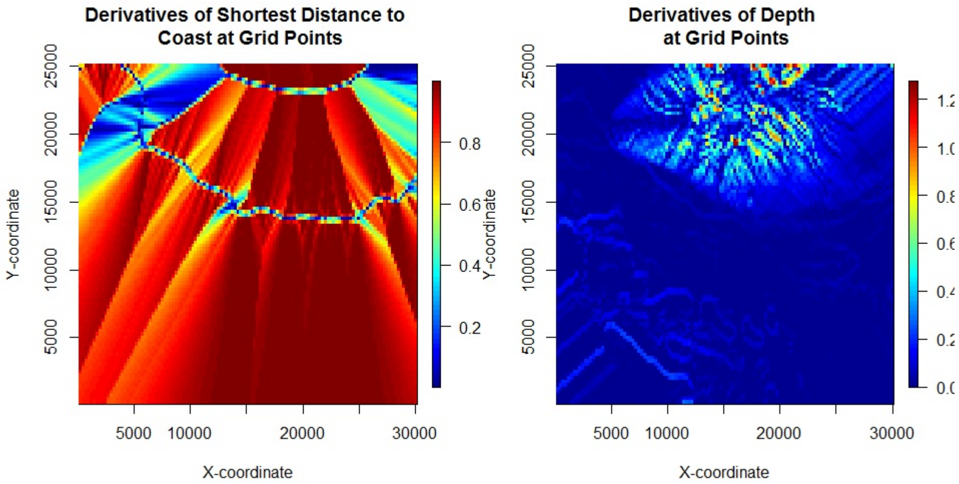


Figure 3?? Derivatives of shortest distance and depth

B.4 Fit Smooth Surface to Shortest Distances at Grid Points

```
rm(list = ls(all.names = TRUE))

# loads data frames of grid points and observations
load("G:/StrongMeganHumpback/Books/grid.dist.Rdata")
load("G:/StrongMeganHumpback/Books/np.dist.Rdata")

xcoords<-sort(grid.dist$X)
ycoords<-sort(grid.dist$Y)

# matrix containing x coords of every point on grid
x.whales<-matrix(xcoords,byrow=T,nrow=120)
# matrix containing y coords of every point on grid
y.whales<-matrix(ycoords,byrow=F,nrow=120)

# sorts grid data frame by x and then y coordinates
grid.dist<-grid.dist[order(grid.dist$X,grid.dist$Y),]

# radial distance from observer to every grid point
rd.whales<-matrix(grid.dist$RDIST,byrow=T,nrow=120)

# vector of unique values in x.whales
xuniq<-sort(unique(grid.dist$X))
# vector of unique values in y.whales
yuniq<-sort(unique(grid.dist$Y))
# Cartesian coordinates of observer
obsxpos<-15000
obsypos<-10000
# max (abs) distance along x axis to any sighting (from observer)
maxx<-max(abs(np.dist$x-15000))
# max offshore distance (i.e. from origin) of any sighting
maxy<-max(np.dist$y)
# max and min dist
maxd<-max(np.dist$z)
mind<-min(np.dist$z)

# loads required library; mcgv allows us to fit a smooth surface
library(mgcv)
```

```
# defines surface; we want to predict DIST from X and Y values of grid point
mod<-gam(SHORTDIST~s(X,Y),data=grid.dist)
# plot the contours of the smooth surface
plot(mod)

# creates 12000 x 1 matrix b containing shortest distances to Maui coastline as
# predicted from smooth surface
tmp.grid<-predict(mod)
pred<-matrix(tmp.grid,byrow=T,ncol=1)

# predicts derivatives (of dist WRT y at each grid point) from smooth surface
delta<-0.01
tmp.deriv<-abs(predict(mod,newdata=list(X=grid.dist$X,Y=grid.dist$Y+delta))-
predict(mod,newdata=list(X=grid.dist$X,Y=grid.dist$Y-delta)))/(2*delta)
# returns 12000 x 1 matrix containing derivatives
zGradwhales<-matrix(tmp.deriv,byrow=T,ncol=1)

# creates 12000 x 1 matrix derivs containing derivatives predicted from smooth surface
derivs<-as.list(zGradwhales)
derivs<-matrix(derivs,byrow=T,ncol=1)

grid.dist<-cbind(grid.dist[,1:3],pred,zGradwhales,grid.dist[,6])
colnames(grid.dist)<-c("X","Y","RDIST","SHORTDIST","DZDYD","SREGION")

# where grid point does not lie in survey region, NA is assigned to SHORTDIST column
grid.dist$SHORTDIST[grid.dist$SREGION==0]<-NA

# creates 120 x 100 matrix containing predicted shortest distance at each gridpoint
pred.dist<-as.list(grid.dist[,4])
z.whales<-matrix(unlist(pred.dist),byrow=T,ncol=100)

# creates 120 x 100 matrix containing predicted derivative at each gridpoint
pred.deriv<-as.list(grid.dist[,5])
zGradwhales<-matrix(unlist(pred.deriv),byrow=T,ncol=100)

# replaces original shortest distance and derivative columns in grid point data
# frame with those calculated from smooth surface
obs.dist<-predict(mod,newdata=list(X=np.dist$x,Y=np.dist$y))
obs.deriv<-abs(predict(mod,newdata=list(X=np.dist$x,Y=np.dist$y+delta))-
predict(mod,newdata=list(X=np.dist$x,Y=np.dist$y-delta)))/(2*delta)

# replaces original shortest distance and derivative columns in observations data
# frame with those calculated from smooth surface
np.dist<-cbind(np.dist[,1:3],obs.dist,obs.deriv,np.dist[,6])
colnames(np.dist)<-c("x","y","r","z","dzdy","obs.period")
```

