

Logistic modeling of point process: a Bayesian approach for presence-only data

Tese de doutorado

Aluno: Guido Alberti Moreira

Orientador: Dani Gamerman



Universidade Federal do Rio de Janeiro
Instituto de Matemática
Departamento de Métodos Estatísticos

February 9, 2020

Abstract

Species distribution models (SDMs) are extremely useful for determining preferences and habitats for different species. Appropriate estimation of species distribution depends on the adequate random sampling scheme which is not always available. Instead, data is frequently composed of georeferenced locations where the species has been observed, which is commonly referred to as presence-only (PO) data.

The statistical modelling of PO type data through Inhomogeneous Poisson Processes (IPP) was proposed by Fithian and Hastie (2013). As has already been noted (Fithian et al, 2015), PO type data presents bias in its sampling pattern, which must be addressed. A natural way to model this bias under IPP is through thinning of the process, which is easily performed using pertinent covariates.

A different model for the intensity is proposed using a logistic link function. It maintains the already established flexibility, while adding extra flexibility in the choice of covariates. Therefore, it is possible to have correlated and even the same covariates in the intensity and thinning components of the model. This is shown through simulated results. Additionally, it provides computational advantages for handling integrals that appear in the likelihood without resorting to approximations.

Finally, all the inference is based on the Bayes paradigm, which provides a way to consider the uncertainty inherent to the unknown components in the analysis.

Keywords: Point Process; Presence-only; Spatial Statistics; Bayesian Analysis; Species Distribution Model.

Contents

1	Introduction	4
1.1	Presence-only data modeling	4
1.2	Theoretical review	6
1.2.1	History	6
1.2.2	Point Process	6
1.2.3	Poisson Process	9
2	Proposal description and inference	19
2.1	Literature context review	19
2.1.1	Preferential sampling in spatial data	19
2.1.2	Presence-only data methods	19
2.2	Proposed model	20
2.3	Inference	23
2.3.1	Bayesian inference	23
2.3.2	Markov Chain Monte Carlo	24
2.3.3	Prior and Full conditionals	26
2.4	Forecasting and accuracy	29
2.4.1	Forecasting	30
2.4.2	Accuracy	30
2.5	Model comparison	31
2.5.1	Proposal vs. existing models	31
2.5.2	Using presence-only model to predict presence-absence data	31
2.5.3	ROC curve	32
2.5.4	AUC	34
3	Data analysis	36
3.1	Simulated data	36
3.1.1	Estimation under correct specification of the model	36
3.1.2	Estimation under misspecification of the model	45
3.2	Real data	45
3.2.1	Eucalyptus sparsifolia	46
3.2.2	Anadenanthera colubrina	53
4	Model extensions	57
4.1	Multiple species modeling	58
4.2	Spatial interaction beyond the covariates	60
5	Conclusion	61

A	Codes used for examples	63
A.1	Generating figures in the introduction	63
A.2	Generating ROC and AUC examples	67
B	Code used for the data	72
B.1	Generating simulated data sets	72
B.2	MCMC for the simulated data	75
	Bibliography	90

Chapter 1

Introduction

This chapter provides the motivation for the proposal of the thesis in Section 1.1 and a review of the statistical and probabilistic theory used in Section 1.2.

1.1. PRESENCE-ONLY DATA MODELING

It is a common practice in analytical biology to study and predict the distribution of the occurrence of biological species in specified regions, using so-called Species Distribution Models (SDMs). It has application in conservation and reserve planning, ecology, evolution, epidemiology, invasive-species management and other fields (Phillips et al., 2006). It is commonly assumed that the occurrence of different species might be influenced by various environmental characteristics inherent to each location, which may either facilitate or discourage its proliferation such as soil pH, ambient temperature or some pertinent chemical component. This type of modeling is particularly useful in conservation biology: to save a threatened species, one first needs to know where it prefers to live, and what its requirements are for survival, i.e., its ecological niche (Phillips et al., 2004).

Ideally, the data for estimating SDMs is obtained through randomized sampling of the region, which allows the analysis using various techniques such as logistic regression of the covariates on occurrence. However, it is not uncommon that the available data is not randomized, but instead consists of georeferenced occurrence locations, that is, a set of geographic coordinates where the species has been observed. Such data is referred to as presence-only (PO) and is usually obtained through opportunistic surveys, meaning that it can result from reports of citizen science programs when people come across a specimen. Another common source of PO data is herbariums where a biologist collects specimen for study, yet goes to and records locations where they know the species exists. Despite the PO data's less than ideal nature, it has the advantage that there is a massive increase in the collection of these data on a growing number of species (Giraud et al. (2016)).

The aim of this work is to model PO data using available environmental covariates which will be called intensity variables for their influence on the intensity function of the species occurrence. Furthermore, it is assumed that the observed occurrences are not selected randomly in space, as is the case in Royle et al. (2012). Instead, observability variables are considered which influence the probability that a species will be observed in a location given that it occurred there, such as proximity to roads or urban centers. In other words, survey bias is assumed to be explained

by the observability variables.

The use of covariates to explain survey bias has already been proposed in works such as Fithian and Hastie (2013), Giraud et al. (2016), Dorazio (2014) or Fithian et al. (2015). Some of those works have suggested the modeling of observability as is seen here. In contrast the intensity in this work is treated as in Gonçalves and Gamerman (2018). The modeling is done through the introduction of a parameter that represents the supreme of the intensity in the region, and a latent process to allow for exact Bayesian computation, which is also a step away from existing likelihood based methods.

The computation of the likelihood function of a Poisson Process depends on the evaluation of an integral of the intensity function over the studied region. However such an integral is analytically unsolvable in most problems. Current methods that require the computation of the likelihood function commonly use a numerical approximation of the aforementioned integral, usually through quadrature points (Fithian and Hastie, 2013). The work of Renner et al. (2015) propose a method to choose quadrature points so that the computation of the integral in the likelihood function is adequate and not too computationally demanding. However, such method is still approximate and prone to error. The approach taken in this thesis is based on Gonçalves and Gamerman (2018) and requires no approximation.

Despite the variety of available solutions to PO data modeling, each has a different problem inherent to it. Most of the solutions, in fact, fail to recognize a critical aspect of data driven analysis, which is the sampling of the full domain of results possibilities.

That means that the sampled locations need to be compared with ones where the species has not been observed. Since the dataset is comprised exclusively of positive observations (presences), then the negative observations (absences) must not be assumed to be known or sampled. A basic notion with regard to presence-only data is that it is so abundant that the required amount of information should be available, despite the less than ideal sampling scheme. While this can be true, the sampling scheme must not be ignored.

The pseudo-absence logistic model (Pearce and Boyce, 2006; Elith and Leathwick, 2007) selects random locations in the region and associates absences to them. This is dangerous, as those absences are not confirmed. One could guess or “hope” that a large portion of those random locations would predominantly have no species presence, but there is no guarantee to that, especially since there is no indication of what the species prevalence might be.

The MaxLike solution (Royle et al., 2012), though clever, makes an assumption that might not be true. The solution treats the presence locations as a uniform random sample of all presence locations. The observability discussion present in the works mentioned above and in Phillips et al. (2009), however, reaches the conclusion that the locations are not uniform. In other words, there are locations where the presence is more noticeable than others. In addition, Fithian et al. (2015) provide a simulation study where the MaxLike solution gravely mis-estimates the species prevalence, which is supposed to be one of the promises of the method. Their study takes advantage of the fact that MaxLike requires the relationship between the presence locations and the covariates to be logit-linear and a small deviance from this relationship misleads the method.

The maximum likelihood estimator of the unthinned Inhomogeneous Poisson Process model (Warton and Shepherd, 2010) has been proven in Fithian and Hastie (2013) to be mathematically

equivalent to the MaxEnt solution, so they suffer from the same problem. To say that the PO sample follows an unthinned Poisson Process is the same as saying that the sample includes all the specimen in the region. Although that might yield an adequate estimation of effect parameters, there is nothing in the model to accommodate the fact that the species was sampled according to an imperfect sampling scheme. That can distort the prediction of unobserved presences.

The thinned Poisson Process (Fithian and Hastie, 2013; Fithian et al., 2015; Dorazio, 2014) accounts for the sampling scheme, and is what this work is based on. However, a few problems remain. Firstly, there is the untractable integral in the likelihood function, as mentioned above. Secondly, there is an identifiability issue with the usual parametrization of the model. Both issues are treated in the proposed model of this thesis.

1.2. THEORETICAL REVIEW

The thesis uses a parametrization of the bi-dimensional Poisson Process, which is a model for Point Processes, also called Spatial Point Pattern Processes. The Poisson Process is worked in detail in Kingman (1993), Cressie (1993), Møller and Waagepetersen (2005) and Diggle (2013). In this thesis, the history of Point and Poisson Processes is presented in Section 1.2.1, the theory for Point Processes is introduced in Section 1.2.2 and for Poisson Processes in Section 1.2.3.

1.2.1. History

1.2.2. Point Process

Point Processes are probabilistic objects used to model random occurrence quantity and locations of certain events in a particular bounded region. Common examples include the location of a certain type of tree. Questions arise such as ‘is there a biological significance to the clustering of trees?’ or ‘Do large trees interact with small trees?’. Point Processes are meant to address these types of questions and they are described below.

A Point Process (PoP) X is a stochastic process whose support is a particular type of countable subsets of Space \mathcal{S} where a Borel σ -algebra can be equipped. Henceforth throughout this thesis, the space $\mathcal{S} = \mathbb{R}^2$ is considered, thus the process is called bi-dimensional. The particular type of subsets is called *locally finite*, meaning that the studied subset has finite cardinality in any bounded region of \mathbb{R}^2 . Since PoP’s are used to model data in a bounded region \mathcal{D} , the data imply a finite number of points in \mathcal{D} .

Note that a realization x of the studied process is any finite subset of \mathcal{D} , which implies that it is a set of points. Also, the randomness of X can be separated in two features, that is, the number of points N_X , meaning the cardinality of the locally finite subset, and the position of the points. It is worthwhile to mention that if the number of points were not random and fixed at n_x , then only the position of the points would be unknown and it would be sufficient to describe the joint distribution of the n_x points over their common support \mathcal{D} .

As a consequence, describing the distribution of a PoP is equivalent to identifying the joint distribution of the number of points that fall in all Borelian $B_i \subset \mathcal{D}, i = 1, \dots, m, \forall m \in \mathbb{N}$ (Møller and Waagepetersen, 2005, Appendix B.2). The notation $N_X(B)$ for any Borelian

$B \subset \mathcal{D}$ will be used for the number of points of the process X that fall in B . Note that the quantity $N_X(B) \in \mathbb{N}$ is a random variable induced by the process X . The special case $N_X(\mathcal{D})$ will be exchanged by N_X for simplicity. The realization of these random variables will be denoted by the lower case $n_x(B)$ or simply n_x for $B \equiv \mathcal{D}$.

The equivalence between the distribution of a PoP and the joint distribution of $N_X(B), \forall B \subset \mathcal{D}$ comes from the fact that knowing the count of points of the process X in all Borelian B implies (and is implied by) knowing the point locations. Therefore, a Point Process can also be defined by a set of indexed random count variables. Furthermore, Møller and Waagepetersen (2005) show that it is enough to determine the probability of void events, that is, to determine $P(N_X(B) = 0)$ for all Borelian B . Throughout this thesis, the Process realization x will still represent the finite set of locations where the points occurred. This definition is detailed below.

Let $(\Omega, \mathcal{A}, \mathcal{P})$ be a probability space. Also, let Φ be a collection of locally finite counting measures on \mathcal{D} . Define \mathcal{N} on Φ as the smallest σ -algebra generated by sets of the form $\{\phi \in \Phi : \phi(B) = n\}$ for all Borelian $B \subset \mathcal{D}$ and $n \in \mathbb{N}$. Then a Point Process X on \mathcal{D} is a measurable mapping of (Ω, \mathcal{A}) on (Φ, \mathcal{N}) . The process X also induces a probability measure $\mathcal{P}_X(M) = \mathcal{P}(N \in M) \equiv \mathcal{P}(\omega : N(\omega) \in M), M \in \mathcal{N}$. At the same time, N_X is a measurable mapping of (Φ, \mathcal{N}) for $B \equiv \mathcal{D}$ to $(\mathbb{N}, \mathcal{A}_{\mathbb{N}})$, where $\mathcal{A}_{\mathbb{N}}$ is an appropriate σ -algebra on the natural numbers. Also, N_X induces a probability measure $\mathcal{P}_{N_X}(L), L \in \mathcal{A}_{\mathbb{N}}$.

Two Point Processes X and Y are defined as *identically distributed* when $\mathcal{P}_X(M) = \mathcal{P}_Y(M)$, for all $M \in \mathcal{N}$. A Point Process is said to be *simple* if $\phi(s) \in \{0, 1\}$, for all $s \in \mathcal{D}$ and almost all $\phi \in \Phi$.

More on probability and measures can be read in Billingsley (1995).

Special cases and examples

The examples shown here have been generated using the procedures and codes detailed in Appendix Section A.1.

A few special cases of Point Processes can be named. Those are the completely random, clustered, regular and marked Point Processes. The completely random process presents no probabilistic pattern to the points, that is, the generating process provides marginal independence between the points. Since there is no interaction between the points, it is possible that some form of pattern arises in the process realization, which may mislead the analyst into thinking the process is *not* completely random.

If there is no prior indication to a particular pattern to the point occurrences, the completely random process is a good benchmark to compare to the data. In other words, the question whether the points generation process is completely random is natural. The reader can find discussions with tests and statistics for this verification in Cressie (1993), Møller and Waagepetersen (2005) and Diggle (2013) if they further want to read on this subject. However, this is not the theme of this thesis and will not be further mentioned.

The clustered process presents attraction between the points, meaning that points tend to occur close to each other. Cluster processes are useful for analyzing clustered bacteria colonies

or specific types of species that clutter for mutual benefit, maybe even symbiotic.

The regular process, on the other hand, provides repulsion between the points, which tends to make the points not occur close to each other, which causes a seemingly almost regular pattern. These processes can be used to model competing species or any type of events which tend to occur at some distance from each other.

Figure 1.1 exemplifies these patterns.

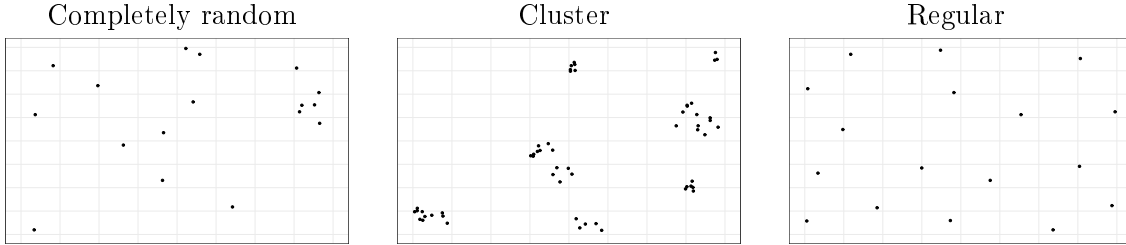


Figure 1.1: Examples of Point Processes for different patterns

The marked PoP is a special case where each point event has some associated real value called mark. This type of process can be used to model simple marks, such as height of trees, but can also be used to broaden the usefulness of the PoP. If points occur at a specific point in time, for example, the mark can be the time of occurrence, which implies that spatio-temporal data can be modeled with such a process. Also, a PoP which is not simple, i.e., that can have more than one point per location, can be modeled as a simple marked PoP where the amount of points that happen in any given location is the mark. Note that the marked PoP can also be modeled as an unmarked PoP whose occurrence space is $S' = S \times \mathbb{R}$. Figure 1.2 exemplifies a marked process.

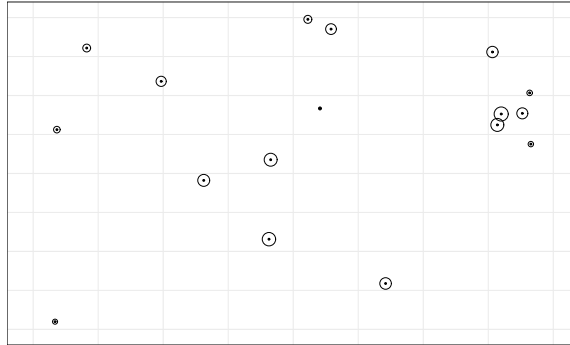


Figure 1.2: Example of a marked Point Process - disk sizes represent the marks

Despite the marked PoP significance, it will not be used in this thesis.

Operations on Point Processes

It is possible to perform operations on Point Processes. Two operations are described here, as they are important throughout the thesis.

Thinning

This operation consists of the elimination or thinning of the points. A thinned Point Process, therefore, is another Point Process with at most the same amount of points as the original Process and at least zero points. A very common thinning operation involves a probability field $p : \mathcal{D} \rightarrow [0, 1]$. Then given a PoP X , the thinned PoP is constituted by keeping each point $s \in X$ with probability $p(s)$ and thinning it with complementary probability.

If the thinning operation of the PoP X with probability field $p(\cdot)$ is independent of the process X and within itself, then it will be denoted as $\tau(X, p(\cdot))$. Figure 1.3 exemplifies the thinning of the realization of a PoP.

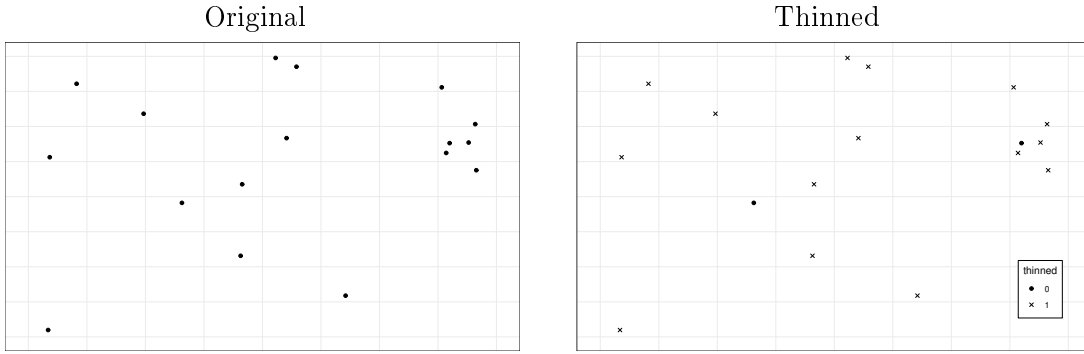


Figure 1.3: Example of thinning of a PoP with only 2 retained points out of the original 17

Superposition

The superposition consists of the union of the points of two PoPes into one, meaning that the resulting process has an amount of points equal to the sum of the amounts of the two original processes. In other words, if Z is the result of the superposition of processes X and Y , then $N_Z(B) = N_X(B) + N_Y(B)$ for all Borelian sets $B \in \mathcal{D}$.

If the superposition operation is performed on two independent processes X and Y , then it will be denoted as $X \cup Y$. Figure 1.4 exemplifies the superposition of the realizations of two independent PoPes.

Note that the two original processes can be viewed as the retained and thinned points of the superposed process. In this sense, superposing can be viewed as the inverse operation of thinning.

1.2.3. Poisson Process

A Poisson Process (PP) is a special case of a Point Process presented in Section 1.2.2. As noted, all that is required for the distribution of a PoP X is the definition of the joint distribution of $N_X(B)$ for all Borelian $B \subset \mathcal{D}$. For this purpose, a Poisson Process is parameterized by a positive functional $\lambda : \mathcal{D} \rightarrow \mathbb{R}^+$, such that $\int_{\mathcal{D}} \lambda(s) ds < \infty$. The PoP X is Poisson if

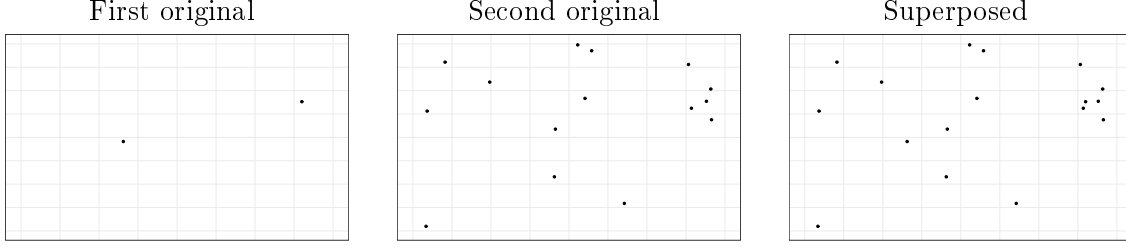


Figure 1.4: Example of superposing two processes

- $N_X(B) \sim \text{Poisson}(\int_B \lambda(s)ds)$ for all Borelian $B \subset \mathcal{D}$
- $N_X(B)$ and $N_X(B')$ are independent for all Borelian $B \cap B' = \emptyset$, $B, B' \subset \mathcal{D}$

The Poisson Process is called homogeneous (HPP) if $\lambda(s) = \lambda, \forall s \in \mathcal{D}$ and inhomogeneous (IPP) otherwise. Note that in the HPP $\int_B \lambda ds = \lambda|B|$.

The functional $\lambda(s)$ represents the point-wise increment to the occurrence rate per unit area at point s .

Some results

All results presented here are more formally shown in Møller and Waagepetersen (2005).

Equivalence of HPP and completely random Point Process

The HPP is equivalent to the completely random (simple) process described in Section 1.2.2. Although an informal proof is presented here, (Kingman, 1993, Chapter 8) provides a formal proof. First, that the HPP implies a completely random PoP is shown.

To characterize a completely random process, only two conditions are required. The first is that the point counts in disjoint Borelian sets are independent, which is automatic from the definition of the Poisson Process. The second is that, given two very small sets B and B' where the area of B' is γ times the area of B , denoted as $|B'| = \gamma|B|$, then the probability of having a point in B' should be γ times the probability of having a point in B . Very small sets are required so that the probability of having more than one point in one of them is negligible.

Thus, note that, if X is an HPP where $\lambda(s) = \lambda, \forall s \in \mathcal{D}$, then

$$\begin{aligned}
 P(N_X(B) = n) &= \frac{e^{-\lambda|B|}(\lambda|B|)^n}{n!} \\
 P(N_X(B') = n) &= \frac{e^{-\lambda|B'|}(\lambda|B'|)^n}{n!} = \frac{e^{-\lambda\gamma|B|}(\lambda\gamma|B|)^n}{n!} \\
 \frac{P(N_X(B') = n)}{P(N_X(B) = n)} &= e^{-\lambda|B|}\gamma^n
 \end{aligned} \tag{1.1}$$

Since every case for which n is larger than one is negligible, we have that

$$\lim_{|B| \rightarrow 0} \frac{P(N_X(B') = n)}{P(N_X(B) = n)} = \gamma, \tag{1.2}$$

as was required. \square

Now, it is shown that a completely random process implies an HPP, taken from (Kingman, 1993, Chapter 1). For simplicity in this proof, consider $N_X(B) \equiv N(B)$. It must be shown that if X is a completely random process then N_X has a Poisson distribution. Consider the set B_t , a circle with radius t inside \mathcal{D} . It is enough to show that B_t has a Poisson distribution, since any other Borelian is generated by a countable set of disjoint circles **Proof or reference missing** and the sum of Poisson random variables is also Poisson, which can be easily verified by its moments generating function. Now define

$$\begin{aligned} p_n(t) &= P(N(B_t) = n) \\ q_n(t) &= P(N(B_t) \leq n). \end{aligned} \tag{1.3}$$

Note that $N(B_t)$ is monotonically non-decreasing in t , which means that $q_n(t)$ monotonically decreasing in t . Also note that both $p_n(t)$ and $q_n(t)$ are differentiable almost everywhere.

The random variable $N(B_t)$ jumps from n to $n + 1$ when its enlarging boundaries crosses one of the random points. The probability that this occurs between t and $t + h$ is the probability that there is a point in the ring between B_t and B_{t+h} , which can be assumed small when h is small. Just as above, consider the probability that more than one point in this ring is negligible. The probability of a point in the aforementioned ring can be defined as $\mu(t + h) - \mu(t)$, where $\mu(t) = E[N(B_t)]$. As is assumed, this number is independent of $N(B_t)$, then

$$q_n(t) - q_n(t + h) = p_n(t)[\mu(t + h) - \mu(t)], \tag{1.4}$$

since the difference in the left-hand side of equation (1.4) is only different from zero for the cases where $N(B_t) = n$ and $N(B_{t+h}) - N(B_t) = 1$ and the right-hand side is the joint probability that $N(B_t) = n$ and that $N(B_{t+h}) - N(B_t) = 1$, multiplied due to the independence. By making $h \rightarrow 0$ in equation (1.4),

$$-\frac{dq_n}{dt} = p_n \frac{d\mu}{dt}. \tag{1.5}$$

For the case $n = 0$, note that $p_0 = q_0$, so

$$-\frac{dp_0}{dt} = p_0 \frac{d\mu}{dt}, \tag{1.6}$$

which is easily solvable to find

$$\frac{d}{dt}(\log p_0 + \mu) = 0. \tag{1.7}$$

Since $p_0(0) = 1$ and $\mu(0) = 0$, it is possible to write

$$\log p_0 + \mu = 0, \tag{1.8}$$

which means that

$$p_0(t) = e^{-\mu(t)}. \tag{1.9}$$

Now for the case $n \geq 1$, note that $p_n = q_n - q_{n-1}$, and write

$$\frac{dq_n}{dt} = \frac{dq_{n-1}}{dt} + \frac{dp_n}{dt}. \quad (1.10)$$

Therefore, substituting in equation (1.5),

$$\begin{aligned} -\frac{dq_{n-1}}{dt} - \frac{dp_n}{dt} &= p_n \frac{d\mu}{dt} \\ -\frac{dq_{n-1}}{dt} - p_n \frac{d\mu}{dt} &= \frac{dp_n}{dt}. \end{aligned} \quad (1.11)$$

Using equation (1.5) on $\frac{dq_{n-1}}{dt}$ for $n-1$ yields

$$\frac{dp_n}{dt} = (p_{n-1} - p_n) \frac{d\mu}{dt}. \quad (1.12)$$

This can be rewritten as

$$\frac{d}{dt}(p_n e^\mu) = p_{n-1} e^\mu \frac{d\mu}{dt}. \quad (1.13)$$

Since $p_n(0) = 0$ for $n \geq 1$, it follows from the calculus fundamental theorem that

$$p_n(t) = e^{-\mu(t)} \int_0^t p_{n-1}(s) e^{\mu(s)} \frac{d\mu}{ds} ds. \quad (1.14)$$

The conclusion is that $N(B_t)$ has a Poisson distribution with mean $\mu(t)$, which is achieved by induction, starting by the number zero, at equation (1.9). For the induction step, assume that $p_{n-1}(t) = e^{-\mu(t)} \frac{\mu(t)^{n-1}}{(n-1)!}$, then

$$\begin{aligned} p_n(t) &= e^{-\mu(t)} \int_0^t e^{-\mu(s)} \frac{\mu(s)^{n-1}}{(n-1)!} e^{\mu(s)} \frac{d\mu}{ds} ds = \\ &= \frac{e^{-\mu(t)}}{(n-1)!} \int_0^t \mu(s)^{n-1} \frac{d\mu}{ds} ds = \\ &= \frac{e^{-\mu(t)}}{(n-1)!} \frac{\mu(t)^n}{n} = e^{-\mu(t)} \frac{\mu(t)^n}{n!} \end{aligned} \quad (1.15)$$

□

Closeness of the Poisson Process under independent thinning and superposition

Poisson Processes are closed with respect to these operations. To show the superposition result, let $X \sim PP(\lambda_X(\cdot))$ and $Y \sim PP(\lambda_Y(\cdot))$. Then, for $Z = X \cup Y$, note that it is sufficient to show that

- $N_Z(B) \sim \text{Poisson}(\cdot)$ for all Borelian $B \subset \mathcal{D}$
- $N_Z(B)$ and $N_Z(B')$ are independent for all Borelian $B \cap B' = \emptyset$, $B, B' \subset \mathcal{D}$

It is straightforward to see that $N_Z(B) = N_X(B) + N_Y(B)$ for all B . Then for $B \cap B' = \emptyset$, $N_Z(B) = N_X(B) + N_Y(B)$ and $N_Z(B') = N_X(B') + N_Y(B')$. But since X and Y are PPs and are independent, then $N_X(B)$ is independent of $N_X(B')$ and of $N_Y(B')$, as is $N_Y(B)$. Therefore, the independence holds.

