Università di Pisa

Dipartimento di Matematica
Corso di Laurea in Matematica

Tesi di Laurea Magistrale

Data-driven models and random generators of rainfall

Candidata:

Debora Grando

Relatori:

Prof. Marco Romito
Prof. Omar Lakkis

Controrelatore:

Prof. Dario Trevisan

Anno Accademico 2018/2019

Contents

Introduction

A recurrent issue encountered in environmental, ecological or agricultural impact studies in which climate is an important driving force is to provide fast and realistic simulations of atmospheric variables such as precipitation at a few specific locations, at daily or hourly temporal scales. In particular, daily and hourly stochastic rainfall models provide useful supporting roles in the analysis of risk and vulnerability within hydrological and hydraulic systems. The goal of this thesis is to study rainfall models, capable of simulating rainfall data to serve as input for PDE-models aimed at evaluating flood and hydrogeological risks. Part of the work presented has been developed during an internship at the University of Sussex, where such models are currently studied.

First of all, a space-time point process model for rainfall is fully described and analysed. It is developed by P. Cowpertwait in [10] and it is based on an ad-hoc discs structure that tries to emulate physical features observed and measured in precipitation fields. In Cowpertwait's model, rain occurs in two dimensional discs, further related to bigger storm discs aimed to introduce correlations between close spatial points. More precisely, storm origins occur in a space-time Poisson process. Each storm origin is of random type and has an associated exponential-distributed radius, so storms are in fact discs in the plane. Associated with a storm origin is a further system of rain discs: rain cell origins occur at any location in the plane according to a spatial Poisson process and their radius is an independent exponential random variable. The arrival times of cell origins follow a Neyman-Scott point process. Moreover, each cell has an associated lifetime and an intensity, both taken to be independent exponential random variables. The overall rain intensity at time t and in a point $p \in \mathbb{R}^2$, is the sum of intensities of all cells alive at time t and overlapping p. The model is illustrated in Chapter 3, after a brief introduction to point processes and a description of the Nelder-Mead method, used by Cowpertwait in the fitting procedure.

Starting from Cowpertwait's cell system, we then develop a new model for rainfall based on Hidden Markov models. In the first place, the new model

emulates Cowpertwait's discs structure, though abolishing the double discs system. Correlations, which were originally induced by bigger storm discs, are now intended to be enclosed in the hidden Markov chain mechanism. The new model is described in Chapter 4, as well as some issues relating the fitting procedure.
A primary process $\{I_n(x)\}_{n\in\mathbb{N}}$ is defined for each point $x \in A$, where $A \subset \mathbb{R}^2$ is a given spatial region of interest; the process is meant to give the rainfall intensity in x for integer times (that can be thought to be days). Parameters that appear in the process' definition depend on a Markov chain, not directly observed. Specifically, for $n \geq 1$ and $x \in A$, we define the intensity process as

$$I_n(x) := \sum_{i=1}^{N^{(n)}} \sigma_i^{(n)} \mathbb{I}_{D(x_i^{(n)}, R_i^{(n)})}(x),$$

where

- $D(x_i^{(n)}, R_i^{(n)})$ is a rain disc with centre in $x_i^{(n)}$ and radius $R_i^{(n)}$, with $R_i^{(n)} \sim \exp(\xi)$ and $x_i^{(n)} \sim \mathrm{U}(A)$;
- $\sigma_i^{(n)} \sim \exp(\tau)$ is a random variable that gives the rain intensity;
- $N^{(n)} \sim \mathrm{Pois}(\lambda)$ represents the number of rain discs generated.

So, even in this case, for every time $n \geq 1$ rain occurs in cell discs, each of which has a constant intensity on its area. Furthermore, $\{I_n(x)\}_{n\in\mathbb{N}}$ is the observable process of a Hidden Markov model; in fact, the parameters $\lambda, \xi, \tau \in \mathbb{R}_+$ are taken to depend on a hidden Markov chain $(J_n)_{n\in\mathbb{N}}$, with state space $\{1, \ldots, m\}$, where $m \in \mathbb{N}$ is a fixed parameter. That is, for each $j = 1, \ldots, m$, given the chain is in state j at time n, $\{J_n = j\}$, we have certain values for the distribution parameters of N, σ and R (and so, parameters are vectors in $\mathbb{R}_+^m$).
In order to calibrate the model parameters to data, a daily rainfall time series for $L = 30$ spatial points (sites) across Germany is used; each point corresponds to the location of a rain gauge from which observed rainfall intensity are available for each day for 70 years. The question of parameter estimation in Hidden Markov model context is widely treated in Chapter 1, where Hidden Markov models are introduced. All the techniques illustrated involve maximum likelihood estimation, both by direct numerical maximization and by specific algorithm (i.e., the Expectation Maximization algorithm). However, in this context, we have to deal with two issues. The first one is the high dimensionality of the problem, since $m^2 + 4m$ parameters have to be estimated, but mainly the absence of an explicit expression for the likelihood. In particular, it is infeasible to obtain an explicit expression for the

state-dependent probability density functions p_j of I_n associated with state $j \in \{1, \dots, m\}$, which play a key role in the standard estimation procedures. Two different methods to solve this second problem are proposed and examined. In the most natural way, the first one is using a Monte Carlo simulation to find a suitable approximation of the state-dependent probability density functions. Thus, maximum likelihood estimation is performed by usual methods. Nevertheless, this procedure results to be computationally expensive, so an alternative approach is considered. The second strategy is using a Bayesian method, described in Chapter 2 of this thesis, that has the notable advantage of not requiring likelihood evaluation: that is the Approximate Bayesian Computation. In applying it to our rainfall Hidden Markov model, we can then bypass the evaluation of the state-dependent probability density functions. Still, from a practical point of view, some precautions and preliminary analysis need to be taken and done.
In both cases, it becomes computationally prohibitive estimating all the necessary parameters, so only few of them are effectively calibrated.
Some numerical results, obtained by experimentation on the software R, are shown in Chapter 5.

Chapter 1

Hidden Markov models

As said in the introduction, the aim of this thesis is generalizing a rainfall model based on a space-time point process. In doing so, we will make large use of the so-called Hidden Markov models. In this chapter we then develop in detail this mathematical tool, focusing both on theoretical properties and more practical issues. Results presented herein are mainly drawn from [24]. The Hidden Markov models (HMMs) are double stochastic processes with one underlying process $(J_n)_{n\in\mathbb{N}}$ (state sequence) that is not observable but may be estimated through a process $(X_n)_{n\in\mathbb{N}}$ that produces a sequence of observations in discrete time n. The processes are related by the following relations:

$$\mathbb{P}(J_n|J_0,\ldots,J_{n-1}) = \mathbb{P}(J_n|J_{n-1})$$
$$\mathbb{P}(X_n|X_0,J_0,\ldots,X_{n-1},J_{n-1},J_n) = \mathbb{P}(X_n|J_n),$$

for all $n \geq 1$. The first relation, in particular, exploits the fact that the underlying process is a Markov chain. Informally, Hidden Markov models can then be thought as processes which are directly observable, but that are in fact ruled by a hidden Markov chain.

1.1 Definition

We now give the formal definition of Hidden Markov model. In this section, we also present some of their main features, directly resulting from the definition.

Definition 1.1.1. A Hidden Markov model is a 5-tuple $\Delta = (E, \delta, \Gamma, \mathcal{X}, p)$ where:

- $E = \{1, \dots, m\}$ with $m \in \mathbb{N}$, $\Gamma = (\gamma_{ij})_{i,j \in E}$ and $\delta = (\delta_1, \dots, \delta_m)$ are respectively the state space, the transition probability matrix and the initial distribution of a (hidden) discrete-time homogeneous Markov chain $(J_n)_{n \in \mathbb{N}}$, namely for $n > 1$ and $i, j \in E$

$$\delta_j = \mathbb{P}(J_1 = j) \text{ and } \gamma_{ij} = \mathbb{P}(J_n = j | J_{n-1} = i);$$

- $\mathcal{X}$ is a set of possible observations for the so-called "state-dependent process" $(X_n)_{n \in \mathbb{N}}$, it may be discrete $\mathcal{X} = \{x_1, \dots, x_k\} \subset \mathbb{Z}$ or continuous $\mathcal{X} \subset \mathbb{R}^k$, for some $k \geq 1$;

- for the discrete case, $p = (p_j(x))_{x \in \mathcal{X}, j \in E}$ is the state conditional probability of the observation process $(X_n)_{n \in \mathbb{N}}$, defined by

$$p_j(x_n) := \mathbb{P}(X_n = x_n | J_n = j),$$

  for continuous observations, $p_j(x)$ is defined as the probability density function of X_n given the chain is in state j.

In general, at each instant of time n, the model is in one of the states i, $1 \leq i \leq m$. It outputs X_n according to a discrete probability (or a continuous density function) $p_i(\cdot)$ and then jumps to state j, $1 \leq j \leq m$ with probability γ_{ij}.
In the following, we mainly consider the discrete case, i.e., $p_j(x_n)$ are considered to be probabilities. However, unless differently stated, all the results presented still apply in the continuous case.
A Hidden Markov model can be represented by a directed acyclic graph. Nodes are the random variables J_n and X_n; further, for all $n \in \mathbb{N}$, there is an edge from J_n to X_n and one from J_{n-1} to J_n. These links are intended to exploit the dependencies within the Hidden Markov model.
This representation gives an easy way of calculating many of the distributions related to a Hidden Markov model. Indeed, in any directed graphical model, the joint distribution of a set of random variable V_i is given by the following relation

$$\begin{aligned} \mathbb{P}(V_1, \dots, V_n) &= \mathbb{P}(V_1 \mid V_2, \dots, V_n)\mathbb{P}(V_2, \dots, V_n) \\ &= \mathbb{P}(V_1 \mid V_1^{(l)})\mathbb{P}(V_2, \dots, V_n) \\ &= \dots \\ &= \prod_{i=1}^{n} \mathbb{P}(V_i \mid V_i^{(l)}), \end{aligned} \tag{1.1}$$

where $V_i^{(l)}$ denotes the set of all nodes of the graph linked to V_i, i.e.,

$$V_i^{(l)} := \{V \in \{V_1, \dots, V_n\} : \text{there exists an edge between } V \text{ and } V_i\}.$$

Notation. In the following, unless stated otherwise, we will use notations given in Definition 1.1.1 to refer to a general Hidden Markov model. For the sake of simplicity, when the values $x \in \mathcal{X}$ and $j = 1, \ldots, m$ are not relevant, we will denote the probability $\mathbb{P}(X_n = x | J_n = j)$ as $\mathbb{P}(X_n | J_n)$; with obvious extension to other probabilities.

Next proposition gives a first notable property of Hidden Markov model: it asserts that, conditional on the current state of the Markov chain, observations are independent.

Proposition 1.1.2. *Let Δ be a HMM and $T > 0$ be an integer. The vectors of random variables $(X_1, \ldots, X_n)$ and $(X_{n+1}, \ldots, X_T)$ are conditionally independent given J_n, for $n = 1, \ldots, T-1$. Namely, for $n = 1, \ldots, T-1$, we have*

$$\mathbb{P}(X_1, \ldots, X_T | J_n) = \mathbb{P}(X_1, \ldots, X_n | J_n) \mathbb{P}(X_{n+1}, \ldots, X_T | J_n).$$

Proof. Firstly, we note that by repeated applications of equation (1.1), we have

$$\begin{aligned} &\mathbb{P}(X_1, J_1, \ldots, X_T, J_T) \\ &= \mathbb{P}(X_1, J_1, \ldots, X_n, J_n) \frac{1}{\mathbb{P}(J_n)} \mathbb{P}(J_n, X_{n+1}, J_{n+1}, \ldots, X_T, J_T). \end{aligned}$$

Then, summing over all possible values for $(J_1, \ldots, J_{n-1})$ and $(J_{n+1}, \ldots, J_T)$ yields

$$\mathbb{P}(J_n, X_1, \ldots, X_T) = \mathbb{P}(X_1, \ldots, X_n, J_n) \frac{1}{\mathbb{P}(J_n)} \mathbb{P}(J_n, X_{n+1}, \ldots, X_T),$$

from which the result is immediate. □

Working with Hidden Markov models and applying them to real world -issues need three main problems to be solved. The first of them is the so-called *evaluation* problem: given an observation sequence $x = (x_1, \ldots, x_T)$ up to an integer time $T > 0$ and a Hidden Markov model Δ, we may need to know which is the probability to obtain the sequence x under the model. In fact, the problem is how to calculate the likelihood $\mathbb{P}(x|\Delta)$. The solution to this problem provides a score of how x belongs to the model Δ and allows us to select the competing model that best matches the observation sequence. The second problem, the *decoding* one, attempts to uncover the hidden part of the stochastic model. From a given sequence of observations $x = (x_1, \ldots, x_T)$ and a Hidden Markov model Δ, we want to find the sequence of hidden states $j = (j_1, \ldots, j_T)$ that best explains the observations.

The last issue, the *learning* problem, is how to adjust the model parameters in order to maximize $\mathbb{P}(x|\Delta)$, where $x = (x_1, \dots, x_T)$ is a given sequence of observations. The solution to this problem attempts to optimize the model parameters to best describe the observation sequence. Furthermore, the solution allows us to adapt the model parameters, according to the observed training data sequence.
All these tasks are strictly linked to the likelihood function of the Hidden Markov model. So, before studying possible ways of solving them, let us develop a formula for the likelihood.

1.2 The likelihood

In this section, we give an expression for the likelihood of a Hidden Markov model given an observation sequence $x = (x_1, \dots, x_T)$ and a model $\Delta = (E, \delta, \Gamma, \mathcal{X}, p)$. We also define some of the main quantities linked to the likelihood and their properties.

Notation. We indicate as j_n the state of the chain $(J_n)_{n\in\mathbb{N}}$ at time $n \in \mathbb{N}$.

Proposition 1.2.1. *Given a Hidden Markov model Δ and an observation sequence $(x_1, \dots, x_T)$ up to a fixed time $T > 0$, the likelihood is given by*

$$L_T(x_1, \dots, x_T) = \sum_{j_1,\dots,j_T=1}^{m} \delta_{j_1} p_{j_1}(x_1) \gamma_{j_1 j_2} \dots \gamma_{j_{T-1} j_T} p_{j_T}(x_T). \tag{1.2}$$

Proof. We present only the case of discrete observations. The following equalities hold

$$\begin{aligned} L_T(x_1, \dots, x_T) &= \mathbb{P}(X_1 = x_1, \dots, X_T = x_T) \\ &= \sum_{j_1,\dots,j_T=1}^{m} \mathbb{P}(X_1 = x_1, J_1 = j_1, \dots, X_T = x_T, J_T = j_T) \end{aligned}$$

$$\begin{aligned}
&\overset{\circ}{=} \sum_{j_1,\dots,j_T=1}^{m} \mathbb{P}(J_1 = j_1) \prod_{k=2}^{T} \mathbb{P}(J_k = j_k | J_{k-1} = j_{k-1}) \\
&\qquad\qquad \times \prod_{k=1}^{T} \mathbb{P}(X_k = x_k | J_k = j_k) \\
&= \sum_{j_1,\dots,j_T=1}^{m} (\delta_{j_1} \gamma_{j_1 j_2} \gamma_{j_2 j_3} \dots \gamma_{j_{T-1} j_T})(p_{j_1}(x_1) p_{j_2}(x_2) \dots p_{j_T}(x_T)) \\
&= \sum_{j_1,\dots,j_T=1}^{m} \delta_{j_1} p_{j_1}(x_1) \gamma_{j_1 j_2} \dots \gamma_{j_{T-1} j_T} p_{j_T}(x_T),
\end{aligned}$$

where the equality $\overset{\circ}{=}$ follows from equation (1.1). □

As the following remark clarifies, it is possible to write the likelihood in a more compact form.