B.5 Fit Smooth Surface to Depths at Grid Points

```
rm(list = ls(all.names = TRUE))

# loads data frames of grid points and observations
load("G:/StrongMeganHumpback/Books/grid.depth.Rdata")
load("G:/StrongMeganHumpback/Books/np.depth.Rdata")

grid.depth$DEPTH<--grid.depth$DEPTH

xcoords<-sort(grid.depth$X)
ycoords<-sort(grid.depth$Y)
```

```
# matrix containing x coords of every point on grid
x.whales<-matrix(xcoords,byrow=T,nrow=120)
# matrix containing y coords of every point on grid
y.whales<-matrix(ycoords,byrow=F,nrow=120)

# sorts grid data frame by x and then y coordinates
grid.depth<-grid.depth[order(grid.depth$X,grid.depth$Y),]

# radial distance from observer to every grid point
rd.whales<-matrix(grid.depth$RDIST,byrow=T,nrow=120)

# vector of unique values in x.whales
xuniq<-sort(unique(grid.depth$X))
# vector of unique values in y.whales
yuniq<-sort(unique(grid.depth$Y))
# Cartesian coordinates of observer
obsxpos<-15000
obsypos<-10000
# max (abs) distance along x axis to any sighting (from observer)
maxx<-max(abs(np.depth$x-15000))
# max offshore distance (i.e. from origin) of any sighting
maxy<-max(np.depth$y)
# max and min depth
maxd<-max(np.depth$z)
mind<-min(np.depth$z)

# loads required library; mcgv allows us to fit a smooth surface
library(mgcv)
# defines surface; we want to predict DEPTH from X and Y values of grid point
mod<-gam(DEPTH~s(X,Y),data=grid.depth,gamma=100000)
# plot the contours of the smooth surface
plot(mod)

# creates 12000 x 1 matrix b containing depths as predicted from smooth surface
tmp.grid<-predict(mod)
pred<-matrix(tmp.grid,byrow=T,ncol=1)

# predicts derivatives (of depth WRT y at each grid point) from smooth surface
delta<-0.01
tmp.deriv<-abs(predict(mod,newdata=list(X=grid.depth$X,Y=grid.depth$Y+delta))-
predict(mod,newdata=list(X=grid.depth$X,Y=grid.depth$Y-delta)))/(2*delta)
# returns 12000 x 1 matrix containing derivatives
zGradwhales<-matrix(tmp.deriv,byrow=T,ncol=1)

# creates 12000 x 1 matrix derivs containing derivatives predicted from smooth surface
derivs<-as.list(zGradwhales)
derivs<-matrix(derivs,byrow=T,ncol=1)

# replaces original depth and derivative columns in grid point data frame with those
# calculated from smooth surface
grid.depth<-cbind(grid.depth[,1:3],pred,zGradwhales,grid.depth[,6])
colnames(grid.depth)<-c("X","Y","RDIST","DEPTH","DZDYD","SREGION")

# where grid point does not lie in survey region, NA is assigned to DEPTH column
grid.depth$DEPTH[grid.depth$SREGION==0]<-NA

# creates 120 x 100 matrix containing predicted depth at each gridpoint
pred.depth<-as.list(grid.depth[,4])
z.whales<-matrix(unlist(pred.depth),byrow=T,ncol=100)

# creates 120 x 100 matrix containing predicted derivative at each gridpoint
pred.deriv<-as.list(grid.depth[,5])
```

```
zGradwhales<-matrix(unlist(pred.deriv),byrow=T,ncol=100)

# predicts depth and derivative of depth at each observation from the smooth surface
obs.depth<-predict(mod,newdata=list(X=np.depth$x,Y=np.depth$y))
obs.deriv<-abs(predict(mod,newdata=list(X=np.depth$x,Y=np.depth$y+delta))-
predict(mod,newdata=list(X=np.depth$x,Y=np.depth$y-delta)))/(2*delta)

# replaces original depth and derivative columns in observations data frame with those
# calculated from smooth surface
np.depth<-cbind(np.depth[,1:3],obs.depth,obs.deriv,np.depth[,6])
colnames(np.depth)<-c("x","y","r","z","dzdy","obs.period")
```