For the other item, $N_X(B) + N_Y(B)$ is the sum of two independent Poisson random variables, which is also Poisson as was mentioned in Section 1.2.3. Furthermore, $N_Z(B) \sim \text{Poisson}(\int_B \lambda_X(s) + \lambda_Y(s)ds)$, which means that $Z \sim PP(\lambda_X(\cdot) + \lambda_Y(\cdot))$. \square

To show the thinning result, it is noted that the distribution of a PoP is completely characterized by the probabilities of void events, as was noted in Section 1.2.2. The probability of a void event of a PP X with intensity function $\lambda(\cdot)$ is the probability function of the Poisson distribution applied to the value zero, as seen below

$$P(N_X(B) = 0) = \frac{e^{-\int_B \lambda(s)ds} \left(\int_B \lambda(s)ds\right)^0}{0!} = e^{-\int_B \lambda(s)ds}. \quad (1.16)$$

Now, let $Y = \tau(X, p(\cdot))$, which means that for the fixed realization x ,

$$P(N_Y(B) = 0 | x, n_x(B) = n) = \prod_{s \in (x \cap B)} (1 - p(s)). \quad (1.17)$$

Note that the joined events x and $n_x(B) = n$ are redundant since x implies $n_x(B)$ for all B , but they are important to put in evidence, since the locations of x may not be fixed when n_x is, which means that $1 - p(s)$ needs to be integrated (averaged) in B when the locations are not fixed. So, for the marginal probability of void events for Y ,

$$\begin{aligned} P(N_Y(B) = 0) &= \sum_{n=0}^{\infty} P(N_Y(B) = 0, N_X(B) = n) = \\ &= \sum_{n=0}^{\infty} P(N_Y(B) = 0 | n_x(B) = n) P(N_X(B) = n) = \\ &= \sum_{n=0}^{\infty} \prod_{\substack{s \\ n_x = n}} (1 - p(s)) \frac{e^{-\int_B \lambda(s)ds} \left(\int_B \lambda(s)ds\right)^n}{n!} = \\ &= e^{-\int_B \lambda(s)ds} \sum_{n=0}^{\infty} \frac{\left(\int_B (1 - p(s))\lambda(s)ds\right)^n}{n!} = \\ &= e^{-\int_B \lambda(s)ds} e^{\int_B \lambda(s)(1-p(s))ds} = \\ &= e^{-\int_B \lambda(s)p(s)ds} \end{aligned} \quad (1.18)$$

The transition from the third to the fourth row happens due to the locations s in the product being undefined, and so they become defined in the integral. This shows that the independent thinning of a PP with intensity $\lambda(\cdot)$ and probability field $p(\cdot)$ is marginally a PP with intensity $\lambda(\cdot)p(\cdot)$. \square

Conditional distribution of the points locations given the quantity of points

In a bi-variate Poisson Process with intensity function $\lambda(\cdot)$ and known quantity of points n_x , the points locations is an independent and identically distributed random sample of a continuous distribution with support in \mathcal{D} and probability density function (p.d.f.)

$$f(s) = \frac{\lambda(s)}{\int_{\mathcal{D}} \lambda(t)dt} \mathbb{1}_{\mathcal{D}}(s), \quad (1.19)$$

where $\mathbb{1}_A(x)$ represents the indicator function which takes the value 1 if $x \in A$ and 0 otherwise. This means that the probability distribution of the location for each point has a R-N derivative with respect to the Lebesgue measure as shown in equation (1.19).

To prove this, it is enough to show that $f(s) \propto \lambda(s)$ in \mathcal{D} , since the normalization constant is only there to ensure that $f(s)$ is indeed a well defined probability density function, which must integrate to 1 in its support. Note that $\int_{\mathcal{D}} \lambda(s) < \infty$ as per its definition in Section 1.2.3.

Note that the p.d.f. of the points locations, represented by the random variable S , can be found by

$$f(s) = \lim_{h \rightarrow 0} \frac{P(S \in B(s, h))}{h}, \quad (1.20)$$

where $B(s, h)$ represents an open ball centered on s with radius h . Now note that $f(s)$ models the location of each one point and therefore, for small enough h , the value of $P(S \in B(s, h))$ must be proportional to the probability that there is 1 point of the PP in $B(s, h)$. For that, note that $\int_{B(s, h)} \lambda(t) dt \rightarrow 0$ when h goes to zero. For small enough h , consider that the probability that the Poisson Process X has more than one point in $B(s, h)$ is negligible. Then the probability that it has one point is

$$P(N_X(B(s, h)) = 1) = e^{-\int_{B(s, h)} \lambda(s) ds} \int_{B(s, h)} \lambda(s) ds. \quad (1.21)$$

Thus, the result follows

$$\begin{aligned} f(s) &\propto \lim_{h \rightarrow 0} \frac{P(N_X(B(s, h)) = 1)}{h} = \\ &= \lim_{h \rightarrow 0} \frac{e^{-\int_{B(s, h)} \lambda(t) dt} \int_{B(s, h)} \lambda(t) dt}{h} = \\ &= \lim_{h \rightarrow 0} \frac{\int_{B(s, h)} \lambda(t) dt}{h} = \\ &= \frac{d}{dt} \left(\int \lambda(t) dt \right) \Big|_s = \lambda(s) \end{aligned} \quad (1.22)$$

□

Density function of a Poisson Process

A density probability function is defined as a Radon-Nikodym (R-N) derivative of the probability distribution with respect to some dominating measure. The probability measure needs to be absolutely continuous with respect to the dominating measure. The discussion around this matter in Gonçalves and Gamerman (2018) discuss the choice of the dominating measure for the construction of the likelihood function of the model. It can also be applied here using the last result of Section 1.2.3.

The most adequate dominating measure to use is the product measure of a counting measure with the Lebesgue measure in \mathbb{R}^2 . As such the density function for the PP X with N_X points and intensity $\lambda(\cdot)$ is constructed using the definition of conditional probability, that is,

$$P(N_X = n) = \frac{e^{-\int_{\mathcal{D}} \lambda(s) ds} \left(\int_{\mathcal{D}} \lambda(s) ds\right)^n}{n!} \quad (1.23)$$

$$f(\{s_1, \dots, s_n\} \mid n_x = n) = \prod_{i=1}^n \frac{\lambda(s_i)}{\int_{\mathcal{D}} \lambda(s) ds}.$$

The first equation in (1.23) comes from the definition of the Poisson Process using the Borelian $B \equiv \mathcal{D}$. The second equation stems from equation (1.19) and the fact that the points locations form an independent and identically distributed random sample of a distribution with such a p.d.f. The indicator function has been omitted but should be implied. The joint density, then, should represent the density for the PP, that is,

$$\begin{aligned} f(n, \{s_1, \dots, s_n\}) &= P(N_X = n) f(\{s_1, \dots, s_n\} \mid n_x = n) \\ f(n, \{s_1, \dots, s_n\}) &= \frac{e^{-\int_{\mathcal{D}} \lambda(s) ds} \left(\int_{\mathcal{D}} \lambda(s) ds\right)^n}{n!} \prod_{i=1}^n \frac{\lambda(s_i)}{\int_{\mathcal{D}} \lambda(s) ds} = \\ &= \frac{e^{-\int_{\mathcal{D}} \lambda(s) ds}}{n!} \prod_{i=1}^n \lambda(s_i). \end{aligned} \quad (1.24)$$

This density can also be written in terms of the process realization x , that is

$$f(x) = \frac{e^{-\int_{\mathcal{D}} \lambda(s) ds}}{n_x!} \prod_{s \in x} \lambda(s). \quad (1.25)$$

The derivation in equation (1.24) is also found more formally in (Cressie, 1993, Section 8.5.1 - equation 8.5.4), albeit adding the trivial case for which $n = 0$, i.e.,

$$f(0, \emptyset) = e^{-\int_{\mathcal{D}} \lambda(s) ds}. \quad (1.26)$$

It is possible to find in the literature (Streit, 2010) the format for the density

$$f(x) = e^{-\int_{\mathcal{D}} \lambda(s) ds} \prod_{s \in x} \lambda(s), \quad (1.27)$$

for which the justification for the canceling of $n_x!$ is that the second equation in (1.23) must be multiplied by $n!$ to account for all the permutations of $\{s_1, \dots, s_n\}$, thus achieving a density for an unordered set of points. This argument is unsound since the joint p.d.f. of an n sized independent and identically distributed sample is not the n -fold multiplication of the uni-variate p.d.f.s multiplied by $n!$.

The development below is displayed to show a situation where the PP density cannot be as in equation (1.27) and is meant as a counterexample to it. Let X be a PP with intensity function $\lambda(\cdot)$, where $\lambda(s) > 1, \forall s \in \mathcal{D}$.

Such a process is not hard to achieve, since the intensity function depends on the scale of the Lebesgue measure (region area in case of the bi-variate PoP). So changing an area measure from, say, squared meters to kilometers increases the scale of the function $\lambda(\cdot)$, since 1 occurrence per meter is equivalent to 1000 occurrences per kilometer. Then all that is required for rescaling the intensity so that $\lambda(s) > 1$ for all s is that $\lambda(s) > s_0$ for all s and some $s_0 > 0$.

Then, assume that the density of X is as in equation (1.27). For every realization x of the process, consider another realization $x' = x \cup t$ such that $t \in \mathcal{D} \setminus x$, where $\mathcal{D} \setminus x = \{s : s \in \mathcal{D}, s \notin x\}$. Then

$$\begin{aligned} f(x') &= e^{-\int_{\mathcal{D}} \lambda(s) ds} \prod_{s \in x'} \lambda(s) = \\ &= \lambda(t) e^{-\int_{\mathcal{D}} \lambda(s) ds} \prod_{s \in x} \lambda(s) = \\ &= \lambda(t) f(x) > f(x). \end{aligned} \tag{1.28}$$

Also, remember in the notation of Section 1.2.2 that N_X is a measurable mapping of the process X to the natural numbers which induces a probability measure. This means that, for events $N \in \mathcal{N}$ and $L \in \mathcal{A}_{\mathbb{N}}$ the relationship

$$\mathcal{P}_X(N) = \mathcal{P}_{N_X}(L) \tag{1.29}$$

must imply that all the point counts of the events in N must be in L . Let m be the mode of the marginal distribution of N_X , that is the value with the largest probability. If m is finite, then $P(N_X = m) = \mathcal{P}_X(N^{(m)})$, where $N^{(m)}$ is the set of all process realizations with m points. Since N_X has a Poisson distribution in a PP, then m is finite, but according to the development in (1.28), the set N' which is composed of all the realizations in N with any one single point added has larger probability than N , which is absurd, as the induced $P(N_X = m + 1)$ must have smaller probability. \square

Sampling from a Poisson Process

Although many sampling methods are available in Møller and Waagepetersen (2005), one in particular stands out for sampling from an IPP. The method used throughout this thesis is the one from Lewis and Shedler (1979). Drawing a random sample from a PP X with intensity function $\lambda(\cdot)$ is described in Algorithm 1.

Algorithm 1

0. Choose large value for λ^*
1. Simulate $N_X \sim \text{Poisson}(\lambda^*)$;
2. Spread n_x points (candidate events) uniformly and independently along \mathcal{D} ;
3. Evaluate $\frac{\lambda(s)}{\lambda^*}$ for each point s from step 2;
4. Re-sample the n_x points by retaining each with probability $\frac{\lambda(s)}{\lambda^*}$ to acquire x .

The proof of this method is very straightforward. Note that the points in step 2 of Algorithm 1 are distributed according to an HPP with intensity λ^* . The points X are the independent thinning of these points with probability $\lambda(\cdot)/\lambda^*$ and thus is a PP with intensity $\lambda^* \lambda(\cdot)/\lambda^* = \lambda(\cdot)$.

Modeling the intensity function

From Section 1.2.3, it is noticeable that a Poisson Process is completely characterized by its intensity function. Then modeling point data with a Poisson Process means describing this function. Different formats of the intensity may lead to very different types of patterns. A constant function leads to a completely random pattern, as commented in Section 1.2.3. Also, a process with high intensity in few sub-regions may mislead one to think the process is a clustered process, commented in Section 1.2.2. Note that a PP can never be a clustered process, since the former implies no interaction between the points and the latter requires the opposite.

Figure 1.5 shows the realization of an IPP whose intensity function is high only near four specific regions and low everywhere else. Therefore the occurrence of points is high in these four regions. The example shows no occurrence outside of these regions which may increase the impression that it is a clustered process. The difference is that the points occurred in regions where they are more likely regardless of the presence or absence of other points, and the areas with high intensity are small compared to the total region. In a clustered process, on the other hand, the points occur close to each other, not necessarily in some place or places in the region over others.

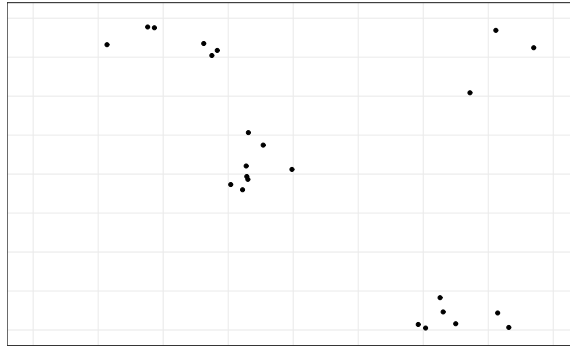


Figure 1.5: Realization of an IPP which may give the wrong impression that the underlying process is a cluster process

The distinction between an IPP and a clustered process may be important to note if there are areas where the points occur more often and/or whether points tend to cluster.

This example is useful to note that an adequate description of the intensity function of a PP is of high value to an analysis. Whenever there is information that may be added towards that end should be included, which means that covariates can greatly help the modeling of the Point Process, particularly if they are available in all locations throughout the region. As in Diggle (2013), a common way to model this is by writing

$$\log \lambda(s) = Z(s)\beta, \forall s \in \mathcal{D}, \quad (1.30)$$

where $Z(s)$ is a $1 \times q_Z$ row vector of intensity covariates at location s and β is a $q_Z \times 1$ column vector of unknown intensity effects parameters. If likelihood based analysis is desired, the corresponding likelihood function for the model parameters β is

$$L_x(\beta) \propto e^{-\int_{\mathcal{D}} e^{Z(s)\beta} ds} \prod_{s \in x} e^{Z(s)\beta}. \quad (1.31)$$

A common problem in maximizing equation (1.31) for β is that the integral $\int_{\mathcal{D}} e^{Z(s)\beta} ds$ cannot be solved analytically, as it becomes necessary to know the functional form of $Z(s)$ throughout \mathcal{D} . A common way to address this is to approximate the integral by some quadrature discretization which is sound since \mathcal{D} is necessarily bounded.

Chapter 2

Proposal description and inference

Presence-only data analysis is the name of methods aimed at relating covariates with reported observations of a certain species over a region. These observations do not compose the full set of specimen in the area and are often considered to be opportunistic, in the sense that they occur mostly in more accessible areas. This thesis treats the problem as the Point Process discussed in Section 1.2.2 framework, which fits the problem very well, since what is random and under study is the prevalence and locations of a particular species in a bounded region. Some works such as Warton and Shepherd (2010) and Fithian and Hastie (2013) also discuss that this approach is well suited for this type of data.

2.1. LITERATURE CONTEXT REVIEW

2.1.1. Preferential sampling in spatial data

2.1.2. Presence-only data methods

The IPP approach to presence-only data appears in a time when biologists were predominantly using the method known as MaxEnt (Phillips et al., 2004) which in turn gained popularity after the pseudo-absence logistic model (Pearce and Boyce, 2006; Elith and Leathwick, 2007) was mostly used.

The latter is a simple logistic that considers the observations as successes and chooses random locations in the unobserved portion of the region as failures. Thus a simple logistic model is fit with ones and zeros. The main problem with this is that the sampled pseudo-absences are not guaranteed absences and can potentially fail to correctly relate the covariates with the data. On the other hand, MaxEnt is a Machine Learning approach to the problem. It attempts to model the underlying distribution of presence locations. For that intent, it forces the difference between the theoretical expected value of the underlying covariates distribution to be equal or close to the average observed values at the presence locations, and in every other aspect the distribution has maximum entropy.

Despite its popularity, MaxEnt provides no statistical evaluation of the model and parameters estimation. However, Renner and Warton (2013) show that the numerical evaluation of MaxEnt provides the same numerical result for the model parameters as the maximum likelihood estimator of an IPP with log-linear intensity with respect to the covariates, as in equation (1.31), when using a particular quadrature approximation of the intractable integral,

as discussed in Section 1. Furthermore, Fithian and Hastie (2013) show the same result, with the difference that the equivalence holds for the theoretical solutions. Therefore the IPP becomes more interesting since it allows for statistical evaluation of the method. In Renner and Warton (2013), for example, the statistical adequacy of a PP to a particular dataset is analyzed, in terms of lack of interaction. They conclude that some sort of interaction is necessary for the data, making the PP no longer adequate and also MaxEnt given their mathematical equivalence.

The IPP, however, has a particular problem, which is that it assumes that all occurrences have been observed. Since the opportunistic sampling is of great importance to the analysis of presence-only data, it must be taken into account. Fortunately the thinning operator as discussed in Section 1.2.2 can take care of that. So if all the species occurrences, observed or not, are distributed according to a Point Process, it is reasonable to say that the observed specimen form another Point Process which is the result of the thinning of the original process. This solution is discussed in Fithian and Hastie (2013), Dorazio (2014) and Fithian et al. (2015). However, they note a particular problem with the thinning, which is also modeled in terms of covariates, in a logistic form. Then the resulting thinned IPP has intensity

$$\lambda(s)p(s) = e^{Z(s)\beta} \frac{e^{W(s)\delta}}{1 + e^{W(s)\delta}}, \quad (2.1)$$

where $W(s)$ represents a $1 \times q_W$ row vector of observability covariates at location s and δ is a $q_W \times 1$ column vector of unknown observability effects parameters. A problem arises due to the fact that equation (2.1) is approximately $e^{Z(s)\beta + W(s)\delta}$ when the observability probability is low, i.e., the thinning probability $1 - p(\cdot)$ is high. In that case, β and δ will have unidentifiable parameters if covariates of $Z(\cdot)$ and $W(\cdot)$ are highly correlated. It is certainly a problem for the intercept terms of these covariates sets.

The proposed model presented in Section 2.2 addresses the intractable integral mentioned in Section 1 and reduces the identifiability issue from equation (2.1).

One last method used in modeling presence-only data is the one known as MaxLike (Royle et al., 2012), which considers the observed locations in a discretized region to be a random sample for all presence locations in the region. The method considers all presence locations to be equally probable to be selected, but that is not true due to the opportunistic sampling discussed above. Also, the model promises to be able to estimate species prevalence, but Hastie and Fithian (2013) provide some simulations to show that this estimation is highly dependent on the proposed relationship to be logit-linear, so that a slight deviation from this causes a large mis-estimation of the prevalence.

2.2. PROPOSED MODEL

The model follows the idea of Gonçalves and Gamerman (2018) to avoid having to approximate the intractable integral in equation (1.25). They, in turn, based themselves on the idea towards the same purpose in Adams et al. (2009). The version presented here is a further modification to include a probabilistic object to represent the unobserved occurrences.

Let Y be the PP that represents all occurrences of the species. Let $\lambda(\cdot)$ be the intensity of Y and assume that $\lambda(s) \leq \lambda^*$, for all $s \in \mathcal{D}$. Here, λ^* represents the supreme of the

occurrences intensity. It is not inviable to say that $\lambda(\cdot)$ is bounded since it is modeled as a function of covariates which are features of the region and are in their turn bounded by physics.

Then instead of the log-linear form of equation (1.31), model the intensity by

$$\lambda(s) = q(s)\lambda^*, \quad (2.2)$$

where $q(s)$ is a function between zero and one. Although any function with such constraint can be used, this thesis models it as

$$\begin{aligned} \text{logit } q(s) &= Z(s)\beta \\ q(s) &= \frac{e^{Z(s)\beta}}{1 + e^{Z(s)\beta}} = \left(1 + e^{-Z(s)\beta}\right)^{-1}, \end{aligned} \quad (2.3)$$

where $W(s)\delta$ represent the impact of intensity covariates on Y . Although $q(\cdot)$ is a functional field that assumes values between 0 and 1, it is not interpreted as a probability in the model. However, it is noted that Y can be constructed as the independent thinning of an HPP with intensity λ^* using a probability field $q(\cdot)$.

Since the observations are only part of Y , it is reasonable to divide Y between these parts, that is, X represents the observed occurrences and X' the unobserved ones. This is modeled as independent thinning, thus

$$\begin{aligned} X &= \tau(Y, p(\cdot)) \\ \text{logit } p(s) &= W(s)\delta \end{aligned} \quad (2.4)$$

Thus, marginally, X is an IPP with intensity $q(\cdot)p(\cdot)\lambda^*$. It is noteworthy that, conversely, X' is independent of X and has intensity $q(\cdot)(1 - p(\cdot))\lambda^*$. One more component is added, which is called U , another IPP, independent of Y with intensity $(1 - q(\cdot))\lambda^*$. Although U has no physical interpretation, its creation is vital to achieve the desired result below. Thus the proposed model is

$$\begin{aligned} \mathcal{Y} &\sim PP(\lambda^*) \\ Y &= \tau(\mathcal{Y}, q(\cdot)) \\ \text{logit } q(s) &= Z(s)\beta \\ U &= \mathcal{Y} \setminus Y \\ X &= \tau(Y, p(\cdot)) \\ \text{logit } p(s) &= W(s)\delta \\ X' &= Y \setminus X, \end{aligned} \quad (2.5)$$

where \mathcal{Y} is used to represent a complete HPP which is the result of the independent superposition of X , X' and U . All that is needed for the density of the model is the joint density of X , X' and U which achieved by the multiplication of their marginal densities, given their mutual independence.

$$\begin{aligned}
f(x, x', u | \beta, \delta, \lambda^*) &= f(x | \beta, \delta, \lambda^*) f(x' | \beta, \delta, \lambda^*) f(u | \beta, \delta, \lambda^*) = \\
&= \frac{e^{-\int_{\mathcal{D}} q(s)p(s)\lambda^*}}{n_x!} \frac{e^{-\int_{\mathcal{D}} q(s)(1-p(s))\lambda^*}}{n_{x'}!} \frac{e^{-\int_{\mathcal{D}} (1-q(s))\lambda^*}}{n_u!} \prod_{s \in x} q(s)p(s)\lambda^* \times \\
&\times \prod_{s \in x'} q(s)(1-p(s))\lambda^* \prod_{s \in u} (1-q(s))\lambda^* = \\
&= \frac{e^{-\int_{\mathcal{D}} \lambda^* ds}}{n_x! n_{x'}! n_u!} \prod_{s \in x} q(s)p(s)\lambda^* \prod_{s \in x'} q(s)(1-p(s))\lambda^* \prod_{s \in u} (1-q(s))\lambda^* = \\
&= \frac{e^{-|\mathcal{D}|\lambda^*}}{n_x! n_{x'}! n_u!} \prod_{s \in x} q(s)p(s)\lambda^* \prod_{s \in x'} q(s)(1-p(s))\lambda^* \prod_{s \in u} (1-q(s))\lambda^*,
\end{aligned} \tag{2.6}$$

where $|\mathcal{D}|$ represents the Lebesgue measure of \mathcal{D} , its area in case of the bi-variate process. Note that, by evaluating the joint density of the three processes, there is no integral to evaluate. Hence, if a likelihood function is constructed based on equation (2.6), there will be no need to approximate any integral. The trade off is the creation of two latent processes, one of which doesn't have a physical interpretation.

An important thing to notice about this model is that there is no spatial relationship between the processes points *except* for the relationship encoded in the covariates. This means that, if the covariates values are known at every location in a continuous fashion, then the model only depends on the coordinates to determine the covariates values. The model could include extra spatial dependence by the introduction of some spatial model in the covariates sets $Z(\cdot)$ or $W(\cdot)$.

A less evident advantage of this approach is with regard to the identifiability issue. Note that the form shown in equation (2.1) has changed. The evaluation of the observations now depends on

$$q(s)p(s) = \frac{e^{Z(s)\beta}}{1 + e^{Z(s)\beta}} \frac{e^{W(s)\delta}}{1 + e^{W(s)\delta}}, \tag{2.7}$$

which means that both $q(s)$ and $p(s)$ need to be small simultaneously in order for the identifiability to be an issue.

On the other hand, there may be a problem with either one of the intercepts in β or δ with λ^* , since there is little information to separate the two, meaning that decreasing the intercepts and increasing λ^* should yield similar intensity values. This is discussed in Gonçalves and Gamerman (2018) where they propose the inclusion of prior information on λ^* to help with this issue. However, this issue brings forth an interesting result.

The IPP with the traditional log-linear intensity that yields the likelihood function shown in equation (1.31) is a limit of an IPP with intensity $q(\cdot)\lambda^*$ with logit-linear $q(\cdot)$. To show that, let β_0 be the intercept of the regression predictor $Z(s)\beta$. Let β_0 and λ^* move jointly by the relationship

$$\left(1 + e^{-\beta_0}\right)^{-1} \lambda^* = \frac{e^{\beta_0}}{1 + e^{\beta_0}} \lambda^* = c, \tag{2.8}$$

for some positive constant $c < \lambda^*$. The result shows that the intensity function $(1 + e^{-Z(\cdot)\beta})^{-1} \lambda^*$ goes to $e^{Z(\cdot)\beta}$ when λ^* goes to infinity and β_0 goes to negative infinity in a rate that maintains equation (2.8) true. First, find β_0 as a function of λ^* and c so that the limit only needs to be done once. Then

$$(1 + e^{-\beta_0})^{-1} \lambda^* = c \Rightarrow e^{-\beta_0} = \frac{\lambda^* - c}{c} \quad (2.9)$$

Now apply the limit, denoting by $\tilde{Z}(\cdot)\tilde{\beta}$ the covariates times the effects minus the intercept β_0 ,

$$\begin{aligned} \lim_{\lambda^* \rightarrow \infty} \frac{\lambda^*}{1 + e^{-Z(\cdot)\beta}} &= \lim_{\lambda^* \rightarrow \infty} \frac{\lambda^*}{1 + e^{-\beta_0 - \tilde{Z}(\cdot)\tilde{\beta}}} = \\ &= \lim_{\lambda^* \rightarrow \infty} \frac{\lambda^*}{1 + e^{-\beta_0} e^{-\tilde{Z}(\cdot)\tilde{\beta}}} = \\ &= \lim_{\lambda^* \rightarrow \infty} \frac{\lambda^*}{1 + \frac{\lambda^* - c}{c} e^{-\tilde{Z}(\cdot)\tilde{\beta}}} = \\ &= \lim_{\lambda^* \rightarrow \infty} \frac{1}{\frac{e^{-\tilde{Z}(\cdot)\tilde{\beta}}}{c}} = \\ &= c e^{\tilde{Z}(\cdot)\tilde{\beta}}, \end{aligned} \quad (2.10)$$

where the fourth line comes from L'Hôpital's rule. Now call $c = e^{\beta'_0}$ and result follows where β'_0 is the intercept of the log-linear model. \square

This result provides an interesting observation when estimating the proposed model. If there is an indication that λ^* is finite, then the proposed logit-linear model is more adequate than the traditional log-linear one.

2.3. INFERENCE

The proposed model derived in equation (2.6) leads to a likelihood function which may be evaluated analytically, however it depends on two latent processes, X' and U , which means that their realizations are required for this evaluation. As in Adams et al. (2009) and Gonçalves and Gamerman (2018), Bayesian inference via MCMC provides an instant solution to this. Bayesian inference is described in Section 2.3.1 and MCMC in Section 2.3.2.

2.3.1. Bayesian inference

A detailed description of Bayesian inference can be found in Gelman et al. (2013). In the Bayesian paradigm all inference is based on the posterior distribution. Given a set of parameters Θ , the posterior distribution is obtained by use of the Bayes theorem, which states that, for two events A and B ,

$$P(A | B) = \frac{P(B | A)P(A)}{P(B)}. \quad (2.11)$$

To achieve this notion in a probabilistic model, call A the parameter set Θ and the collected or measured data B . The goal is to describe the conditional probability of the parameters given the data. Since the data are stochastically described conditional to the parameters by the model's density function, the formulation is

$$\pi(\Theta | x) = \frac{L_x(\Theta)\pi(\Theta)}{f(x)} \propto L_x(\Theta)\pi(\Theta), \quad (2.12)$$

where $\pi(\Theta | x)$ represents the posterior density, $\pi(\Theta)$, the prior density, $L_x(\Theta)$ the model's likelihood function and $f(x)$ the marginal likelihood. The prior distribution, which yields the prior density, contains all information about the parameters prior to the data collection or measurement. The proportionality in equation (2.12) stems from the fact that $f(x)$ is a constant with respect to Θ and is commonly suppressed in most Bayesian inference procedures. In the Bayesian framework, the prior information about the parameters is said to be updated by the likelihood function, that is, by the data.

Then, in order for the inference to be made, it is required that not only the model is chosen, but also the prior distribution.

A common problem with regard to acquiring the posterior distribution is its complexity. In most non-trivial problems, this multivariate distribution is not known in closed form. Although analytical approximations are available, such as R-INLA (Rue et al., 2009), the focus of the thesis is inference via Markov Chain Monte Carlo.

2.3.2. Markov Chain Monte Carlo

The solution known as MCMC has had four major breakthroughs which led to its widespread use in Statistics. The first was Metropolis et al. (1953) which introduced its first format, known as the Metropolis method. The second was Hastings (1970) who provided a faster method than Metropolis', despite being based on the same principle. His updated method became known as the Metropolis-Hastings method. The third was Geman and Geman (1984) who applied the idea to image processing which eliminated an acceptance-rejection step of the Metropolis-Hastings method. Their method became known as the Gibbs sampler, but it only began an exponential growth in the field of Statistics after Gelfand and Smith (1990).

Although a more detailed description of these methods can be found in Gamerman and Lopes (2006), it is briefly introduced here. The method considers the Monte Carlo approximation to an integral. Let $\{x_1, \dots, x_n\}$ represent an n sized random sample of a distribution with probability measure $\mu(\cdot)$ with support in a space \mathcal{S} . Then for any μ -integrable function $h(\cdot)$,

$$\int_{\mathcal{S}} h(x) d\mu(x) \approx \frac{1}{n} \sum_{i=1}^n h(x_i), \quad (2.13)$$

where the integral with respect to $\mu(x)$ is the Lebesgue-Stieltjes integral. The result converges to equality as n goes to infinity. In terms of Statistical inference, this is a very powerful result, since many distribution summaries can be achieved when the appropriate function $h(\cdot)$ is used. Table 2.3.2 provides some useful examples. Let $\mathcal{S} = \mathcal{S}' + \mathcal{S}'_{\perp}$ where \mathcal{S}' and \mathcal{S}'_{\perp} are perpendicular subspaces of \mathcal{S} generated from a two-fold partition of the dimensions of \mathcal{S} .