Remark 1.2.2. As usual, let δ and Γ be, respectively, the initial distribution and the transition probability matrix of the hidden Markov chain and let $P(x_n) \in \mathbb{R}^{m \times m}$ denotes the diagonal matrix whose elements are the state-dependent probabilities given the observation x_n at time n, namely

$$P(x_n) := \begin{bmatrix} p_1(x_n) & & \\ & \ddots & \\ & & p_m(x_n) \end{bmatrix}.$$

Then, we can write the next matrix form expression for the likelihood, which follows easily from equation (1.2),

$$L_T(x_1, \dots, x_T) = \delta P(x_1) \Gamma P(x_2) \dots \Gamma P(x_T) \mathbf{e},$$

where $\mathbf{e} = (1, \dots, 1)^T \in \mathbb{R}^m$.

In Hidden Markov models, likelihood evaluation usually passes trough the so-called forward and backward probabilities. They are at first defined by matrices products; but, as we will see in Section 1.4, forward and backward probabilities turn out to be joint and conditional probabilities, respectively. Furthermore, it is easy to show recursive relations for this quantities, which play a key role not only in likelihood evaluation and hence parameter estimation, but also in decoding and model checking.
From now on, let $T > 0$ be a fixed integer time.

Definition 1.2.3. For each $n = 1, \dots, T$ the vector of **forward probabilities** $\alpha_n \in \mathbb{R}^m$ is defined by

$$\alpha_n := \delta P(x_1) \Gamma P(x_2) \cdot \dots \cdot \Gamma P(x_n) = \delta P(x_1) \prod_{s=2}^{n} \Gamma P(x_s).$$

Definition 1.2.4. For each $n = 1, \dots, T$ the vector of **backward probabilities** $\beta_n \in \mathbb{R}^m$ is defined by

$$\beta_n := \Gamma P(x_{n+1}) \Gamma P(x_{n+2}) \cdot \ldots \cdot \Gamma P(x_T)\mathbf{e} = \left(\prod_{s=n+1}^{T} \Gamma P(x_s) \right) \mathbf{e},$$

where $\mathbf{e} = (1, \dots, 1)^T \in \mathbb{R}^m$ and with the convention that an empty product is the identity matrix.

Even if it is not specified, the vector of forward probabilities is a row vector, whilst the vector of backward probabilities is a column vector.
We denote as $\alpha_n(j)$ and $\beta_n(j)$ for $j = 1, \dots, m$ each component of the vector of forward probabilities α_n and the vector of backward probabilities β_n.

Remark 1.2.5. An immediate consequence of the above definitions are the following recursive relations for forward and backward probabilities.
The vectors of forward probabilities are linked by the equations:

$$\alpha_1 = \delta P(x_1) \quad \text{and} \quad \alpha_{n+1} = \alpha_n \Gamma P(x_{n+1}) \quad \text{for} \quad n = 2, \dots, T.$$

Or, in scalar form,

$$\alpha_{n+1}(j) = \left(\sum_{i=1}^{m} \alpha_n(i) \gamma_{ij} \right) p_j(x_{n+1}).$$

Similarly, the vectors of backward probabilities can be written as

$$\beta_T = \mathbf{e} \quad \text{and} \quad \beta_n = \Gamma P(x_{n+1}) \beta_{n+1} \quad \text{for} \quad n = T-1, \dots, 1.$$

Or, in scalar form,

$$\beta_n(j) = \sum_{i=1}^{m} \gamma_{ji} p_i(x_{n+1}) \beta_{n+1}(i).$$

1.3 Evaluation problem

We have now developed the main tools to solve the three problems stated in Section 1.1. We begin by considering the first problem presented: the evaluation problem. The formula for the likelihood proved in Proposition 1.2.1 obviously constitutes our starting point for this analysis.
Let $x = (x_1, \dots, x_T)$ be a sequence of observations up to an integer time $T > 0$ in which the likelihood has to be evaluated. For the sake of simplicity,

in the following, we will omit the dependency from that sequence in the likelihood expression, that is, we set $L_T := L_T(x_1, \ldots, x_T)$.
First of all, we observe that from Equation (1.2) and the definition of the forward probability the likelihood is given by $L_T = \alpha_T \mathbf{e}$. The easiest way of computing it, is then recursively via

$$\alpha_1 = \delta P(x_1) \qquad \text{and} \qquad \alpha_n = \alpha_{n-1} \Gamma P(x_n) \qquad \text{for } n = 2, \ldots, T.$$

The number of operations involved is of order Tm^2, making the evaluation of the likelihood quite feasible even for large T. However, in the case of discrete state-dependent distributions, the elements of α_n, being made up of products of probabilities, become progressively smaller as n increases, and are eventually rounded to zero. In fact, with probability 1 the likelihood approaches 0 (or possibly ∞ in the continuous case) exponentially fast. In fact, one can show that, with probability 1, $\mathbb{P}(X_1, \ldots, X_n) = \sum_{j=1}^{m} \alpha_j(n)$ is approximately equal to e^{-nH} for large n, where H is the entropy of the process (see [18] for details). So, $\sum_{j=1}^{m} \alpha_j(n)$ converges to 0 or ∞ at an exponential rate. Typically, all of $\alpha_n(1), \ldots, \alpha_n(m)$ are about the same order of magnitude, so the same conclusion applies to each $\alpha_n(j)$ individually. The remedy is the same both for under-flow and over-flow, but we restrict our attention to the former one.
Since the likelihood is a product of matrices, not of scalars, it is not possible to circumvent numerical underflow simply by computing the log of the likelihood as the sum of logs of its factors. To solve the problem, we compute the log-likelihood by using a strategy of scaling the vector of forward probabilities α_n. Effectively we scale the vector α_n at each time n so that its elements add to 1, keeping track of the sum of the logs of the scale factors thus applied.
Firstly, we define for $n = 1, \ldots, T$, the vector $\phi_n \in \mathbb{R}^m$ such that

$$\phi_n = \frac{\alpha_n}{w_n},$$

where $w_n = \sum_{j=1}^{m} \alpha_n(j) = \alpha_n \mathbf{e} \in \mathbb{R}$. It is immediate to note the following consequences:

$$\begin{aligned}
&w_1 = \alpha_1 \mathbf{e} = \delta P(x_1) \mathbf{e};\\
&\phi_1 = \delta P(x_1)/w_1;\\
&w_n \phi_n = w_{n-1} \phi_{n-1} \Gamma P(x_n);\\
&L_T = \alpha_T \mathbf{e} = w_T(\phi_T \mathbf{e}) = w_T.
\end{aligned}$$

Hence,

$$L_T = w_T = w_1 \prod_{n=2}^{T} \frac{w_n}{w_{n-1}},$$

and by multiplying by $\mathbf{e}$ each side of the third equation above, it follows that

$$w_n = w_{n-1}(\phi_{n-1}\Gamma P(x_n)\mathbf{e}).$$

So, we conclude that

$$\log L_T = \log w_1 + \sum_{n=2}^{T} \log\left(\frac{w_n}{w_{n-1}}\right) = \sum_{n=2}^{T} \log(\phi_{n-1}\Gamma P(x_n)\mathbf{e}).$$

We can therefore summarize in the following algorithm the computation of log-likelihood. In order to calculate ϕ_n at each iteration, we introduce the auxiliary vector $v \in \mathbb{R}^m$ and the scalar $u \in \mathbb{R}$, which represents the quotient w_n/w_{n-1} at each step. The log-likelihood is then saved in the variable $logL$.

$w_1 \leftarrow \delta P(x_1)\mathbf{e}$;
$\phi_1 \leftarrow \delta P(x_1)/w_1$;
$logL \leftarrow \log(w_1)$;
for $n = 2, \ldots, T$
    $v \leftarrow \phi_{n-1}\Gamma P(x_n)$
    $u \leftarrow v\mathbf{e}$
    $logL \leftarrow logL + \log(u)$
    $\phi_n \leftarrow v/u$
return $logL$.

In the following, we refer to this procedure as the *scaling algorithm*. It will be used in numerical experimentation shown in Chapter 5.

1.4 Learning problem

We now focus on the learning problem. In this section, we examine the leading techniques used to maximize the likelihood for Hidden Markov models. Direct maximization and numerical methods are used in this context. As they are standard approaches, we only introduce briefly, at the end of this section, some of the main issues of numerical methods used in Hidden Markov models.
Instead, we present in detail the so-called *Expectation Maximization algorithm*. It is a common method adopted when some data are missing; when applied to Hidden Markov models, the algorithm makes strong use of the forward and backward probabilities and their properties. In the first part of this section we exploit and prove these features.

As anticipated before, forward and backward probabilities can be written, in a more interpretable way, as joint and conditional probabilities, respectively. The following propositions present these alternative forms.

Proposition 1.4.1. *For $n = 1, \dots, T$ and $j = 1, \dots, m$ it holds*

$$\alpha_n(j) = \mathbb{P}(X_1 = x_1, \dots, X_n = x_n, J_n = j).$$

Proof. The proof proceeds by induction on n.
The case $n = 1$ is straightforward. Indeed, since $\alpha_1 = \delta P(x_1)$ by definition, we have, for all $j = 1, \dots, m$,

$$\alpha_1(j) = \delta_j p_j(x_1) = \mathbb{P}(J_1 = j)\mathbb{P}(X_1 = x_1 | J_1 = j) = \mathbb{P}(X_1 = x_1, J_1 = j).$$

We now show that, if the proposition holds for some $n \in \mathbb{N}$, then it also holds for $n + 1$:

$$\begin{aligned}
\alpha_{n+1}(j) &= \sum_{i=1}^{m} \alpha_n(i)\gamma_{ij} p_j(x_{n+1}) \quad \text{(by remark 1.2.5)} \\
&= \sum_{i=1}^{m} \mathbb{P}(X_1 = x_1, \dots, X_n = x_n, J_n = i)\mathbb{P}(J_{n+1} = j | J_n = i) \\
&\qquad \times \mathbb{P}(X_{n+1} = x_{n+1} | J_{n+1} = j) \\
&\overset{\circ}{=} \sum_{i=1}^{m} \mathbb{P}(X_1 = x_1, \dots, X_{n+1} = x_{n+1}, J_n = i, J_{n+1} = j) \\
&= \mathbb{P}(X_1 = x_1, \dots, X_{n+1} = x_{n+1}, J_{n+1} = j),
\end{aligned}$$

where the equality $\overset{\circ}{=}$ follows from equation (1.1). Indeed, from (1.1) the joint distribution of $(X_1, \dots, X_n)$ and $(J_1, \dots, J_n)$ is given by

$$\mathbb{P}(X_1, J_1, \dots, X_n, J_n) = \mathbb{P}(J_1) \prod_{k=2}^{n} \mathbb{P}(J_k | J_{k-1}) \prod_{k=1}^{n} \mathbb{P}(X_k | J_k).$$

This one and the analogous expression for $\mathbb{P}(X_1, J_1, \dots, X_{n+1}, J_{n+1})$ imply that

$$\mathbb{P}(X_1, J_1, \dots, X_{n+1}, J_{n+1}) = \mathbb{P}(J_{n+1} | J_n)\mathbb{P}(X_{n+1} | J_{n+1})\mathbb{P}(X_1, J_1, \dots, X_n, J_n),$$

and summing over all possible values of $(J_1, \dots, J_{n-1})$ gives the equality $\overset{\circ}{=}$. □

In order to establish the analogous proposition for the backward probabilities, we need two simple relations stated in the following lemma.

Lemma 1.4.2. *For any fixed integer $T > 0$,*

(*i*) *for $n = 0, \ldots, T-1$,*

$$\mathbb{P}(X_{n+1}, \ldots, X_T | J_{n+1}) = \mathbb{P}(X_{n+1}|J_{n+1})\mathbb{P}(X_{n+2}, \ldots, X_T|J_{n+1});$$

(*ii*) *for $n = 1, \ldots, T-1$,*

$$\mathbb{P}(X_{n+1}, \ldots, X_T | J_{n+1}) = \mathbb{P}(X_{n+1}, \ldots, X_T | J_n, J_{n+1}).$$

Proof. (*i*) This is an easy consequence of Equation (1.1). Indeed, we have

$$\begin{aligned}
&\mathbb{P}(X_{n+1}, J_{n+1}, \ldots, X_T, J_T) \\
=&\mathbb{P}(X_{n+1}|J_{n+1}) \left(\mathbb{P}(J_{n+1}) \prod_{k=n+2}^{T} \mathbb{P}(J_k|J_{k-1}) \prod_{k=n+2}^{T} \mathbb{P}(X_k|J_k) \right) \quad \text{(by (1.1))} \\
=&\mathbb{P}(X_{n+1}|J_{n+1})\mathbb{P}(X_{n+2}, J_{n+2}, \ldots, X_T, J_T),
\end{aligned}$$

then, summing over all possible values for $(J_{n+2}, \ldots, J_T)$ and dividing by $\mathbb{P}(J_{n+1})$ led to the statement (*i*).

(*ii*) In order to prove the second relation, we show that the expressions at each side of the equality to be proved can be reduced to the same one. The right-hand side of the equation can be written as

$$\begin{aligned}
&\frac{\mathbb{P}(X_{n+1}, \ldots, X_T, J_n, J_{n+1})}{\mathbb{P}(J_n, J_{n+1})} \\
&= \frac{1}{\mathbb{P}(J_n, J_{n+1})} \sum_{j_{n+2}, \ldots, j_T = 1}^{m} \mathbb{P}(J_n, X_{n+1}, J_{n+1}, \ldots, X_T, J_T),
\end{aligned}$$

which, by equation (1.1), reduces to

$$\sum_{j_{n+2}, \ldots, j_T = 1}^{m} \prod_{k=n+2}^{T} \mathbb{P}(J_k|J_{k-1}) \prod_{k=n+1}^{T} \mathbb{P}(X_k|J_k).$$

The left-hand side is

$$\begin{aligned}
&\frac{1}{\mathbb{P}(J_{n+1})} \sum_{j_{n+2}, \ldots, j_T = 1}^{m} \mathbb{P}(X_{n+1}, J_{n+1}, \ldots, X_T, J_T) \\
&= \frac{1}{\mathbb{P}(J_{n+1})} \sum_{j_{n+2}, \ldots, j_T = 1}^{m} \mathbb{P}(J_{n+1}) \prod_{k=n+2}^{T} \mathbb{P}(J_k|J_{k-1}) \prod_{k=n+1}^{T} \mathbb{P}(X_k|J_k) \quad \text{(by (1.1))},
\end{aligned}$$

which is the same expression.

□

Proposition 1.4.3. *For $n = 1, \dots, T-1$ and $j = 1, \dots, m$, it holds*

$$\beta_n(j) = \mathbb{P}(X_{n+1} = x_{n+1}, \dots, X_T = x_T | J_n = j),$$

provided that $\mathbb{P}(J_n = j) > 0$.