The plots below show the resulting derivatives of both shortest distance to coast and depth at each grid point, after smooth surfaces were fitted.

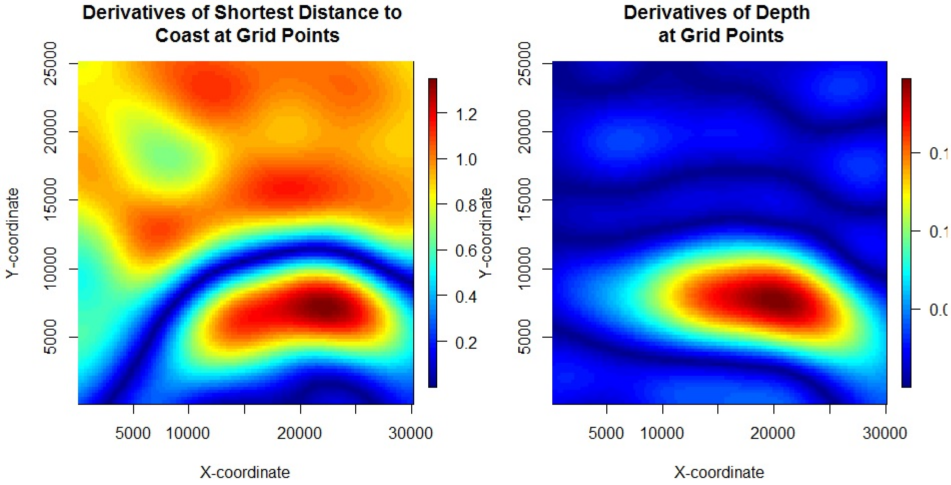


Figure 3?? Derivatives of shortest distance and depth from smooth surface

B.6 `nupoint`; Model Fitting - Shortest Distance to Coast

`nupoint.env.fit` within the `nupoint` library was used to fit a number of density and detection functions, using maximum likelihood methods to estimate the parameters of both functions. For the detection function, a half normal form was considered, and for the density function, six candidate forms were considered; normal, log-normal, beta, uniform, 2-mixture normal and 3-mixture normal. The most parsimonious model was chosen on the basis of the AIC (the model with the smallest AIC was selected). The code below documents how these models were fitted and how model diagnostics were produced.

```
# loads grid data frame and data frame of sightings
load("G:/StrongMeganHumpback/Books/grid.dist.Rdata")
load("G:/StrongMeganHumpback/Books/np.dist.Rdata")

# loads required library
library(nupoint)

# specifies form of density and detection functions as normal and half normal,
# respectively
norm.grad.type <- "NORM"
norm.det.type <- "HNORM"
# list of starting values for parameters; mean and standard error of density
# function, followed by standard deviation of detection function
```

```
#starting values of parameters to be estimated
norm.pars <- c(5000,5000,4000)
# upper and lower bounds of parameters
norm.LB <- c(-2000,1,1)
norm.UB <- c(20000,15000,25000)

# fits distribution model
norm.fit<-nupoint.env.fit(pars=norm.pars, # starting paramater values
                          z=np.dist$z, # shortest distances to sightings
                          rd=np.dist$r, # radial distances to sightings
                          dzdy=np.dist$dzdy, # derivatives at sightings
                          z.mat=z.whales, # shortest distance to grid points
                          dzdy.mat=zGradwhales, # derivatives at grid points
                          rd.mat=rd.whales, # radial distances to grid points
                          minz=mind, # min shortest distance to Maui
                                     # coastline of any observation
                          wx=maxx, # max absolute x distance from observer
                                   # of any sighting
                          wy=maxy, # max offshore distance of any sighting
                          wz=maxd, # max shortest distance to Maui coastline
                                   # of any observation
                          grad.type=norm.grad.type, # density fn type
                          det.type=norm.det.type, # detection fn type
                          lower.b=norm.LB, # specifies lower param bounds
                          upper.b=norm.UB, # specifies upper param bounds