The different methods mentioned above which came to be known as MCMC are ways to draw samples from the desired posterior distribution. For that effect, the theory of Markov Chain is used (Gamerman and Lopes, 2006). The theory of Markov Chains will not be reviewed here,

$h(x) =$	Summary
x	Expected value
$(x - \bar{x})^2$	Variance
$\mathbb{1}_{(-\infty, a)}(x)$	$P(X \leq a)$
$(\mathbb{1}_{(-\infty, \cdot)}(q))^{-1}(x)$	q^{th} quantile
Retaining \mathcal{S}' dimensions of x	Marginalizing with respect to \mathcal{S}'_{\perp}
Density kernel	Distribution's p.d.f.

Table 2.1: Usable functions that lead to interesting integrals to be estimated by the Monte Carlo method

but the main result is presented.

A Markov Chain has transition probabilities for all its possible states that describe the transition of the state at one time point to the state in the next. Its characterizing feature is that given all the history of the chain, the next step is only stochastically dependent on the last step. The transition probabilities define the properties of the chain. A state is said to be ergodic if the chain returns to it after leaving it at a given step lesser than infinity with probability larger than zero and if the set of the lengths of possible paths leaving the state which return to it have greatest common divisor 1. If all states in a chain are ergodic, then the chain is ergodic.

An ergodic chain necessarily has a stationary distribution, which means to say that a random variable from it, when updated with the chain's transition probabilities still has marginally the stationary distribution. From any starting point, a Markov Chain converges to the stationary distribution, which means that after applying the transition to any starting point, the states will be distributed according to it. All that is left is to find a Markov Chain that has the posterior as its stationary distribution.

The proposal of Metropolis et al. (1953) finds such a chain whose transition is to iterate at every coordinate or Θ in equation (2.12). Draw $p^{(*)}$ from a uniform distribution centered on the previous point p at that coordinate. If the resulting posterior is larger than what it was, accept the new point, else accept it with probability $\pi(p^{(*)} | *) / \pi(p | *)$ where $\pi(\cdot | *)$ represents the posterior density varying at the corresponding coordinate and fixing the other ones, called the full conditional. Since the posterior distribution is only used in that ratio and in its relative comparing, then the marginal likelihood is canceled out.

It was updated by Hastings (1970) who proposed the sampling of an undefined proposal distribution $q(\cdot)$ which may improve the acceptance rate. The new acceptance probability is $\pi(p^{(*)} | *) / \pi(p | *) \times q(p) / q(p^{(*)})$. Finally, Gelfand and Smith (1990) noted from Geman and Geman (1984) that, if $\pi(\cdot | *)$ is the kernel of a distribution from which it is known how to sample, then it is enough to sample from it, without requiring an acceptance step.

The Gibbs sampler and the Metropolis-Hastings (M-H) method can be joined in the sense that, if the full conditional is known in closed form at some coordinates, then the Gibbs sampler is used, but at the coordinates where it is not known, then a M-H step is used. This is also known as the Metropolis(-Hastings)-within-Gibbs algorithm.

Other more advanced Monte Carlo methods are already available, such as Stan (Stan

Development Team, 2019) and adaptive MC (Haario et al., 1999). Although Stan is a very powerful tool, it is reliant on posterior gradients which are not trivial in the proposed model, since it would be necessary to differentiate the posterior also with respect to the latent processes X' and U . The difficulty arises from the fact that it is hard to define distance between two Point Process realizations, and deriving gradients is dependent on distances. The adaptive MC provides interesting methods to evolve the M-H proposal distribution over the iterations, however the acceptance rate has not been an issue in the implementation of the model.

Therefore, the proposal uses standard Metropolis-within-Gibbs to draw samples from the posterior. The exact algorithm is detailed in Section 2.3.3.

2.3.3. Prior and Full conditionals

In order to fully define the model and make inference, a prior distribution must be defined, as reviewed in Section 2.3.1. The distribution chosen must describe the prior knowledge about the parameters, but it is useful to choose it so as to facilitate calculations. In this case, the choice is made to achieve easily sampled full conditionals, as described in Section 2.3.2.

The parameter vector for the model (2.5) is $\Theta = (X', U, \beta, \delta, \lambda^*)$, where β itself is a q_Z long vector and δ is a q_W long vector. Also note that X' and U are stochastic processes, therefore the parameter vector is infinite-dimensional.

The prior is specified so that X' , U , β , δ and λ^* are all pairwise independent.

Since the stochastic description of X' and U is completely done in the model, then they don't require further prior specification. The other parameters, however, do, although the likelihood function extracted from equation (2.6) permits some convenient definitions. So, if $\lambda^* \sim \text{Gamma}(a, b)$, whose density is

$$\pi(\lambda^*) = \frac{b^a}{\Gamma(a)} \lambda^{*a-1} e^{-b\lambda^*} \mathbb{1}_{(0, \lambda_{max})}(\lambda^*), \quad (2.14)$$

then the full conditional of λ^* will also be Gamma with parameters $a + n_x + n_{x'} + n_u$ and $b + |\mathcal{D}|$. In equation (2.14) the value λ_{max} is ∞ when the prior is not truncated and a positive fixed value otherwise. Therefore, the Gamma distribution is used with choices of a and b very small, so that the prior distribution has very large variance, so as to not induce any non-existing prior information. In all instances throughout the applications in the thesis, the values chosen were 10^{-4} .

For the β and δ parameter vectors, it is interesting to note that, given the realizations of X' , U and λ^* , the likelihood function resembles the product of two logistic regressions. The first, considering β , treats $x \cup x'$ as successes and u as failures, whilst considering δ , treats x as successes and x' as failures.

So given their prior independence, the full conditionals of β and δ will also be independent and resemble the posterior distributions of two separate logistic regression. Therefore, multivariate Normal distributions with zero mean and diagonal covariance matrices with large variances are used for these two vectors for their prior.

Despite their conditional independence, β and δ are updated in the MCMC in a single step to accelerate computations.

Then, the algorithm for the MCMC procedure is

Algorithm 2

- 0-0. Initialize the chains by setting initial coordinates for X' and U and determine their covariates and call them iteration 0
- 0-1. Further initialize the chains by setting initial values for β , δ and λ^* and call them iteration 0
- 1. Draw a realization of two independent PPes with adequate intensity functions for X' and U given the parameters in the last iteration using Algorithm 1
- 2. Draw a sample from $\text{Gamma}(a + n_x + n_{x'} + n_u, b + |\mathcal{D}|)$ for λ^* using the PPes realizations from the last step
- 3. Use a Metropolis-Hastings acceptance-rejection step with Gaussian centered on the last step as proposal for the independent logistic regressions of β and δ using the PPes realizations from step 1
- 4. Decide if convergence has been achieved AND if enough draws from the posterior have been made to perform a good enough Monte Carlo approximation. If so, stop program. If not, add 1 to the current value of the iteration and return to step 1

There are a few things to note. Firstly, step 0-0. is not required, as the initiated values aren't used since the processes are sampled at the first step. Secondly, the sampling of λ^* does not depend on the sampled values of β and δ . This, however, does not mean that they are independent when marginalized with respect to the latent processes in the posterior distribution.

In fact, the intercept coordinates of both β and δ vectors can have an identifiability issue with λ^* , as discussed in Gonçalves and Gamerman (2018). Figure 2.1 illustrates the relationship between these parameters in one of the applications. In it, δ_0 represents the intercept component of the vector δ .

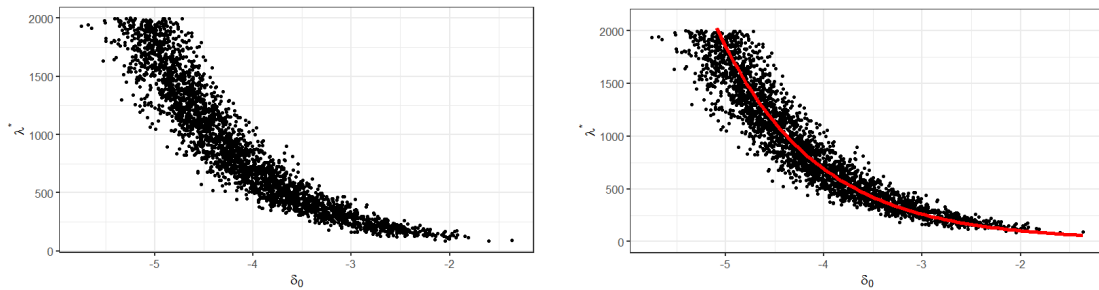


Figure 2.1: Posterior relation between intercept and λ^*

The red line on the graphic on the right-hand side of Figure 2.1 has been drawn using the same relationship of equation (2.8).

$$\left(1 + e^{-\delta_0}\right)^{-1} \lambda^* = c \quad (2.15)$$

for some positive constant c , which has been chosen as the posterior expected value for this relation, found by Monte Carlo integration.

This means that a viable reparameterization for the model which might reduce posterior parameter dependence is to exchange λ^* for, say, $\gamma = \left(1 + e^{-\beta_0}\right)^{-1} \left(1 + e^{-\delta_0}\right)^{-1} \lambda^*$. Although only one of the intercepts, i.e. either β_0 or δ_0 , yield the observed relationship with λ^* , both have been included since it is not yet clear what causes one or the other intercept to be related to λ^* . The posterior density calling $pts = n_x + n_{x'} + n_u$, adjusted by the Jacobian, becomes

$$\begin{aligned} \pi(x', u, \beta, \delta, \gamma | x) &\propto \frac{e^{-(b+|\mathcal{D}|)(1+e^{-\beta_0})(1+e^{-\delta_0})\gamma}}{n_{x'}!n_u!} \gamma^{a+pts-1} \left[\left(1 + e^{-\beta_0}\right) \left(1 + e^{-\delta_0}\right) \right]^{a+pts} \times \\ &\quad \times \prod_{s \in x} \left(1 + e^{-Z(s)\beta}\right)^{-1} \left(1 + e^{-W(s)\delta}\right)^{-1} \times \\ &\quad \times \prod_{s \in x'} \left(1 + e^{-Z(s)\beta}\right)^{-1} \left(1 + e^{W(s)\delta}\right)^{-1} \times \\ &\quad \times \prod_{s \in u} \left(1 + e^{Z(s)\beta}\right)^{-1} \times \\ &\quad \times \phi\left(\Sigma_\beta^{-1/2}\beta\right) \phi\left(\Sigma_\delta^{-1/2}\delta\right), \end{aligned} \quad (2.16)$$

where $\phi\left(\Sigma_\beta^{-1/2}\beta\right)$ represents the multivariate Normal density with mean zero and covariance matrix Σ_β . $\phi\left(\Sigma_\delta^{-1/2}\delta\right)$ is analogous.

Note that this causes β , δ and γ to *not* be conditional independent given the latent processes, but it should reduce the marginal dependence between the parameters. However, the full conditional for γ is still a Gamma distribution and the ones for β and δ are no longer independent. Since in Algorithm 2 they were sampled in the same M-H step, then little needs to be changed. However, in order to improve the acceptance rate, the proposal distribution was chosen as a multivariate normal in dimension $q_Z + q_W$ centered on the last step and with covariance matrix equal to the negative inverted hessian of the full conditional. Algorithm 3 describes the new sampling procedure.

Algorithm 3

0. Initialize the chains by setting initial values for β , δ and γ and call them iteration 0
1. Draw a realization of two independent PPes with adequate intensity functions for X' and U given the parameters in the last iteration using Algorithm 1
2. Draw a sample from $Gamma((a + pts, (b + |\mathcal{D}|)(1 + e^{-\beta_0})(1 + e^{-\delta_0}))$ for γ using the PPes realizations from the last step
3. Use a Metropolis-Hastings acceptance-rejection step with Gaussian centered on the last step and covariance matrix equal to the negative inverted hessian matrix of the full conditional as proposal for the joint sampling of β and δ using the PPes realizations from step 1 and γ from step 2
4. Decide if convergence has been achieved AND if enough draws from the posterior have been made to perform a good enough Monte Carlo approximation. If so, stop program. If not, add 1 to the current value of the iteration and return to step 1

Algorithm 3 is the one used throughout this thesis.

Computation time

A common issue when using methods that rely on heavy computation is the computation time. It is usual to code programs like MCMC in lower level programming languages like C++, Ox or, more recently, Julia. All MCMC calculations in this thesis has been done using Ox version 7.20 (Doornik, 2017). The reason to use lower level languages is that the computation time is usually much lower. Appendix Section B.2 presents part of the code used to draw samples of the posterior. It also references an online resource to the rest of the code.

There is an issue that arises with the proposed model estimated with Algorithm 3. The number of points for latent processes X' and U is random and depend on the model parameter λ^* . As the number of points of these processes vary, so do the number of calculations the computer must make, changing not only the thinning process of the sampling algorithm in Algorithm 1, but also the number of elements to be multiplied (or summed in the log scale) of the full conditional for β and δ .

This means that as the Markov Chain moves through the parameter space, higher values of λ^* in the chain imply longer computation time. A clear weakness of the model, then, is taking long to converge when λ^* visits high values prior to convergence. Also, behavior such as seen in Figure 2.1 means that the Markov Chain has high auto-correlation, taking it even longer to fully visit the full region of posterior mass.

2.4. FORECASTING AND ACCURACY

The most important utility of a model for presence-only data is its ability to correctly identify where the species occurred everywhere in the region where it wasn't observed. In the proposed

model, this is represented by the latent process X' , that is, all unobserved species occurrences.

2.4.1. Forecasting

It is hard to predict the realization of an unobserved process such as X' . The best possible forecasting in this scenario is to be able to draw the intensity function of the occurrences over the region, such as with a heat map. Note that in order for this model to be estimated, Algorithm 3 requires the ability to evaluate the intensity function, and therefore the covariate values, at every location of the region in a continuous manner.

In any situation where this is possible, such as in simulated data, then the heat map is a good forecast. Furthermore, if the simulated data comes from the same model as the one being used to estimate it, then comparing the parameters is enough, since correctly estimating the parameters means that the intensity function was correctly estimated as well.

In a situation where the covariates can only be measured up to a fine lattice over the region, then some sort of approximation is required for the model. The most important one is about the covariate values. The most intuitive way to input a covariate in a point outside the lattice vertices is to use some sort of interpolation. It is important, however, to know that any interpolation assumes that the covariates vary smoothly to some degree. In most situations this is reasonable, even an interpolation as simple as taking the covariate value at the closest vertex available. This procedure is equivalent to sampling the vertices themselves with reposition when applying Algorithm 1 to sample PPes.

On the other hand, a fine lattice of points also limits the possibilities for species occurrences, since now they can only occur in a finite set of points. This means that, at every iteration of the MCMC, every lattice location can be marked as one if X' occurred there or zero otherwise. At the end of the MCMC, it is possible to count how often X' occurred at each location, providing a posterior probability of occurrence for it.

2.4.2. Accuracy

The accuracy of a given forecast, however, is harder to obtain. Presence-only data is lacking in information by its nature. So separating a data set as training and test data will further reduce the information available for model estimation. Also, the thinning field $p(\cdot)$ as in equation (2.4) exists for the sole purpose of explaining the observation or not of a given occurrence, so an arbitrary elimination of observed occurrences might and probably will affect the model estimation.

Nevertheless, there are some cases where model accuracy can be provided. Some species that have presence-only data available also have presence-absence data available in the same region. What is meant by presence-absence is that a randomized study has been performed, where locations were pre-selected for sampling and the presence or absence of the species was recorded at every such location. Therefore, presence-absence data is considered to have *unbiased* sampling.

Thus a given presence-only (PO) model which has already been estimated tries to predict the

presence-absence (PA) locations and wherever the PO model correctly predicts a PA observation, a true positive is recorded, and so forth. The procedure for this is detailed in Section 2.5.2.

2.5. MODEL COMPARISON

2.5.1. Proposal vs. existing models

2.5.2. Using presence-only model to predict presence-absence data

It is not straightforward to use the estimation resulting from a presence-only (PO) model to predict presence-absence (PA) data. According to Gelfand and Shirota (2018), this is due to the fact that a presence-absence dataset is composed of 1's and 0's of pre-defined locations, while a presence-only dataset is composed of random locations over the region. The incompatibility appears as a consequence that the random component of each dataset is different, one being a random value 0 or one, and the other, random locations.

However, a unifying theory can be developed, provided that the biological realizations are the same, that being the occurrence of the species. One way to see it is that the incompatibility stems from comparing an aggregated regions with PA data with continuous points with PO data.

However, it is important to remember that the distribution of a Point Process (PoP) is completely specified by the joint distribution of aggregated areas, as seen in Section 1.2.2. Therefore, for each sub-region where the PA data is recorded, the probability of presence can be derived.

The PA data is constituted of m sites labeled by 0's and 1's, then it is safe to assume that the sub-regions are small enough that the probability of there being more than 1 occurrence is very small. So, for each sub-region ds , the calculation desired is, for PoP X , $P(N_X(ds) = 1 | N_X(ds) \leq 1)$. Furthermore, for each entry of the PA data, a single value for each covariate is given, which makes it safe to assume that the covariates are approximately constant within the sub-regions.

Furthermore, since the models being used are Poisson Processes, the random variables $N_X(ds_i), \forall i = 1, \dots, m$ which represent the occurrence counts for all the observed sub-regions in the PA data are independent, since ds_i are disjoint.

So, the following result is helpful for the required forecasting. Let X be a Poisson Process in \mathcal{D} with intensity function $\lambda(\cdot)$. Then for a Borelian A ,

$$P(N_X(A) = n) = \frac{e^{-\int_A \lambda(s)ds} \left(\int_A \lambda(s)ds\right)^n}{n!}. \quad (2.17)$$

Now let $Y(A)$ be a random variable so that

$$Y(A) = \begin{cases} 1, & \text{if } N_X(A) \geq 1 \\ 0, & \text{if } N_X(A) = 0. \end{cases} \quad (2.18)$$

Then, it is easy to see that $P(Y(A) = 1) = 1 - P(N_X(A) = 0) = 1 - e^{-\int_A \lambda(s)ds}$. Furthermore,

$$\begin{aligned}
P(Y(A) = 1 | N_X(A) \leq 1) &= \frac{P(Y(A) = 1, N_X(A) \leq 1)}{P(N_X(A) \leq 1)} = \\
&= \frac{P(Y(A) = 1, N_X(A) = 1)}{P(N_X(A) \leq 1)} = \\
&= \frac{P(N_X(A) = 1)}{P(N_X(A) \leq 1)} = \\
&= \frac{e^{-\int_A \lambda(s) ds} \int_A \lambda(s) ds}{e^{-\int_A \lambda(s) ds} (1 + \int_A \lambda(s) ds)} = \\
&= \frac{\int_A \lambda(s) ds}{1 + \int_A \lambda(s) ds},
\end{aligned} \tag{2.19}$$

where the second equality stems from the fact that $Y(A) = 1$ and $N_X(A) \leq 1$ can only happen simultaneously if $N_X(A) = 1$, and the third is due to the fact that $N_X(A) = 1$ implies $Y(A) = 1$.

This result becomes useful by combining it with the assumption that, for every sub-region of the PA data $ds_i, i = 1, \dots, m$, the covariates are approximately constant, which means that the intensity function is approximately constant. Then, $\lambda(s)$ is approximately fixed in each ds_i . As a consequence, it is possible to write $\int_{ds_i} \lambda(s) ds = \lambda_i |ds_i|$, where λ_i is the constant intensity of ds_i . Therefore,

$$P(Y(ds_i) = 1 | N_X(ds_i) \leq 1) = \frac{\lambda_i |ds_i|}{1 + \lambda_i |ds_i|}. \tag{2.20}$$

In summary, an estimated presence-only model can provide prediction to PA data by evaluating the probability of presence according to equation (2.20).

2.5.3. ROC curve

Comparing the efficacy of models is important for ranking methods in many fields, particularly when forecasting is involved. Goodness-of-fit methods like DIC are relevant in many situations, but when even the data is viewed as different objects, as is the case in pseudo-absence models, MaxLike and IPP, then DIC are no longer necessarily comparable between methods.

Since good forecasting is a desired feature of a PO method, they should be compared by forecasting capability. As mentioned in Section 2.4.2, accuracy can be measured by correctly forecasting a presence-absence data set. The procedure for this is provided in Section 2.5.2. Note that the resulting prediction is encoded as true/false positives and negatives, which implies a confusion matrix, as shown in Table 2.2.

		Reality	
		Positive	Negative
Predicted	Positive	<i>TP</i>	<i>FP</i>
	Negative	<i>FN</i>	<i>TN</i>

Table 2.2: Confusion matrix

TP stands for true positive, *FP* for false positive, *FN* for false negative and *TN* for true negative. Note that $TP + FN$ is the total of positive values in reality and $FP + TN$ is the total

of negative values. A good model will have large TP relative to FN and large TN relative to FP . To measure this, two relative values are used. They are

$$\begin{aligned}\text{Sensitivity} &= \frac{TP}{TP + FN} \\ \text{Specificity} &= \frac{TN}{TN + FP}\end{aligned}\tag{2.21}$$

Note that both these metrics vary between 0 and 1. While analyzing sensitivity and specificity of a confusion matrix is of great value in terms of comparing models accuracy, a problem remains. Most binary outcome models usually output a value between 0 and 1, and the positive/negative classification happens by assigning positives to points whose output exceeds a certain threshold, and negative otherwise.

The threshold is in many cases taken as the value 0.5, as it provides no preference towards any side, however the cases where another threshold is more valuable must be considered. Some examples can be quickly cited, such as if a given plant is being studied and it is imperative that it is preserved against deforestation, that is, it is better to over-estimate its presence. This means that the cost of a false negative is much higher than that of a false positive. Then a lower threshold will achieve this.

On the other hand, if a plant is hard and expensive to reach, then an expedition which is required to extract it should be really sure that the plant is there before expending the necessary resources to reach it. Thus a false positive has higher cost than a false negative, and a high threshold should be used.

It is significant to note that a confusion matrix changes with the threshold. The higher the threshold, the more the method classifies points as negatives, increasing TN and FN . In the extremes, a threshold of 0 yields sensitivity 1 and specificity 0 and vice-versa.

In order to compare models between multiple thresholds, the receiving operator characteristic (ROC) curve is used. This technique is widely used in ecology and medicine. It is a plot of sensitivity and 1-specificity for all threshold varying from 1 to 0. Figure 2.2 illustrates a ROC curve. The procedure and code used to generate this example is detailed in Appendix Section A.2.

A dashed line has been included in the figure to represent a random classifier, that is, a classifier which assigns positives with probability t independently to all points, where $1 - t$ is the threshold. This means that the probability that a given positive will be correctly assigned is t , which is also the expected sensitivity. Also, t is the probability of a point to be falsely assigned as a positive, i.e. the expected 1-specificity. This means that the ROC curve of this classifier is the identity line in the 0-1 square.

A perfect classifier, on the other hand, must have at least one threshold that leads to sensitivity = specificity = 1, or sensitivity = 1 and 1-specificity = 0. Since the ROC curve is monotonically non-decreasing, this implies that the ROC curve of the perfect classifier starts at the origin then moves vertically to the coordinates (0, 1), then horizontally to (1, 1).

The ROC line of the random classifier is usually included when analyzing a model since it is desirable that the proposed method performs better than the random classifier. It is also

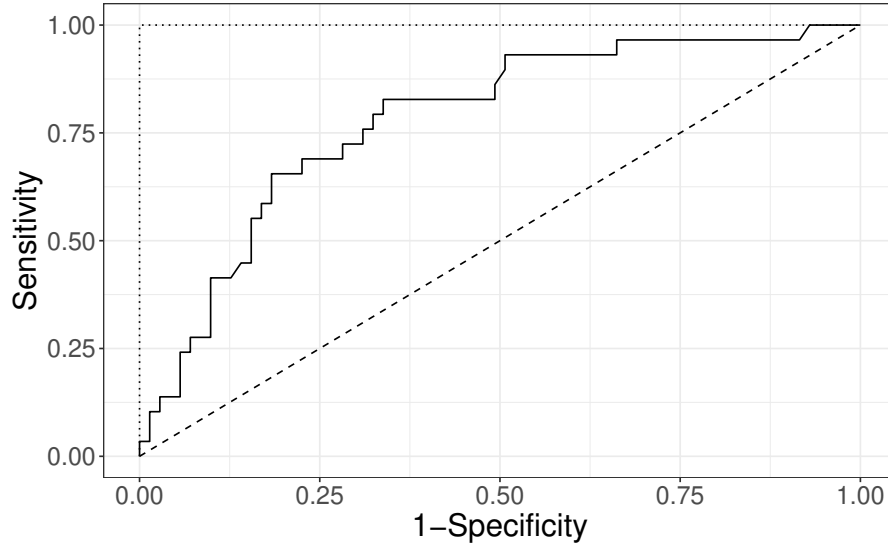


Figure 2.2: ROC curve example - dashed line represents the random classifier - dotted line represents the perfect classifier

expected that any method for non-trivial data will perform more poorly than the perfect classifier, which is usually not included given its obvious position.

Therefore the ROC curve of a method usually lies between the random and perfect classifiers. The higher it goes, the better it is, meaning that it is desired that the curve goes as close to the perfect classifier as possible. It also means that two methods can be compared by how “high” their curves go, and this comparison is unconditional to output threshold. A more detailed description of the ROC curve in the medical and ecological context, as well as other metrics based on a confusion matrix can be found in Fielding and Bell (1997). They explain that the name receiver operating characteristic is related to signal processing where the receiver is a person or computer who has a characteristic capability of identifying a signal.

2.5.4. AUC

Although useful and widely used, the ROC curve presented in Section 2.5.3 is based on graphical comparison. Therefore it is possible to encounter a situation where the better model is not evident in the graphic, such as when the ROC curves cross each other. Figure 2.3 illustrates such a situation where the dot-dashed line represents a competing model.

The graphic gives the slight impression that the solid line provides a better model, given that it seems to be above the dot-dashed line more often than not, roughly between 1-specificity values ≈ 0.15 and ≈ 0.5 , evidenced by the light green vertical lines. However, a “slight impression” hardly constitutes scientifically sound argument.

For that reason, a metric called Area Under ROC Curve (AUC) is used for an objective comparison of ROC curves. As its name suggests, it is merely calculated by the area under the ROC curve for the model, so that a scalar metric for classification methods can be used. It is easy to realize that the AUC of the perfect classifier is 1 and that of the random classifier is

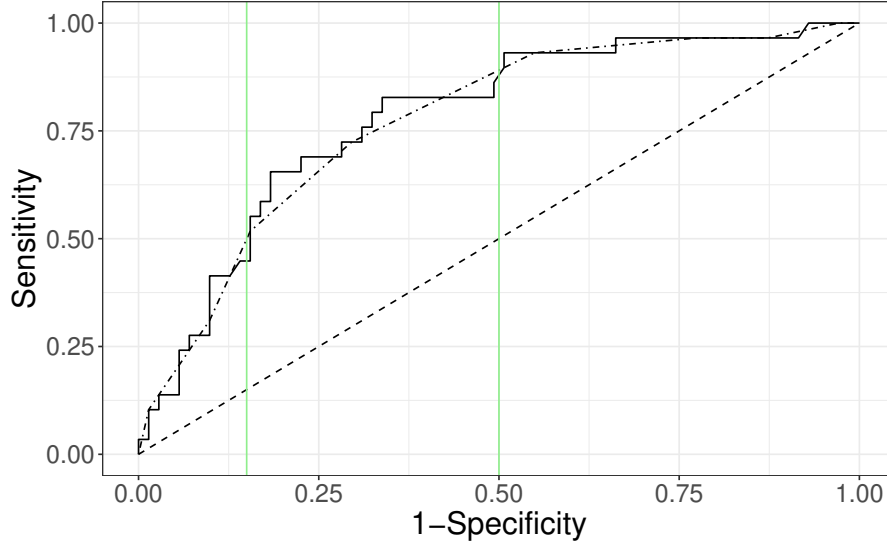


Figure 2.3: Competing ROC curves

0.5, so a given model typically has AUC between these values.

For the example of Figure, 2.3, the solid line has AUC 0.7797475 and the dash-dotted line has AUC 0.7727052. This means that the solid line provides, indeed, better classification, albeit barely.

The procedure used to calculate these values is detailed in Appendix Section A.2.

AUC in Bayesian inference

Although not immediately evident, the metric AUC has a particular feature which causes it to be analyzed in an interesting way under the Bayesian approach. Given the true values of the points, a model whose parameters are fixed yields a particular AUC. This can be alternatively viewed as, given the true values and a Statistical model, different sets of parameter values lead to different determined AUCs, which means that the AUC is a deterministic function of the parameters (under the true values and a model), or $AUC = AUC(\theta)$.

This becomes particularly interesting when looking at Monte Carlo integration, as in equation (2.13). In the adequate notation, set $AUC(\theta) = h(\theta)$. In terms of inference, $AUC(\theta)$ can be viewed as a transformed parameter, and its marginal posterior distribution can be obtained by Monte Carlo integration.

This becomes useful to put together its marginal posterior density. Since other methods provide point-wise estimates to their parameters, their single-valued AUC can be compared with the AUC (marginal) posterior of the proposed model under the Bayesian approach.