Proof. The proof is done by induction using results of Lemma 1.4.2. We first establish validity for $n = T - 1$.
Since $\beta_{T-1} = \Gamma P(x_T)\mathbf{e}$, we have

$$\beta_{T-1}(j) = \sum_{i=1}^{m} \mathbb{P}(J_T = i | J_{T-1} = j)\mathbb{P}(X_T = x_T | J_T = i).$$

Now, by the second statement of Lemma 1.4.2,

$$\mathbb{P}(J_T | J_{T-1})\mathbb{P}(X_T | J_T) = \mathbb{P}(J_T | J_{T-1})\mathbb{P}(X_T | J_{T-1}, J_T) = \frac{\mathbb{P}(X_T, J_{T-1}, J_T)}{\mathbb{P}(J_{T-1})},$$

so, by substituting it in the previous equality we obtain

$$\begin{aligned}\beta_{T-1}(j) &= \frac{1}{\mathbb{P}(J_{T-1} = j)} \sum_{i=1}^{m} \mathbb{P}(X_T = x_T, J_{T-1} = j, J_T = i) \\ &= \frac{\mathbb{P}(X_T = x_T, J_{T-1} = j)}{\mathbb{P}(J_{T-1} = j)} \\ &= \mathbb{P}(X_T = x_T | J_{T-1} = j),\end{aligned}$$

as required.
To prove the inductive step (i.e., validity for $n+1$ implies validity for n), we note that the recursion of Remark 1.2.5 for β_n and the inductive hypothesis establish that

$$\begin{aligned}\beta_n(j) = \sum_{i=1}^{m} \gamma_{ji} \mathbb{P}(X_{n+1} = x_{n+1} | J_{n+1} = i) \\ \times \mathbb{P}(X_{n+2} = x_{n+2}, \dots, X_T = x_T | J_{n+1} = i).\end{aligned}$$

Then, Lemma 1.4.2 implies that

$$\mathbb{P}(X_{n+1} | J_{n+1})\mathbb{P}(X_{n+2}, \dots, X_T | J_{n+1}) = \mathbb{P}(X_{n+1}, \dots, X_T | J_n, J_{n+1}).$$

Now, if we substitute the last relation in the previous one the result is

$$\begin{aligned}\beta_n(j) &= \sum_{i=1}^{m} \mathbb{P}(J_{n+1}=i|J_n=j) \\ &\qquad \times \mathbb{P}(X_{n+1}=x_{n+1},\dots,X_T=x_T|J_n=j,J_{n+1}=i) \\ &= \frac{1}{\mathbb{P}(J_n=j)} \sum_{i=1}^{m} \mathbb{P}(X_{n+1}=x_{n+1},\dots,X_T=x_T,J_n=j,J_{n+1}=i) \\ &= \frac{\mathbb{P}(X_{n+1}=x_{n+1},\dots,X_T=x_T,J_n=j)}{\mathbb{P}(J_n=j)},\end{aligned}$$

which is the required conditional probability. □

We now establish an interesting result relating the product of each component of the forward and backward probabilities; further, it gives an alternative way to express the likelihood. It is used in applying the Expectation Maximization algorithm to Hidden Markov models.

Proposition 1.4.4. *Given a Hidden Markov model Δ and an integer $T>0$, for $n=1,\dots,T$ and $j=1,\dots,m$, it holds*

$$\alpha_n(j)\beta_n(j) = \mathbb{P}(X_1=x_1,\dots,X_T=x_T,J_n=j).$$

Consequently $\alpha_n\beta_n = \mathbb{P}(X_1=x_1,\dots,X_T=x_T) = L_T$ for each such n.

Proof. By the relations established in Proposition 1.4.1 and in Proposition 1.4.3, we have

$$\begin{aligned}\alpha_n(j)\beta_n(j) &= \mathbb{P}(X_1,\dots,X_n,J_n=j)\mathbb{P}(X_{n+1},\dots,X_T|J_n=j) \\ &= \mathbb{P}(J_n=j)\mathbb{P}(X_1,\dots,X_n|J_n=j)\mathbb{P}(X_{n+1},\dots,X_T|J_n=j).\end{aligned}$$

Now, we apply the conditional independence of $(X_1,\dots,X_n)$ and $(X_{n+1},\dots,X_T)$ given J_n, stated in Proposition 1.1.2; we obtain

$$\begin{aligned}\alpha_n(j)\beta_n(j) &= \mathbb{P}(J_n=j)\mathbb{P}(X_1,\dots,X_n,X_{n+1},\dots,X_T|J_n=j) \\ &= \mathbb{P}(X_1,\dots,X_T,J_n=j).\end{aligned}$$

Summation of this equation over $j=1,\dots,m$ yields the second conclusion. □

Finally, we prove two particular relations, concerning the conditional probability of the hidden chain given the observed process. Even if they can appear meaningless from a practical point of view, they result to have a central role in applying the Expectation Maximization algorithm to Hidden Markov models.

Proposition 1.4.5. *Let Δ be a Hidden Markov model and let $T > 0$ be a fixed integer. For $n = 1, \dots, T$, for $j = 1, \dots, m$, it holds*

$$\mathbb{P}(J_n = j | X_1 = x_1, \dots, X_T = x_T) = \frac{\alpha_n(j)\beta_n(j)}{L_T}.$$

Moreover, for $n = 2, \dots, T$, for $j, k \in \{1, \dots, m\}$, we have

$$\mathbb{P}(J_{n-1} = j, J_n = k | X_1 = x_1, \dots, X_T = x_T) = \frac{\alpha_{n-1}(j)\gamma_{jk}p_k(x_n)\beta_n(k)}{L_T}.$$

Proof. The first assertion follows immediately from the Proposition 1.4.4 above applying the definition of conditional probability.
For the second one, the proof proceeds as follows.

$$\mathbb{P}(J_{n-1} = j, J_n = k | X_1 = x_1, \dots, X_T = x_T)$$

$$= \frac{\mathbb{P}(J_{n-1} = j, J_n = k, X_1 = x_1, \dots, X_T = x_T)}{L_T}$$

$$= \frac{\mathbb{P}(X_1, \dots, X_{n-1}, J_{n-1} = j)\mathbb{P}(J_n = k | J_{n-1} = j)\mathbb{P}(X_n, \dots, X_T | J_n = k)}{L_T}$$

(by equation (1.1))

$$= \frac{\alpha_{n-1}(j)\gamma_{jk}\mathbb{P}(X_n | J_n = k)\mathbb{P}(X_{n+1}, \dots, X_T | J_n = k)}{L_T}$$

(by Lemma 1.4.2)

$$= \frac{\alpha_{n-1}(j)\gamma_{jk}p_k(x_n)\beta_n(k)}{L_T}.$$

□

1.4.1 The Expectation Maximization algorithm

The Expectation Maximization algorithm (EM algorithm) is a general iterative method for performing maximum likelihood estimation when the observations can be viewed as incomplete data. The term "incomplete data", in its general form, implies the existence of two sample spaces; the observed data are realisations of one of them, while the others are not observed directly, but only indirectly trough the first ones.
The algorithm exploits the fact that the complete-data log-likelihood may be straightforward to maximize even if the likelihood of the observed data is

not, where by "complete-data log-likelihood" we mean the log-likelihood of the parameters of interest $\theta \in \Theta$ (for a certain parameter space Θ), based on both the observed data and the missing data.
We now present an informal description of the algorithm. The first step is choosing starting values for the parameters θ to be estimated. Then, repeating the following steps until some convergence criterion has been satisfied, for example, until the resulting change in θ is less than some threshold.

E step Compute the expectations of those functions of the missing data that appear in the complete-data log-likelihood, given the observations and given the current estimates of θ.

M step Maximize, with respect to θ, the complete-data log-likelihood with the functions of the missing data replaced in it by their expectations previously found.

The key idea of Expectation Maximization algorithm is that it is not (necessarily) the missing data themselves that are replaced in the complete-data log-likelihood by their expectations, but those functions of the missing data that appear in it.
Before illustrating how this algorithm works for Hidden Markov models, we analyse the Expectation Maximization algorithm in its general form, also mentioning some of the convergence properties ([19]).
In the following, we denote respectively as y, y_{obs} and y_{mis} the complete data, the observed data and the missing data, so that $y = y_{obs} \cup y_{mis}$. Then, let $f(y|\theta)$ be the density of complete data y given the parameters θ.
From the definition of conditional probability density function, we can show it can be factored as

$$f(y|\theta) = f(y_{obs}, y_{mis}|\theta) = f(y_{obs}|\theta) f(y_{min}|y_{obs}, \theta),$$

where $f(y_{obs}|\theta)$ is the density of the observed data and $f(y_{min}|y_{obs}, \theta)$ is the conditional density of the missing data given the observed data and the parameters. Moreover, if $l(\theta|\ \cdot\ ) := \log(f(\ \cdot\ |\theta))$ denotes the log-likelihood, we have that the corresponding decomposition of log-likelihood $l(\theta|y)$ is then

$$l(\theta|y) = l(\theta|y_{obs}, y_{mis}) = l(\theta|y_{obs}) + \log(f(y_{min}|y_{obs}, \theta)).$$

In practical applications, the objective is to estimate θ by maximizing the incomplete-data log-likelihood $l(\theta|y_{obs})$ with respect to θ for the available observations y_{obs}; this task, however, can be difficult to accomplish directly and the following procedure gives a way to circumvent the problem.
The idea used is quite simple: we begin by writing

$$l(\theta|y_{obs}) = l(\theta|y) - \log f(y_{mis}|y_{obs}, \theta), \tag{1.3}$$

and attempting to maximize the right hand side of the equation. This is made through the steps of the Expectation Maximization algorithm. Specifically, let $\theta^{(t)}$ be the current estimate of the parameter, at each iteration $t \geq 0$ of the algorithm. In the E step, we compute the expectation of the complete-data log-likelihood $l(\theta|y)$ if θ were $\theta^{(t)}$, which is given by

$$Q(\theta|\theta^{(t)}) := \mathbb{E}[l(\theta^{(t)}|y_{obs}, y_{mis})] = \int l(\theta|y) f(y_{min}|y_{obs}, \theta^{(t)}) dy_{mis}.$$

Then the M step determines a new value for θ by maximizing this expected complete-data log-likelihood: that is,

$$\theta^{(t+1)} = \underset{\theta}{\operatorname{argmax}} \ \ Q(\theta|\theta^{(t)}).$$

In other words, in the M step, we look for $\theta^{(t+1)}$ such that

$$Q(\theta^{(t+1)}|\theta^{(t)}) \geq Q(\theta|\theta^{(t)}), \qquad \text{for all } \theta.$$

Different ways of performing this optimization are available and they depend on the specific context.
The following considerations show how this procedure gives, at each iteration, a greater value for the log-likelihood of observations $l(\theta|y_{obs})$.
If we compute the expectation of both sides of expression (1.3) over the distribution of the missing data y_{mis}, given the observed data y_{obs} and given a current estimate of θ, $\theta^{(t)}$, we obtain

$$l(\theta|y_{obs}) = Q(\theta|\theta^{(t)}) - H(\theta|\theta^{(t)}),$$

where

$$\begin{aligned} H(\theta|\theta^{(t)}) &:= \mathbb{E}[\log f(y_{mis}|y_{obs}, \theta^{(t)})] \\ &= \int \log f(y_{mis}|y_{obs}, \theta) f(y_{mis}|y_{obs}, \theta^{(t)}) dy_{mis}. \end{aligned}$$

Moreover, it is easy to show that the following property holds.

Remark 1.4.6. By Jensen's inequality, we have that for every θ,

$$H(\theta|\theta^{(t)}) \leq H(\theta^{(t)}|\theta^{(t)}),$$

with equality if and only if $f(y_{mis}|y_{obs}, \theta^{(t)}) = f(y_{mis}|y_{obs}, \theta)$.

Now, let us consider a sequence of iterates $\theta^{(0)}, \theta^{(1)}, \ldots$, where for each $t \geq 0$, $\theta^{(t)}$ is the parameter estimate obtained at iteration t of the algorithm

and, so, $\theta^{(t+1)}$ is a function of $\theta^{(t)}$, say $\theta^{(t+1)} = M(\theta^{(t)})$. The differences in value of $l(\theta|y_{obs})$ at successive iterates are then given by

$$\begin{aligned} l(\theta^{(t+1)}|y_{obs}) - l(\theta^{(t)}|y_{obs}) =& [Q(\theta^{(t+1)}|\theta^{(t)}) - Q(\theta^{(t)}|\theta^{(t)})] \\ & - [H(\theta^{(t+1)}|\theta^{(t)}) - H(\theta^{(t)}|\theta^{(t)})]. \end{aligned} \tag{1.4}$$

Due to the fact that the Expectation Maximization algorithm described above chooses $\theta^{(t+1)}$ to maximize $Q(\theta|\theta^{(t)})$ with respect to θ, the difference of Q functions in equation (1.4) is non-negative. Furthermore, the difference in the H functions is negative by remark 1.4.6. So,

$$l(\theta^{(t+1)}|y_{obs}) - l(\theta^{(t)}|y_{obs}) \geq 0,$$

i.e., the change from $\theta^{(t)}$ to $\theta^{(t+1)}$ increases the log-likelihood.
We have thus proved the following theorem.

Theorem 1.4.7. *The Expectation Maximization algorithm increases the incomplete log-likelihood $l(\theta|y_{obs})$ at each iteration, that is,*

$$l(\theta^{(t+1)}|y_{obs}) \geq l(\theta^{(t)}|y_{obs}),$$

with equality if and only if $Q(\theta^{(t+1)}|\theta^{(t)}) = Q(\theta^{(t)}|\theta^{(t)})$ and $f(y_{mis}|y_{obs}, \theta^{(t)}) = f(y_{mis}|y_{obs}, \theta)$.

Theorem 1.4.7 implies that the log-likelihood is non decreasing after each iteration of the Expectation Maximization algorithm, and is strictly increasing after any iteration such that Q increases.
For the sake of knowledge, we also state the following theorem which gives a remarkable convergence property of the Expectation Maximization algorithm.

Theorem 1.4.8. *Suppose that $\theta^{(t)}$ for $t \geq 0$ is an instance of the Expectation Maximization algorithm such that:*

(i) the sequence $l(\theta^{(t)}|y_{obs})$ is bounded;

(ii) $Q(\theta^{(t+1)}|\theta^{(t)}) - Q(\theta^{(t)}|\theta^{(t)}) \geq \lambda(\theta^{(t+1)} - \theta^{(t)})$ for some scalar $\lambda > 0$ and all $t > 0$.

Then the sequence $\theta^{(t)}$ converges to some θ^* in the closure of the parameter space Θ.

A proof can be found in [15].

1.4.2 The Expectation Maximization algorithm for Hidden Markov model

We now show how the Expectation Maximization algorithm can be extended to maximize the likelihood of a Hidden Markov model Δ. Since the sequence of states occupied by the Markov chain of Δ is not directly observed, a very natural approach for parameter estimation is to treat those states as missing data and to employ this algorithm in order to find the parameters which realize the likelihood maximum. As anticipated, results given at the beginning of this section play a key role in the algorithm for Hidden Markov models.
We begin by considering the complete-data log-likelihood of a Hidden Markov model, i.e., the log-likelihood of an observation sequence $(x_1, \ldots, x_T)$ together with the missing data $(j_1, \ldots, j_T)$:

$$\begin{aligned}
&\log \mathbb{P}(X_1 = x_1, J_1 = j_1, \ldots, X_T = x_T, J_T = j_T) \\
&= \log \left(\delta_{j_1} \prod_{n=2}^{T} \gamma_{j_{n-1} j_n} \prod_{n=1}^{T} p_{j_n}(x_n) \right) \\
&= \log(\delta_{j_1}) + \sum_{n=2}^{T} \log(\gamma_{j_{n-1} j_n}) + \sum_{n=1}^{T} \log(p_{j_n}(x_n)),
\end{aligned}$$

where we have used the relation $\stackrel{\circ}{=}$ from the proof of Proposition 1.2.1.
In this context, it is convenient to represent the sequence of state $j_1, \ldots, j_T$ followed by the Markov chain by the following random variables, defined for $j, k = 1, \ldots, m$:

$$\begin{aligned}
u_j(n) &:= \mathbb{I}_{\{j_n = j\}} && \text{for } n = 1, \ldots, T; \\
v_{jk}(n) &:= \mathbb{I}_{\{j_{n-1} = j\} \cap \{j_n = k\}} && \text{for } n = 2, \ldots, T.
\end{aligned}$$

With this notation, the complete-data log-likelihood of a Hidden Markov model becomes

$$\begin{aligned}
&\log \mathbb{P}(X_1 = x_1, J_1 = j_1, \ldots, X_T = x_T, J_T = j_T) \\
&= \sum_{j=1}^{m} u_j(1) \log \delta_j + \sum_{j=1}^{m} \sum_{k=1}^{m} \left(\sum_{n=2}^{T} v_{jk}(n) \right) \log \gamma_{jk} + \sum_{j=1}^{m} \sum_{n=1}^{T} u_j(n) \log(p_j(x_n)).
\end{aligned} \tag{1.5}$$

Thus, the states of the Markov chain only appear in the new random variables introduced, so $u_j(n)$ and $v_{jk}(n)$ are the functions of the missing data to be replaced by their expectations in applying the E step of the Expectation Maximization algorithm. More precisely, at each iteration, the algorithm proceeds in this case as follows:

E step Replace all the quantities $u_j(n)$ and $v_{jk}(n)$ by their expectations, computed given the observations $(x_1, \ldots, x_T)$ and the current parameter estimates. They are given by

$$\begin{aligned}\hat{u}_j(n) &:= \mathbb{E}[u_j(n)|X_1 = x_1, \ldots, X_T = x_T] \\ &= \mathbb{P}(J_n = j|X_1 = x_1, \ldots, X_T = x_T) \\ &= \frac{\alpha_n(j)\beta_n(j)}{L_T};\end{aligned}$$

and

$$\begin{aligned}\hat{v}_{jk}(n) &:= \mathbb{E}[v_{jk}(n)|X_1 = x_1, \ldots, X_T = x_T] \\ &= \mathbb{P}(J_{n-1} = j, J_n = k|X_1 = x_1, \ldots, X_T = x_T) \\ &= \frac{\alpha_{n-1}(j)\gamma_{jk}p_k(x_n)\beta_n(k)}{L_T};\end{aligned}$$

where the last equalities of both relations follow from Proposition 1.4.5.