)

# produces model diagnostics
nupoint.env.gof(norm.fit$par,rd.whales,z.whales,mind,maxd,np.dist$z,
norm.grad.type,norm.det.type,intervals=10,plot=T,dzdy.mat=zGradwhales)



# specifies form of density and detection functions as log normal and half
# normal, respectively
lnorm.grad.type <- "LOGNORM"
lnorm.det.type <- "HNORM"
lnorm.pars <- c(5,6,2000)
lnorm.LB <- c(2,1,10) #lower parameter space bound.
lnorm.UB <- c(25,25,25000)

# fits distribution model
lnorm.fit<-nupoint.env.fit(pars=lnorm.pars,
                           z=np.dist$z,
                           rd=np.dist$r,
                           dzdy=np.dist$dzdy,
                           z.mat=z.whales,
                           dzdy.mat=zGradwhales,
                           rd.mat=rd.whales,
                           minz=mind,
                           wx=maxx,
                           wy=maxy,
                           wz=maxd,
                           grad.type=lnorm.grad.type,
                           det.type=lnorm.det.type,
                           lower.b=lnorm.LB,upper.b=lnorm.UB,
)

# produces model diagnostics
nupoint.env.gof(lnorm.fit$par,rd.whales,z.whales,mind,maxd,np.dist$z,
lnorm.grad.type, lnorm.det.type,intervals=10,plot=T,dzdy.mat=zGradwhales)
```

```
# specifies form of density and detection functions as beta and half
# normal, respectively
beta.grad.type <- "BETA"
beta.det.type <- "HNORM"
beta.pars <- c(1.5,2,6000)
beta.LB<- c(1,1,10)
beta.UB <- c(10,40,20000)

# fits distribution model
beta.fit<-nupoint.env.fit(pars=beta.pars,
                          z=np.dist$z,
                          rd=np.dist$r,
                          dzdy=np.dist$dzdy,
                          z.mat=z.whales,
                          dzdy.mat=zGradwhales,
                          rd.mat=rd.whales,
                          minz=mind,
                          wx=maxx,
                          wy=maxy,
                          wz=maxd,
                          grad.type=beta.grad.type,
                          det.type=beta.det.type,
                          lower.b=beta.LB,upper.b=beta.UB,
)

# produces model diagnostics
nupoint.env.gof(beta.fit$par,rd.whales,z.whales,mind,maxd,np.dist$z,
beta.grad.type,beta.det.type,intervals=10,plot=T,dzdy.mat=zGradwhales)

-------------------------------------------------------------------------
Warning messages:
1: In nupoint.env.fit(pars = beta.pars, z = np.dist$z, rd = np.dist$r,  :
  environ.fit.f: convergence error code returned by optim
2: In nupoint.env.fit(pars = beta.pars, z = np.dist$z, rd = np.dist$r,  :
  environ.fit.f: one or more parameter estimates equals starting value.
Check convergence.
-------------------------------------------------------------------------
```

```
# specifies form of density and detection functions as uniform and half
# normal, respectively
unif.grad.type <- "UNIFORM"