Chapter 3

Data analysis

The proposal from equation (2.5) and its advantageous qualities need to be verified with data. First, some scenarios may be simulated to check how it fares in real situations which may arise. Then, it needs to be tested on real data and compared with other existing methods. The AUC metric discussed in Section 2.5.4 will be useful for that.

In order to circumvent chain auto-correlation, a common trick is to discard a fixed amount of samples. All chains in this thesis used 500,000 iterations, retaining 1 in every 500, so that in the end, 1,000 draws of the posterior are available for Monte Carlo integration. Also, it is common to discard a part of the draws starting from the beginning of the chain, which is a part where the chain hasn't yet converged to the stationary distribution yet. This part is usually called burn-in period. In this thesis, the burn-in period varied from 100,000 to 2,000,000 iterations.

Three independent chains are run to increase confidence in the MCMC convergence. Convergence is declared when the three chains intertwine in the same region.

3.1. SIMULATED DATA

Simulated data is useful to create situations to test how the model fares. It is also readily available and only has missing or misinput data when so desired. Appendix Section B.1 presents the code for simulation of the data.

The simulations used different values for β and δ . In all cases, λ^* was set to 1000. Unless noted otherwise, covariates are drawn at each location following independent standard Normal distributions. This is due to the fact that, since all spatial dependence is assumed to be in the covariates, then knowing the covariates value is enough for the model. Nevertheless, a case where there is spatial correlation between the covariates is tested in Section 3.1.2.

3.1.1. Estimation under correct specification of the model

A few cases were tested to see if the model can estimate itself. The simplest model has only 1 covariate in the intensity set $Z(\cdot)$ and in $W(\cdot)$, so that each β and δ has 2 parameters, including the intercepts.

In total, four models were simulated, each with different values for β and δ . In all of them, λ^* was set at 1000. The chosen values are shown in Table 3.1.

Model	β_0	β_1	δ_0	δ_1
1	-1	2	-3	4
2	-1	2	3	4
3	2	2	-3	4
4	2	2	3	4

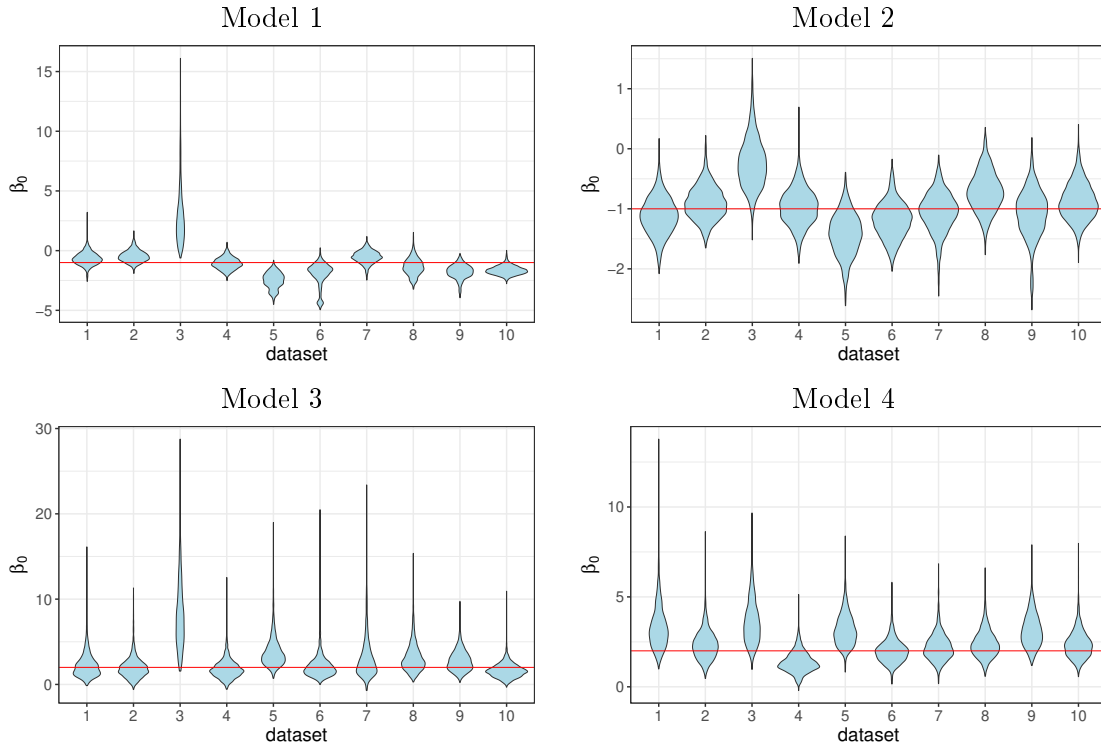
Table 3.1: Chosen values for the parameters to simulate data

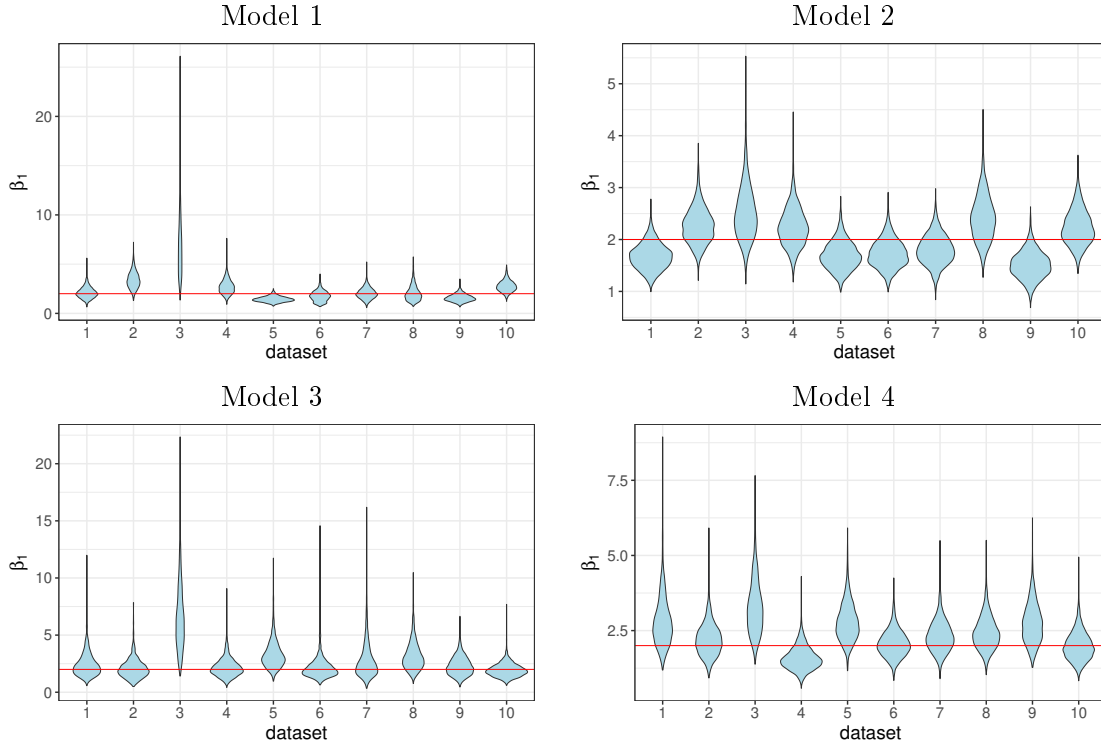
Each model was simulated 10 times. Only the retained Presence-Only points and their respective covariates values were used for the estimation. Table 3.2 shows how many observed points were present in each

		Dataset										Average
		1	2	3	4	5	6	7	8	9	10	
Model	1	73	83	73	86	81	92	82	107	89	85	85.1
	2	260	257	230	237	254	264	263	288	252	287	259.2
	3	178	189	186	213	180	202	197	210	205	193	195.3
	4	586	583	605	626	566	635	588	618	642	571	602

Table 3.2: Number of Presence Only points in each dataset drawn from each simulation

Figures 3.1, 3.2, 3.3, 3.4 and 3.5 show the violin plots of the marginal posterior, for each dataset, of the different models, respectively for β_0 , β_1 , δ_0 , δ_1 and λ^* . The violin plot has been chosen since it provides more information than the boxplot.

Figure 3.1: Estimation of β_0 for the 4 models

Figure 3.2: Estimation of β_1 for the 4 models

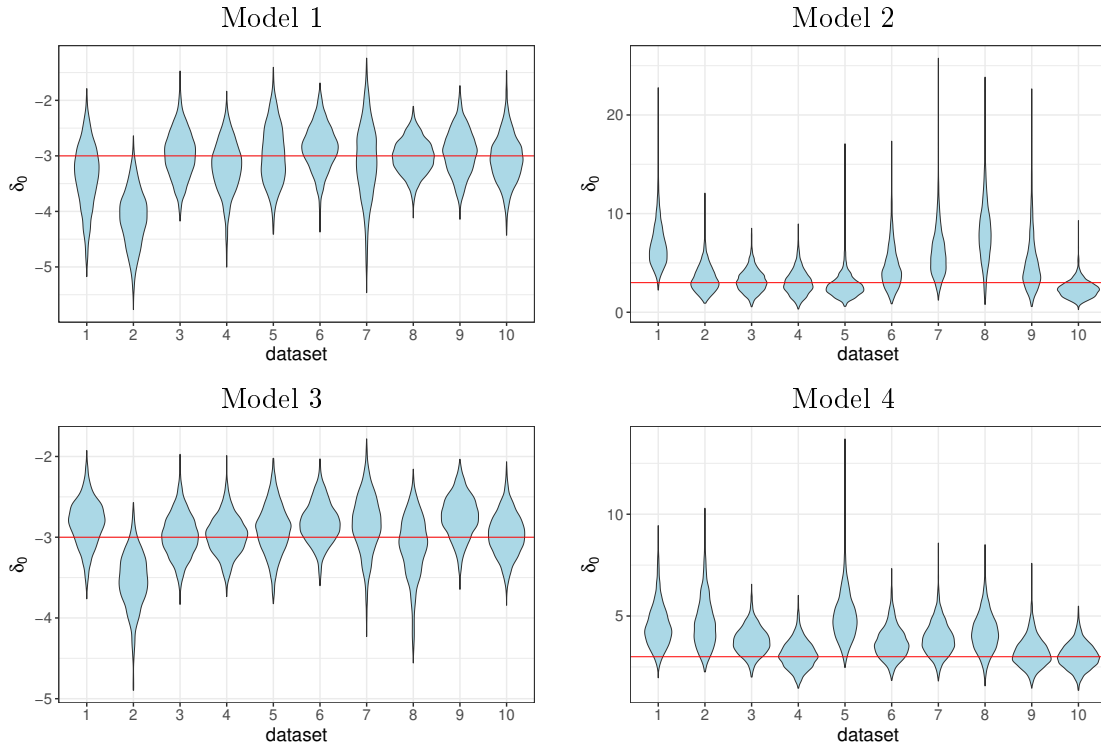
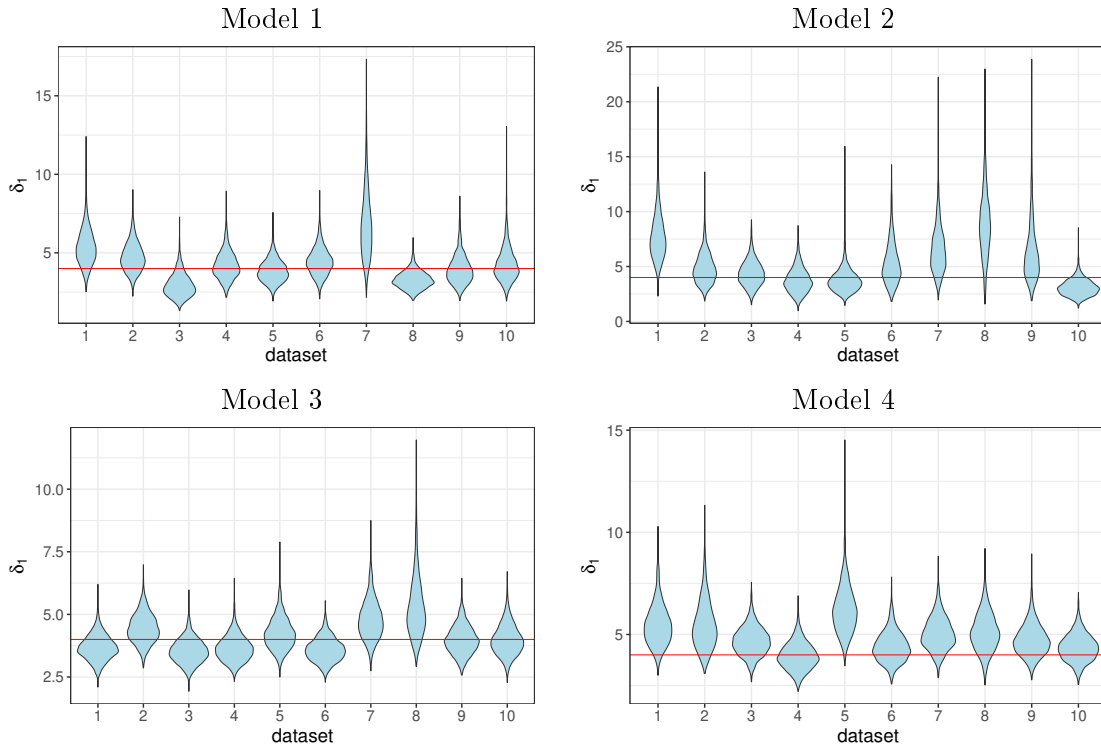
Some interesting things can be noted from these results. Firstly, the parameter λ^* seems very unstable for model 1, and that is mainly associated with the low point count of the datasets. The other models managed to estimate this parameter much better, meaning that a certain amount of points is needed for the proper estimation of λ^* .

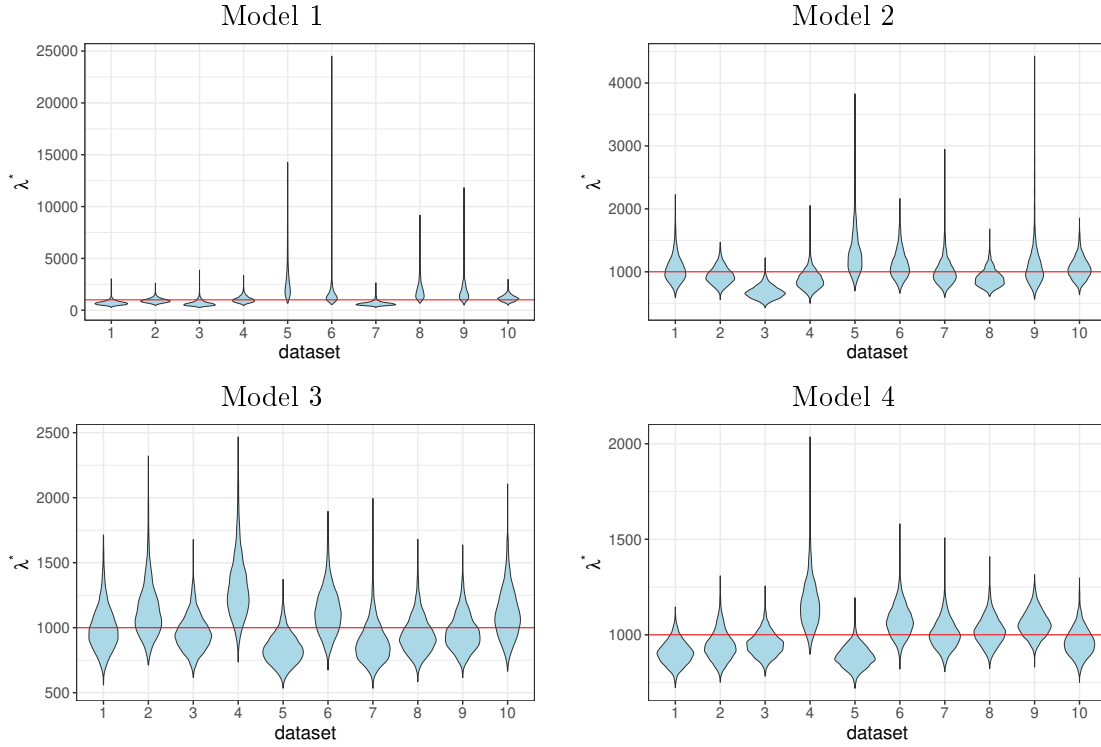
Also, model 2 seems to estimate the β vector very well, while models 1 and 3 seem to have very heavy tails on the posterior. Model 4 also has longer tails, yet not as much as 1 and 3. This situation is different for vector δ . Curiously, the only difference between these models is the sign of the intercept. This could give the indication that a negative true value of the intercept helps the estimation of that vector. The exception is model 1 for δ_1 , but the low point count could be the culprit again.

Generally, the model is capable of estimating the parameters, and it seems to does so consistently. A certain amount of points in the observations is necessary for the proper estimation of λ^* . Although the marginal posterior mode is often close to the true value of the parameter, the distribution can be heavily tailed to the right, that is, to larger values.

The joint posterior also yields interesting graphics in terms of two-by-two interaction. However, there would be 400 graphics to look at, since it's 4 models times 10 datasets times 10 two-by-two interactions. In order to choose which graphics to look more closely, correlations become interesting. Figure 3.6 shows a graphic for each interaction, separated by model. Each point is the correlation between the two parameters in a given dataset.

Some interesting correlations come into view. In particular, $\beta_0 \times \beta_1$, $\beta_0 \times \lambda^*$, $\beta_1 \times \lambda^*$ and

Figure 3.3: Estimation of δ_0 for the 4 modelsFigure 3.4: Estimation of δ_1 for the 4 models

Figure 3.5: Estimation of λ^* for the 4 models

$\delta_0 \times \delta_1$. The latter seems especially odd, as it seems that the sign of δ_0 determines the sign of the correlation. For parsimony reasons, not all datasets will be shown for each interaction. The pairs $\beta_0 \times \beta_1$ and $\delta_0 \times \delta_1$ display two graphics, whilst $\beta_0 \times \lambda^*$ and $\beta_1 \times \lambda^*$ display only one graphic with the highest absolute correlation. The pair $\beta_0 \times \beta_1$ has been chosen with the smallest and largest absolute correlation and the pair $\delta_0 \times \delta_1$ with the largest absolute correlation with opposite signs. They are displayed in Figure 3.7.

The first thing to note is that β_0 and λ^* have a relationship that resembles the one in Figure 2.1, so it is not unexpected. The relationship between β_1 and λ^* is probably only caused by the correlation between β_0 and β_1 .

However, it is not clear when β_0 and β_1 have high correlation, but it did happen often, particularly for model 4, where they are almost not identifiable. That also happened with δ_0 and δ_1 , but only when δ_0 was positive. When it was negative, then there was correlation too, only negative and not so strong.

Another way to test the model under correct specification is to put more covariates and see if the model can estimate them too. The same procedure as before was performed using 7 covariates in both the intensity set $Z(\cdot)$ and observability set $W(\cdot)$. The total number of points of all datasets are seen in Table 3.3.

The marginal posterior for the 5 parameters can be seen in Figure 3.8. The true values (used to simulate the data) are evidenced by the horizontal red line.

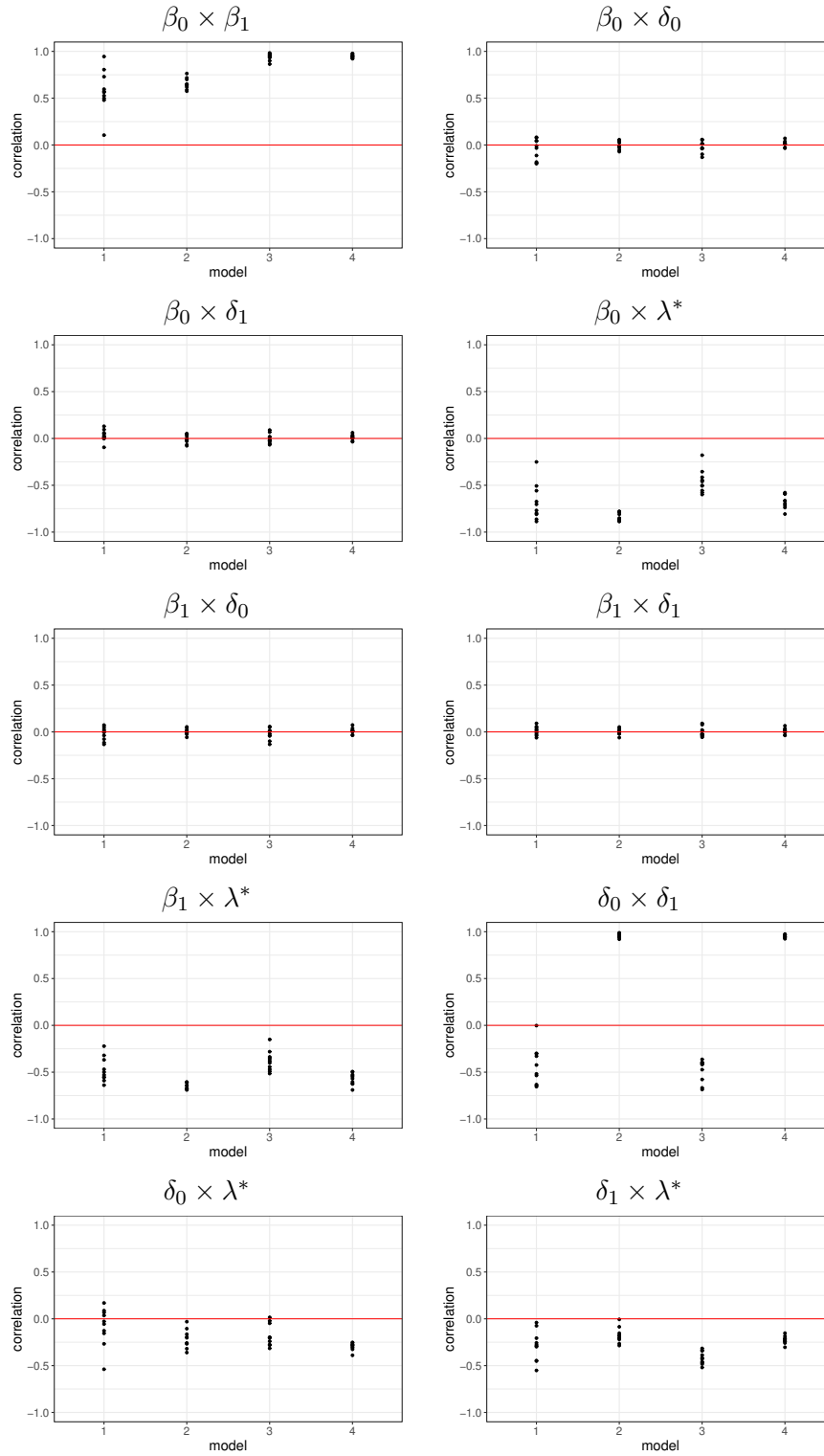


Figure 3.6: Correlations

Again, the model is capable of estimating the parameters, however maintaining the heavy tails

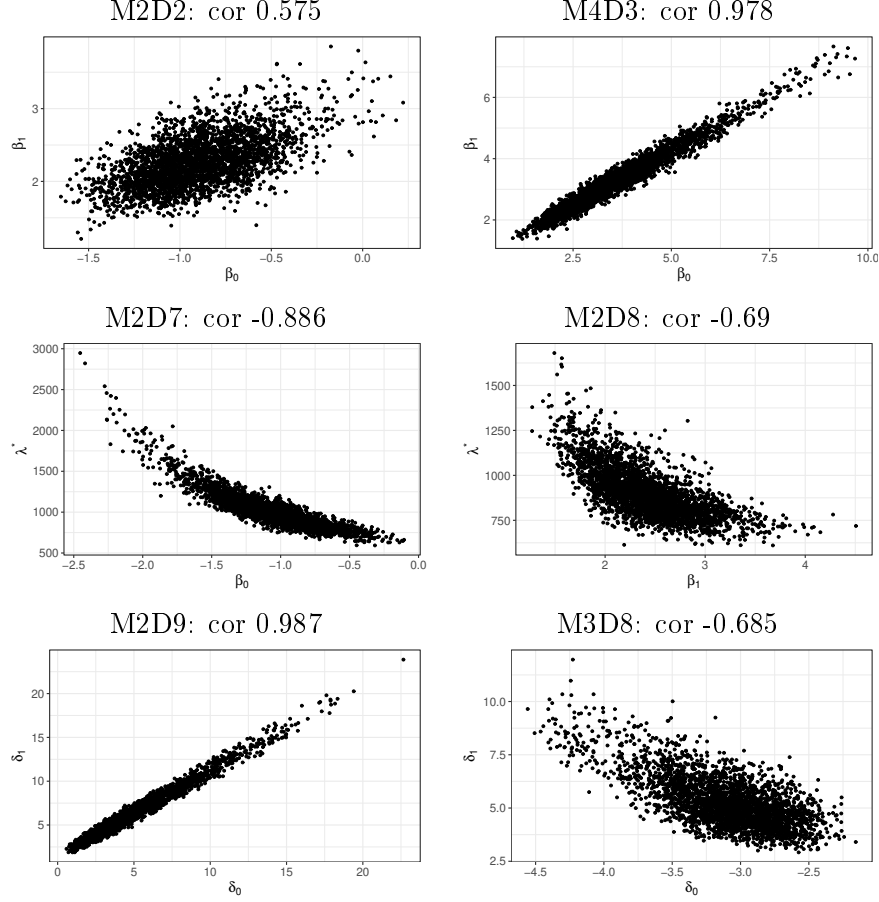


Figure 3.7: Bi-variate plots. MaDb means model a, dataset b. Parameters are seen in the axis labels

Dataset										Average
1	2	3	4	5	6	7	8	9	10	
241	259	258	240	246	225	262	245	234	226	243.6

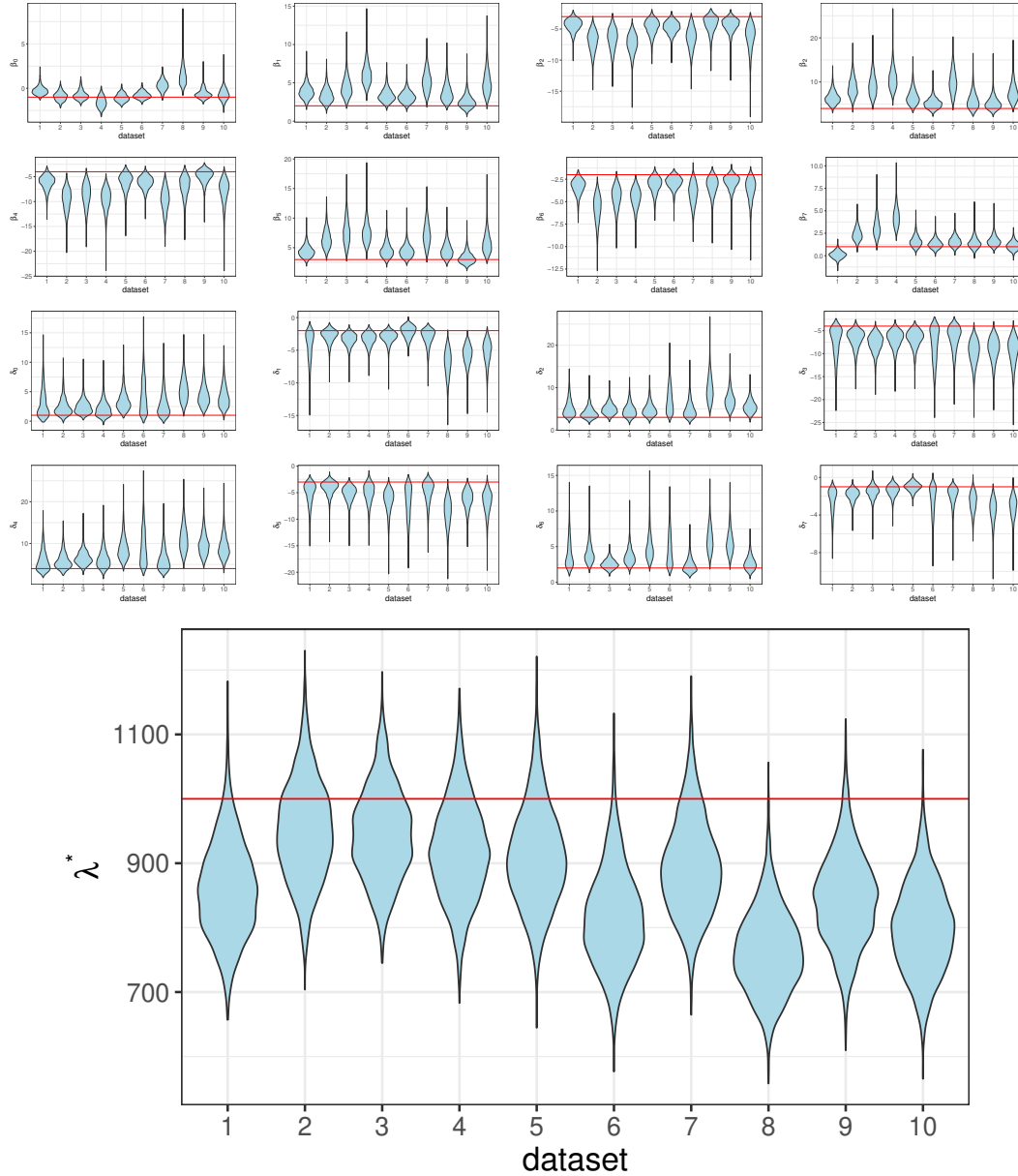
Table 3.3: Number of Presence Only points in each dataset drawn from each simulation

in the posterior. The estimation of λ^* was also adequate, however slightly underestimating it.