M step Having replaced $u_j(n)$ and $v_{jk}(n)$ by their expectations, maximize the complete-data log-likelihood (1.5) with respect to the three sets of parameters: the initial distribution δ, the transition probability matrix Γ and the parameters of the state-dependent distributions.

We note that equation (1.5) is the sum of three different terms in each of which only one of the parameters to be estimated appears. Furthermore, since each term is sum of the product of indicator functions and logarithm of probabilities, the complete-data log-likelihood turns out to be the sum of negative quantities, so the M step can be effectively split into three separate maximizations. The first and the second term are maximized with respect to δ and Γ, respectively. This can be easily achieved using a Lagrange multiplier, which eventually gives:

$$\delta_j = \hat{u}_j(1) \quad \text{and} \quad \gamma_{jk} = f_{jk} / \sum_{k=1}^{m} f_{jk} \qquad \text{for } j, k = 1, \ldots, m, \tag{1.6}$$

where $f_{jk} = \sum_{n=2}^{T} \hat{v}_{jk}(n)$.

The maximization of the third term may be easy or difficult, depending on the nature of the state-dependent distributions assumed. It is essentially the standard problem of maximum likelihood estimation for the distribution concerned.

1.4.3 Direct maximization of the likelihood

We have already discussed in Section 1.3 the problem of likelihood evaluation and the related techniques adopted to prevent under-flow (or, over-flow). However, in performing numerical methods for likelihood maximization, some others difficulties arise: mainly, parameters to be estimated are usually subject to constraints and, further, multiple local maxima in the likelihood function may exist.
In general, there are two groups of constraints: those that apply to the parameters of the state-dependent distributions and those that apply to the parameters of the Markov chain. The latter are common in all the Hidden Markov models: the transition probability matrix Γ must be a stochastic matrix and the vector δ must be a distribution, i.e., $\delta_j \geq 0$, $\gamma_{ij} \geq 0$, $\sum_{j=1}^{m} \delta_j = 1$ and $\sum_{j=1}^{m} \gamma_{ij} = 1$ for $i, j \in \{1, \ldots, m\}$. The former depend instead on which state-dependent distributions are chosen and, so, on the nature of the parameters to be estimated. There are several special-purpose software to maximize a function of several variables which are subject to constraints. However, they can be very slow and alternative strategies can be adopted: in this case, constraints can be imposed by making transformations on the parameters. As an example, we illustrate how the re-parametrization of Γ can be accomplished. First of all, the transition probability matrix has m^2 entries, but only $m(m-1)$ free parameters as the sum over each row must be equal to 1. So, it is possible to transform the m^2 constrained entries γ_{ij} in $m(m-1)$ unconstrained real numbers τ_{ij}, with $i \neq j$.
We begin by considering a strictly increasing function $g : \mathbb{R} \mapsto \mathbb{R}^+$, typical choices for g are

$$g(x) = e^x \qquad \text{or} \qquad g(x) = \begin{cases} e^x & \text{if } x \leq 0, \\ x + 1 & \text{if } x \geq 0. \end{cases}$$

We continue by defining

$$\nu_{ij} = \begin{cases} g(\tau_{ij}) & \text{for } i \neq j, \\ 1 & \text{for } i = j. \end{cases}$$

We then set $\gamma_{ij} = \nu_{ij} / \sum_{k=1}^{m} \nu_{ik}$ for $i, j = 1, \ldots, m$ and $\Gamma = (\gamma_{ij})_{i,j=1,\ldots,m}$. The maximization procedure is thus performed by unconstrained numerical methods with respect to the modified parameters τ_{ij} for $i, j = 1, \ldots, m$, $i \neq j$. Subsequently, the natural parameters are recovered by the inverse of the transformation made for the elements with $i \neq j$ and by using the restriction on the sum of the row of Γ for diagonal elements.
Another issues that worth mentioning is that the likelihood of a Hidden

Markov model is a complicated function of the parameters and frequently has several local maxima. The goal of course is to find the global maximum, but there is no simple method of determining in general whether a numerical maximization algorithm has reached the global maximum. Depending on the starting values, it can easily happen that the algorithm identifies a local, but not the global, maximum. A sensible strategy is therefore to use a range of starting values for the maximization, and to see whether the same maximum is identified in each case.
Finally, in the case of Hidden Markov models with continuous state-dependent distributions, it may happen that the likelihood is unbounded in the vicinity of certain parameter combinations. For instance, in case of a normal state-dependent density function, the likelihood becomes arbitrarily large if one sets a component mean equal to one of the observations and allows the corresponding variance to tend to zero. This problem would not arise if we replace each density value in the likelihood by the probability of the interval corresponding to the recorded value. Indeed, in this case, the likelihood would be a probability and so it would be in $[0, 1]$. So, one can replace the state-dependent density functions $p_j(x_n)$ for $j = 1, \ldots, m$ and $n = 1, \ldots, T$ that appear in the likelihood expression (1.2), by

$$p_j(x_n) \approx \int_{a_n}^{b_n} p_j(x)dx,$$

where the interval (a_n, b_n) consists of those values which, if observed, would be recorded as x_n.

1.5 Decoding problem

In this last section, we focus on the problem of determine the states of the Markov chain that are most likely (under the fitted model) to have given rise to the observation sequence; even if, we will not use it in the following, this is of great interest in many applications. Given a sequence of observations $x = (x_1, \ldots, x_T)$ we can distinguish two different kinds of decoding: *local decoding* of the state at time n refers to the determination of that state which is most likely, for the sequence x at that time; in contrast, *global decoding* refers to the determination of the most likely sequence of states $(j_1, \ldots, j_T)$. The goal of local decoding is then determining, given the observations up to time T, $(x_1, \ldots, x_T)$, the most likely state j_n^* for each $n = 1, \ldots, T$. It is defined as

$$j_n^* := \underset{j=1,\ldots,m}{\operatorname{argmax}} \ \mathbb{P}(J_n = j | X_1 = x_1, \ldots, X_T = x_T).$$

The probability on the right hand side can be obtained by mean of the forward and backward probabilities by Proposition 1.4.5:

$$\mathbb{P}(J_n = j | X_1 = x_1, \ldots, X_T = x_T) = \frac{\alpha_n(j)\beta_n(j)}{L_T}.$$

Here L_T can be computed by the scaling method described in Section 1.3. Scaling is also necessary in order to prevent numerical underflow in the evaluation of the product $\alpha_n(j)\beta_j(n)$. Furthermore, for each $n = 1, \ldots, T$, we can determine the distribution of the state J_n, given the observations $(x_1, \ldots, x_T)$, which for m states is a discrete probability distribution with support $\{1, \ldots, m\}$.
Nevertheless, in applications, one is usually interested not so much in the most likely state for each separate n (as provided by local decoding) but in the most likely sequence of hidden states. Instead of maximizing over j $\mathbb{P}(J_n = j | X_1 = x_1, \ldots, X_T = x_T)$ for each n, one seeks that sequence of states $j_1, \ldots, j_T$ which maximizes the conditional probability

$$\mathbb{P}(J_1 = j_1, \ldots, J_T = j_T | X_1 = x_1, \ldots, X_T = x_T); \tag{1.7}$$

or equivalently, the joint probability

$$\mathbb{P}(J_1 = j_1, \ldots, J_T = j_T, X_1 = x_1, \ldots, X_T = x_T) = \delta_{j_1} \prod_{n=2}^{T} \gamma_{j_{n-1} j_n} \prod_{n=1}^{T} p_{j_n}(x_n).$$

This is the basics of the so-called global decoding. The results of local and global decoding are often very similar but not identical.
To determine the most likely sequence of states, one can use an efficient dynamic programming algorithm, the so-called *Viterbi algorithm*. Indeed, maximizing (1.7) over all possible states $j_1, \ldots, j_T$ is feasible only for small T as it would involve m^T function evaluations.

1.5.1 The Viterbi algorithm

The Viterbi algorithm was conceived by Andrew Viterbi in 1967 ([23]) as a decoding algorithm for convolution codes over noisy digital communication links. It is a dynamic programming algorithm and it has now a lot of applications.
In applying the algorithm to find the most likely sequence of hidden states of a Hidden Markov model which have given rise to a sequence of observations $(x_1, \ldots, x_T)$, we begin by defining T auxiliary variables for each state of the

chain. These give the maximum values for the joint probability of partial observations sequence and state sequence up to time n when the current state is j:

$$\xi_{1j} := \mathbb{P}(J_1 = j, X_1 = x_1) = \delta_j p_j(x_1),$$

and, for $n = 2, \ldots, T$,

$$\xi_{nj} := \max_{j_1,\ldots,j_{n-1}} \mathbb{P}(X_1 = x_1, J_1 = j_1, \ldots, J_{n-1} = j_{n-1}, X_n = x_n, J_n = j).$$

The following recursion follows easily by induction on n using the definitions of Markov chain and Hidden Markov models. It holds for $n = 2, \ldots, T$ and $i = 1, \ldots, m$:

$$\xi_{ni} = \left(\max_j (\xi_{n-1,j} \gamma_{ji}) \right) p_i(x_n). \tag{1.8}$$

The idea of the Viterbi algorithm is to use the above relation to compute, for each $n = T, \ldots, 1$, the state j_n which gives the maximum of ξ_{nj}, i.e., $j_n = \underset{j=1,\ldots,m}{\operatorname{argmax}} \ \xi_{nj}$. The algorithm keeps a pointer to the "winning state" in the maximum finding operation. It results in state j_T, where $j_T = \underset{j=1,\ldots,m}{\operatorname{argmax}} \ \xi_{Tj}$, then it starts from this state and back-track the sequence of states as the pointer in each state indicates. At the end, we get the required set of states. Formally, relation (1.8) provides an efficient mean of computing the $T \times m$ matrix of values ξ_{nj}, as the computational effort is linear in T. In fact, the required maximization sequence $j_1, \ldots, j_T$ can be determined recursively from

$$j_T = \underset{j=1,\ldots,m}{\operatorname{argmax}} \ \xi_{Tj},$$

and, for $n = T - 1, \ldots, 1$,

$$j_n = \underset{j=1,\ldots,m}{\operatorname{argmax}} \ (\xi_{nj} \gamma_{j,j_{n+1}}).$$

We can note that, since the quantity to be maximized in global decoding is simply a product of probabilities (as opposed to a sum of such products), one can choose to maximize its logarithm, in order to prevent numerical underflow.

Chapter 2

Approximate Bayesian Computation

In all model-based statistical inference, the likelihood function is of central importance, as, inter alia, the most popular methods for fitting models involve maximum likelihood estimation. However, for an increasing range of scientific problem, it may be difficult to obtain an analytical formula or the likelihood function may be computationally very expensive to evaluate. Even in Hidden Markov model's context we may not be able to write down an explicit expression for the likelihood due to, for example, the lack of a definite relation for the state-dependent distributions. Approximate Bayesian Computation (ABC) constitutes a class of Bayesian methods, which has emerged as an effective and intuitively accessible way of performing parameters estimation and model selection, without directly estimating likelihood. As being part of the class of the so-called "likelihood-free" methods, Approximate Bayesian Computation considers an approximation to the preferred model, so that modelling realism is maintained at the expense of some approximation error.
In Bayesian inference, complete knowledge about a vector of model parameters, $\theta \in \Theta$, obtained by fitting a model M, is contained in the posterior distribution. Here, prior beliefs about the model parameters, as expressed through the prior distribution, $\pi(\theta)$, are updated by observing data y_{obs} through the likelihood function $l(y_{obs}|\theta) := \mathbb{P}(y_{obs}|\theta)$ of the model. [1]
Using Bayes' theorem, the resulting posterior distribution:

$$\pi(\theta|y_{obs}) = \frac{l(y_{obs}|\theta)\pi(\theta)}{\int_\Theta l(y_{obs}|\theta)\pi(\theta)d\theta},$$

[1] Note that here $l(y_{obs}|\theta)$ denotes the likelihood and not the log-likelihood as in Chapter 1.

contains all necessary information required for analysis of the model. Typically, due to the complexity of the model or prior, the posterior distribution is not available in closed form, so numerical methods are used to proceed with the inference. In order to enumerate the necessary integrals, Monte Carlo integration is often used. This essentially relies on the ability to draw samples $\theta^{(1)}, \ldots, \theta^{(N)} \sim \pi(\theta|y_{obs})$ from the posterior distribution and to use them to approximate the true posterior distribution by the law of large numbers. There are a number of popular algorithms available for generating samples from posterior distributions, such as importance sampling, Markov Chain Monte Carlo (MCMC) and sequential Monte Carlo (SMC). The problem is that each of them requires evaluation of the likelihood function, but, its numerical evaluation is often either computationally prohibitive, or simply not possible. Examples can occur where the size of the observed dataset, say y_{obs}, is sufficiently large that, in the absence of low dimensional sufficient statistics, evaluating the likelihood function even once is impracticable. "Likelihood-free" models are intended to circumvent likelihood evaluation; in particular, Approximate Bayesian Computation are those likelihood-free methods that produce an approximation to the posterior distribution resulting from the imperfect matching of data. In the next section, we illustrate, following the analysis in [2, 21], the first example of likelihood-free algorithm, that is then generalized and extended to give its Approximate Bayesian Computation version.

2.1 Likelihood-free rejection sampling algorithm

The basic form of likelihood-free method is the *rejection sampling algorithm*. The goal of the algorithm is producing samples θ in a fixed space Θ, from a known density function (for which direct sampling is infeasible), by mean of another density function. Its main scheme can be informally described as follows. At first, a set of parameter points is first sampled from the prior distribution chosen. Then, given a sampled parameter point θ, a dataset y is simulated under the statistical model M specified by θ. Finally, the sampled parameter values are either discarded or accepted according whether sample data match the observed ones y_{obs} or not. Let us explain it in detail.

For the sake of simplicity, for the moment, we assume that data generated under the model under consideration are discrete. At first, we consider the standard rejection sampling algorithm from a general density $f(\theta)$. The key requirement of the method is the availability of another probability density function $g(\theta)$ whose functional form is known and from which independent

and identically distributed sampling is readily feasible.
This algorithm receives as inputs the target density $f(\theta)$, a sampling density $g(\theta)$, from which we *can* sample values for θ, such that $g(\theta) > 0$ if $f(\theta) > 0$ and an integer $N > 0$. Then, for each $i = 1, \ldots, N$ the following steps are repeated:

1. generate $\theta^{(i)} \sim g(\theta)$, from the sampling density g;
2. accept $\theta^{(i)}$ with probability $\frac{f(\theta^{(i)})}{Kg(\theta^{(i)})}$, where $K \geq \max_{\theta} \frac{f(\theta)}{g(\theta)}$, else go to (1).

The output is a set of parameter vectors $\theta^{(1)}, \ldots, \theta^{(N)}$ which are samples from $f(\theta)$.