unif.det.type <- "HNORM"
unif.pars<-3000 #starting values for parameter vector to be estimated
unif.LB<-500 #lower parameter space bound.
unif.UB<-22000

# fits distribution model
unif.fit<-nupoint.env.fit(pars=unif.pars,
                          z=np.dist$z,
                          rd=np.dist$r,
                          dzdy=np.dist$dzdy,
                          z.mat=z.whales,
                          dzdy.mat=zGradwhales,
                          rd.mat=rd.whales,
                          minz=mind,
                          wx=maxx,
                          wy=maxy,
                          wz=maxd,
                          grad.type=unif.grad.type,
                          det.type=unif.det.type,
                          lower.b=unif.LB,upper.b=unif.UB,
)

# produces model diagnostics
nupoint.env.gof(unif.fit$par,rd.whales,z.whales,mind,maxd,np.dist$z,
unif.grad.type,unif.det.type,intervals=10,plot=T,dzdy.mat=zGradwhales)



det.type<-"HNORM"
sigma.r<-3000 # detection function parameter
grad.type<-"MNORM"
nDist<-2 # number of normal distributions in mixture model

wt<-rep(1/nDist,nDist) # assign weights to each normal distn
mu<-seq(mind,maxd,length=nDist) # means of mixture distns
sigma<-rep(5000,nDist) # standard errors of mixture distns

# starting values of parameters
pars<-as.vector(matrix(c(mu,sigma,wt),ncol=nDist,byrow=TRUE))
pars<-c(pars[-length(pars)],sigma.r)

# lower bounds of parameters
mumin<-rep(-3000,nDist)
sigmamin<-rep(1,nDist)
alphamin<-rep(0,nDist)
sigma.rmin<-1
lower.b<-as.vector(matrix(c(mumin,sigmamin,alphamin),ncol=nDist,byrow=TRUE))
lower.b<-c(lower.b[-length(lower.b)],sigma.rmin)

# upper bounds of parameters
mumax<-rep(5e+4,nDist)
sigmamax<-rep(2e+5,nDist)
alphamax<-rep(1,nDist)
sigma.rmax<-1e+5
upper.b<-as.vector(matrix(c(mumax,sigmamax,alphamax),ncol=nDist,byrow=TRUE))
upper.b<-c(upper.b[-length(upper.b)],sigma.rmax)

# fits distribution model
mn.environ.fit<-nupoint.env.fit(pars=pars,
                                z=np.dist$z,
                                rd=np.dist$r,
                                dzdy=np.dist$dzdy,
                                z.mat=z.whales,
                                dzdy.mat=zGradwhales,
                                rd.mat=rd.whales,
                                minz=mind,
                                wx=maxx,
                                wy=maxy,
                                wz=maxd,
                                grad.type=grad.type,
                                det.type=det.type,
                                n=nDist,
                                lower.b=lower.b,
                                upper.b=upper.b
)

# produces model diagnostics
nupoint.env.gof(mn.environ.fit$par,rd.whales,z.whales,mind,maxd,np.dist$z,
grad.type,det.type,intervals=10,plot=T,dzdy.mat=zGradwhales,n=nDist)
```