It is also important to simulate a model with a covariate that has spatial correlation. The next exercise does so. The intensity and observability sets only have one covariate each, and they are drawn no longer independently from the standard Normal distribution, but from two independent Gaussian Processes (Cressie, 1993). The chosen correlation kernel, denoted as $\rho(\cdot)$, was the Gaussian one:

$$\rho(d) = \exp \left\{ -\frac{d^2}{\phi} \right\} \quad (3.1)$$

To estimate this model, step 1 from Algorithm 3 would require calculating the distances matrix of the proposed points between themselves and between the observations. This would be too computationally intensive as it must happen every iteration of the MCMC, so instead, a fine

Figure 3.8: Estimation of β and δ for the model with more covariates

regular lattice is generated when the data is simulated and the respective covariates are stored in a file on the computer drive.

The estimating model will then read the file whenever it needs to retrieve covariate values. The number of points in the lattice is chosen depending on the known parameter ϕ in equation (3.1), so as to guarantee that any point in \mathcal{D} has correlation of at least 0.99 with its closest point in the lattice. This closest point is used as a surrogate for generating the covariates every iteration.

Until now, showing the generated observations would have been meaningless, since there is no spatial structure in the points. However, with spatially dependent data, the spatial pattern can

be graphically observed.

The last exercise for estimating the model under correct specification is to test the identifiability issue mentioned in Section 2.2. In particular, each covariate set $Z(\cdot)$ and $W(\cdot)$ has two covariates, however, one of them is the same for both sets. This means these covariates have correlation 1 and according to Fithian and Hastie (2013), Fithian et al. (2015) and Dorazio (2014), their respective effects are not identifiable. To make it harder for the model, the components of β and δ relative to these covariates have the same sign, but different values in the generating model. Results can be found in Figure 3.9.

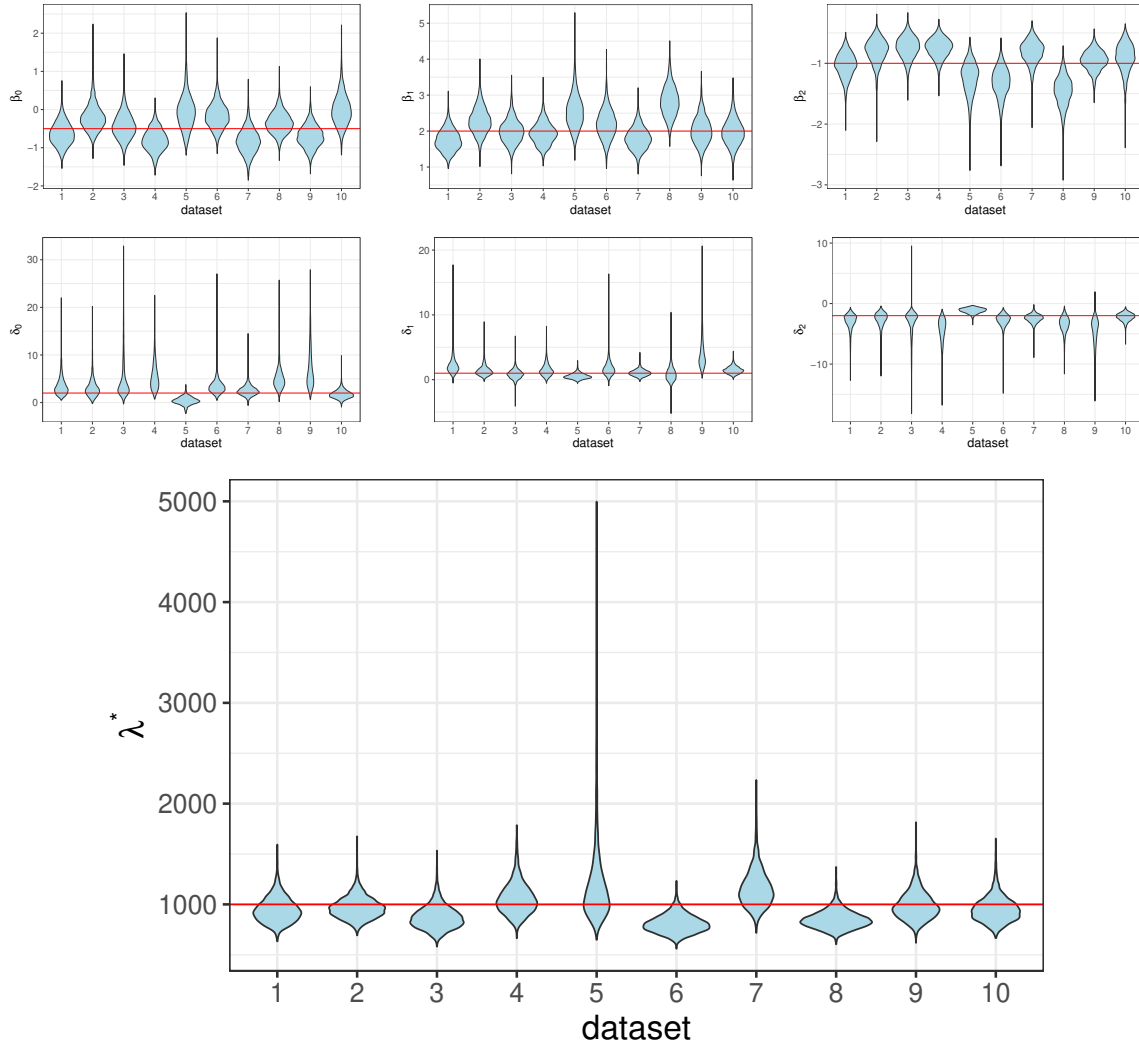


Figure 3.9: Correlations

The covariates relative to β_2 and δ_2 are the same. However, the model correctly estimated these parameters. This shows that the β and δ identifiability issue doesn't affect the proposed model.

3.1.2. Estimation under misspecification of the model

By remembering that the chosen model is usually not a perfect representation of the underlying generating process in real data, the model needs to be put through some tests of misspecification to check its stability in different situations.

For example, a covariate that has no effect on the data may be used in the estimation, or an important covariate may be not included. As a middle point of these situations, a covariate may be included but it is not as strong for explaining the data.

The link function chosen in equation (2.4) may be not ideal, or worse yet, the best model could be the traditional log-linear model in equation (1.30). Finally, Hastie and Fithian (2013) provide a simulation study which shows evidence to the fact that the MaxLike solution is very dependent on the true relationship between the data and the covariates to be logit-linear. Some deviation from this breaks the effectiveness of the model.

All these situations are tested in this Section. The first one is found by setting one of the generating coordinates of β to zero. Its results are seen in Figure 3.10.

The model has correctly estimated the parameter as 0.

The second situation is when an important covariate is not included in the covariates set. The chosen set for this simulation was the intensity set.

To test a less than ideal covariate to be a part of the model, a proxy covariate is generated with decreasing correlation with the correct one.

Next, the simulated dataset come from the traditional log-linear model. Given that this model is a limiting result of the proposed model, as shown in Section 2.2, the expected behaviour of the Markov Chain is to increase λ^* and decrease at least one of the intercepts without bound. Finally, the same exercise of Hastie and Fithian (2013) is applied to the generated data and the estimation is supposed to correctly estimate the effects.

3.2. REAL DATA

The model has performed as expected under different simulated circumstances in Section 3.1. It must still be put against other models commented in Section 2.1.2 to be considered competitive as a Presence-Only data method. This comparison must be done in real data.

Despite the abundance of PO datasets, there aren't as many available which also include readily usable observability covariates. Two datasets are used. The first is the same one in Renner et al. (2015). It includes two observability covariates. An important advantage of this dataset is the existence of presence-absence data, which can be used to evaluate model accuracy as described in Section 2.4.2 and model comparison by AUC as described in Section 2.5.4.

The second is a dataset of Brazilian data which has been analyzed by Mazzochini et al. (2019). This dataset is different, as the scale of the analysis is much bigger, continent-wise. This dataset does not have presence-absence data to match, so model accuracy cannot be checked and compared.

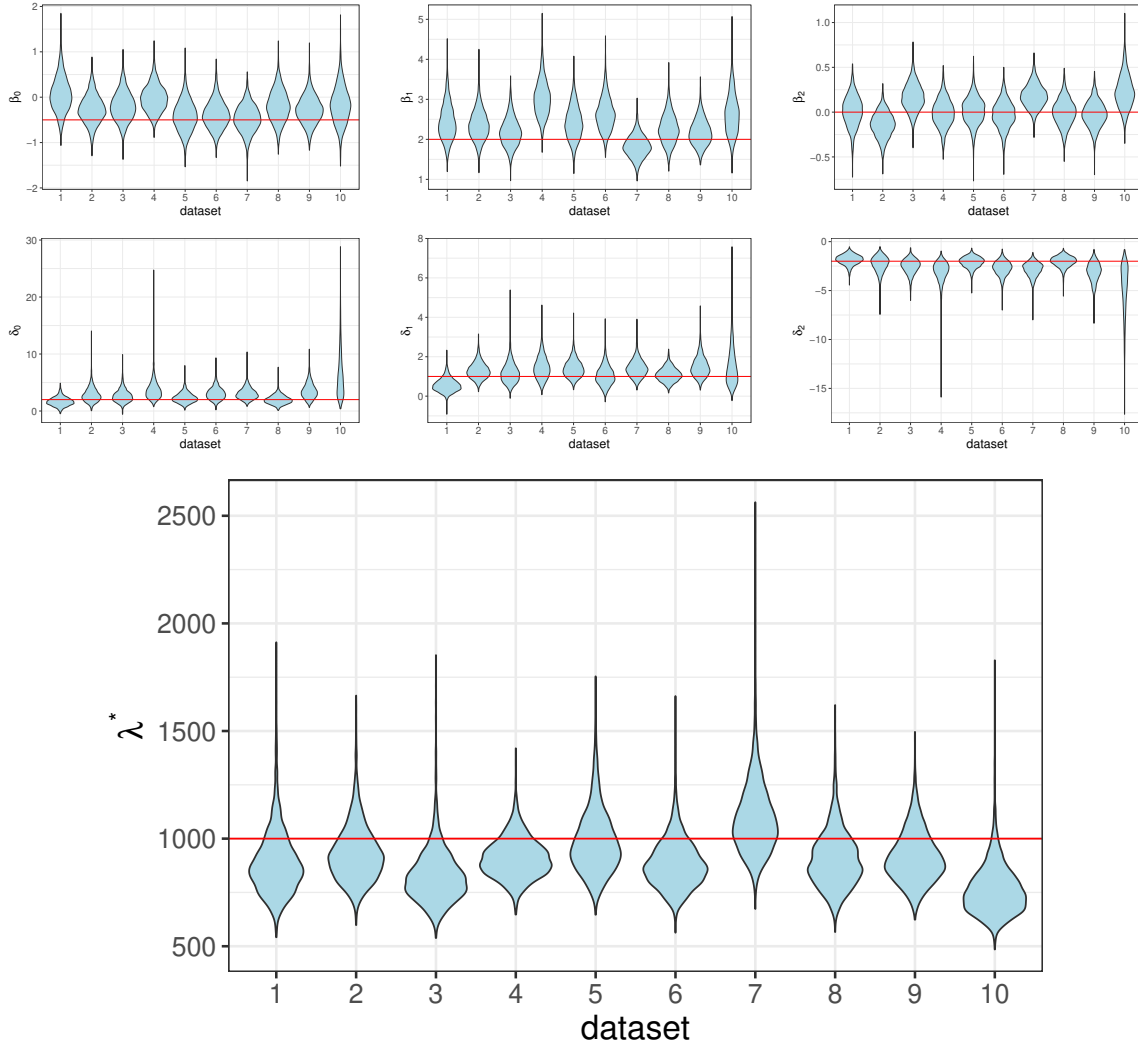


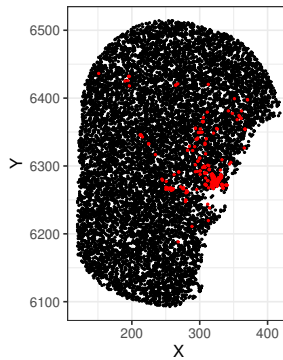
Figure 3.10: Correlations

The MCMC for the real datasets had 2,000,000 iterations for the burn-in period and 500,000 iterations for the posterior sample, using only 1 in every 500 iterations. In the end, the posterior had a sample size of 1,000.

3.2.1. Eucalyptus sparsifolia

This dataset, also used in Renner et al. (2015), models the tree *Eucalyptus sparsifolia* in the Greater Blue Mountains World Heritage Area near Sydney, Australia, and a surrounding 100 km buffer zone. Figure 3.11 shows a specimen of this tree.

The presence-only dataset is composed of 230 location points and 10,000 background points spread through the region. Each point has in average a 16 km² area. The data and background points can be seen in Figure 3.12. The presence points were collected from incidental sightings since 1972.

Figure 3.11: *Eucalyptus sparsifolia*Figure 3.12: Presence points of *Eucalyptus sparsifolia* can be seen in red. Black points are background where covariates are available

From Figure 3.12 it is possible to see that most observations happen near the Sydney bay area. Some covariates have been deemed important for the occurrence of the tree and a others, for the observation of occurrences.

The intensity covariates were number of fires since 1943, minimum and maximum annual temperature, annual rainfall, and a categorical soil variable. Observability covariates were distance to main roads and distance to urban areas.

For the analysis, it is common to use quadratic and interaction effects, as well as linear effects Phillips et al. (2006). This has been done in the model, for both covariates sets. Also, the 5 observability covariates (linear, quadratic and interaction effects) have *not* been included in the intensity set. Although the model should be able to estimate them, the computation time becomes much larger in this case.

It took a long time for the MCMC to run the model as it was, over a month of calculations for each chain on an Intel® Core™ i5-7400 CPU @ 3.00GHz \times 4 with 8 GB memory computer running Linux Ubuntu 18.04. After this time, the results are presented below.

All covariates were standardized with respect to the background points average and standard deviations safe for the dummy variables. Also, the prior for the parameters β and δ have been modified so that each individual effect had prior variance equal to 10. The reason for this is that there is a prior probability of 0.997 that the effect will lie between -3 and 3. The standardized covariate, which will usually vary between -3 and 3, will then cause a variation of the logistic function of at most 0.999, roughly all there is to vary in this scale. This is done to help the estimation.

The parameters and effects they represent are presented in Table 3.4.

β_1 Fire ²	β_2 Fire	β_3 MinT ²	β_4 MinT	β_5 MaxT ²
β_6 MaxT	β_7 Rain ²	β_8 Rain	β_9 Fire x MinT	β_{10} MinT x MaxT
β_{11} MaxT x Rain	β_{12} Fire x MaxT	β_{13} MinT x Rain	β_{14} Fire x Rain	β_{15} Soil 1
β_{16} Soil 2	β_{17} Soil 3	β_{18} Soil 4	β_{19} Soil 5	
δ_1 D.MainRoads ²	δ_2 D.MainRoads	δ_3 D.Urban ²	δ_4 D.Urban	δ_5 D.MainRoads x D.Urban

Table 3.4: Parameter encoding of the covariates effects

Figure 3.13 shows the estimation of β .

Figure 3.14 shows the estimation of δ .

Figure 3.15 shows the estimation of λ^* .

There are a few other interesting results that are worth mentioning. First is the relationship between the intercepts β_0 and δ_0 . Figure 3.16 illustrates this. At first glance, the bivariate plot of the posterior seems to indicate that they are not identifiable. The density of their sum, however, indicates that this is not completely true, as the density seems to be multimodal, giving the impression that they are nor completely unidentifiable, despite being heavily correlated.

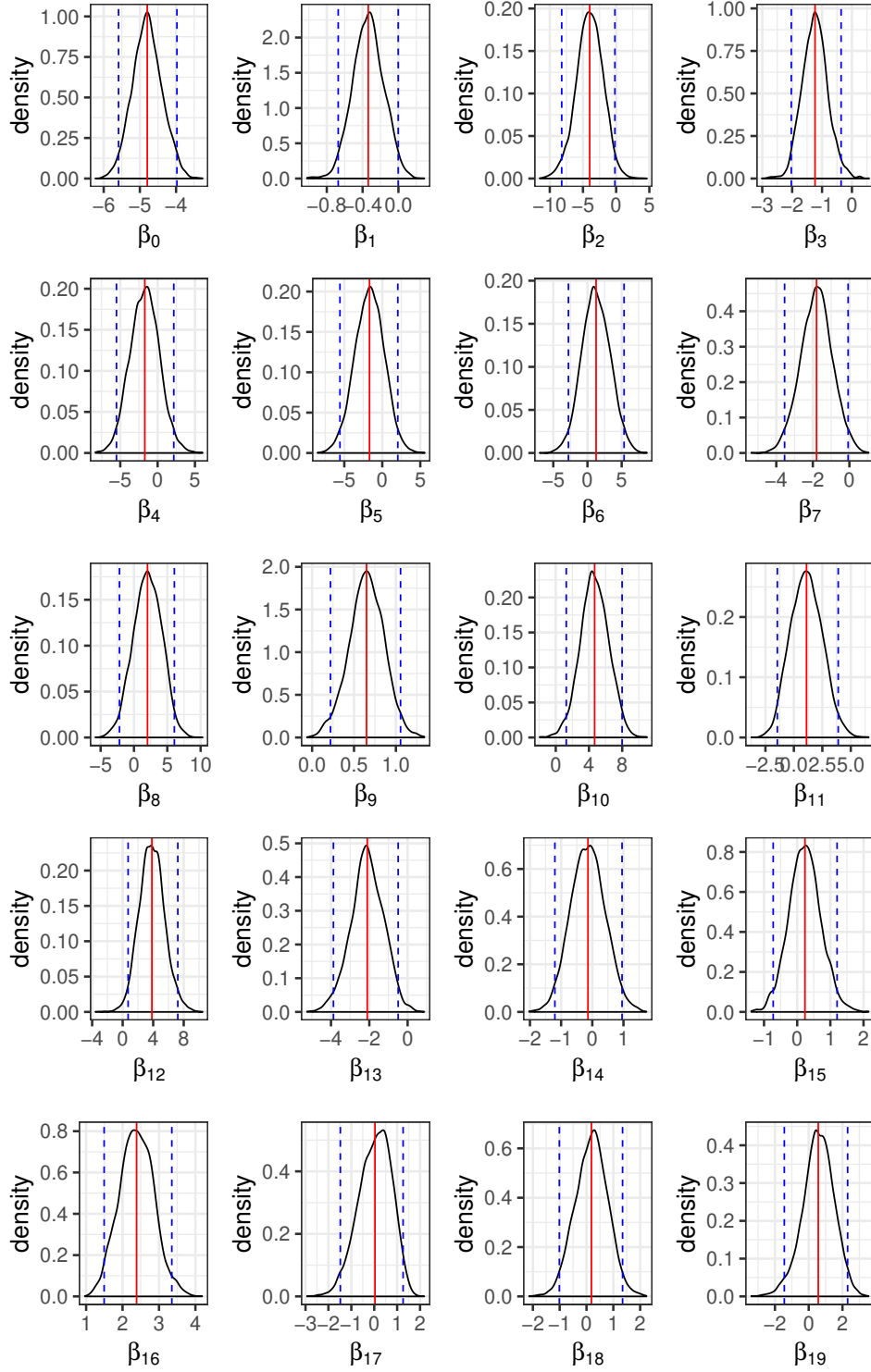


Figure 3.13: Estimation of β for the *Eucalyptus sparsifolia* data. Red solid lines are the posterior mean and blue dashed lines mark 0.95 intervals

Model comparison

Comparisons will be done with MaxLike and a log-linear IPP using the method described in 2.5.2. No comparison with MaxEnt is necessary since it is mathematically equivalent to the

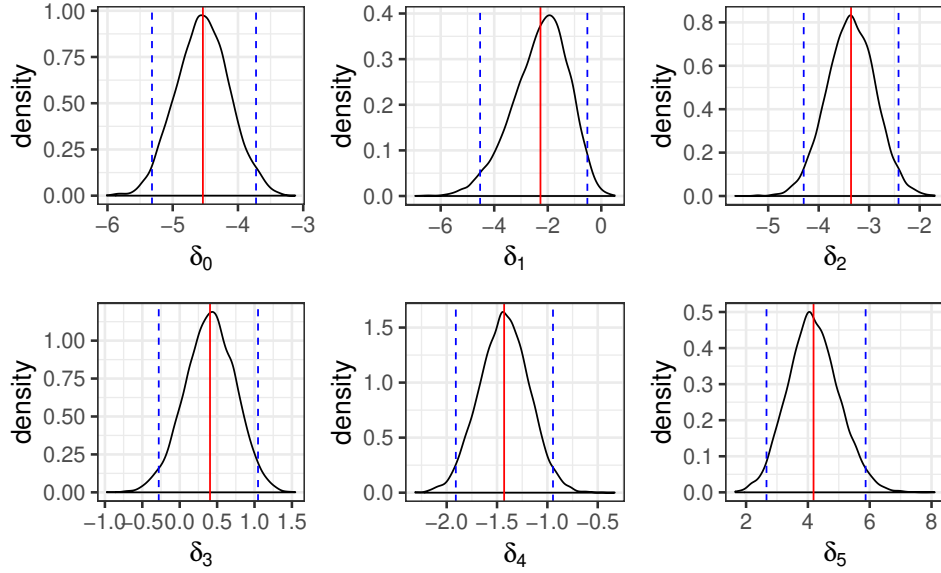


Figure 3.14: Estimation of δ for the *Eucalyptus sparsifolia* data. Red solid lines are the posterior mean and blue dashed lines mark a 0.95 interval

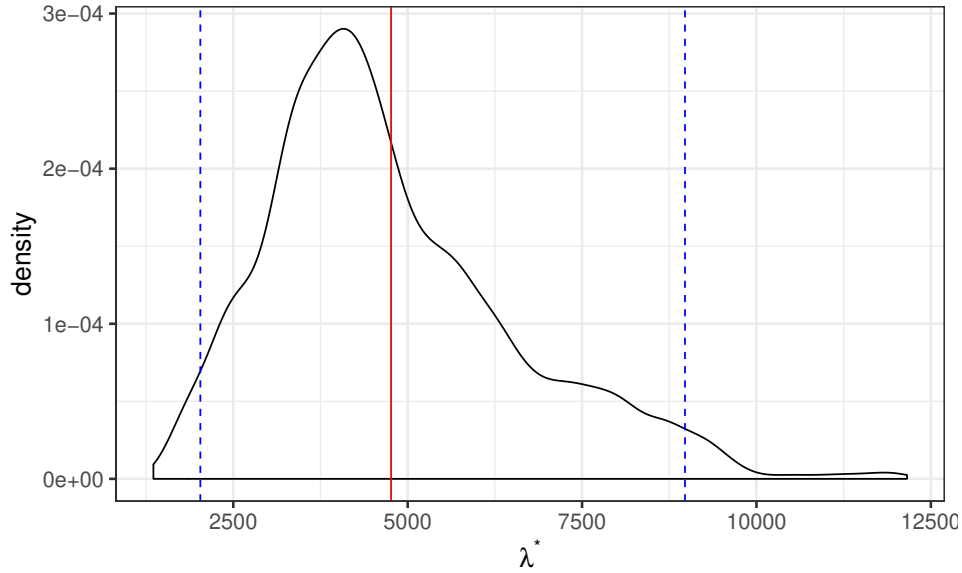


Figure 3.15: Estimation of λ^* for the *Eucalyptus sparsifolia* data. Red solid line is the posterior mean and blue dashed lines mark a 0.95 interval

latter. The comparison can be done either via ROC curve or AUC, whereas the latter is more definitive. The ROC curves in Figure 3.17 were created using the package `maxlike` and the code provided by Renner et al. (2015) (Supplementary material). The curve for the proposed model was created using the average value for each threshold.

For the AUC, as discussed in Section 2.5.4, a posterior sample is obtained and compared with the AUC of the competing methods. The result is presented in Figure 3.18.

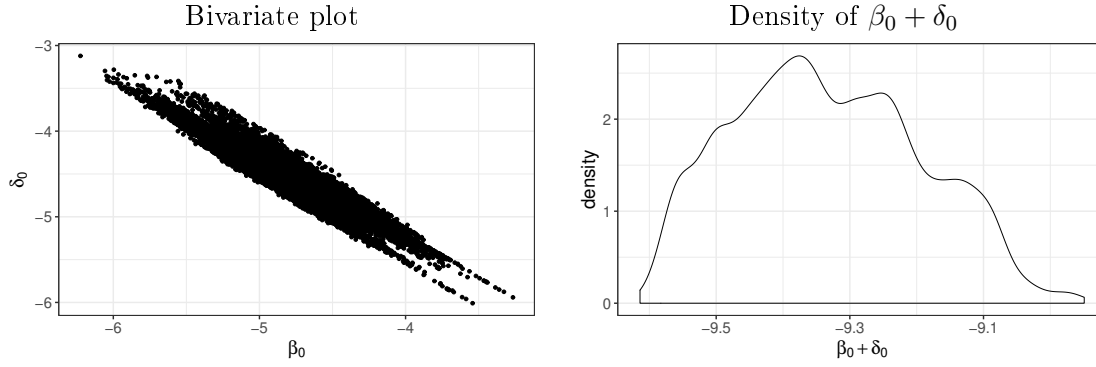
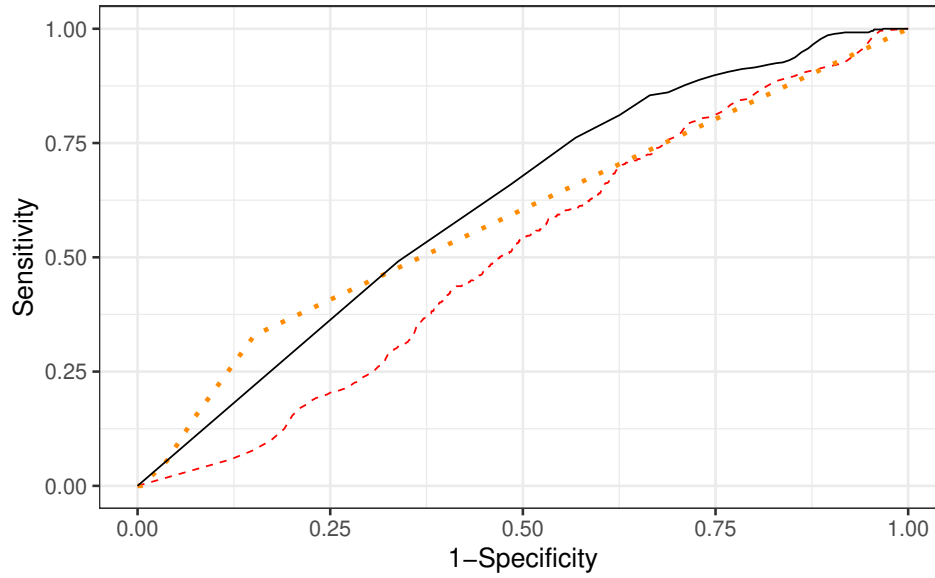
Figure 3.16: Relationship between β_0 and δ_0 

Figure 3.17: ROC curve comparison between models - black solid curve represents the proposed model, red dashed curve represents maxlike and the orange dotted curve represents the traditional log-linear IPP/MaxEnt

As Figures 3.17 and 3.18 show, the proposed methodology achieves very similar results to the traditional IPP, while being vastly superior to the MaxLike method. Figure 3.18 in particular is interesting since the AUC achieved using the posterior mean as a fixed value for the parameters is almost to the right-hand side tail of the marginal posterior of the AUC, the 82th percentile to be exact. This value of 0.687 is just over 0.01 different from the value from the traditional IPP, 0.698.

Since the uncertainty of the AUC for the traditional IPP isn't available, it's hard to state which is better. A clear advantage of the traditional IPP over the proposal at its current state is computation time. The traditional IPP takes only a few seconds to be fitted, which is in another order of magnitude when compared to the weeks long time it took for the proposal.

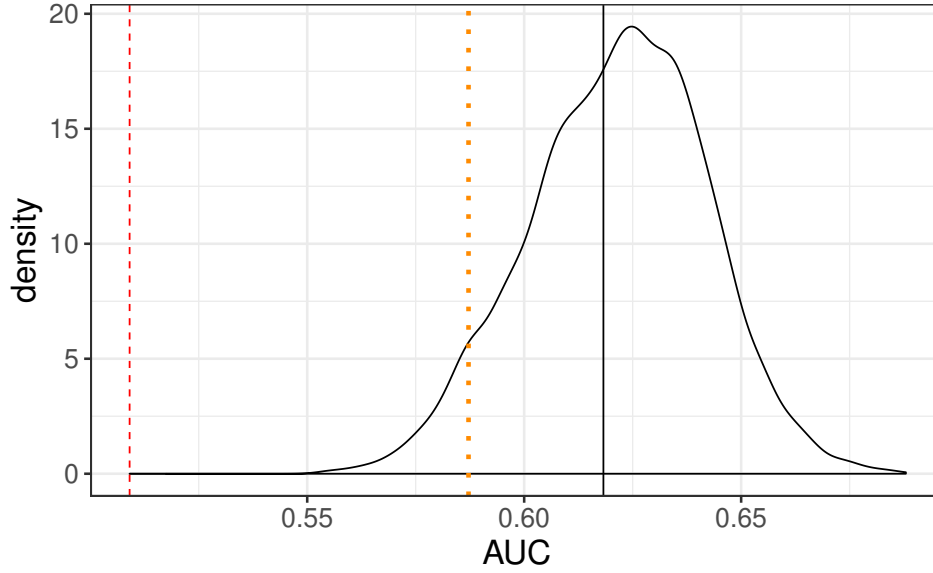


Figure 3.18: AUC comparison between models - black solid density represents the proposed posterior AUC density while the vertical line represents the AUC using the posterior mean, red dashed vertical line represents maxlike and the orange dotted vertical line represents the traditional log-linear IPP/MaxEnt

However, a few issues are present in the traditional IPP. Firstly, the likelihood function needs to be approximated, as discussed in Section 1. Secondly, take notice of the parameter estimations seen in Table 3.5.