In heuristic terms, in performing the algorithm we assume that a constant $K > 1$ exists such that,

$$Kg(\theta) \geq f(\theta),$$

for all $\theta \in \Theta$. Then the idea is sampling uniformly under the graph of $Kg(\theta)$ and accepting only those samples that fall under the graph of f. As clarified later, this is achieved, in practice, by accepting θ with probability

$$\frac{f(\theta^{(i)})}{Kg(\theta^{(i)})}.$$

In fact, it is straightforward to see, that given such a K, this second operation is the same as generating a realization of an uniform random variable on $[0, 1]$, $u \sim U([0, 1])$, independently from θ, and accepting the pair (θ, u) if

$$uKg(\theta) \leq f(\theta).$$

According to that, the algorithm's steps can be re-written as follows:

1. generate, independently, $\theta \sim g(\theta)$ and $u \sim U([0, 1])$;
2. accept the pair (θ, u) if and only if $u \leq \frac{f(\theta)}{Kg(\theta)}$.

This second version allows to prove the actual efficacy of the rejection sampling algorithm in sampling from a known density function. Proposition 2.1.1 and Lemma 2.1.2 below, stated in general, make this procedure rigorous.

Proposition 2.1.1. *Let ξ be a random variable with density f with respect to a measure λ on Θ and U be an independent real random variable uniformly distributed on the interval $[0, K]$. Then the pair $(\xi, Uf(\xi))$ of random variables is uniformly distributed under the graph of $Kf(\theta)$, that is*

$$\mathcal{G}_{f,K} := \{(\theta, u) \in \Theta \times \mathbb{R}^+ : 0 < u < Kf(\theta)\},$$

with respect to $\lambda \otimes \lambda^{leb}$, where λ^{leb} denotes the Lebesgue measure.
Conversely, if a random vector (ξ, U) of $\Theta \times \mathbb{R}^+$ is uniformly distributed on $\mathcal{G}_{f,K}$, then ξ admits f as marginal probability density function.

Proof. First of all, we note we can assume $K = 1$ without loss of generality, indeed if Proposition 2.1.1 is true for some value K_0, then both claims also hold for all values of $K > 0$ simply by scaling the ordinate by K/K_0.
In order to prove the first statement, let $B \subset \mathcal{G}_{f,1}$ a measurable subset and let us consider its section in θ, that is $B_\theta := \{u \in \mathbb{R}^+ : (\theta, u) \in B\}$. It is trivial to prove that $Uf(\xi)$ has density $\frac{1}{f(\theta)}$ with respect to the Lebesgue measure, so we have

$$\mathbb{P}((\xi, Uf(\xi)) \in B) = \int_{\theta \in \Theta} \int_{u \in B_\theta} \frac{1}{f(\theta)} \lambda^{leb}(du) f(\theta) \lambda(d\theta) = \int \int_B \lambda^{leb}(du) \lambda(d\theta),$$

which gives the thesis.
For the second statement, let us consider a measurable subset $A \subset \Theta$ and set $\bar{A} := \{(\theta, u) \in A \times \mathbb{R}^+ : 0 \leq u \leq f(\theta)\}$. Then, using the hypothesis of uniformity, we have

$$\mathbb{P}(\xi \in A) = \mathbb{P}((\xi, U) \in \bar{A}) = \frac{\int \int_{\bar{A}} \lambda^{leb}(du) \lambda(d\theta)}{\int \int_{\mathcal{G}_{f,1}} \lambda^{leb}(du) \lambda(d\theta)} = \int_A f(\theta) \lambda(d\theta).$$

□

Lemma 2.1.2. *Let $\{V_n\}_{n \geq 1}$ be a sequence of independent and identically distributed random variables taking values in a measurable space $(V, \mathcal{V})$ and $B \in \mathcal{V}$ a set such that $\mathbb{P}(V_1 \in B) = p > 0$ (and so, $\mathbb{P}(V_i \in B) = p$ for all $i \geq 1$).*
The integer-valued random variable $\sigma := \inf\{k \geq 1 : V_k \in B\}$ is geometrically distributed with parameter p, i.e., for all $i \geq 0$, it holds

$$\mathbb{P}(\sigma = i) = (1 - p)^{i-1} p.$$

Moreover, the random variable $V := V_\sigma \mathbb{I}_{\{\sigma < \infty\}}$ is distributed according to

$$\mathbb{P}(V \in A) = \frac{\mathbb{P}(V_1 \in A \cap B)}{p},$$

for all $A \in \mathcal{V}$.

We adopt the convention that $\inf \emptyset = +\infty$.

Proof. The first relation follows easily from the definition of σ, using the independence of V_i:

$$\mathbb{P}(\sigma = i) = \mathbb{P}(V_1 \notin B, V_2 \notin B, \dots, V_{i-1} \notin B, V_i \in B) = (1-p)^{i-1}p.$$

In particular, it implies that the waiting time σ is almost surely finite, namely

$$\mathbb{P}(\sigma < \infty) = \sum_{i=1}^{+\infty} \mathbb{P}(\sigma = i) = \frac{p}{1-p} \sum_{i=1}^{+\infty} (1-p)^i = 1.$$

In order to prove the second part of the lemma, we consider $A \in \mathcal{V}$ and we have

$$\begin{aligned}
\mathbb{P}(V \in A) &= \sum_{i=1}^{+\infty} \mathbb{P}(V_1 \notin B, V_2 \notin B, \dots, V_{i-1} \notin B, V_i \in A \cap B) \\
&= \sum_{i=1}^{+\infty} (1-p)^{i-1} \mathbb{P}(V_1 \in A \cap B) \\
&= \frac{1}{p} \mathbb{P}(V_1 \in A \cap B),
\end{aligned}$$

where we have used the fact that the $V_1, V_2, \dots, V_i$ are independent and identically distributed. □

Hence, by Proposition 2.1.1, the intermediate pairs $(\theta^{(i)}, u_i)$ generated in the second form of the algorithm at the i-th iteration are such that $(\theta^{(i)}, Ku_i g(\theta^{(i)}))$ are uniformly distributed under the graph of $Kg(\theta)$. By Lemma 2.1.2, if we set

$$V_k = (\theta^{(k)}, U_k) \quad \text{and} \quad B := \{(\theta, u) \in \Theta \times [0, K] : uKg(\theta) \leq f(\theta)\},$$

where B is the acceptance region, we have that the accepted pair (θ, u) is uniformly distributed under the graph of $f(\theta)$. Further, using Proposition 2.1.1, we have that θ is marginally distributed according to f, as we wanted. The probability p of acceptance is equal to

$$\mathbb{P}\left(U_1 \leq \frac{f(\theta_1)}{Kg(\theta_1)}\right) = \mathbb{P}((\theta_1, KU_1 g(\theta_1)) \in \mathcal{G}_{f,K}) = \frac{\int_\Theta f(\theta)\lambda(d\theta)}{\int_\Theta Kg(\theta)\lambda(d\theta)} = \frac{1}{K}.$$

Finally, we note that the choice

$$K = \max_\theta \frac{f(\theta)}{g(\theta)},$$

as stated in the first algorithm formulation, is the optimal one as it maximizes the acceptance probability for a given g and therefore minimizes the average required computational effort.

The rejection sampling algorithm can be easily used for sampling from the posterioir distribution. Specifically, if we specify $f(\theta) = \pi(\theta|y_{obs})$ and suppose that the prior is used as the sampling distribution, then the acceptance probability is proportional to the likelihood, as then

$$\frac{f(\theta)}{Kg(\theta)} = \frac{\pi(\theta|y_{obs})}{K\pi(\theta)} \propto l(y_{obs}|\theta).$$

The remarkable fact in this context is that it is possible to stochastically determine whether or not to accept or reject a draw from the sampling density, without numerical evaluation of the acceptance probability. A feature that turns to be essential when the likelihood is computationally intractable. This can be achieved by noting that the acceptance probability is proportional to the probability of generating the observed data, y_{obs}, under the model for a fixed parameter vector, θ. That is, for a fixed θ, if we generate a dataset from the model, then the probability of generating our observed dataset exactly, so that $y = y_{obs}$, is precisely $l(y_{obs}|\theta)$. From this observation, we can use the Bernoulli event of generating $y = y_{obs}$ (or not) to determine whether to accept (or reject) a draw from the sampling distribution, in lieu of directly evaluating the probability $l(y_{obs}|\theta) = \mathbb{P}(y_{obs}|\theta)$. We can then rewrite the simple rejection sampling algorithm for sampling from the posterior distribution without directly evaluating the likelihood.
In doing that, choosing the sampling distribution to be the prior is not essential: sampling may be done from a general density, $g(\theta)$, as well. In this case, the acceptance probability is proportional to $l(y_{obs}|\theta)\pi(\theta)/g(\theta)$ and whether to accept a draw form $g(\theta)$ can be split in two stages: as before, if we generate y such that $y \neq y_{obs}$, we reject the draw from $g(\theta)$. If instead $y = y_{obs}$, then we accept the draw from $g(\theta)$ with probability $\pi(\theta)/[Kg(\theta)]$, where $K \geq \max_{\theta} (\pi(\theta)/g(\theta))$.
The likelihood-free rejection sampling algorithm gets as inputs the target posterior density $\pi(\theta|y_{obs})$, consisting of a prior distribution $\pi(\theta)$ and a procedure of generating data under the model M, a proposal density $g(\theta)$, with $g(\theta) > 0$ if $\pi(\theta) > 0$ and an integer $N > 0$. Then, as in the standard case, the following steps are repeated for each $i = 1, \ldots, N$:

1. generate $\theta^{(i)} \sim g(\theta)$ from the sampling density $g(\theta)$;

2. generate data y under the model M with parameter vector $\theta^{(i)}$; in fact, it corresponds to generate $y \sim l(y|\theta^{(i)})$ from the likelihood;

3. if $y = y_{obs}$, then accept $\theta^{(i)}$ with probability $\frac{\pi(\theta^{(i)})}{Kg(\theta^{(i)})}$, where $K \geq \max_{\theta} \frac{\pi(\theta)}{g(\theta)}$, else go to (1).

As before, the output is a set of parameter vectors $\theta^{(1)}, \ldots, \theta^{(N)}$ which are samples from $\pi(\theta|y_{obs})$.

2.1.1 Refinements: introduction of a threshold

In practice, the likelihood-free rejection sampling algorithm as described above turns out to be very inefficient. It is not only due to a possible mismatch between prior and posterior distributions, but mainly it is because of the request of generating data from the model that exactly match the observed data y_{obs}. Indeed, the probability of generating data such that $y = y_{obs}$ can be very low and become zero if the data generated under the model are continuous, which is likely to be the case in general. In order to alleviate such computational overheads, one possible variation on the likelihood-free rejection algorithm would be to adjust the requirement that $y = y_{obs}$ exactly. Instead, the acceptance criterion could require that the generated data are simply "close" to the observed data. A standard way to do that would be to require that $\rho(y, y_{obs}) \leq \epsilon$ for some fixed threshold $\epsilon > 0$ and distance measure ρ. So, in the algorithm's statement, a metric ρ and a positive value ϵ are given as further inputs. Then, the third step is substituted by

(3) if $\rho(y, y_{obs}) \leq \epsilon$, then accept $\theta^{(i)}$ with probability $\frac{\pi(\theta^{(i)})}{Kg(\theta^{(i)})}$, where $K \geq \max_{\theta} \frac{\pi(\theta)}{g(\theta)}$, else go to (1).

This would also permit a relaxation of our previous assumption that data generated under the model are discrete. In this way, the output would no longer be drawn from $\pi(\theta|y_{obs})$ unless $\epsilon = 0$, but will instead be drawn from an approximation of it, which will be closer to the true posterior as ϵ gets smaller. On the other hand, increasing ϵ will considerably improve the acceptance rate of the algorithm and so its computational efficiency.
Before going on, we now give an alternative formal proof of the fact that, for $\epsilon = 0$, the likelihood-free rejection algorithm draws samples (θ, y), for which the marginal distribution of the parameter vector is the true posterior $\pi(\theta|y_{obs})$.
We begin by observing that the overall procedure of rejection sampling (steps (1)-(3)) is finally equivalent to drawing sample (θ, y) from a joint distribution proportional to

$$\mathbb{I}_{\{\rho(y,y_{obs})\leq\epsilon\}} l(y|\theta)g(\theta).$$

This sample (θ, y) is further accepted with probability proportional to $\pi(\theta)/g(\theta)$, so the algorithm is truly sampling from a joint distribution proportional to:

$$\mathbb{I}_{\{\rho(y,y_{obs})\leq\epsilon\}} l(y|\theta) g(\theta) \frac{\pi(\theta)}{g(\theta)} = \mathbb{I}_{\{\rho(y,y_{obs})\leq\epsilon\}} l(y|\theta)\pi(\theta). \tag{2.1}$$

This can be considered as the approximation of the true posterior distribution from which the algorithm is sampling. Now if $\epsilon = 0$, then the θ marginal of (2.1) equals the true posterior distribution, as:

$$\begin{aligned} &\lim_{h\to 0} \int \mathbb{I}_{\{\rho(y,y_{obs})\leq\epsilon\}} l(y|\theta)\pi(\theta)dy \\ &= \int \delta_{obs}(y) l(y|\theta)\pi(\theta)dy \\ &= l(y_{obs}|\theta)\pi(\theta), \end{aligned}$$

where $\delta_{\cdot}(\cdot)$ is the Dirac function.

2.1.2 Refinements: use of summary statistics

In applying this last version of likelihood-free rejection algorithm to real data it may still result that the the true parameter vector is not even close to the estimated posterior samples. The main cause of this can usually be found in the high dimension of the comparison $\rho(y, y_{obs})$. If the observed data y_{obs} are a high-dimensional vector, the chance of generating a simulated vector y, with the same dimension, that is close to y_{obs} is vanishing small, even if we use the true parameter vector. This means that ϵ must be relatively large, which results in accepting samples $\theta^{(i)}$ that generate data $y^{(i)}$ that are not actually close to y_{obs}, and thereby producing a poor approximation to $\pi(\theta|y_{obs})$. In order to avoid this problem one need to reduce the dimension of the data comparison $\rho(y, y_{obs})$. If lower dimensional statistics $s := S(y)$ and $s_{obs} := S(y_{obs})$, such that $\dim(S(y)) \ll \dim(y)$ are available, then $\rho(y, y_{obs})$ may be replaced by $\rho(s, s_{obs})$. In doing so, we do not want to have much loss of information, therefore the statistics $S(y)$ must be highly informative, for θ under the model. A proper choice would be a sufficient statistic, defined, in Bayesian context, as follows.

Definition 2.1.3. A statistic S is said to be a *Bayes sufficient statistic* for the parameter model θ if, for every prior distribution $\pi(\theta)$, there exist versions of the posterior $\pi(\theta|y)$ and $\pi(\theta|S(y))$ such that $\pi(\theta|y) \equiv \pi(\theta|S(y))$ for almost any y.

Sufficient statistics thus allow not to lost information by passing from the high-dimensional data to lower-dimensional information.
The third step of the likelihood-free rejection sampling algorithm is finally replaced by:

(3) compute $s = S(y)$. If $\rho(s, s_{obs}) \leq \epsilon$, then accept $\theta^{(i)}$, with probability $\frac{\pi(\theta^{(i)})}{Kg(\theta^{(i)})}$, where $K \geq \max_{\theta} \frac{\pi(\theta)}{g(\theta)}$, else go to (1);

where the statistic S, the distance measure ρ and the threshold ϵ are given as inputs.