B.7 Density Estimate

The following code was used to calculate the density of whale groups per km^2 at any random point in time within the survey period. This was estimated to be 0.0208 whales.

```
# model parameters of fitted model
mpar<-norm.fit$par
mu<-mpar[1]
sigma<-mpar[2]
# calculate estimated density at each grid point
dens<-dnorm(z.whales,mu,sigma)
# find average of these densities
dens.gpt<-sum(na.omit(as.numeric(dens)))/6359
# divide by total number of snapshots to obtain an estimate of average density
# per metre squared per snapshot
dens.pm.ps<-dens.gpt/1581
# average density per km squared per snapshot
dens.pkm.ps<-dens.pm.ps*1000000
a.dens<-dens.pkm.ps
```

B.8 Bootstrapping - Density Estimates

In order to calculate the variance and 95% confidence intervals for the density estimate, non parametric bootstrapping was carried out. This involved sampling 30 days with replacement from the data set, refitting the model given the data within this sample, and obtaining an estimate of average density. This was repeated 2000 times, using the code documented below.

```
# creates vector of length 2000; for each of the bootstrap resamples, we
# will obtain one estimate of density (per km^2 per snapshot); these
# estimates will be placed into this vector
a.dens<-numeric(2000)

for (j in 1:2000) {

    # sample with replacement from the unique days
    days<-unique(np.dist$obs.period)
    samp<-sample(days,30,replace=T)
    np.dist1<-np.dist[which(np.dist$obs.period %in% samp),]

    # for each day in sample, find number of snapshots that day
    # NOTE; snapsperday is a data frame containing two columns; the
    # first contains the index relating to day, and the second contains
    # the number of snapshots on that day
    snapshots<-as.data.frame(snapsperday)
    snaps<-numeric(30)
    for (i in 1:30) {which<-which(snapshots$INDEX==samp[i])
                     snaps[i]<-snapshots[which,2]}

    # total number of snapshots across all sampled days
    totalsnaps<-sum(snaps)

    # max absolute distance in x direction to any sighting from observer
    maxy<-max(np.dist1$y)
    # max offshore distance (in y direction) of any sighting
    maxx<-max(abs(np.dist1$x-15000))
    # max and min dist
    maxd<-max(np.dist1$z)
    mind<-min(np.dist1$z)

    # fit density model to this new sample and use nupoint.env.fit as
    # before to estimate model parameters

    # specifies form of density and detection functions as normal and half normal,
    # respectively
    norm.grad.type <- "NORM"
    norm.det.type <- "HNORM"
    # list of starting values for parameters; mean and standard error of density
    # function, followed by standard deviation of detection function
    #starting values of parameters to be estimated
    norm.pars <- c(5000,5000,4000)
    # upper and lower bounds of parameters
    norm.LB <- c(-2000,1,1)
    norm.UB <- c(20000,15000,25000)

    # fits distribution model
    norm.fit<-nupoint.env.fit(pars=norm.pars, # starting parameter values
                             z=np.dist1$z, # shortest distances to sightings
                             rd=np.dist1$r, # radial distances to sightings
                             dzdy=np.dist1$dzdy, # derivatives at sightings
                             z.mat=z.whales, # shortest distance to grid points
                             dzdy.mat=zGradwhales, # derivatives at grid points
                             rd.mat=rd.whales, # radial distances to grid points
                             minz=mind, # min shortest distance to Maui
                                        # coastline of any observation
                             wx=maxx, # max absolute x distance from observer
                                      # of any sighting
                             wy=maxy, # max offshore distance of any sighting
                             wz=maxd, # max shortest distance to Maui coastline
                                      # of any observation
                             grad.type=norm.grad.type, # density fn type
                             det.type=norm.det.type, # detection fn type
                             lower.b=norm.LB, # specifies lower param bounds
                             upper.b=norm.UB, # specifies upper param bounds
    )

    # parameters of normal model
    p<-norm.fit$par[1:2]
    mu<-p[1]
    sigma<-p[2]

    # creates matrix containing estimate of density at each grid point
    dens<-dnorm(z.whales,mu,sigma)
    # sums across these densities and divides by the total number of grid
    # points in the survey region, namely 6359
    # returns average density at any grid point (i.e. per metre squared)
    dens.gpt<-sum(na.omit(as.numeric(dens)))/length(na.omit(as.numeric(dens)))
    # divides by the total number of snapshots to obtain an estimate of
    # average density per metre squared per snapshot
    dens.pm.ps<-dens.gpt/totalsnaps
    # multiplies by 1,000,000 to get average density per km^2 per snapshot
    dens.pkm.ps<-dens.pm.ps*1000000

    a.dens[j]<-dens.pkm.ps

}
```