Fire ²	Fire	MinT ²	MinT	MaxT ²
-0.5	-5.1	-3.3	-51.8	-74.3
MaxT	Rain ²	Rain	Fire x MinT	MinT x MaxT
88.1	-17	69	0.4	45.6
MaxT x Rain	Fire x MaxT	MinT x Rain	Fire x Rain	Soil 1
-46.2	4.5	10.7	0.5	14.3
Soil 2	Soil 3	Soil 4	Soil 5	
16.2	14	14.2	14.3	
D.MainRoads ²	D.MainRoads	D.Urban ²	D.Urban	D.MainRoads x D.Urban
-2.6	-2.7	0.6	-1.7	3.8

Table 3.5: Maximum likelihood estimation of the traditional log-linear IPP

These values are unrealistic to be used in the exponential scale with standardized covariates. Looking at Maximum Temperature in particular in Figure 3.19, one can see that its value vary from -2 to 2. Multiplied by its estimated effect in the exponential scale, its contribution to the intensity function varies from 3×10^{-77} to 3.3×10^{76} . Those numbers should cast doubt on the estimation.

This discussion brings confidence to the proposed model since it brings equivalent prediction capabilities to the traditional models with the added advantages discussed throughout this thesis.

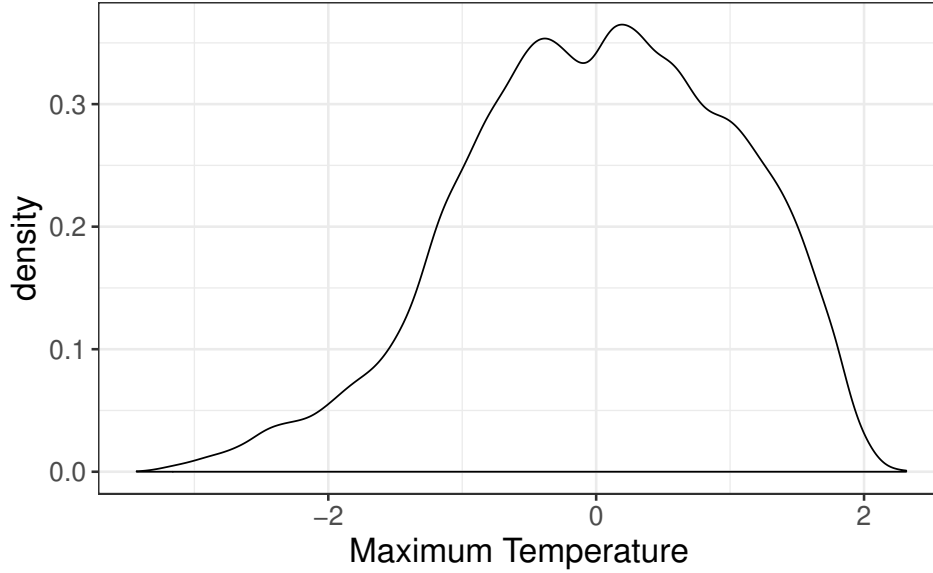


Figure 3.19: Standardized values of the maximum temperature

3.2.2. *Anadenanthera colubrina*

This dataset, analyzed in Mazzochini et al. (2019), is composed of Brazilian data. Many species of semi-arid areas are analyzed. In particular, the species *Anadenanthera colubrina* with 544 presence points is evaluated. There are two main differences between this dataset and the one studied in Section 3.2.1. The first difference is that there is no presence-absence data that can be used to evaluate AUC and thus, compare models. The second difference is that the scale of the region is much bigger. Each point in the region has 1 km² area. The data has a total of 23 million points.

The intensity variables are displayed were Altitude, Slope, Annual Mean Temperature, Mean Temperature of Warmest Quarter, Mean Temperature of Coldest Quarter, Annual Precipitation, Precipitation of Wettest Month, Precipitation of Driest Month, Precipitation Seasonality (Coefficient of Variation), Precipitation of Wettest Quarter, Precipitation of Driest Quarter, Precipitation of Warmest Quarter, Precipitation of Coldest Quarter, Mean Diurnal Range (Mean of monthly*(max temp - min temp)), Isothermality (Mean Diurnal Range/Temperature Annual Range) (* 100), Temperature Seasonality (standard deviation *100), Max Temperature of Warmest Month, Min Temperature of Coldest Month, Temperature Annual Range (BIO5-BIO6), Mean Temperature of Wettest Quarter, Mean Temperature of Driest Quarter.

The observability covariates were 'distance to closest research institute' and 'distance to closest semi-arid region'.

Since there are 21 intensity covariates, the combination of all quadratic and interaction effects would result in a total of 252 dimensions of β (253 including the intercept) to be estimated. For a 544 points dataset, there are just too many parameters, so only the linear effects are added in the intensity set. Furthermore, the linear effects of the observability covariates were also

included in the intensity set. The observability set had quadratic and interaction parameters since there are only two variables.

Table 3.6 maps the effects to their parameter names.

β_1 Alt	β_2 Slope	β_3 AnnMeanTemp	β_4 MeanTempWarm	β_5 MeanTempCold
β_6 AnnPrecip	β_7 PrecipWetMo	β_8 PrecipDryMo	β_9 PrecipSeason	β_{10} PrecipWetQua
β_{11} PrecipDryQua	β_{12} PrecipWarm	β_{13} PrecipCold	β_{14} MeanDiurnRange	β_{15} Isothem
β_{16} TempSeason	β_{17} MaxTempWarmMo	β_{18} MinTempColdMo	β_{19} TempAnnualRange	β_{20} MeanTempWetQua
β_{21} MeanTempDryQua	β_{22} D.Inst	β_{23} D.Arid		
δ_1 D.Inst ²	δ_2 D.Inst	δ_3 D.Arid ²	δ_4 D.Arid	δ_5 D.Inst x D.Arid

Table 3.6: Parameter encoding of the covariates effects

All estimations can be seen in Figure 3.20 and 3.21.

Many values of β are centered on zero, which suggests that these covariates do not impact the occurrence of the tree. Namely,

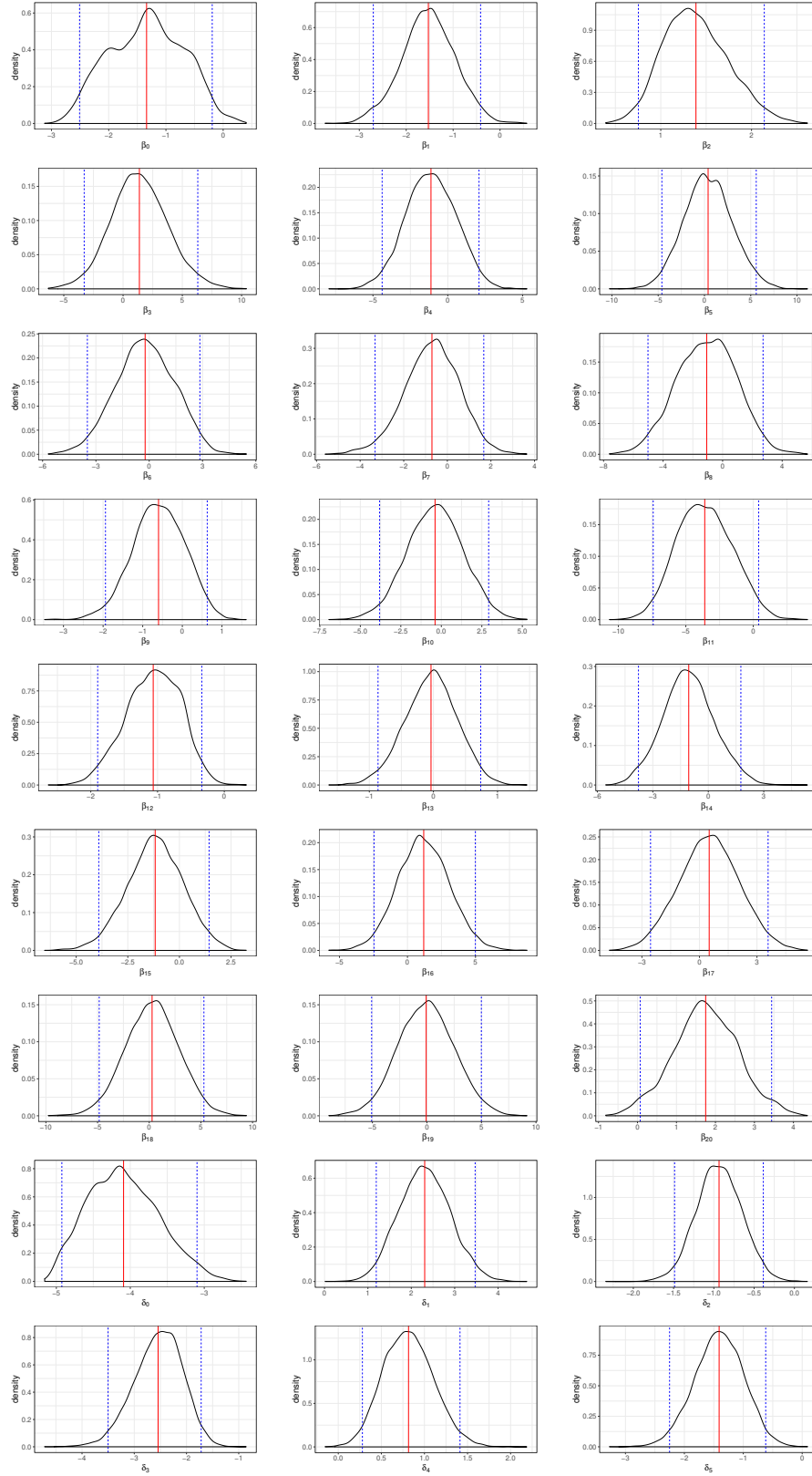
Following the analysis from Section 3.2.1, it is interesting to compare the intercepts β_0 and δ_0 . This is done in Figure 3.22.

Differently from Figure 3.16, the bivariate posterior distribution of β_0 and δ_0 does not resemble a line, being more curvy. This is interesting because the parameter $\beta_0 + \delta_0$ no longer necessarily needs to be a constant.

It is also interesting to analyze parameters β_{22} with δ_2 and β_{23} with δ_4 since they quantify the effects of the same covariates in both sets. The analysis can be seen in Figure ??.

It can be seen that there is practically no posterior relationship between these parameters, meaning that they are very identifiable. This feat cannot be achieved by the traditional IPP model.

The density of λ^* , in its turn, seems well defined if not for the apparent bimodality. Below are the estimations of the traditional IPP model.

Figure 3.20: Estimation of β and δ for the *Anadenanthera colubrina* data

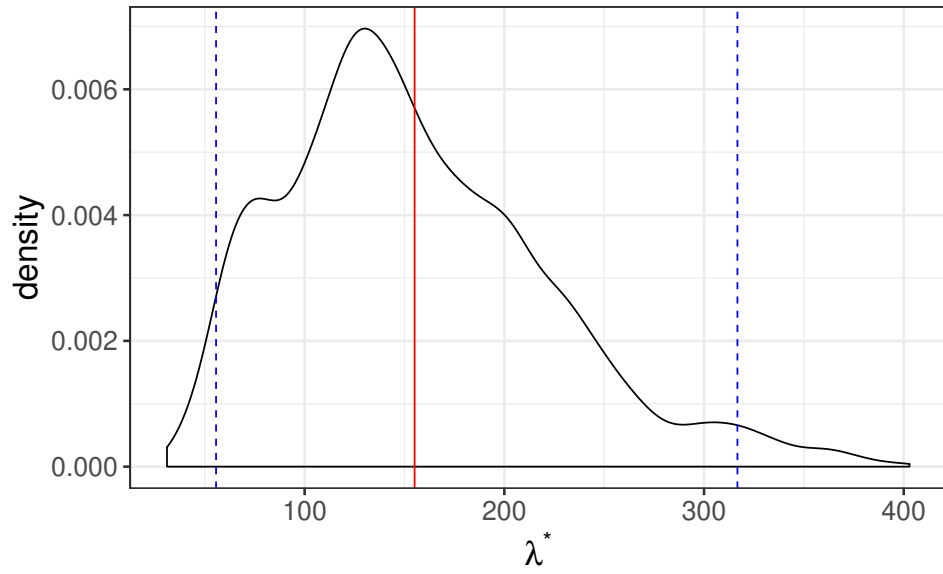


Figure 3.21: Estimation of λ^* for the *Anadenanthera colubrina* data

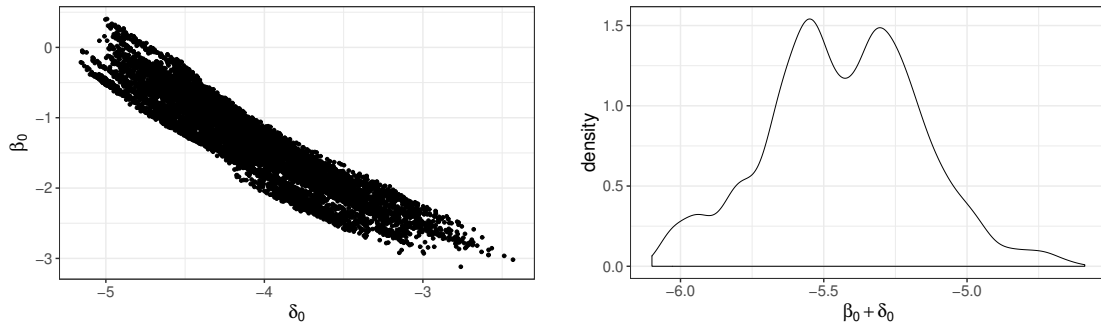


Figure 3.22: Comparing the marginal joint posterior of β_0 and δ_0 .

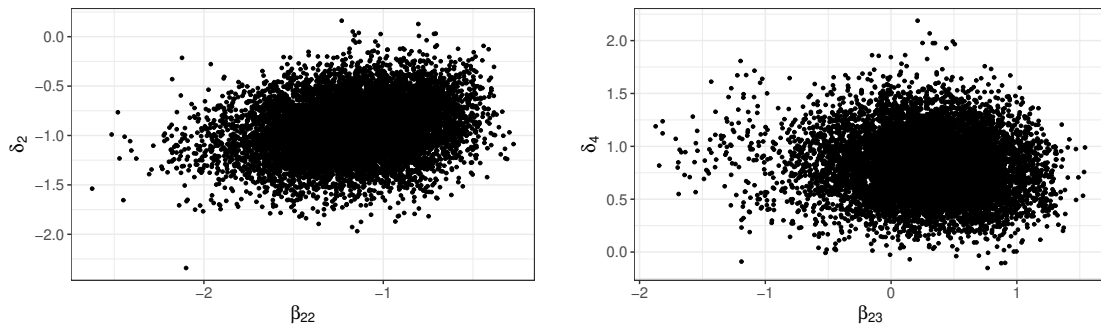


Figure 3.23: Comparing the marginal joint posterior of β_{22} with δ_2 and of β_{23} with δ_4 .

Chapter 4

Model extensions

A model for an application as wide as Presence-only data can clearly benefit from elaborations. A few have been explored in the literature. Since the proposal and the traditional IPP have been built over the same theory, the same extensions can be applied to both, and what has been developed for one can be used on the other.

An important extension comes from an issue brought up in Renner and Warton (2013). In it, they perform a needed validation analysis of a particular IPP model assumption, that the entirety of the spatial dependence on the data is mapped by the covariates. In the particular data (*Corymbia eximia*) where they test this, they verify that further spatial dependence is needed.

This is an important point as this assumption can not be tested in the MaxEnt method, and its invalidity is also applicable to MaxEnt given the mathematical equivalence between the methods (Fithian and Hastie, 2013). A practical way to address this is to include spatially dependent noise in the linear predictors $Z(\cdot)\beta$ and $W(\cdot)\delta$. This is done in Renner et al. (2015) by adding a Gaussian Process in the linear predictor. The same idea can be applied to the proposal of this thesis in the future. This is addressed in Section 4.2

Also, Fithian et al. (2015) propose an interesting way of adding information to the model. Their reasoning is that the processes of observability for different species should be similar in a given common region to some degree, since site accessibility should roughly be the same. Then, they propose to use data for a species that has both presence-only (PO) and presence-absence (PA) to estimate region observability by using the PA data to estimate species intensity without bias and then the PO data to use that estimation to have an idea of observability. Then, for a species which only has PO data, the observability would be already given.

However, on the particular dataset (different species of Eucalyptus) they use, they verified that the observability parameters δ is not the same across species. However they verify good AUC results when pooling various (36) species. This can be achieved in the proposal with varying commitments, by including prior correlation between the corresponding δ parameters of different species when they are modeled together. Renner et al. (2019) expand on this idea by checking spatial dependence and applying an ℓ_1 regularization to the maximum likelihood estimation in order to address overfitting.

A discussion is raised by Gelfand and Shirota (2018) with respect of the difference between PA and PO is unprecedented. They argue that in PA data, the locations are fixed and what is random is the occurrence or not of the species whilst in the PO data, the locations themselves are random. This brings a problem to comparing methods and datasets of the two different kinds.

Another interesting idea in ecology is species interaction. It is known that many species interact in the same region, be it by attraction or repulsion. A very good way to do this would be to use one IPP as a covariate to another, which is covered in Section 4.1.

4.1. MULTIPLE SPECIES MODELING

Although the literature has only addressed multiple species modeling by attempting to pool the information towards the observability thinning parameters, there has yet to be a model for species interaction.

One possibility considers the occurrence of one IPP to be covariates for the another. This model is presented in equation (4.1). Let X be the IPP for the observed occurrences of a species, X' the unobserved occurrences and U_X its latent completion process. At the same time, let Y be the observed occurrences of the other species and, equivalently, Y' and U_Y are the unobserved occurrences and the completion of the second species. Then

$$\begin{aligned}
\mathcal{X} &\sim PP(\lambda_X^*) \\
(X \cup X') &= \tau(\mathcal{X}, q_X(\cdot)) \\
\text{logit } q_X(s) &= Z_X(s)\beta_X \\
U_X &= \mathcal{X} \setminus (X \cup X') \\
X &= \tau((X \cup X'), p_X(\cdot)) \\
\text{logit } p_X(s) &= W_X(s)\delta_X \\
X' &= (X \cup X') \setminus X \\
\mathcal{Y} &\sim PP(\lambda_Y^*) \\
(Y \cup Y') &= \tau(\mathcal{Y}, q_Y(\cdot)) \\
\text{logit } q_Y(s) &= Z_Y(s)\beta_Y + d(s, (x \cup x'))\beta_{dist} \\
U_Y &= \mathcal{Y} \setminus (Y \cup Y') \\
Y &= \tau((Y \cup Y'), p_Y(\cdot)) \\
\text{logit } p_Y(s) &= W_Y(s)\delta_Y \\
Y' &= (Y \cup Y') \setminus Y,
\end{aligned} \tag{4.1}$$

where $d(\cdot, (X \cup X'))$ is some function of distance of a point to the process realization of $X \cup X'$. This creates an interaction between the processes. This also means that the MCMC procedure needs to be updated. Calling $pts_X = n_x + n_{x'} + n_{u_X}$ and $pts_Y = n_y + n_{y'} + n_{u_Y}$, the posterior density is

$$\begin{aligned}
\pi(x', u_X, \beta_X, \delta_X, \lambda_X^*, y', u_Y, \beta_Y, \beta_{dist}, \delta_Y, \lambda_Y^*) &\propto \frac{e^{-|\mathcal{D}|(\lambda_X^* + \lambda_Y^*)}}{n_{x'}! n_{u_X}! n_{y'}! n_{u_Y}!} \times \\
&\times \lambda_X^{*pts_X} \lambda_Y^{*pts_Y} \prod_{s \in x} q_X(s) p_X(s) \times \\
&\times \prod_{s \in x'} q_X(s) (1 - p_X(s)) \prod_{s \in u_X} (1 - q_X(s)) \times \\
&\times \prod_{s \in y} q_Y(s) p_Y(s) \times \\
&\times \prod_{s \in y'} q_Y(s) (1 - p_Y(s)) \prod_{s \in u_Y} (1 - q_Y(s)) \times \\
&\times \pi(\beta_X, \delta_X, \lambda_X^*, \beta_Y, \beta_{dist}, \delta_Y, \lambda_Y^*).
\end{aligned} \tag{4.2}$$

The MCMC procedure is similar to the one in Section 2.3.3. To make the procedure simpler, note that once the distances $d(\cdot, (X \cup X'))$ have been evaluated, they can be considered to be a simple covariates with effect β_{dist} . Under this approach, the only full conditional that changes is the one for X' , which can be seen in the equation below.

$$\pi(x'|\cdot) \propto \frac{1}{n_{x'}!} \prod_{s \in x'} q_X(s) (1 - p_X(s)) \lambda_X^* \prod_{s \in (y \cup y')} q_Y(s) \prod_{s \in u_Y} (1 - q_Y(s)). \tag{4.3}$$

If not for the last product, this would be a Poisson Process. Its inclusion makes it so it isn't, so a M-H step is used, as described in Section 2.3.2. The distribution used as a proposal $q(\cdot)$ is the prior, that is an IPP with intensity $q_X(\cdot)(1 - p(\cdot))\lambda_X^*$. The acceptance probability is the minimum between 1 and

$$\begin{aligned}
\frac{\pi(x'^{(prop)}|\cdot)q(x'^{(prev)})}{\pi(x'^{(prev)}|\cdot)q(x'^{(prop)})} &= \frac{\prod_{s \in (y \cup y')} \left(1 + e^{-Z(s)\beta_Y - d(s, (x \cup x'^{(prop)})}\beta_{dist}\right)^{-1}}{\prod_{s \in (y \cup y')} \left(1 + e^{-Z(s)\beta_Y - d(s, (x \cup x'^{(prev)})}\beta_{dist}\right)^{-1}} \times \\
&\times \frac{\prod_{s \in u_Y} \left(1 + e^{Z(s)\beta_Y + d(s, (x \cup x'^{(prop)})}\beta_{dist}\right)^{-1}}{\prod_{s \in u_Y} \left(1 + e^{Z(s)\beta_Y + d(s, (x \cup x'^{(prev)})}\beta_{dist}\right)^{-1}} = \\
&= \frac{\prod_{s \in (y \cup y')} \left(1 + e^{-Z(s)\beta_Y - d(s, (x \cup x'^{(prev)})}\beta_{dist}\right)}{\prod_{s \in (y \cup y')} \left(1 + e^{-Z(s)\beta_Y - d(s, (x \cup x'^{(prop)})}\beta_{dist}\right)} \times \\
&\times \frac{\prod_{s \in u_Y} \left(1 + e^{Z(s)\beta_Y + d(s, (x \cup x'^{(prev)})}\beta_{dist}\right)}{\prod_{s \in u_Y} \left(1 + e^{Z(s)\beta_Y + d(s, (x \cup x'^{(prop)})}\beta_{dist}\right)}.
\end{aligned} \tag{4.4}$$

In order to test this, a small simulation study has been done using $d(\cdot, (X \cup X')) = \min(\|s - (X \cup X')\|)$ where $\|\cdot\|$ represents the Euclidean distance. The results are shown below.

4.2. SPATIAL INTERACTION BEYOND THE COVARIATES

An important assumption of the model is that the covariates encodes all the spatial interaction between the points. However, Renner and Warton (2013) explore a dataset where this is not true. The work of Renner et al. (2015) has an addition to the model which includes a spatial Gaussian Process in the linear predictor to account for such occasion. They argue that the resulting Point process is no longer Poisson, despite being conditionally Poisson.

It is not complicated to augment the proposed model with such an extension. Let $V(s), s \in \mathcal{D}$ be the value of a Gaussian Process (GP) at point s . The GP is characterized by the fact that any finite-dimensional set of values in the studied region is a multivariate Normal random vector. It is customary to set the mean vector of this multivariate distribution to zero.

The spatial interaction is introduced by the correlation of the random vector. The simplest and most common form of this is to make the correlation between two points to not depend on where they are nor on the direction of the line that crosses them. The properties that refer to these are called, respectively, stationarity and isometry. A stationary and isometric GP has correlation between the points that only depend on the distance between the points. The most used distance is the Euclidean one.

Then let $V(s), s \in \mathcal{D}$ be a stationary and isometric GP with some correlation function $\rho(s, s') \equiv \rho(\|s - s'\|)$. Then the IPP for the presence-only data is presented below.

$$\begin{aligned}
\mathcal{Y} &\sim PP(\lambda^*) \\
Y &= \tau(\mathcal{Y}, q(\cdot)) \\
\text{logit } q(s) &= Z(s)\beta + V(s) \\
U &= \mathcal{Y} \setminus Y \\
X &= \tau(Y, p(\cdot)) \\
\text{logit } p(s) &= W(s)\delta \\
X' &= Y \setminus X.
\end{aligned} \tag{4.5}$$

Another GP can be equivalently added to $\text{logit } p(\cdot)$, describing a spatial interaction of the observability process. While this exercise has not yet been performed in this work, it is an important extension to provide a potentially more adequate model for the data.

In fact, the proposed model in equation (2.5) can be seen as a particular case of equation (4.5) where $\rho(\|s - s'\|) = 0$ and the variance of any multivariate Normal that comes from $V(\cdot)$ is zero.

Chapter 5

Conclusion

A model has been presented to address a presence-only data problem. Although other methods exist in the literature, they suffer from some problems. An identifiability issue prevents the use of all covariates which could be important to the data, and the likelihood function needs to be approximated, whereas the proposed model works around them.

Bayesian inference is not only used, but the solution to evaluating the likelihood function exactly relies on MCMC which is an important tool of Bayesian inference. This brings forth, at the same time, full information on estimation uncertainty through the posterior distribution.

A discussion raised by Gelfand and Shirota (2018) which can be applied to the use of a presence-only model to make predictions on presence-absence data is taken into account when comparing prediction capabilities of the proposed model with other methods.

The proposed methodology proved to be capable of everything it promises to do. The identifiability issue with regard to parameters of correlated covariates is diminished. The inference is done so that the only approximation done is Monte Carlo integration via MCMC.

As argued by Warton and Shepherd (2010), Fithian and Hastie (2013) and Dorazio (2014), the Point Process makes for an appropriate model for presence-only data. It treats the data as they are, that is, random number and locations for presence of the species. The thinned PoP is adequate since it accounts for biased observation of the presences, to which has been referred in this thesis as observability process. The data, as it is, also invalidates an important premise of the MaxLike method, that the presences are randomly selected to be observed with constant probability.

The effect of this invalidation can be verified by the model comparison in Section 3.2.1. The proposal and the traditional IPP yielded similar prediction metrics, with former being slightly better, however the latter provided unrealistic estimates for some parameters. This makes the proposal more attractive as a more rounded method.

Although showing evidence as an advancement in the modeling of presence-only data, there is much room to grow in terms of computation time. At the moment of writing, there was no alternative to provide faster convergence of the MCMC chain. Being the only current method to treat the completion latent process, MCMC takes a long time to run. A perhaps greater

problem is that the total time of computing is not fixed, since the total amount of computation is not fixed, given that computation depends on the latent processes which have random size.

This can be a major impediment for the proposal to be used by ecologists. The analysis of Section 3.2.1 required over a month of run time on a home computer per chain. While it can be possible to improve computation time by making use of more powerful computers, the proposal would benefit the most by some sort of analytical efficiency. Recent developments in Hamiltonian Monte Carlo (HMC) are powerful, yet the theory depends on energy conservation in a given space and the space of a Point Process when viewed as a subspace of \mathbb{R}^n has variable dimension n . It is thus harder to guarantee conserving energy.

Another important discussion is with regard to the estimation of λ^* . In their work, they imply that some prior information can help in estimating it, particularly with respect to areas where the data happen more often, meaning that the intensity function of the points should be close to its supreme, encoded by λ^* . However, when testing different sets of covariates for the data from Section 3.2.1, the estimation of λ^* was not consistent throughout the different sets, meaning that this parameter is not only a physical feature of the data, but also of the model being estimated.

More straightforward next steps involve improving the model itself. As discussed in Chapter 4, a Gaussian Process can be added to the linear predictors, that is, both the ones for intensity and observability. This would improve the model to incorporate spatial dependence beyond the covariates. The beauty of this solution is that, if there is no such dependence, the model should be able to estimate little spatial correlation. Furthermore, if there is no need for extra variability in the linear predictor if, for example, the covariates provide sufficient explanatory information, then the Gaussian Process will be estimated to have little variance.

Another important extension is multiple species modeling. The literature treats this by setting equal observability parameters in Fithian et al. (2015) and Renner et al. (2019). This can be achieved in the proposal by setting prior correlation between these parameters. However, more can be done as the proposal permits defining interaction as exemplified by the simulation study in Section 4.1, where the presence of a species influences the intensity function of others.

Appendix A

Codes used for examples

Here, the examples used throughout the thesis are explained in more detail and the codes used to generate figures is exposed. All codes are in the R language version 3.6.2 (R Core Team, 2019).