2.2 ABC rejection sampling algorithm

After the intuitive idea of likelihood-free methods of previous sections, we now aim to give further details and describe the exact form of the Approximate Bayesian Computation for the rejection sampling algorithm, that gives an approximation to the posterior distribution.
In order to obtain the final form of the ABC rejection sampling, further improvements are made starting from the following consideration. As stated so far, the algorithm does not discriminate between those samples, θ, for which the associated dataset y exactly equals the observed data y_{obs}, and those samples, θ for which the associated dataset is further away from y_{obs}, i.e., $\rho(y, y_{obs}) = \epsilon$. Therefore, one may need to introduce a more continuous scaling from 1, the exact match $y = y_{obs}$, to 0, when $\rho(y, y_{obs})$ is large. This can be achieved by replacing the indicator function with a standard smoothing kernel function defined as follows:

Definition 2.2.1.

$$K_\epsilon(u) := \frac{1}{\epsilon} K\left(\frac{u}{\epsilon}\right),$$

where $u = \rho(y, y_{obs})$ and $K(\cdot) : \mathbb{R} \mapsto \mathbb{R}$ is a *kernel function*.
Namely:

- K is symmetric, i.e., $K(u) = K(-u)$ for each $u \in \mathbb{R}$;
- $K(u) \geq 0$ for all $u \in \mathbb{R}$;
- $\int_{\mathbb{R}} K(u)du = 1$;
- $\int_{\mathbb{R}} uK(u)du = 0$;
- $\int_{\mathbb{R}} u^2K(u)du < \infty$.

Moreover, following the usual convention, we define $\lim_{\epsilon \to 0} K_\epsilon(u)$ as a point mass at the origin $u = 0$.
Typical choices for the kernel function are

$$K(u) = \begin{cases} \frac{1}{2}\mathbb{I}_{\{|u| \leq 1\}} & \text{(uniform kernel);} \\ (1 - |u|)\mathbb{I}_{\{|u| \leq 1\}} & \text{(triangular kernel);} \\ \frac{1}{\sqrt{2\pi}} e^{-\frac{1}{2}u^2} & \text{(Gaussian kernel).} \end{cases}$$

Substituting the kernel function into the likelihood-free rejection algorithm results in the ABC rejection sampling algorithm.
Inputs:

- a target posterior density $\pi(\theta|y_{obs}) \propto l(y_{obs}|\theta)\pi(\theta)$ (a prior distribution $\pi(\theta)$ and a procedure for generating data under the model);
- a proposal density $g(\theta)$ with $g(\theta) > 0$ if $\pi(\theta|y_{obs}) > 0$;
- an integer $N > 0$;
- a kernel function $K_\epsilon(u)$ and a scale parameter ϵ.

Sampling:
For $i = 1, \ldots, N$:

1. generate $\theta^{(i)} \sim g(\theta)$ from sampling density g;
2. using the current draw of the parameter $\theta^{(i)}$, generate a simulated dataset y; that is, generate $y \sim l(y|\theta^{(i)})$ from the likelihood;
3. accept $\theta^{(i)}$ with probability $\frac{K_\epsilon(\rho(y, y_{obs}))\pi(\theta^{(i)})}{Kg(\theta^{(i)})}$, where $K \geq \left(K_\epsilon(0) \max_\theta \frac{\pi(\theta)}{g(\theta)} \right)$, else go to (1).

Output:
a set of parameter vectors $\theta^{(1)}, \ldots, \theta^{(N)}$ generated from $\pi_{ABC}(\theta|y_{obs})$, the ABC approximation of the true posterior distribution.

Similarly to what done for the likelihood-free rejection sampling algorithm we want to determine the form of the target distribution, $\pi_{ABC}(\theta|y_{obs})$, of this algorithm. We can follow the same argument as before. It turns out the algorithm generate samples (θ, y) from the joint distribution:

$$\pi_{ABC}(\theta, y|y_{obs}) \propto K_\epsilon(\rho(y, y_{obs}))l(y|\theta)\pi(\theta). \tag{2.2}$$

Accordingly, we define the ABC approximation to the true posterior distribution as

$$\pi_{ABC}(\theta|y_{obs}) := \int \pi_{ABC}(\theta, y|y_{obs})dy,$$

where $\pi_{ABC}(\theta, y|y_{obs})$ is given by (2.2). Firstly we note that, even in this case, it is easy to show that samples from the true posterior distribution are obtained as $\epsilon \to 0$. Indeed, we have

$$\lim_{\epsilon\to 0} \pi_{ABC}(\theta, y|y_{obs}) \propto \lim_{\epsilon\to 0} K_\epsilon(\rho(y, y_{obs}))l(y|\theta)\pi(\theta) = \delta_{y_{obs}}(y)l(y|\theta)\pi(\theta),$$

and so

$$\lim_{\epsilon\to 0} \pi_{ABC}(\theta|y_{obs}) \propto \int \delta_{y_{obs}}(y)l(y|\theta)\pi(\theta)dy = l(y_{obs}|\theta)\pi(\theta).$$

The next natural task is then evaluating the level of accuracy of this approximation.
In doing this, we note that we can define the ABC approximation to the true likelihood $l(y|\theta)$, for a fixed value of θ, simply re-writing relation (2.2) without the prior distribution $\pi(\theta)$. It is given by

$$l_{ABC}(y_{obs}|\theta) = \int K_\epsilon(\rho(y, y_{obs}))l(y|\theta)dy.$$

In this way, we can then check for the pointwise bias in the likelihood approximation. For the sake of simplicity, we will consider the univariate case, so that $y, y_{obs} \in \mathbb{R}$ and $\rho(y, y_{obs}) = |y - y_{obs}|$. Besides, we assume that the likelihood is infinitely differentiable. We thus obtain

$$\begin{aligned}
l_{ABC}(y_{obs}|\theta) &= \int K_\epsilon(|y - y_{obs}|)l(y|\theta)dy \qquad \text{(substituting } u = (y - y_{obs})/\epsilon) \\
&= \int K(u)l(y_{obs} + u\epsilon|\theta)du \\
&= \int K(u)\left[l(y_{obs}|\theta) + u\epsilon l'(y_{obs}|\theta) + \frac{u^2\epsilon^2}{2}l''(y_{obs}|\theta) + o(u^3\epsilon^3)\right]du \\
&= l(y_{obs}|\theta) + \frac{1}{2}\epsilon^2 l''(y_{obs}|\theta)\int u^2K(u) + o(u^3\epsilon^3)du;
\end{aligned}$$

where we have used a Taylor expansion of $l(y_{obs} + u\epsilon|\theta)$ around the point y_{obs} and the kernel function properties of Definition 2.2.1.
Then, the pointwise bias in the likelihood $b_\epsilon(y|\theta) := l_{ABC}(y|\theta) - l(y|\theta)$ as a function of y for fixed θ, can be written to second order as:

$$\hat{b}_\epsilon(y|\theta) = \frac{1}{2}\epsilon^2\sigma_K^2 l''(y|\theta),$$

where $\sigma_K^2 = \int u^2 K(u) du$ is the variance of the kernel function. In terms of the quality of approximation the choice of scale parameter results to be more important than the choice of kernel function and the bias is reduced if ϵ is small, corresponding to better approximations. We can derive an analogous expression when y and y_{obs} are multivariate.
Besides, in a similar manner we can determine the pointwise bias in the resulting ABC posterior approximation

$$a_\epsilon(\theta|y_{obs}) := \pi_{ABC}(\theta|y_{obs}) - \pi(\theta|y_{obs}),$$

as a function of θ. From the definition of $b_\epsilon(y_{obs}|\theta)$, we have

$$\begin{aligned} b_\epsilon(y_{obs}|\theta)\pi(\theta) &= l_{ABC}(y_{obs}|\theta)\pi(\theta) - l(y_{obs}|\theta)\pi(\theta) \\ &= \pi_{ABC}(\theta|y_{obs})c_{ABC} - \pi(\theta|y_{obs})c, \end{aligned}$$

where $c_{ABC} = \int l_{ABC}(y_{obs}|\theta)\pi(\theta)d\theta > 0$ and $c = \int l(y_{obs}|\theta)\pi(\theta)d\theta > 0$. Rearranging above terms, we obtain:

$$\begin{aligned} a_\epsilon(\theta|y_{obs}) &= \frac{b_h(y_{obs}|\theta)\pi(\theta) + \pi(\theta|y_{obs})c}{c_{ABC}} - \pi(\theta|y_{obs}) \\ &= \frac{b_\epsilon(y_{obs}|\theta)\pi(\theta)}{c_{ABC}} + \pi(\theta|y_{obs})\left(\frac{c}{c_{ABC}} - 1\right). \end{aligned}$$

It is easy to conclude that as $\epsilon \to 0$, then $a_\epsilon(\theta|y_{obs}) \to 0$.
Indeed, from (2.2), we have that $b_\epsilon(y_{obs}|\theta)$ goes to 0 as $\epsilon \to 0$. So, for fixed θ, $l_{ABC}(y_{obs}|\theta) \to l(y_{obs}|\theta)$ and $\frac{c}{c_{ABC}} \to 1$ as $\epsilon \to 0$.

The ABC posterior approximation just developed is, however, rarely used in practice, because it is highly unlikely that $y \sim y_{obs}$ can be generated from $l(y|\theta)$ from any choices of θ for realistic datasets, except in very specific scenarios where, for example, y_{obs} is very low dimensional. This results in the need to use a large value of the kernel scale parameter ϵ in order to achieve viable rejection sampling algorithm acceptance rates (or a similar loss of performance in other algorithms), and in doing so produce poorer ABC posterior approximations. Then, as in the case of the likelihood-free sampling rejection algorithm, a typical ABC analysis will involve specification of a vector of summary statistics $s = S(y)$, where $\dim(s) \ll \dim(y)$. The ABC rejection sampling algorithm will then compare s with $s_{obs} = S(y_{obs})$, rather than y and y_{obs}, so in the algorithm procedure, before deciding whether to accept θ or not, the summary statistic (which is part of the input) is computed.
The dimension of the summary statistic should be large enough so that it contains as much information about the observed data as possible, but also

low enough so that the curse-of-dimensionality of matching s and s_{obs} is avoided. The most efficient choice for S results to be the minimal sufficient statistic (i.e., a sufficient statistic such that for every sufficient statistic S' there exists a function f such that $S = f(S')$). However, this may be non-viable in practice: identification of any low-dimensional sufficient statistic (except the trivial full dataset y_{obs}) may be impossible and they may not even exist. Moreover, for some models it may be the case that the dimension of the minimal sufficient statistic is equal to that of the original dataset. As this will cause curse-of-dimensionality problems in matching s with s_{obs}, it is likely that a more accurate ABC posterior approximation can be achieved by using a lower-dimensional non-sufficient statistic, rather than remaining within the class of sufficient statistics.
We can show the properties proved above still apply in such a modified version of the algorithm. In a similar fashion to the previous section, it can be seen that the ABC posterior approximation has the form

$$\pi_{ABC}(\theta|s_{obs}) \propto \int K_\epsilon(\rho(s, s_{obs}))l(s|\theta)\pi(\theta)ds,$$

where $l(s|\theta) = \int \delta_s(S(y))l(y|\theta)dy$ is the likelihood function of the summary statistic $s = S(y)$. Once again, if we let $\epsilon \to 0$, using analogous considerations as before, we obtain

$$\lim_{\epsilon \to 0} \pi_{ABC}(\theta|s_{obs}) = l(\theta|s_{obs})\pi(\theta);$$

that is, samples from the distribution $\pi(\theta|s_{obs})$ are obtained as $\epsilon \to 0$. This shows the reason for which is desirable for S to be sufficient for the model parameter: in such a case, $\pi(\theta|s_{obs}) \equiv \pi(\theta|y_{obs})$, by definition 2.1.3, and so samples are produced from the true posterior distribution.

Chapter 3

A space-time model for rainfall

In this chapter, we present in detail a first model for rainfall, proposed by P. Cowpertwait in [10]. The model aims to reproduce rainfall patterns at each point of a fixed region during a time period. It is a relatively simple space-time model that has rain cells arriving in a cluster point process and has parameters that can be related, at least in some broad sense, to physical features observed and measured in precipitation fields. Before giving model's details, we briefly introduce some of the mathematical tools used by Cowpertwait. In particular, we state the major definitions and we outline some of the main aspects related to the theory of point processes. We also give the basics of the Nelder-Mead algorithm, used by Cowpertwait in the fitting procedure to minimize a cost function.

3.1 A brief introduction to point processes

In this section we give the basics of the point process theory, in order to understand the first model for rainfall we will consider subsequently. We only present essential definitions and some useful properties, without giving details of proofs. Results provided herein are fully explained in [12, 13].
Informally, a point process can be succinctly described as a random collection of points, whether in time, space, or time and space together. To put this concept into a formal mathematical framework, the process can be described as a random counting measure, say $N(\cdot)$, allocating integer values $N(A)$, representing the number of points falling in A, to sufficiently regular sets A of the relevant state space, for example a Borel set.
More precisely, let $\mathcal{X}$ be an arbitrary complete separable metric space and $\mathcal{B}(\mathcal{X})$ the σ-field of its Borel sets. Let $\mathcal{N}_{\mathcal{X}}$ denote the space of all boundedly finite integer-valued measures on $\mathcal{B}(\mathcal{X})$, i.e., all Borel measures on $\mathcal{X}$ that are

finite on every bounded Borel set. We say that a sequence of boundendly finite measures $\{\mu_n\}_n$ w^* - convergences to a boundedly finite measure μ if and only if $\int f d\mu_n \mapsto \int f d\mu$ for all bounded continuous function f in $\mathcal{X}$ vanishing outside a bounded set.
One can show that under the w^*-topology, $\mathcal{N}_\mathcal{X}$ is a complete separable metric space in its own right and the corresponding Borel σ-algebra, $\mathcal{B}(\mathcal{N}_\mathcal{X})$ is the smallest σ-algebra on $\mathcal{N}_\mathcal{X}$ with respect to which the mappings $\mu \mapsto \mu(A)$ are measurable for all $A \in \mathcal{B}(\mathcal{X})$ (see [13] for details). In this context, a point process is then defined as follows.

Definition 3.1.1. Let $(\Omega, \mathcal{F}, \mathbb{P})$ be a probability space. A *point process* N on state space $\mathcal{X}$ is a measurable mapping from $(\Omega, \mathcal{F}, \mathbb{P})$ into $(\mathcal{N}_\mathcal{X}, \mathcal{B}(\mathcal{N}_\mathcal{X}))$. A point process N is *simple* when $N(\{x\}) \in \{0, 1\}$ almost surely for all $x \in \mathcal{X}$.

One may show the following proposition holds.

Proposition 3.1.2. *A map $N : \Omega \mapsto \mathcal{N}_\mathcal{X}$ is a point process if and only if $N(A)$ is a random variable for each bounded $A \in \mathcal{B}_\mathcal{X}$.*

In this scenario, $N(A)$ can be thought as the random variable that gives the number of points falling in A.
Key concepts in point process theory are the distribution of a point process and its finite-dimensional distributions, defined as follows.

Definition 3.1.3. The *distribution* of a point process is the probability measure it induces on $(\mathcal{N}_\mathcal{X}, \mathcal{B}(\mathcal{N}_\mathcal{X}))$.

Definition 3.1.4. The *finite-dimensional distributions* of a point process N are the joint distributions, for all finite families of bounded Borel sets $A_1, \ldots, A_k$ of the random variables $N(A_1), \ldots, N(A_k)$.

These distributions are strictly related, as the following proposition asserts. A complete proof of this can be found in [13].

Proposition 3.1.5. *The distribution of a point process is completely determined by its finite-dimensional distributions, that is, if two point processes give the same values for the finite-dimensional distributions, then their distributions coincide.*

We now turn our attention to one of the principal point process: the Poisson point process. It has a great importance in point process theory, especially it serves in illustrating and motivating more general results. It is defined as follows:

Definition 3.1.6. A point process N is a *Poisson process* on $\mathcal{X}$ if the following conditions hold:

- for each finite family of bounded, disjoint Borel sets $\{A_i\}_{i=1,\dots,k}$, the random variables $N(A_1), \dots, N(A_k)$ are mutually independent;
- there exists a boundedly finite Borel measure $\Lambda(\cdot)$ such that for each A in $\mathcal{B}(\mathcal{X})$ and $k \in \mathbb{N}$

$$\mathbb{P}(N(A) = k) = e^{-\Lambda(A)} \frac{[\Lambda(A)]^k}{k!}.$$

  In other words, N has Poisson finite-dimensional distributions.