B.9 Bootstrapping - Parameter Estimates

The following code uses the nupoint.env.boot function within the nupoint library to estimate the variance of the parameter estimates of both the density and detection functions.

```
# creates list in form required for nupoint.env.boot function
sightwhales<-list(sighting.mat=np.dist,x.mat=x.whales,y.mat=y.whales,rd.mat=rd.whales,
z.mat=z.whales,zGradmat=zGradwhales,x=xuniq,y=yuniq,obsx=obsxpos,obsy=obsypos,wx=maxx,
wy=maxy,wz=maxd,minz=mind)

# returns nboot estimates of density and detection function parameters
boot.res<-nupoint.env.boot(sightings=sightwhales,nboot=200,blockVar='obs.period',
                           initial.pars=c(9159.997,11065.62,4167.013),
                           grad.type='NORM',det.type='HNORM',lower.b=c(-2000,1,1),
                           upper.b=c(10000,10000,10000))

# loads required library; raster allows us to calculate coefficient of variation of
# each parameter estimate
library (raster)
# calculates cv (as a percentage) for each of the parameter estimates
cv1<-cv(boot.res[,1])
cv2<-cv(boot.res[,2])
cv3<-cv(boot.res[,3])
```

C Model Diagnostics - χ^2 Goodness-of-fit Test

```
-------------------------------------------------------
1D Chi-squared Goodness-of-Fit results
-------------------------------------------------------
    bin.min  bin.max     mids expected observed Chisq
1    255.97  1512.29   884.13   480.81      406 11.64
2   1512.29  2768.61  2140.45   767.20      802  1.58
3   2768.61  4024.93  3396.77   651.10      728  9.08
4   4024.93  5281.25  4653.09   510.48      430 12.69
5   5281.25  6537.57  5909.41   349.21      364  0.63
6   6537.57  7793.89  7165.73   228.04      201  3.21
7   7793.89  9050.21  8422.05   116.55      148  8.49
8   9050.21 10306.53  9678.37    54.60       63  1.29
9  10306.53 11562.85 10934.69    23.76       40 11.10
10 11562.85 12819.17 12191.01     9.24        8  0.17
-------------------------------------------------------
Chi-squ. statistic   =  59.86865
Number of parameters =  3
Chi-squ. df          =  6
Chi-squ. GoF p-value =  4.786216e-11
-------------------------------------------------------
```