A.1. GENERATING FIGURES IN THE INTRODUCTION

For the generation of the examples of Point Processes, Diggle (2013) was used. In all cases, the region \mathcal{D} is taken to be the 0-1 square. The following code was run before the graphics.

```
library(dplyr)
library(ggplot2)
```

```
# Golden ratio
gr = (sqrt(5)+1)/2
standWidth = 8
```

The completely random, clustered and regular processes are in Figure 1.1. The completely random process is generated as a homogeneous Poisson Process.

```
# Completely random process
set.seed(123)
lambda = 20
n_x = rpois(1, lambda)
pts = cbind(runif(n_x), runif(n_x))
g = data.frame(pointsx = pts[,1], pointsy = pts[,2]) %>%
  ggplot(aes(x=pointsx, y=pointsy))+
  theme_bw()+theme(axis.text.x=element_blank(),
    axis.text.y=element_blank(),
    axis.ticks.x=element_blank(),
    axis.ticks.y=element_blank())+
  xlim(0,1)+ylim(0,1)+xlab("")+ylab("")+
  geom_point()
g
ggsave("figs/compRand.eps", plot=g, device="eps", width=standWidth, height=standWidth/g
```

The cluster process is generated by selecting random centroids according to a completely random process. Then an inhomogeneous Poisson Process is generated for each centroid, whose intensity function is proportional to a bi-variate Normal distribution centered on the centroid and covariance matrix is diagonal with entries 0.03 on the diagonal. Strictly speaking, the generation of these points should follow a Normal truncated to the 0-1 square. For simplicity's sake, points falling outside the region are excluded from the figure (when their coordinates exceed 1) or pushed back in (when their coordinates are negative), which means that the realized points are slightly different from what is intended, but are still good for the example.

```
# Cluster process
set.seed(123)
lambda = 11
cs = rpois(1,lambda)
centroids = cbind(runif(cs),runif(cs))
l = 5
n_c = rpois(cs,l)
pts = NULL
for (i in 1:cs)
  pts = abs(rbind(pts,
    cbind(rnorm(n_c[i],sd=0.03),rnorm(n_c[i],sd=0.03))+
    matrix(centroids[i,],n_c[i],2,byrow=T)))
g = data.frame(pointsx = pts[,1],pointsy = pts[,2]) %>%
  ggplot(aes(x=pointsx,y=pointsy))+
  theme_bw()+theme(axis.text.x=element_blank(),
    axis.text.y=element_blank(),
    axis.ticks.x=element_blank(),axis.ticks.y=element_blank())+
  xlim(0,1)+ylim(0,1)+xlab("")+ylab("")+
  geom_point()
g
ggsave("figs/cluster.eps",plot=g,device="eps",
  width=standWidth,height=standWidth/gr)
```

The regular process is generated like so: First generate the number of points. Then add each point individually so that every point must fall at least some pre-specified squared distance away from every other existing point. This is done by acceptance-rejection. Repeat until the generated number of points have been placed. The pre-specified distance has been fine tuned to achieve the desired effect, and depends on the number of points. The larger the amount of points, the smaller the distance must be.

```
# Regular process
set.seed(123)
# This function calls itself until it can generate a valid point
samplePT = function(existing,maxDist = 0.02,count = 0){
  maxCount = 50
  propo = runif(2)
  if (count == maxCount){
    stop("Attempts_reached_maximum_allowed_value.")
  }
```

```

} else if (is.null(existing))
  return(propo)

minD = min(as.vector(diag(existing %*% t(existing))) +
  as.vector(propo %*% propo) -
  as.vector(2*existing %*% propo))
if (minD<maxDist)
  return(samplePT(existing ,maxDist ,count+1))
else
  return(rbind(existing ,propo))
}
lambda = 20
n_x = rpois(1,lambda)
pts = NULL
for (i in 1:n_x)
  pts = samplePT(pts,0.05)
g = data.frame(pointsx = pts[,1],pointsy = pts[,2]) %>%
  ggplot(aes(x=pointsx,y=pointsy))+
  theme_bw()+theme(axis.text.x=element_blank(),
    axis.text.y=element_blank(),
    axis.ticks.x=element_blank(),axis.ticks.y=element_blank())+
  xlim(0,1)+ylim(0,1)+xlab("")+ylab("")+
  geom_point()
g
ggsave("figs/regular.eps",plot=g,device="eps",
  width=standWidth,height=standWidth/gr)

```

The marked process in Figure 1.2 is a simple completely random Process whose marks are independent Normal with mean 0 and standard deviation 50, truncated to positive values.

```

# Marked process
set.seed(123)
lambda = 20
n_x = rpois(1,lambda)
pts = cbind(runif(n_x),runif(n_x))
marks = rnorm(n_x,sd=50)
g = data.frame(pointsx = pts[,1],
  pointsy = pts[,2],marked=abs(marks)) %>%
  ggplot(aes(x=pointsx,y=pointsy))+
  theme_bw()+theme(axis.text.x=element_blank(),
    axis.text.y=element_blank(),
    axis.ticks.x=element_blank(),axis.ticks.y=element_blank(),
    legend.position="none")+
  xlim(0,1)+ylim(0,1)+xlab("")+ylab("")+
  geom_point(size=0.5)+
  geom_point(aes(size=marked),shape=1)
g

```

```
ggsave("figs/marked.eps", plot=g, device="eps",
       width=standWidth, height=standWidth/gr)
```

The thinned and superposed processes in Figures 1.3 and 1.4 are mirrors of each other and come from the thinning of a homogeneous Poisson Process. The superposed processes are just the thinned and retained processes.

```
# Thinned process
set.seed(123)
lambda = 20
n_x = rpois(1, lambda)
pts = cbind(runif(n_x), runif(n_x))
thin = rbinom(n_x, 1, 0.5)
g = data.frame(pointsx = pts[,1], pointsy = pts[,2], thinned=as.factor(1-thin)) %>%
  ggplot()+theme_bw()+
  theme(legend.position = c(0.9, 0.2),
        legend.background = element_rect(size=0.5, linetype="solid",
        colour = "black"),
        axis.text.x=element_blank(), axis.text.y=element_blank(),
        axis.ticks.x=element_blank(), axis.ticks.y=element_blank())+
  xlim(0,1)+ylim(0,1)+xlab("")+ylab("")+
  geom_point(aes(x=pointsx, y=pointsy, shape=thinned))+
  scale_shape_manual(values=c(19, 4))
g
ggsave("figs/thinned.eps", plot=g, device="eps",
       width=standWidth, height=standWidth/gr)

# Superposed process
set.seed(123)
lambda = 20
n_x = rpois(1, lambda)
pts = cbind(runif(n_x), runif(n_x))
thin = rbinom(n_x, 1, 0.5)
g = data.frame(pointsx = pts[,1], pointsy = pts[,2]) %>%
  dplyr::filter(thin==1) %>%
  ggplot()+theme_bw()+
  theme(axis.text.x=element_blank(), axis.text.y=element_blank(),
        axis.ticks.x=element_blank(), axis.ticks.y=element_blank())+
  xlim(0,1)+ylim(0,1)+xlab("")+ylab("")+
  geom_point(aes(x=pointsx, y=pointsy))
g
ggsave("../figs/superposed1.eps", plot=g, device="eps",
       width=standWidth, height=standWidth/gr)
g = data.frame(pointsx = pts[,1], pointsy = pts[,2]) %>%
  dplyr::filter(thin==0) %>%
  ggplot()+theme_bw()+
```

```

theme(axis.text.x=element_blank(),axis.text.y=element_blank(),
      axis.ticks.x=element_blank(),axis.ticks.y=element_blank())+
xlim(0,1)+ylim(0,1)+xlab("")+ylab("")+
geom_point(aes(x=pointsx,y=pointsy))
g
ggsave("figs/superposed2.eps",plot=g,device="eps",
       width=standWidth,height=standWidth/gr)

```

The IPP which resembles a cluster process in Figure 1.5 is an inhomogeneous Poisson Process generated by Algorithm 1. The intensity function used was calculated like so: Four points P were randomly placed in the region and the intensity function $\lambda(s)$ is proportional to $\min_{p \in P} \exp\{-d(s,p)/0.01\}$ where $d(s,p)$ is the squared euclidean distance between s and p .

```

# Fake cluster process (IPP)
set.seed(123)
tops = 4
tops_coord = cbind(runif(tops),runif(tops))
int_func = function(s,topses,phi=1){
  dist_to_closest = min(as.vector(s %*% s) +
                        as.vector(diag(topses %*% t(topses))) -
                        as.vector(2*topses %*% s))
  return(exp(-dist_to_closest/phi))
}
lambda_star = 200
n_x_prop = rpois(1,lambda_star)
x_prop = cbind(runif(n_x_prop),runif(n_x_prop))
accept_prob = apply(x_prop,1,
  function(s)
    200*int_func(s,tops_coord,0.01)/lambda_star)
x = x_prop[which(rbinom(n_x_prop,1,accept_prob)==1),]
n_x = nrow(x)
g = data.frame(pointsx = x[,1],pointsy = x[,2]) %>%
  ggplot(aes(x=pointsx,y=pointsy))+
  theme_bw()+theme(axis.text.x=element_blank(),
    axis.text.y=element_blank(),
    axis.ticks.x=element_blank(),axis.ticks.y=element_blank())+
  xlim(0,1)+ylim(0,1)+xlab("")+ylab("")+
  geom_point()
g
ggsave("figs/fakeCluster.eps",plot=g,device="eps",
       width=standWidth,height=standWidth/gr)

```

A.2. GENERATING ROC AND AUC EXAMPLES

The examples have been generated with simulated data from the model:

$$Y_i \sim \text{Bernoulli}(p_i), \quad i = 1, \dots, 100$$

$$\text{logit } p_i = -1 + x_i. \quad (\text{A.1})$$

The covariates x_i were simulated independently from a Normal distribution with parameters 0 and 1. For simplicity, the predictions were done in sample.

The evaluation of the area under the ROC curve (AUC) is done as follows. The curve is the result of the application of the model on a discrete set of points. Then the curve increases either horizontally, vertically or diagonally in a line. In order to find the area under it, it is simpler to realize there are rectangles and trapezoids as seen in Figure A.1. Their respective areas can be found and added in order to find the total area.

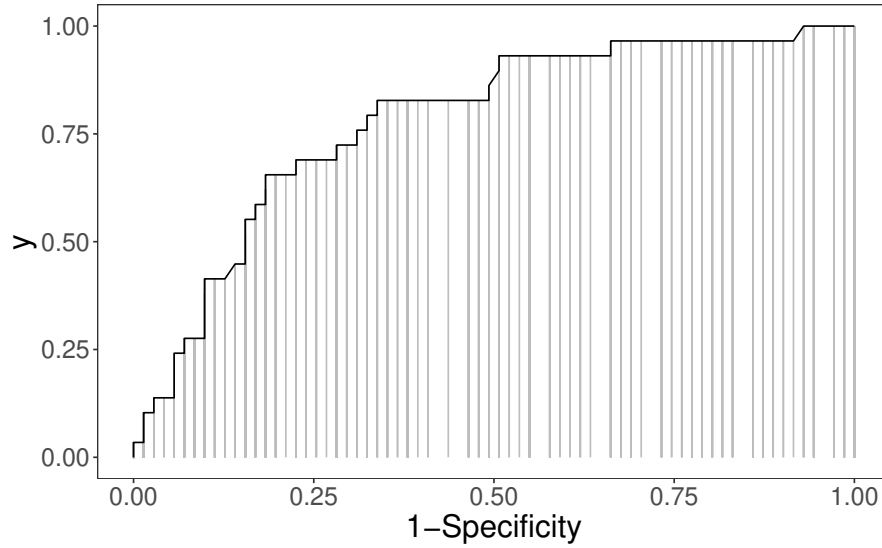


Figure A.1: Rectangles and trapezoids which form the ROC

Note that the thresholds start at 1 and decrease to 0. For every decreasing step, the confusion matrix, as described in Table 2.2, changes. For a single point, the decrease of the threshold may change its classification from negative to positive. On the one hand, if its true value is positive, that means an increase in sensitivity, while maintaining the specificity. On the other hand, if its true value is negative, the specificity decreases, that is, 1-specificity increases, while maintaining sensitivity stable.

So at every decreasing step of the threshold, there are four possibilities for the sensitivity with 1-specificity combination. Either none change, one of them changes or both change, the latter being the case where multiple points change classification with varying true values. To find the area under the ROC curve, the areas of rectangles and trapezoids must be added. A rectangle is formed if there is an increase in 1-specificity whilst sensitivity remains stable, and it is a trapezoid if both change. Note that if there is no change in 1-specificity, then no area is to be added.

The calculations use a T -dimensional vector for the thresholds. For each threshold $i = 2, \dots, T$

call $\delta_{Se}^{(i)} = Se_i - Se_{i-1}$ where Se_i represents the sensitivity resulting from threshold i . Likewise, call $\delta_{CSp}^{(i)} = CSp_i - CSp_{i-1}$ where $1 - CSp_i$ is the specificity resulting from threshold i .

Then, set $AUC_1 = 0$ and for $i = 2, \dots, T$ find AUC_i with

$$AUC_i = AUC_{i-1} + \begin{cases} 0 & , \text{ if } \delta_{CSp}^{(i)} = 0 \\ \delta_{CSp}^{(i)} Se_i & , \text{ if } \delta_{Se}^{(i)} = 0 \text{ and } \delta_{CSp}^{(i)} \neq 0 \\ \delta_{CSp}^{(i)} Se_{i-1} + \delta_{CSp}^{(i)} \delta_{Se}^{(i)} / 2 & , \text{ if } \delta_{Se}^{(i)} \neq 0 \text{ and } \delta_{CSp}^{(i)} \neq 0. \end{cases} \quad (A.2)$$

Note that in the case of a rectangle ($\delta_{Se}^{(i)} = 0$ and $\delta_{CSp}^{(i)} \neq 0$), the sensitivity doesn't change, so $\delta_{CSp}^{(i)} Se_i = \delta_{CSp}^{(i)} Se_{i-1}$. The trapezoid increase can be reduced to $\delta_{CSp}^{(i)} (Se_{i-1} + \delta_{Se}^{(i)} / 2)$. Then, equation (A.2) can be reduced to

$$AUC_i = AUC_{i-1} + \delta_{CSp}^{(i)} \left(Se_{i-1} + \frac{1}{2} \delta_{Se}^{(i)} \right), \quad (A.3)$$

since all possibilities are covered. The AUC for the model is AUC_T . The code for the R software version 3.6.2 (R Core Team, 2019) below was used to generate the examples for the ROC curves and their respective AUC.

```
library(ggplot2)
```

```
# Golden ratio
```

```
gr = (sqrt(5)+1)/2
```

```
standWidth = 8
```

```
# Simulating data
```

```
set.seed(456)
```

```
n = 100
```

```
inter = 1000
```

```
x = rnorm(n)
```

```
ps = (1+exp(1-x))^( -1)
```

```
y = rbinom(n,1,ps)
```

```
total = sum(y)
```

```
# ROC curve for the generating model
```

```
thresh = seq(1,0,len=inter)
```

```
sens = rep(0,inter)
```

```
spec = rep(0,inter)
```

```
for (i in 1:inter){
```

```
  sens[i] = sum((ps>=thresh[i])[which(!y)]) / total
```

```
  spec[i] = sum((ps<thresh[i])[which(!y)]) / (n-total)
```

```
}
```

```
df.ROC = data.frame(Sensitivity=sens, CompSpecificity=1-spec, Random=thresh)
```

```
g = ggplot(df.ROC)+theme_bw()+labs(x="1-Specificity")+
```

```
  geom_line(aes(x=CompSpecificity, y=Sensitivity))+
```

```

    theme(text = element_text(size=20))+
    geom_line(aes(x=Random,y=Random),
      linetype="dashed")+ # Random classifier
    geom_line(aes(x=perfX,y=perfY),
      linetype="dotted", # Perfect classifier
      data=data.frame(perfX = c(0,0,1),
        perfY = c(0,1,1)))
g

AUC = 0
for (i in 2:inter){
  dSe = df.ROC$Sensitivity[i]-df.ROC$Sensitivity[i-1]
  dCSp = df.ROC$CompSpecificity[i]-df.ROC$CompSpecificity[i-1]
  increment = dCSp*(df.ROC$Sensitivity[i-1] + dSe*0.5)
  AUC = AUC + increment
}
AUC # 0.7797475

ggsave("figs/ROC.eps",plot = g,device="eps",
  width=standWidth,height=standWidth/gr)

# Crossing ROC curves
sens2 = rep(0,inter)
spec2 = rep(0,inter)
p2s = pnorm(1+0.008*x) # Wrong parameters
for (i in 1:inter){
  sens2[i] = sum((p2s>=thresh[i])[which(!y)])/total
  spec2[i] = sum((p2s<thresh[i])[which(!y)])/(n-total)
}

df.ROC = data.frame(Sensitivity=sens, CompSpecificity=1-spec, Random=thresh,
  Sensitivity2=sens2, CompSpecificity2=1-spec2)

g = ggplot(df.ROC)+theme_bw()+labs(x="1-Specificity")+
  geom_line(aes(x=CompSpecificity,y=Sensitivity))+
  theme(text = element_text(size=20))+
  geom_line(aes(x=Random,y=Random),
    linetype="dashed")+ # Random classifier
  geom_line(aes(x=CompSpecificity2,y=Sensitivity2),
    linetype="dotdash")+
  geom_vline(xintercept=0.15,color="lightgreen")+
  geom_vline(xintercept=0.5,color="lightgreen")
g

AUC = 0
for (i in 2:inter){
  dSe = df.ROC$Sensitivity2[i]-df.ROC$Sensitivity2[i-1]
  dCSp = df.ROC$CompSpecificity2[i]-df.ROC$CompSpecificity2[i-1]

```

```
    increment = dCSp*(df.ROC$Sensitivity2[i-1] + dSe*0.5)
    AUC = AUC + increment
  }
AUC # 0.7727052

ggsave("figs/ROC2.eps", plot = g, device="eps",
        width=standWidth, height=standWidth/gr)
```

Appendix B

Code used for the data

Here some code is presented that was used to simulate data and estimate the proposed model in equation (2.5). The full R projects can be found in .

In all cases, the code writes data on storage drive. The reason for this is that all results are immediately written into reliable files, which makes the programs able to pick off where they leave off if there is power outage or any other interrupting event. Another reason to get the chain off the system memory is the latent processes X' and U , since, depending on the value of λ^* , a full chain may take up 30 gigabytes, which most computers cannot withstand.

The simulations have been done in the R Software version 3.6.2 (R Core Team, 2019) and the MCMC chains were programmed in Ox version 7.20 (Doornik, 2017). An interface was created to seamlessly transfer data between these programs. All calculations are done in the log scale for computational stability.

B.1. GENERATING SIMULATED DATA SETS

This Section presents the code used to simulate data and ready it for Ox to read. Function `geraPObin` simulates data and stores it in a binary file, returning the path to that file. Parameter `path` is the directory where the chain is to be located. `b` is the chosen β vector, `d` is the chosen δ vector and `ls` is λ^* . `covcom` varies between 0 and 1. When 1, it forces the last covariate of the $W(\cdot)$ vector to be the same covariate as the last in the $Z(\cdot)$ vector. The link can be logit or log. If log, the generating model has an intensity function as the one in equation (1.30) and there is no true value for λ^* .

Typical use of this function would look like

```
dataPath = geraPObin(pathProj, c(1,1), c(2,1), 1000)
```

It would simulate a PO data set from the model in equation (2.5) with parameters

$$\begin{aligned}\beta &= \begin{pmatrix} 1 \\ 1 \end{pmatrix} \\ \delta &= \begin{pmatrix} 2 \\ 1 \end{pmatrix} \\ \lambda^* &= 1000,\end{aligned}\tag{B.1}$$

where all covariates are independent standard Normal random variables. Command `dataPath$caminho` would return the path to where the data was written. Function definition:

```
geraPObin = function(path,b,d,ls = 1e4,covcom=0,link="logit"){

nb = length(b)
nd = length(d)
## Total points
Nt = rpois(1,ls)
# Spreading points over the region
xT = runif(Nt)
yT = runif(Nt)
pT = cbind(xT,yT)

## Generating covariates Z
ZT = matrix(c(rep(1,Nt),rnorm(Nt*(nb-1))),
  nrow=Nt,ncol=nb,byrow=F)

## Resampling according to the chosen link function
if (link=="logit")
  probY = -log(1+exp(-ZT*%b))
else
  probY = ZT*%b-log(ls)
selY = 1*(log(runif(Nt))<=probY)
Y = pT[which(selY==1),]
ZY = as.matrix(ZT[which(selY==1),])
NY = nrow(Y)
U = pT[which(selY==0),]
ZU = as.matrix(ZT[which(selY==0),])
NU = nrow(U)

## Generating covariates W
if (covcom)
  WY = matrix(c(rep(1,NY),
    ZY[,2],rnorm(NY*(nd-2))),
    nrow=NY,ncol=nd,byrow=F)
else
  WY = matrix(c(rep(1,NY),rnorm(NY*(nd-1))),
    nrow=NY,ncol=nd,byrow=F)

## Resample to separate observed and not observed occurrences (PO sample)
probX = -log(1+exp(-WY*%d))
```

```
selX = 1*(log(runif(NY))<=probX)
X = Y[which(selX==1),]
ZX = as.matrix(ZY[which(selX==1),])
WX = as.matrix(WY[which(selX==1),])
NX = nrow(X)
Xp = Y[which(selX==0),]
ZXp = as.matrix(ZY[which(selX==0),])
WXp = as.matrix(WY[which(selX==0),])
NXp = nrow(Xp)

## Picking a file name and storing the data
if (substr(path,nchar(path),nchar(path))=="/")
  cam = path
else
  cam = paste(path,"/",sep="")
caminho = paste(cam,"dadosPO",
                format.Date(Sys.time(),
                             "%Y%m%d%H%M%S"), ".dat", sep="")
guardaPO = file(caminho,"wb")
writeBin(nb,guardaPO)
writeBin(nd,guardaPO)
writeBin(0,guardaPO) # Let Ox know that there is no phi
                      # (phi is used when covariates are
                      # spatially correlated)
for (i in 1:nb) writeBin(b[i]*1.0,guardaPO)
for (i in 1:nd) writeBin(d[i]*1.0,guardaPO)
writeBin(ls*1.0,guardaPO)
writeBin(link,guardaPO)
writeBin(NU,guardaPO)
writeBin(NX,guardaPO)
writeBin(NXp,guardaPO)
if (NU>0)
  for (i in 1:NU){
    writeBin(U[i,1]*1.0,guardaPO)
    writeBin(U[i,2]*1.0,guardaPO)
    for (j in 1:nb) writeBin(ZU[i,j]*1.0,guardaPO)
  }
  for (i in 1:NX){
    writeBin(X[i,1]*1.0,guardaPO)
    writeBin(X[i,2]*1.0,guardaPO)
    for (j in 1:nb) writeBin(ZX[i,j]*1.0,guardaPO)
    for (j in 1:nd) writeBin(WX[i,j]*1.0,guardaPO)
  }
if (NXp > 0)
  for (i in 1:NXp){
    writeBin(Xp[i,1]*1.0,guardaPO)
    writeBin(Xp[i,2]*1.0,guardaPO)
    for (j in 1:nb) writeBin(ZXp[i,j]*1.0,guardaPO)
```

```

    for (j in 1:nd) writeBin(WXp[i,j]*1.0,guardaPO)
  }
writeBin(covcom*1.0,guardaPO)
close(guardaPO)

return(list(caminho=caminho,zx=ZX,wx=WX))
}

```

B.2. MCMC FOR THE SIMULATED DATA

The following R function writes the MCMC initialization parameters on a temporary file, calls the Ox program from the system shell and passes on the necessary paths on to Ox. The parameter `arquivo` is the path to the PO data. `niter` means the total number of iterations, `burnin` is the number of iterations in the burn-in period and `thin` means that 1 in every `thin` iterations will be retained and the rest discarded.

Parameters `ib`, `id` and `il` are the initial values for β , δ and λ^* , respectively. If the transformed model from Section 2.3.3 is used, then `il` initiates γ instead. `mpb` and `spb` are the prior parameters for β , `mpd` and `spd` the same for δ and `al` and `bl` for λ^* . `aD` is the total area of the region \mathcal{D} . `storeProcess = TRUE` forces the Ox program to store latent process realizations as well as parameters. Its value is set to `FALSE` for efficiency and speed when only the parameter values are of interest. `passa` is the Ox code to be run, `cad` is the path to the MCMC chains where the Ox code will write. Finally, `suf` is a suffix to the MCMC chain file name, and is used when multiple chains are run on the same simulated PO data.

A typical use of this function would be

```

mcmc_a = POxMCMC(dataPath$caminho,500000,100000,500,
  c(0,0),c(0,0),500,rep(0,2),diag(2)*100,
  rep(0,2),diag(2)*100,1e-4,1e-4,suf="a")

```

This would start the Ox program to run the MCMC code on the PO data stored in `dataPath$caminho` with 100,000 burn-in period, 500,000 iterations taking 1 in every 500. The initial values for the chain are at the origin for β and δ . Parameter λ^* starts at 500. The prior for β is a bi-variate Normal centered on 0 and covariance matrix $100I$ where I is the 2 by 2 identity matrix. The same prior is used for δ . The prior for λ^* is a Gamma distribution with parameters 10^{-4} and 10^{-4} . The file that is to be created will have the suffix “a” to identify its chain. Other chains can be run with other starting values to validate convergence according to Gelman and Rubin (1992).

```

POxMCMC = function(arquivo,niter,burnin,thin,ib,id,il,
  mpb,spb,mpd,spd,al,bl,aD=1,storeProcess=TRUE,
  passa = "Arquivos_Interface/MCMCpo.ox",
  cad="Arquivos_Interface/Cadeias/",suf=""){

  path = substr(arquivo,1,max(gregexpr("/",arquivo)[[1]]))
  fimres = substr(arquivo,max(gregexpr("/",arquivo)[[1]])+8,nchar(arquivo))
  tempath = paste(path,"tempR.txt",sep="")

```



```

temp = file(tempath,"w")
cat(length(ib),length(id),format(niter,sci=F),
    format(burnin,sci=F),format(thin,sci=F),
    format(ib,sci=F),format(id,sci=F),format(il,sci=F),
    format(mpb,sci=F),format(spb,sci=F),format(mpd,sci=F),
    format(spd,sci=F),format(al,sci=F),format(bl,sci=F),
    format(aD,sci=F),storeProcess,sep="\n",file=temp)
close(temp)

# Detects the Operating System to call the correct command
sysName = Sys.info()[1]
if (sysName == "Windows"){
  oxRegPath = file.path("Software",
    "OxMetrics7","OxEdit","Modules",fsep="\\")
  options(warn=-1)
  reg=readRegistry(oxRegPath,
    "HCU")$Path0 ## Recovers Ox executable from registry in Windows
  options(warn=0)
  if (!exists("reg")){
    print("Ox_line_program_not_found_in_system._Exiting_code.")
    return()
  }
  oxpath = gsub("\\\\", "/", reg)
} else if (sysName == "Darwin")
  oxpath = "/Applications/OxMetrics7/ox/bin/oxl"
else if (sysName == "Linux")
  oxpath = "oxl64"

cadeias = paste(pathProj, "/", cad, "MCMC", suf, fimres, sep="")
runPassa = paste(pathProj, "/", passa, sep="")

system2(oxpath, paste("\\"", runPassa, "\\"", sep="", collapse=""),
  input=c(arquivo, tempath, cadeias))

return(cadeias)
}

```

Below is the Ox code that is started from R function POoxMCMC. Long lines have been wrapped for readability. They are indicated by the character ~ at its end. Code should be reassembled to work.

```

#include <oxstd.oxh>
#include <oxprob.h>

// Function used to read covariates from a lattice file in case
// they are spatially correlated
readGridCov(const path, const points, const delta, const z, const w)
{

```

```

decl arq, A = zeros(2,1), P, D, p = rows(points),~
    sD, sortD, M, i, j, tempZi, tempWi,~
    tempZ = zeros(rows(z[0]),columns(z[0])),~
    tempW = zeros(rows(w[0]),columns(w[0]));

// Distance matrix D is the product of the steps matrix P
// with the sizes matrix A: PA = D

A[0][0] = 1;
A[1][0] = floor(1/delta)+1;
P = round(points ./ delta)+(ones(p,1)~zeros(p,1));
D = P*A;

arq = fopen(path,"rb");
j = 0;
foreach (i in D)
{
    fseek(arq,'f',2*(i-1));
    fread(arq,&tempZi,'f');
    fread(arq,&tempWi,'f');
    tempZ[j][0] = tempZi;
    tempW[j++][0] = tempWi;
}

fclose(arq);
z[0] = tempZ;
w[0] = tempW;
}

// Generating the full conditionals for X' and U
CCprocessos(const us, const zus, const wus,~
    const xls, const zxls, const wxls, const beta,~
    const delta, const gamma, const cc, const deltaGrid,~
    const gridPath, const aread)
{
    decl nY, Ztotpos, ZZtot, WZtot, tempGridZ,~
    tempGridW, cpZtot, selZ, contaZ, juntaZ, Z,~
    ZZ, WZ, zzb, probu, selU, contaXl, contaU,~
    juntaU, nbeta, ndelta;

    nbeta = rows(beta);
    ndelta = rows(delta);

    // To generate X' and U, a HPP is thinned
    nY = ranpoisson(1,1,gamma*(1+exp(-delta[0]))*(1+exp(-beta[0]))*aread);
    Ztotpos = ranu(nY,2);