The Poisson process is said to be homogeneous if $\Lambda(A)$ is proportional to the volume of A; namely, $\Lambda(A) = \lambda \cdot \lambda^{leb}(A)$, where λ^{leb} denotes the Lebesgue measure and $\lambda \in \mathbb{R}$. Instead, for a non-homogeneous process $\Lambda(\cdot)$ is the integral of some intensity $\lambda(x)$.
An interesting and useful property of Poisson processes is given in the following remark. It also furnishes a way to concretely figure a Poisson process.

Remark 3.1.7. Let N be a Poisson process and $T > 0$. Let us suppose there are M observations on $(0, T]$ at points $t_1, \dots, t_M$. Our aim is obtaining the likelihood of this realization; this can be defined as the probability of obtaining the given number of observations in $(0, T]$, times the joint conditional density for the positions of those observations, given their number.
So, we fix $\Delta > 0$ and we begin by calculating the probability of obtaining single events in $C_i := (t_i - \Delta, t_i]$ for $i = 1, \dots, M$ and no points in the remaining part of $(0, T]$. From Definition 3.1.6, it is given by

$$\begin{aligned}
&\mathbb{P}(N(C_i) = 1 \text{ for } i = 1, \dots, M \text{ and } N((0,T] \setminus \bigcup_{i=1}^{M} C_i) = 0) \\
&= \mathbb{P}((N(C_i) = 1 \text{ for } i = 1, \dots, M)\mathbb{P}(N((0,T] \setminus \bigcup_{i=1}^{M} C_i) = 0) \\
&= \left(\prod_{i=1}^{M} e^{-\lambda\Delta} \lambda\Delta \right) e^{-\lambda[T - \sum_{i=1}^{M} \Delta]} \\
&= e^{-\lambda T} \prod_{i=1}^{M} \lambda\Delta.
\end{aligned}$$

Dividing by Δ^M and letting $\Delta \to 0$, to obtain the density, we find as the required likelihood function

$$L_T(N; t_1, \dots, t_M) = \lambda^M e^{-\lambda T}.$$

By definition 3.1.6, the probability of obtaining precisely M events in T is equal to

$$\frac{e^{-\lambda T}[\lambda T]^M}{M!};$$

this implies that the conditional density of obtaining points at $(t_1, \ldots, t_M)$ given M points in $(0, T]$, is just $M!/T^M$. This corresponds to a uniform distribution over the hyperoctant with edge of length T, that is

$$\{(t_1, \ldots, t_M) \in \mathbb{R}^M : 0 \leq t_1 \leq \cdots \leq t_M \leq T\}.$$

Moreover, the factor $M!$ can be interpreted as the combinatorial factor representing the number of ways M distinct particles can be allocated in the M distinct time points, so the individual particles are to be thought as uniformly and independent distributed over $(0, T]$.

Cowpertwait, in [10], develops a model based on a space-time Poisson process. Space-time point processes are point processes defined on $\mathcal{X} \subset \mathbb{R}^2 \times [0, +\infty)$. They are characterized by the so-called *conditional intensity* $\lambda(t, x, y)$, which can be thought as the frequency with which events are expected to occur around a particular location (t, x, y) in space-time, conditional on the prior history H_t of the point process up to time t. When the process is a Poisson process λ, results to be deterministic, i.e., $\lambda(t, x, y)$ depends only on t, x and y, and not on the previous history. In particular, the simplest model is the stationary Poisson, where the conditional intensity is constant: $\lambda(t, x, y) = \bar{\lambda}$ for all t, x, y. In this case, it coincides with the rate of the Poisson distribution. The latter one is the process considered by Cowpertwait in developing his rainfall model.
Traditionally the points of a space-time point process are thought to be indistinguishable, other than by their times and locations. Often, however, there is other important information to be stored along with each point. In such cases, processes can be viewed as *marked* space-time point processes, i.e., random collection of points, where each point has associated with it a further random variable called a mark. It may often be convenient to view a space-time point process as a purely temporal point process, with spatial mark associated with each point. For the sake of completeness, we concisely characterize such processes rigorously.
In order to formally define a marked point process, we introduce the family $\mathcal{N}_{\mathcal{X}\times\mathcal{K}}$ of all boundedly finite counting measure defined on the product space $\mathcal{B}(\mathcal{X} \times \mathcal{K})$, where $\mathcal{K}$ is a complete separable metric space of marks, subject to the additional requirement that the *ground measure* N_g defined by

$$N_g(A) := N(A \times \mathcal{K}) \quad \text{for all} \quad A \in \mathcal{B}_{\mathcal{X}},$$

is a boundedly finite simple counting measure. Then, we set

Definition 3.1.8. A *marked point process* on $\mathcal{X}$ with marks in $\mathcal{K}$ is a point process N on $\mathcal{B}_{\mathcal{X}\times\mathcal{K}}$ for which

$$\mathbb{P}(N \in \mathcal{N}_{\mathcal{X}\times\mathcal{K}}) = 1;$$

its ground process is given by $N_g(\cdot) := N(\cdot \times \mathcal{K})$.

Poisson processes are not obviously the only point processes used in practice; there a lot of classes of point processes which are adopted in models formulation. Their common theme is the generation of the final model by a two-stage construction: first, the generation of an indexed family of processes, and then an operation applied to members of the family to produce the final process. In this context, we briefly examine cluster processes, extensions of Poisson processes. They are widely used in rainfall models since they are natural patterns for the locations of objects in the plane or in three-dimensional space. The intuitive motivation of such processes involves two components: the locations of clusters and the locations of elements within a cluster. In order to model the cluster elements, we specify a countable family of point process $N(\cdot|y_i)$ indexed by the cluster centres $\{y_i\}_i$; whilst, to model the cluster locations, we suppose there is given a process N_c of cluster centres, often unobserved, whose generic realization consists of the points $\{y_i\} \subset \mathcal{Y}$, where $\mathcal{Y}$ is a complete separable metric space. More formally, we have the following definition.

Definition 3.1.9. N is a *cluster process* on a complete separable metric space $\mathcal{X}$, with centre process N_c on the complete separable metric space $\mathcal{Y}$ and component process the measurable family of point process $\{N(\cdot|y) : y \in \mathcal{Y}\}$, when for every bounded set $A \in \mathbb{B}_{\mathcal{X}}$, it holds

$$N(A) = \int_{\mathcal{Y}} N(A|y) N_c(dy) = \sum_{y_i \in N_c(\cdot)} N(A|y_i) < \infty \qquad \text{a.s.}$$

In practice, the component processes are often required to be mutually independent. In this case, we say that the component processes come from *an independent measurable family* and thereby they define *an independent cluster process.*
An important class of processes, frequently occurring in applications, is the one where

- the class centres are the points of a Poisson process (the so-called *Poisson cluster process*);
- the clusters are independent and finite almost surely.

Among this class, we can distinguish different kinds of processes according to how points of the cluster are distributed around the cluster centre. In particular, we give the definition of the Neyman-Scott process.

Definition 3.1.10. A cluster process is said to be a *Neyman-Scott process* if and only if it is a Poisson cluster process and the points are distributed according to a Poisson distribution around each cluster centre, i.e., $\{N(\cdot|y)$ is a Poisson point process.

Rainfall models based on Neyman-Scott process have been largely developed in literature (e.g., [3, 5]); as we see in detail in next sections, in the model proposed by Cowpertwait in [10], a Neyman-Scott model is used to model occurrence time of rain cells.

3.2 The Nelder-Mead method

This section is committed to presenting the basics of the Nelder-Mead algorithm, also known as simplex method. It is first described in [20] as a method for the minimization of a function of n variables, which depends on the comparison of function values at the $(n+1)$ vertices of a general simplex, without the use of derivatives. The great power of this algorithm relies on its adaptability, as it does not require any constraint or regularity of the function to be minimized. In defining method's steps in [20], particular attention has been given in order to use the algorithm for statistical problems involving the maximization of a likelihood function, in which unknown parameters enter non-linearly.
Algorithm's operations can be succinctly summarised in the following way. An initial simplex S is chosen , then the method performs a sequence of transformations of the working simplex aimed at decreasing the function values at its vertices; at each step, the transformation is determined by computing one or more test points together with their function values and by comparison of these function values with those at the vertices.
Specifically, the first step of the algorithm is constructing the initial simplex; it is usually done by generating $n+1$ vertices $P_0, \ldots, P_n$ around a given input point $P_{inp} \in \mathbb{R}^n$. In practice, the most frequent choice is $P_0 = P_{inp}$. The remaining n vertices are then generated, for example, either by setting $P_j := P_0 + h_j e_j$, for $j = 1, \ldots, n$, where h_j is a stepsize in the direction of unit vector $e_j \in \mathbb{R}^n$ or by imposing that S is a regular simplex, where all the edges have the same specified length. The algorithm proceeds iterating the following steps. The first one is to determine the indices h, s and l of the vertices that have, respectively, the highest, the second highest and the

lowest function value; if we denote as y_i the function value at P_i, that is

$$y_h := \max_{j=1,\dots,n} y_j, \qquad y_s := \max_{\substack{j=1,\dots,n \\ j\neq h}} y_j, \qquad y_l := \min_{j=1,\dots,n} y_j.$$

We refer at these vertices as the *worst*, the *second worst* and the *best* ones. Further, the centroid of the points $\bar{P}$ with $j \neq h$ is computed, i.e.,

$$\bar{P} := \frac{1}{n}\sum_{j\neq h} P_j.$$

It can be thought as the centroid of the *best* side of the simplex, opposite to the worst vertex P_h.
Subsequently the main part of the algorithm is performed: at each stage, P_h is replaced by a new better point by using three different operations: *reflection*,*expansion* and *contraction*. In this procedure, all new test points lie on the line joining P_h and $\bar{P}$ and at most two of them are computed in one iteration. If this succeeds, the accepted point becomes the new vertex of the working simplex; otherwise, the simplex is shrunk towards P_l, so $n+1$ new vertices are computed. Let us describe these passages in detail.
The reflection point P_r of P_h is computed as

$$P_r := \bar{P} + \alpha(\bar{P} - P_h),$$

where $\alpha > 0$ is a constant called the *reflection coefficient*. In this way, P_r is on the line joining P_h and $\bar{P}$, on the far side of $\bar{P}$ from P_h. Now, there are four possibilities for y_r to be considered in order to determine next steps.
If $y_l \leq y_r < y_s$, the algorithm has found a better point than the two worst ones, so the point P_h is replaced by P_r (in this case, we say that the new point is accepted) and the iteration starts again with the new simplex.
If, instead, P_r is a better point than P_l, i.e., $y_r > y_l$, an expansion is made towards P_r in order to further improve the function value. The expansion point P_e is given by

$$P_e := \bar{P} + \gamma(P_r - \bar{P}),$$

where the *expansion coefficient* γ is greater than unity and such that $\gamma > \alpha$. If $y_e < y_r$, i.e., the expansion has effectivley found a better value than the reflection, the algorithm accepts P_e as the new point for the simplex and restarts the iterations, otherwise P_r is accepted and the process starts again. This is the so-called "greedy maximization", which is now used in most implementations and in theory, but it is not the one used by Nelder and Mead in their paper ([20]). They proposed a "greedy expansion" where the expansion point P_e is accepted if $y_e < y_l$ and $y_r < y_l$, regardless of the

relationship between P_r and P_e. In this way, it may happen that $y_r < y_e$, so P_r would be a better new point then P_e, but P_e is still accepted for the new simplex. The reason for that is to keep the working simplex as large as possible, to avoid premature termination of iterations, which is sometimes useful, especially for non-smooth functions.
On the other hand, when the reflected point P_r has a greater function value than P_s, i.e., $y_r \geq y_s$, the algorithm computes the *contraction point* P_c by using the better of the two points P_h and P_r. This means that, if we denote as $P_{h,r}$ the point such that $y_{h,r} = \min\{y_r, y_h\}$, we define the contraction point to be

$$P_c := \bar{P} + \beta(P_{h,r} - \bar{P}),$$

where β is the contraction coefficient and it is chosen to be $\beta \in (0, 1)$. Now, if $y_c > y_{h,r}$ the algorithm has found a better value for P_h, so it is replaced by P_c. Otherwise, the research for a better vertex of the simplex has failed and a new simplex is constructed by replacing all the vertices P_j by $(P_j + P_l)/2$ and restarting the process (*shrink transformation*). This last operation was introduced in the original paper for preventing the rare case of a failed contraction which can occur when a valley is curved and one point of the simplex is much farther from the valley bottom than the others and contraction can cause the reflected point to move away from the valley bottom instead of towards it. In more recent versions of the algorithm, the shrink transformation also depends on a coefficient $\delta \in (0, 1)$; that is, in the last case considered above, all the vertices of the simplex are replaced by $\delta(P_j + P_l)$.
The stopping criterion proposed by Nelder and Mead concerns with the variation in the y values over the simplex; the algorithm stops when the error

$$\sqrt{\sum_{j=1}^{n} \frac{(y_j - \bar{y})^2}{n}}$$

is lower than a fixed threshold.
Even if the method results to perform well in practice, very little is known about its convergence properties, with mainly negative results. Lagarias et al. in [17] present convergence results in one and two dimensions for the original Nelder–Mead algorithm applied to strictly convex functions with bounded level sets. However, almost nothing is known about the behaviour of the method for less smooth or discontinuous functions, except that the method may fail to converge to a minimizing point.

3.3 Formulation of the rainfall model

Cowpertwait presents in [10] a space-time point process model, in which rain occurs in two dimensional cell discs, each of them is related to storm discs of different random types. The double discs structure is an *ad hoc* construction that attempts to emulate some physical issues of rainfall origins and, at the same time, makes interactions between spatial points rigorous from a mathematical point of view. Rainfall features, such as the intensity and the duration of an event, are represented by random variables with established distributions.

Specifically, there is a main space-time Poisson process according to which storm origins centres occur in space and time; it is assumed that this process has constant rate ζ_s, per unit area per unit time. Each storm origin is of random type z, where z is a realization of an independent random variable with continuous probability density function f_z. Ideally, these different random types are meant to simulate different kinds of precipitation. Associated with a type z storm origin is a radius $R_s(z)$, which is assumed to be an independent exponential random variable with parameter $\gamma_s(z)$; in this way storms are discs in the $\mathbb{R}^2$ plane.

Associated with a type z storm origin is a family of cell discs that effectively establishes rainfall amount and duration. Cell centres are points in $\mathbb{R}^2$ that occur according to a spatial Poisson process with rate $\zeta_c(z)$ per unit area; as before, each cell origin has a radius $R_c(z)$ that is an independent exponential random variable with parameter $\gamma_c(z)$, so that cells are discs in the plane. Moreover, it is assumed that the waiting time of a cell origin after a type z storm origin is an independent exponential random variable with parameter $\beta(z)$. Hence, the arrival time of cell origins occur in a Neyman-Scott point process; indeed, cluster's centres are the arrival time of storm origins which are, by assumption, Poisson distributed, points of the clusters have exponential interarrival times and so they globally follow a Poisson process.

Cell discs have an associated duration $L(z)$ and intensity $X(z)$, which represent period and amount of rain that occur in that region. Both $L(z)$ and $X(z)$ are taken to be independent exponential random variables with parameter $\eta(z)$ and $\xi(z)$, respectively. The cell intensity is assumed to be constant throughout the cell lifetime and over the area of the cell disc. In order to make an allowance for the effect of orography, the intensity is scaled by a factor φ_p at point $p \in \mathbb{R}^2$.

The total intensity at time t and a location $x \in \mathbb{R}^2$ is then the sum of the intensities of all cells alive at time t and overlapping x. In order to exploit a compact expression for it, we first introduce some useful variables.