```

```

if (gridPath != "") // If a file with a grid is given
{
  ZZtot = ones(nY,1) ~ rann(nY,nbeta-2);
  WZtot = ones(nY,1) ~ rann(nY,ndelta-2);
  tempGridZ = zeros(nY,1);
  tempGridW = zeros(nY,1);
  readGridCov(gridPath,Ztotpos,deltaGrid,&tempGridZ,&tempGridW);

  ZZtot ~= tempGridZ;
  WZtot ~= tempGridW;
  if (cc)
    WZtot[][1] = ZZtot[][1];
}
else
{
  ZZtot = ones(nY,1) ~ rann(nY,nbeta-1);
  if (cc)
    WZtot = ones(nY,1) ~ ZZtot[][1] ~ rann(nY,ndelta-2);
  else
    WZtot = ones(nY,1) ~ rann(nY,ndelta-1);
}

// cpZtot is the nY long vector with the probabilities that the point
// belongs to neither X' nor U
cpZtot = -log(1+exp(-ZZtot * beta))-~
  log(1+exp(-WZtot * delta));
selZ = log(ranu(nY,1)) .> cpZtot;
contaZ = sumc(selZ)[0][0]-1;
if (contaZ < 0) // Latent processes can be empty sets too
{
  us[0] = <>;
  zus[0] = <>;
  wus[0] = <>;
  xls[0] = <>;
  zxls[0] = <>;
  wxls[0] = <>;
  return;
}
juntaZ = sortbyc((1 - selZ) ~ Ztotpos ~ ZZtot ~ WZtot,0);
Z = juntaZ[0:contaZ][1:2];
ZZ = juntaZ[0:contaZ][3:(2+nbeta)];
WZ = juntaZ[0:contaZ][(3+nbeta):(2+nbeta+ndelta)];

// Calculating the probabilities of U, given that it's either X' or U
zzb = ZZ * beta;
probu = zzb - log(1+exp(WZ * delta)+exp(zzb));
selU = log(ranu((contaZ+1),1)) .> probu;

```

```
contaU = sumc(selU)[0][0]-1;
juntaU = sortbyc((1 - selU) ~ Z ~ ZZ ~ WZ,0);
if (contaU > -1)
{
  us[0] = juntaU[0:contaU][1:2];
  zus[0] = juntaU[0:contaU][3:(2+nbeta)];
  wus[0] = juntaU[0:contaU][(3+nbeta):(2+nbeta+ndelta)];
}
else
{
  us[0] = <>;
  zus[0] = <>;
  wus[0] = <>;
}
if (contaU != contaZ)
{
  xls[0] = juntaU[(contaU+1):contaZ][1:2];
  zxls[0] = juntaU[(contaU+1):contaZ][3:(2+nbeta)];
  wxls[0] = juntaU[(contaU+1):contaZ][(3+nbeta):(2+nbeta+ndelta)];
}
else
{
  xls[0] = <>;
  zxls[0] = <>;
  wxls[0] = <>;
}
}

// Gamma full conditional
CCgamma(const gamma, const pontos, const t,~
const s, const a, const b, const aread)
{
  decl tg;
  tg = rangamma(1,1,a+pontos,(b+aread)*(1+exp(-s))*(1+exp(-t)));

  gamma[0] = tg;
}

// Calculates the density value of the beta and delta
// joining full conditional, up to a constant
calcFCbdDens(const beta, const delta, const zu,~
const zx, const zxl, const wx, const wxl,~
const mb, const md, const sb, const sd,~
const g, const ad, const pts, const ag, const bg)
{
  decl adu = 0, adxp = 0, tr, st;
```

```

st = log(1+exp(-beta [0])) + log(1+exp(-delta [0]));

if (rows(zu))
    adu = sumc(-log(1+exp(zu*beta)));
if (rows(wxl))
    adxp = sumc(-log(1+exp(wxl*delta)));

tr = -g*(ad+bg)*exp(st) + (pts+ag)*st;

return(sumc(-log(1+exp(-(zx | zxl)*beta))) + adu + ~
    sumc(-log(1+exp(-wx*delta))) + adxp + tr - ~
    1/2*((beta-mb)'*sb*(beta-mb))-1/2*((delta-md)'*sd*(delta-md)));
}

// Generating a joint Metropolis-Hastings step for beta and delta
CCbetadelta_max(const beta, const delta, const zu, ~
    const zx, const zxp, const wx, const wxp, ~
    const g, const aread, const pts, const mb, ~
    const md, const sb, const sd, const ag, const bg)
{
    decl H, Ha, U, vec, vecB, vecD, zAll, zbAll, ~
        wAll, wdAll, i, hb, hd; // Hessian variables
    decl nb, nd, prop, bprop, dprop; // M-H proposal variables
    decl probacb, probacd, signal, ldp, lda, ~
        qp=0, qa=0, ret = 0; // Acceptance and return variables

    // Generating proposal based on last step using the
    // negative inverted hessian matrix for covariance matrix
    nb = rows(beta [0]);
    nd = rows(delta [0]);

    // Finding the proposal center value
    vec = beta [0] | delta [0];
    vecB = vec [0:(nb-1)];
    vecD = vec [nb:(nb+nd-1)];
    H = zeros(nb+nd,nb+nd);
    Ha = zeros(nb+nd,nb+nd);
    zAll = zx | zxp | zu;
    wAll = wx | wxp;

    // Hessian for the proposal
    hb = zeros(nb,nb);
    hd = zeros(nd,nd);
    zbAll = zAll*vecB;
    wdAll = wAll*vecD;
    for (i=0;i<pts;i++)
        hb += -exp((-log(1+exp(-zbAll [i])))-~
            log(1+exp(zbAll [i])))) .* zAll [i] [] ' * zAll [i] [];

```

```

for (i=0;i<(rows(wx)+rows(wxp));i++)
  hd += -exp((-log(1+exp(-wdAll[i])))-~
    log(1+exp(wdAll[i]))) .* wAll[i][i] , * wAll[i][i];
H[0:(nb-1)][0:(nb-1)] = hb - sb;
H[nb:(nb+nd-1)][nb:(nb+nd-1)] = hd - sd;
H[0][0] -= g*aread*(1+exp(-vecD[0]))*exp(-vecB[0])+~
  pts*(1+exp(-vecB[0]))^(-1)*(1+exp(vecB[0]))^(-1);
H[nb][nb] -= g*aread*(1+exp(-vecB[0]))*exp(-vecD[0])+~
  pts*(1+exp(-vecD[0]))^(-1)*(1+exp(vecD[0]))^(-1);
H[0][nb] = -g*aread*exp(-vecB[0]-vecD[0]);
H[nb][0] = -g*aread*exp(-vecB[0]-vecD[0]);

// MH proposal
prop = invert(choleski(-H))*rann(nb+nd,1)+vec;
bprop = prop[0:(nb-1)];
dprop = prop[nb:(nb+nd-1)];

// Hessian for the previous step. Used to calculate acceptance prob
hb = zeros(nb,nb);
hd = zeros(nd,nd);
zbAll = zAll*bprop;
wdAll = wAll*dprop;
for (i=0;i<pts;i++)
  hb += -exp((-log(1+exp(-zbAll[i])))-~
    log(1+exp(zbAll[i]))) .* zAll[i][i] , * zAll[i][i];
for (i=0;i<(rows(wx)+rows(wxp));i++)
  hd += -exp((-log(1+exp(-wdAll[i])))-~
    log(1+exp(wdAll[i]))) .* wAll[i][i] , * wAll[i][i];
Ha[0:(nb-1)][0:(nb-1)] = hb - sb;
Ha[nb:(nb+nd-1)][nb:(nb+nd-1)] = hd - sd;
Ha[0][0] -= g*aread*(1+exp(-dprop[0]))*exp(-bprop[0])+~
  pts*(1+exp(-bprop[0]))^(-1)*(1+exp(bprop[0]))^(-1);
Ha[nb][nb] -= g*aread*(1+exp(-bprop[0]))*exp(-dprop[0])+~
  pts*(1+exp(-dprop[0]))^(-1)*(1+exp(dprop[0]))^(-1);
Ha[0][nb] = -g*aread*exp(-bprop[0]-dprop[0]);
Ha[nb][0] = -g*aread*exp(-bprop[0]-dprop[0]);

// Acceptance probability
ldp = logdet(-H,&signal);
lda = logdet(-Ha,&signal);
qp = -1/2 * ((prop - vec)' * (-H) * (prop - vec) - ldp);
qa = -1/2 * ((vec - prop)' * (-Ha) * (vec - prop) - lda);
if (log(ranu(1,1)) <= calcFCbdDens(bprop,dprop,zu,zx,zxp,wx,~
  wxp,mb,md,sb,sd,g,aread,pts,ag,bg)-~
  calcFCbdDens(beta[0],delta[0],zu,zx,~
  zxp,wx,wxp,mb,md,sb,sd,g,aread,pts,ag,bg)+qa-qp)
{

```

```
    beta[0] = bprop;
    delta[0] = dprop;
    ret = 1; // Controlling acceptance rate
}

return(ret);

}

main()
{
    decl pathDPO, pathINIT, pathOUT, arqDPO, arqINIT,~
        arqOUT, pathGRID = "", arqGRID; // File variables
    decl nb, nd, phis, b, phib, d, phid, l, link, nu,~
        nx, nxl, ny, u, zu, x, zx, wx, xl, zxl, wxl,~
        dgrid=1.0, covcom; // Data loading variables
    decl niter, nburnin, nthin, ib, id, ig, mb, sb,~
        md, sd, al, bl, ad, sp; // MCMC initialization variables
    decl i, j, k, temp, iter; // Auxiliary variables
    decl isb, isd, igb, igd, usezx, usewx, iu, izu,~
        iwu, ixl, izxl, iwxl, baceit, iaceit, checkpoint,~
        avgac, tempo, tempo_inc; // MCMC variables
    decl ru, rxl, rx, gaceit, gavgac; // File writing variables

    // Reading paths passed on from the R function
    scan("%z",&pathDPO);
    scan("%z",&pathINIT);
    scan("%z",&pathOUT);

    ranseed(456);

    /***** Loading PO data *****/
    arqDPO = fopen(pathDPO,"rb");
    fread(arqDPO,&nb,'d');
    fread(arqDPO,&nd,'d');
    fread(arqDPO,&phis,'f');
    b = zeros(nb,1);
    for (i=0;i<nb;++i)
    {
        fread(arqDPO,&temp,'f');
        b[i][0] = temp;
    }
    if (phis)
        fread(arqDPO,&phib,'f');
    d = zeros(nd,1);
    for (i=0;i<nd;++i)
    {
        fread(arqDPO,&temp,'f');
```

```
    d[i][0] = temp;
}
if (phis)
    fread(arqDPO,&phid,'f');

fread(arqDPO,&l,'f');
fread(arqDPO,&link,'s');
fread(arqDPO,&nu,'d');
fread(arqDPO,&nx,'d');
fread(arqDPO,&nxl,'d');

if (nu > 0)
{
    u = zeros(nu,2);
    zu = zeros(nu,nb);
    for (i=0;i<nu;++i)
    {
        fread(arqDPO,&temp,'f');
        u[i][0] = temp;
        fread(arqDPO,&temp,'f');
        u[i][1] = temp;
        for (j=0;j<nb;++j)
        {
            fread(arqDPO,&temp,'f');
            zu[i][j] = temp;
        }
    }
}

x = zeros(nx,2);
zx = zeros(nx,nb);
wx = zeros(nx,nd);

for (i=0;i<nx;++i)
{
    fread(arqDPO,&temp,'f');
    x[i][0] = temp;
    fread(arqDPO,&temp,'f');
    x[i][1] = temp;
    for (j=0;j<nb;++j)
    {
        fread(arqDPO,&temp,'f');
        zx[i][j] = temp;
    }
    for (j=0;j<nd;++j)
    {
        fread(arqDPO,&temp,'f');
        wx[i][j] = temp;
    }
}
```



```
    }
}

if (nxl > 0)
{
    xl = zeros(nxl,2);
    zxl = zeros(nxl,nb);
    wxl = zeros(nxl,nd);
    for (i=0;i<nxl;++i)
    {
        fread(arqDPO,&temp,'f');
        xl[i][0] = temp;
        fread(arqDPO,&temp,'f');
        xl[i][1] = temp;
        for (j=0;j<nb;++j)
        {
            fread(arqDPO,&temp,'f');
            zxl[i][j] = temp;
        }
        for (j=0;j<nd;++j)
        {
            fread(arqDPO,&temp,'f');
            wxl[i][j] = temp;
        }
    }
}

if (phis)
{
    fread(arqDPO,&dgrid,'f');
    pathGRID = replace(pathDPO,"dadosPO","covGrade");
}
fread(arqDPO,&covcom);

fclose(arqDPO);

//      Testing variables load
//      println("nb: ",nb,"\nnd: ",nd,"\nphis: ",~
//              phis,"\nb: ",b,"\nphib: ",phib,"\nd: ",d,~
//              "\nphid: ",phid,"\nl: ",l,"\nlink(string): ",~
//              link,"\nnu: ",nu,"\nnx: ",nx,"\nnxl: ",nxl,~
//              "\nu: ",u,"\nzu: ",zu,"\nx: ",x,"\nzx: ",zx,~
//              "\nw: ",w,"\nxl: ",xl,"\nzxl: ",zxl,"\nwxl: ",wxl);

/***** End of Loading PO data *****/

// In case the number of estimated parameters is different from
// the number of generating parameters
```

```
igb = nb;
igd = nd;

/***** Loading MCMC initialization parameters *****/
arqINIT = fopen(pathINIT,"r");
fscan(arqINIT,"%d",&nb);
fscan(arqINIT,"%d",&nd);
fscan(arqINIT,"%d",&niter);
fscan(arqINIT,"%d",&nburnin);
fscan(arqINIT,"%d",&nthin);

/* Loading starting MCMC values */
ib = zeros(nb,1); // Beta
for (i=0;i<nb;i++)
{
    fscan(arqINIT,"%f",&temp);
    ib[i][0] = temp;
}
id = zeros(nd,1); // Delta
for (i=0;i<nd;i++)
{
    fscan(arqINIT,"%f",&temp);
    id[i][0] = temp;
}
fscan(arqINIT,"%f",&ig); // Gamma
/* End of loading starting MCMC values */

/* Loading prior parameters */
mb = zeros(nb,1); // Mu for Beta
for (i=0;i<nb;i++)
{
    fscan(arqINIT,"%f",&temp);
    mb[i][0] = temp;
}
sb = zeros(nb,nb); // Sigma for Beta
for (i=0;i<nb;i++)
    for (j=0;j<nb;j++)
    {
        fscan(arqINIT,"%f",&temp);
        sb[j][i]=temp;
    }
md = zeros(nd,1); // Mu for Delta
for (i=0;i<nd;i++)
{
    fscan(arqINIT,"%f",&temp);
    md[i][0] = temp;
}
sd = zeros(nd,nd); // Sigma for Delta
```

```

for (i=0;i<nd;i++)
  for (j=0;j<nd;j++)
  {
    fscan(arqINIT,"%f",&temp);
    sd[j][i]=temp;
  }
fscan(arqINIT,"%f",&a1); // a for Gamma — can be 0
fscan(arqINIT,"%f",&b1); // b for Gamma — can be 0
/* End of loading prior parameters */

fscan(arqINIT,"%f",&ad); // D region area
fscan(arqINIT,"%s",&sp); // boolean: store the latent processes

fclose(arqINIT);

if (imod(niter,nthin) != 0)
{
  println("Number of iterations must be a multiple of thin value.");
  exit(1);
}

//      Testing parameters load
//      println("nb: ",nb,"\nnd: ",nd,"\nnIter: ",~
//              nIter,"\nnburnin: ",nburnin,"\nnthin: ",~
//              nthin,"\nib: ",ib,"\nid: ",id,"\nil: ",il,~
//              "\nmb: ",mb,"\nsb: ",sb,"\nmd: ",md,"\nsd: ",~
//              sd,"\nal: ",a1,"\nbl: ",b1,"\nad: ",ad);

/***** End of loading MCMC initialization parameters *****/

/***** MCMC calculations *****/
// Control of M-H acceptance rate
isb = invert(sb);
isd = invert(sd);
rx = rows(x);
baceit = zeros(nburnin,3);
daceit = zeros(nburnin,1);
checkpoint = <0.1;0.2;0.3;0.4;0.5;0.6;~
            0.7;0.8;0.9>; // Let user know of progress every 10%
println("Burn in started at: ",strtrim(timestr(today())));
tempo = timer();
tempo_inc = timer();
k=0;

// Dealing with the case where a covariate that was used in the

```

```

// simulation isn't added in the estimation
if (igb <= nb)
    usezx = zx;
else
{
    usezx = zeros(rows(zx),nb);
    for (i=0;i<(nb-2);i++)
        usezx[i][i] = zx[i][i];
    usezx[i][nb-1] = zx[i][igb-1];
}
if (igd <= nd)
    usewx = wx;
else
{
    usewx = zeros(rows(wx),nd);
    for (i=0;i<(nd-2);i++)
        usewx[i][i] = wx[i][i];
    usewx[i][nd-1] = wx[i][igd-1];
}

// Burn-in period. Not stored in file
for (iter=0;iter<nburnin;++iter)
{
    CCprocessos(&iu,&izu,&iwu,&ixl,&izxl,&iwxl,~
        ib,id,ig,covcom,dgrid,pathGRID,ad);
    CCgamma(&ig,nx+rows(izxl)+rows(izu),ib[0],id[0],al,bl,ad);
    baceit[iter][i] = CCbetadelta_max(&ib,&id,izu,zx,izxl,wx,iwxl,~
        ig,ad,rx+rows(izu)+rows(izxl),mb,md,isb,isd,al,bl);
    if (max(iter,== trunc(nburnin*checkpoint)))
    {
        println("Burn in has completed ",trunc(iter/nburnin*100),~
            " \b% of the calculations after ",strtrim(timespan(timer(),~
            tempo)), " with an increment of ",strtrim(timespan(timer(),~
            tempo_inc)), ".");
        tempo_inc = timer();
    }
}
avgac = sumc(baceit[0:(nburnin-1)][])/nburnin;
println("End of burn in calculations with ",~
    avgac[0][2]*100,"% accepted trials of the M-H algorithm.~
    \nThe average acceptance rate was: ",avgac[0][0], " and ",~
    avgac[0][1], ".");
println("Last increment was ",strtrim(timespan(timer(),tempo_inc)));
println(nburnin," iterations of the burn in took:~
    ",strtrim(timespan(timer(),tempo)), ".\nBurn in stopped at: ",~
    strtrim(timestr(today())));
// End of burn-in period
iaceit = zeros(niter,3);

```

```
gaceit = zeros(niter/nthin,3);
tempo = timer();
tempo_inc = timer();
k=0;
// Storing the data first in file
arqOUT = fopen(pathOUT,"wb");
fwrite(arqOUT,rx);
fwrite(arqOUT,igb);
fwrite(arqOUT,igd);
fwrite(arqOUT,nb);
fwrite(arqOUT,nd);
if (sp=="TRUE")
    fwrite(arqOUT,1.0);
else
    fwrite(arqOUT,0.0);
for (i=0;i<rx;++i)
{
    fwrite(arqOUT,x[i][0]);
    fwrite(arqOUT,x[i][1]);
    for (j=0;j<igb;++j)
        fwrite(arqOUT,zx[i][j]);
    for (j=0;j<igd;++j)
        fwrite(arqOUT,wx[i][j]);
}
fwrite(arqOUT,niter/nthin);
fclose(arqOUT);
// Valid MCMC part. Stored in file
for (iter=0;iter<niter;++iter)
{
    CCprocessos(&iu,&izu,&iwu,&ixl,&izxl,&iwxl,~
        ib,id,ig,covcom,dgrid,pathGRID,ad);
    CCgamma(&ig,nx+rows(izxl)+rows(izu),ib[0],id[0],al,bl,ad);
    iaceit[iter][] = CCbetadelta_max(&ib,&id,izu,zx,izxl,wx,iwxl,~
        ig,ad,rx+rows(izu)+rows(izxl),mb,md,isb,isd,al,bl);
    if (max(iter.== trunc(niter*checkpoint)))
    {
        println("MCMC has completed ",trunc(iter/niter*100),~
            " \b% of the calculations after ",strtrim(timespan(timer(),~
            tempo)), " with an increment of ",strtrim(timespan(timer(),~
            tempo_inc)), ".");
        tempo_inc = timer();
    }
}
if (!imod((iter+1),nthin)) // Stores every nthin iterations
{
    ru = rows(iu);
    rxl = rows(ixl);
    arqOUT = fopen(pathOUT,"ab");
    for (i=0;i<nb;++i)
```

```

        fwrite(arqOUT,ib[i][0]);
    for (i=0;i<nd;++i)
        fwrite(arqOUT,id[i][0]);
    fwrite(arqOUT,ig);
    fwrite(arqOUT,ru);
    fwrite(arqOUT,rxl);
    if (sp == "TRUE")
    {
        if (ru)
            for (i=0;i<ru;++i)
            {
                fwrite(arqOUT,iu[i][0]);
                fwrite(arqOUT,iu[i][1]);
                for (j=0;j<nb;++j)
                    fwrite(arqOUT,izu[i][j]);
                for (j=0;j<nd;++j)
                    fwrite(arqOUT,iwu[i][j]);
            }
        if (rxl)
            for (i=0;i<rxl;++i)
            {
                fwrite(arqOUT,ixl[i][0]);
                fwrite(arqOUT,ixl[i][1]);
                for (j=0;j<nb;++j)
                    fwrite(arqOUT,izxl[i][j]);
                for (j=0;j<nd;++j)
                    fwrite(arqOUT,iwxl[i][j]);
            }
    }
    fclose(arqOUT);
    gaceit[k++][i] = iaceit[iter][i];
}

}
avgac = sumc(iaceit)/niter;
gavgac = sumc(gaceit)/(niter/nthin);
println("End of MCMC calculations with ",~
    avgac[0][2]*100,"% accepted trials of the M-H algorithm.~
    \nThe average acceptance rate was: ",avgac[0][0],".");
println("Stored iterations has ",~
    gavgac[0][2]*100,"% accepted trials of the M-H algorithm.~
    \nThe average acceptance rate of stored iterations was: ",gavgac[0][0],".");
println("Last increment was ",strtrim(timespan(timer(),tempo_inc)));
println(niter," iterations of MCMC took:~
    ",strtrim(timespan(timer(),tempo))," seconds.");
/***** End of MCMC calculations *****/

}

```

Bibliography

- Adams, R. P., Murray, I. and Mackay, D. J. C. (2009) Tractable nonparametric bayesian inference in poisson processes with gaussian process intensities. In *International Conference on Machine Learning*.
- Billingsley, P. (1995) *Probability and Measure*. Wiley Series in Probability and Statistics. Wiley.
- Cressie, N. A. C. (1993) *Spatial Point Patterns*. John Wiley and Sons, Inc.
- Diggle, P. (2013) *Statistical Analysis of Spatial and Spatio-Temporal Point Patterns*. Chapman & Hall/CRC Monographs on Statistics & Applied Probability. CRC Press.
- Doornik, J. (2017) *Object-Oriented Matrix Programming Using Ox*. Timberlake Consultants Press and Oxford, London, 3 edn. URL www.doornik.com.
- Dorazio, R. M. (2014) Accounting for imperfect detection and survey bias in statistical analysis of presence-only data. *Global Ecology and Biogeography*, **23**, 1472–1484.
- Elith, J. and Leathwick, J. (2007) Predicting species distributions from museum and herbarium records using multiresponse models fitted with multivariate adaptive regression splines. *Diversity and Distributions*, **13**, 265–275.
- Fielding, A. H. and Bell, J. F. (1997) A review of methods for the assessment of prediction errors in conservation presence/absence models. *Environmental Conservation*, **24**, 38–49.
- Fithian, W., Elith, J., Hastie, T. and Keith, D. A. (2015) Bias correction in species distribution models: pooling survey and collection data for multiple species. *Methods in Ecology and Evolution*, **6**, 424–438.
- Fithian, W. and Hastie, T. (2013) Finite-sample equivalence in statistical models for presence-only data. *Ann. Appl. Stat.*, **7**, 1917–1939.
- Gamerman, D. and Lopes, H. (2006) *Markov Chain Monte Carlo - Stochastic Simulation for Bayesian Inference*. CRC Press.
- Gelfand, A. E. and Shirota, S. (2018) Preferential sampling for presence/absence data and for fusion of presence/absence data with presence-only data.
- Gelfand, A. E. and Smith, A. F. M. (1990) Sampling-based approaches to calculating marginal densities. *Journal of the American Statistical Association*, **85**, 398–409.
- Gelman, A., Carlin, J., Stern, H., Dunson, D., Vehtari, A. and Rubin, D. (2013) *Bayesian Data Analysis, Third Edition*. Chapman & Hall/CRC Texts in Statistical Science. Taylor & Francis.

- Gelman, A. and Rubin, D. B. (1992) Inference from iterative simulation using multiple sequences. *Statistical Science*, **7**, 457–472.
- Geman, S. and Geman, D. (1984) Stochastic relaxation, gibbs distributions, and the bayesian restoration of images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **PAMI-6**, 721–741.
- Giraud, C., Calenge, C., Coron, C. and Julliard, R. (2016) Capitalizing on opportunistic data for monitoring relative abundances of species. *Biometrics*, **72**, 649–658.
- Gonçalves, F. B. and Gamerman, D. (2018) Exact bayesian inference in spatiotemporal cox processes driven by multivariate gaussian processes. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, **80**, 157–175.
- Haario, H., Saksman, E. and Tamminen, J. (1999) Adaptive proposal distribution for random walk metropolis algorithm. *Computational Statistics*, **14**, 375–395.
- Hastie, T. and Fithian, W. (2013) Inference from presence-only data; the ongoing controversy. *Ecography*, **36**, 864–867.
- Hastings, W. K. (1970) Monte carlo sampling methods using markov chains and their applications. *Biometrika*, **57**, 97–109.
- Kingman, J. F. C. (1993) *Poisson processes*, vol. 3 of *Oxford Studies in Probability*. New York: The Clarendon Press Oxford University Press. Oxford Science Publications.
- Lewis, P. A. W. and Shedler, G. S. (1979) Simulation of nonhomogeneous poisson processes by thinning. *Naval Research Logistics Quarterly*, **26**, 403–413.
- Mazzochini, G. G., Fonseca, C. R., Costa, G. C., Santos, R. M., Oliveira-Filho, A. T. and Ganade, G. (2019) Plant phylogenetic diversity stabilizes large-scale ecosystem productivity. *Global Ecology and Biogeography*, **28**, 1430–1439.
- Metropolis, N., Rosenbluth, A. W., Rosenbluth, M. N., Teller, A. H. and Teller, E. (1953) Equation of state calculations by fast computing machines. *The Journal of Chemical Physics*, **21**, 1087–1092.
- Møller, J. and Waagepetersen, R. (2005) Statistical inference and simulation for spatial point processes. *Journal of the Royal Statistical Society: Series A (Statistics in Society)*, **168**, 258–259.
- Pearce, J. L. and Boyce, M. S. (2006) Modelling distribution and abundance with presence-only data. *Journal of Applied Ecology*, **43**, 405–412.
- Phillips, S. J., Anderson, R. P. and Schapire, R. E. (2006) Maximum entropy modeling of species geographic distributions. *Ecological Modelling*, **190**, 231 – 259.
- Phillips, S. J., Dudík, M. and Schapire, R. E. (2004) A maximum entropy approach to species distribution modeling. In *Proceedings of the Twenty-first International Conference on Machine Learning*, ICML '04, 83–. New York, NY, USA: ACM.
- Phillips, S. J., Dudík, M., Elith, J., Graham, C. H., Lehmann, A., Leathwick, J. and Ferrier, S. (2009) Sample selection bias and presence-only distribution models: implications for background and pseudo-absence data. *Ecological Applications*, **19**, 181–197.

- R Core Team (2019) *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria. URL<https://www.R-project.org/>.
- Renner, I. W., Elith, J., Baddeley, A., Fithian, W., Hastie, T., Phillips, S. J., Popovic, G. and Warton, D. I. (2015) Point process models for presence-only analysis. *Methods in Ecology and Evolution*, **6**, 366–379.
- Renner, I. W., Louvrier, J. and Gimenez, O. (2019) Combining multiple data sources in species distribution models while accounting for spatial dependence and overfitting with combined penalised likelihood maximisation. *bioRxiv*.
- Renner, I. W. and Warton, D. I. (2013) Equivalence of maxent and poisson point process models for species distribution modeling in ecology. *Biometrics*, **69**, 274–281.
- Royle, J. A., Chandler, R. B., Yackulic, C. and Nichols, J. D. (2012) Likelihood analysis of species occurrence probability from presence-only data for modelling species distributions. *Methods in Ecology and Evolution*, **3**, 545–554.
- Rue, H., Martino, S. and Chopin, N. (2009) Approximate bayesian inference for latent gaussian models using inte- grated nested laplace approximations (with discussion). *Journal of the Royal Statistical Society, Series B*, **71**, 319–392.
- Stan Development Team (2019) RStan: the R interface to Stan. URL<http://mc-stan.org/>. R package version 2.19.2.
- Streit, R. (2010) *Poisson Point Processes: Imaging, Tracking, and Sensing*. Springer US.
- Warton, D. I. and Shepherd, L. C. (2010) Poisson point process models solve the “pseudo-absence problem” for presence-only data in ecology. *Ann. Appl. Stat.*, **4**, 1383–1402.