Let $dN_s(r, \theta, t)$ denote the number of storm origins in a small interval $(dr, d\theta, dt)$

at the location (r, θ), given in polar coordinates, and a time t. From the definition of intensity of a point process we have that

$$\mathbb{E}[dN_s(r, \theta, t)] = \zeta_s r dr d\theta dt.$$

In the same way, we can consider the number $dN_{z,s}(r, \theta, t)$ of storm origins of type z in the same interval and it holds that

$$\mathbb{E}[dN_{z,s}(r, \theta, t)] = \zeta_s f_z(z) r dr d\theta dt dz.$$

In the same fashion, we denote as $dN_{z,c,u}(r, \theta, v)$ the number of cell origins in a small interval $(dr, d\theta, dv)$, at time v and location (r, θ), due to a type z storm origin at time u, with $u < v$. Now, the expected number of such cell discs is given by

$$\mathbb{E}[dN_{z,c,u}(r, \theta, v)] = \zeta_c(z)\beta(z)e^{-\beta(z)(v-u)} r dr d\theta dv.$$

Moreover, let $\mathbb{I}_{p,z,u}(r_s)$ be an indicator function that takes the value 1 if and only if a storm origin arriving at time u with centre a distance r_s from p overlaps p. Let $X_{p,z,u,v}(r_c, t - v)$ denote the cell intensity at point $p \in \mathbb{R}^2$ and time t due to a cell with centre located at a distance r_c from p and with arrival time v; in this way, if the distance r_c is greater than the cell radius $R_c(z)$ the intensity in p will be zero.
We can now exploit the intensity process in a point $p \in \mathbb{R}^2$ at time t due to a type z storm. It is given by

$$Y_{p,z}(t) = \int_{\theta_s} \int_{r_s} \int_{\theta_c} \int_{r_c} \int_{u=-\infty}^{t} \int_{v=u}^{t} X_{p,z,u,v}(r_c, t - v)\mathbb{I}_{p,z,u}(r_s) \, dN_{z,c,u}(\theta_c, r_c, v) dN_{z,s}(\theta_s, r_s, u). \tag{3.1}$$

Expression (3.1) exploits the fact that in a point p at time t the amount of rain due to a type z storm is the sum of all cells' intensities that are related to that storm, that are active at time t and that overlap p. Then, the overall intensity process $Y_p(t)$ is the superposition of the independent intensity process $Y_{p,z}$ and is therefore given by

$$Y_p(t) = \varphi_p \int_z Y_{p,z}(t).$$

3.4 Model properties

We present in this section some further properties of the above model. They allow for a better understanding of the model together with constituting an essential point in the fitting procedure.

First of all, let us consider a random variable N which represents the number of storm discs that cover an arbitrary point $p \in \mathbb{R}^2$ during a time interval $(0, T)$, where $T > 0$. Due to the fact that storm origins occur in a space-time Poisson process, N is a Poisson random variable. Without loss of generality, we can consider the point p to be the origin, so the probability that a storm disc with centre in (θ, r) and arrival time t overlaps p is given by the probability that the storm radius R_s is greater than r, that is

$$\mathbb{P}(p \in D_s((\theta, r), R_s)) = \mathbb{P}(R_s \geq r) = e^{-\gamma_s r},$$

where $D_s((\theta, r), R_s)$ is the storm disc with centre in (θ, r) and radius R_s. Thus, the expected number of type z storm discs that cover p is

$$\begin{aligned}
\mathbb{E}[N] &= \mathbb{E}\left[\int_0^{2\pi}\int_0^{T}\int_0^{+\infty} \mathbb{I}_{\{p \in D_s((\theta,r),R_s)\}} r\zeta_s f_z(z)\, dz\, dr\, dt\, d\theta\right] \\
&= \int_0^{2\pi}\int_0^{T}\int_0^{+\infty} re^{-\gamma_s r}\zeta_s f_z(z)\, dz\, dr\, dt\, d\theta \qquad (3.2) \\
&= \frac{2\pi T \zeta_s f_z(z) dz}{\gamma_s^2}.
\end{aligned}$$

Hence, storm discs overlap an arbitrary point in $\mathbb{R}^2$ at times that follow a Poisson process with rate λ given by

$$\lambda = 2\pi\zeta_s \int_z \frac{f_z(z)}{\gamma_s^2(z)} dz.$$

The same argument for cell discs gives that the number of cell discs, associated with a type z storm, overlapping an arbitrary point in $\mathbb{R}^2$ is again a Poisson random variable with mean

$$v(z) = \frac{2\pi\zeta_c(z)}{\gamma_c^2(z)}. \qquad (3.3)$$

We now want to give an expression for the expected value of the rainfall intensity in a point p at time t. In doing so, we note that the random variables $\mathbb{I}_{p,z,u}(r_s)$ and $X_{p,z,u,v}(r_c, t-v)$ are independent both of the counting processes that appear in (3.1) and each other. Besides, they have expectations respectively $e^{-\gamma_s(z) r_s}$ and $e^{-\eta(z)(t-v)} e^{-\gamma_c(z) r_c} \xi^{-1}(z)$. The former follows immediately from the definition of $\mathbb{I}_{p,z,u}(r_s)$; the latter is a consequence of the fact that $X_{p,z,u,v}(r_c, t-v)$ can be written as

$$X_{p,z,u,v}(r_c, t-v) = X(z)\mathbb{I}_{\{R_c \geq r_c\}}\mathbb{I}_{\{L(z) \geq t-v\}}.$$

Indeed, the cell intensity at point $p \in \mathbb{R}^2$ and time t due to a cell with centre located at a distance r_c from p and with arrival time v is equal to $X(z)$ if and only if the the cell overlaps p, i.e., the cell radius R_c is greater than r_c, and the cell is still alive at time t, i.e., the cell lifetime $L(z)$ is greater than $t - v$. Hence, form the independent properties stated above, we have

$$\mathbb{E}[X_{p,z,u,v}(r_c, t-v)] = \mathbb{E}[X(z)]\mathbb{P}(R_c \geq r_c)\mathbb{P}(L(z) \geq t-v) = \frac{e^{-\gamma_c(z)r_c}e^{-\eta(z)(t-v)}}{\xi(z)}.$$

The variables $dN_{z,c,u}(\theta_c, r_c, v)$ and $dN_{z,s}(\theta_s, r_s, u)$ are not independent, as the number of cell discs related to a type z storm is correlated to the number of storm discs, but the expectation of their product is independent of both $\mathbb{I}_{p,z,u}(r_s)$ and $X_{p,z,u,v}(r_c, t-v)$. It is given by the following expression:

$$\begin{aligned}
&\mathbb{E}\left[dN_{z,c,u}(\theta_c, r_c, v)dN_{z,s}(\theta_s, r_s, u)\right] \\
&= \mathbb{E}\left[dN_{z,s}(\theta_s, r_s, u)\mathbb{E}\left[dN_{z,c,u}(\theta_c, r_c, v)|dN_{z,s}(\theta_s, r_s, u)\right]\right] \\
&= \zeta_c(z)\beta(z)e^{-\beta(z)(v-u)}r_c dr_c d\theta_c \zeta_s f_z(z) r_s dr_s d\theta_s dv du dz.
\end{aligned}$$

Hence, we can easily calculate the expectation of $Y_p(t)$ which is given by

$$\mathbb{E}\left[Y_p(t)\right] = 2\pi\zeta_s\varphi_p \int_z \frac{v(z)f_z(z)}{\xi(z)\eta(z)\gamma_s^2(z)}dz. \tag{3.4}$$

It is interesting to note that in this model the rainfall intensity in a point $p \in \mathbb{R}^2$ is not independent from the intensity in a point $q \in \mathbb{R}^2$, with $q \neq p$. In order to show it, we consider for the moment a general spatial process of discs in $\mathbb{R}^2$, with independent radii with exponential distribution with parameter γ and centres occurring in a spatial Poisson process with rate ζ. Without loss of generality, let us consider p to be the origin and q to be the point with polar coordinates $(\theta, r) = (0, d)$; so, d is the distance between these two points. Now, let us divide the plane into four quadrants bounded by the perpendicular bisector of the line joining p and q and the x-axis, i.e., bounded by the lines with equations: $2r\cos\theta = d$ and $\theta = 0$. If we consider the quadrant $Q = \{(\theta, r) : 0 \leq \theta \leq \frac{\pi}{2},\ r \geq \frac{d}{2\cos\theta}\}$, we have that a disc with centre in Q overlaps p and q if and only if the disc overlaps point p, due to the fact that every point in Q has a distance from p greater than d. Moreover, due to the symmetry of the problem, the same argument is valid for each quadrants. Therefore, the expected number of discs simultaneously overlapping both p and q, is given by

$$\begin{aligned}
&\mathbb{E}\left[\int_0^{\frac{\pi}{2}}\int_{\frac{d}{2\cos\theta}}^{+\infty} \mathbb{I}_{\{p,q\in D((\theta,r),R)\}} r dr d\theta\right] \\
&= 4\int_0^{\frac{\pi}{2}}\int_{\frac{d}{2\cos\theta}}^{+\infty} \mathbb{P}(p \in D((\theta, r), R), q \in D((\theta, r), R)) r dr d\theta
\end{aligned}$$

$$
\begin{aligned}
&= 4\int_0^{\frac{\pi}{2}} \int_{\frac{d}{2\cos\theta}}^{+\infty} \mathbb{P}(p \in D((\theta,r),R)) r dr d\theta \\
&= 4\int_0^{\frac{\pi}{2}} \int_{\frac{d}{2\cos\theta}}^{+\infty} \zeta r e^{-r\gamma} dr d\theta \\
&= \int_0^{\frac{\pi}{2}} e^{-\frac{d\gamma}{2\cos\theta}} \zeta \left(\frac{2d}{\gamma\cos\theta} + \frac{4}{\gamma^2} \right) d\theta,
\end{aligned}
$$

where $D((\theta,r),R))$ denotes the disc with centre in (θ,r) (in polar coordinates) and radius the random variable R.
Hence, the conditional probability that a disc overlap p given that it overlaps q is given by

$$
\begin{aligned}
&\mathbb{P}(p \in D((\theta,r),R)) |\ \ \exists\ D((\theta,r),R)) : q \in D((\theta,r),R))) \\
&= \frac{\int_0^{\frac{\pi}{2}} e^{-\frac{d\gamma}{2\cos\theta}} \zeta \left(\frac{2d}{\gamma\cos\theta} + \frac{4}{\gamma^2} \right) d\theta}{\frac{2\pi\zeta}{\gamma^2}} \\
&= \frac{2}{\pi} \int_0^{\frac{\pi}{2}} e^{-\frac{d\gamma}{2\cos\theta}} \left(\frac{d\gamma}{2\cos\theta} + 1 \right) d\theta,
\end{aligned}
$$

where we have used the definition of conditional probability and the analogue of relation (3.3). Let us denote this quantity $\kappa_{p,q}[\gamma]$ for for $p,q \in \mathbb{R}^2$ with $d = ||p-q||$ the distance between these points.
Hence, the same argument as in (3.2) gives that the rate at which type z storm discs overlap both p and q is $2\pi\kappa_{p,q}[\gamma_s]\zeta_s\gamma_s^{-2}(z)f_z(z)dz$. This allows to give an expression for the cross-covariance of the intensities at a pair of points $p,q \in \mathbb{R}^2$:

$$
\begin{aligned}
&Cov[Y_p(t), Y_q(t+\tau)] \\
&= 2\pi\varphi_p\varphi_q \int_z \frac{[C_z(\tau) + 2\kappa_{p,q}[\gamma_c(z)]\kappa_{p,q}[\gamma_s(z)]e^{-\eta(z)\tau}]v(z)\zeta_s f_z(z)}{\gamma_s^2(z)\eta(z)\xi^2(z)} dz,
\end{aligned}
$$

where $C_z(\tau) = \beta(z)v(z)[\beta(z)e^{-\eta(z)\tau} - \eta(z)e^{-\beta(z)\tau}][\beta^2(z) - \eta^2(z)]^{-1}$. The complete computation can be found in [7].
Now, let $\{S_{p,i}(h)\}_i$ be a rainfall time series at $p \in \mathbb{R}^2$ aggregated over discrete time intervals of width h, so that $S_{p,i}(h) = \int_{(i-1)h}^{ih} Y_p(t)dt$ and let $S_{p,z,i}(h)$ be the aggregated total due to type z storms. A noticeable property of this model is that it is possible to write down expressions for the moments up to third order of the aggregated time series $S_{p,i}(h)$ and for the covariance in two points $p,q \in \mathbb{R}^2$. For the sake of completeness, as they will be used in the next section, we state their expressions, yet without giving details on how are

computed. The expected value of $S_{p,i}(h)$ follows easily from equation (3.4) and it is given by:

$$\mu_p(h) := \mathbb{E}[S_{p,i}(h)] = 2\pi\varphi_p h \int_z \frac{v(z)\zeta_s f_z(z)}{\gamma_s^2(z)\eta(z)\xi(z)} dz. \tag{3.5}$$

The autocovariance at lag $l \geq 0$ results to be

$$\begin{aligned} \phi_p(h,l) &:= Cov[S_{p,i}(h), S_{p,i+l}(h)] \\ &= 2\pi\varphi_p^2 \int_z \frac{v(z)\zeta_s f_z(z)}{\gamma_s^2(z)\xi^2(z)} F(h,l,\eta(z),\beta(z),v(z)) dz. \end{aligned} \tag{3.6}$$

Finally, we can obtain the following expression for the third moment function:

$$\begin{aligned} \psi_p(h) &:= \mathbb{E}[(S_{p,i}(h) - \mu_p(h))^3] \\ &= 2\pi\varphi_p^3 \int_z \frac{v(z)\zeta_s f_z(z)}{\gamma_s^2(z)\eta^4(z)\xi^3(z)} G(h,l,\eta(z),\beta(z),v(z)) dz. \end{aligned} \tag{3.7}$$

In relations above, $F(h,l,\eta(z),\beta(z),v(z))$ and $G(h,l,\eta(z),\beta(z),v(z))$ are expressions which depend on the parameters within brackets. Their explicit forms can be found in [10].
We can also compute the cross covariance for a pair of two points $a, b \in \mathbb{R}^2$; it is given by

$$\begin{aligned} \phi_{a,b}(h,l) &:= Cov[S_{a,i}(h), S_{b,i+l}(h)] \\ &= \varphi_a\varphi_b\varphi_p^{-2}\phi_p(h,l) \\ &\quad - 4\pi\varphi_a\varphi_b \int_z \frac{v(z)\kappa_{a,b}[\gamma_s(z)]\zeta_s f_z(z)(1-\kappa_{a,b}[\gamma_c(z)])A(z,h,l)}{\gamma_s^2(z)\eta(z)\xi^2(z)} dz, \end{aligned} \tag{3.8}$$

where $A(z,h,l)$ is a function that depends on $\eta(z)$, l and h (see [10]).

3.5 Fitting procedure

In this section we aim to present the procedure used by Cowpertwait to fit the model. In [10], a time series for rainfall in 23 spatial points (sites) is used; each point corresponds to the location of a rain gauge from which observed data are available at the 10 minutes (1/6 h) aggregation level: $\{S_{p_j,i}(h) : j = 1,\ldots,23; h = 1/6\}$. In this work each site is identified by its first two

coordinates $p(x_1, x_2) \in \mathbb{R}^2$.
In order to fit the model to data, a distribution needs to be selected for z and functions chosen for the model parameters. Cowpertwait's choice is to take z to be a uniform random variable, so that $f_z(z) = 1$ for $z \in (0, 1)$, $v(z)$ to be a linear combination of $(1 - z)$ and z, i.e., $v(z) = (1 - z)v_0 + zv_1$ and $\xi(z) = 1/z$, with the remaining parameters set to constants. The cell intensity $X(z)$ is scaled by the factor φ_p so that, under this parametrization, the mean cell intensity is

$$\mathbb{E}[\varphi_p X(z)] = \varphi_p \int_0^1 \int_0^{+\infty} x e^{-\frac{x}{z}} z^{-1} dx dz = \frac{\varphi_p}{2}.$$

The spatial storm radii parameter $\gamma_s(z)$ is a constant γ_s, which implies the temporal rate of storm arrivals at an arbitrary point $p \in \mathbb{R}^2$ is also a constant $\lambda = \frac{2\pi\zeta_s}{\gamma_s^2}$.
The model parameters $\lambda, v_0, v_1, \beta, \eta$ can be estimated by the following dimensionless functions. These can be surely computed for the real data; besides, by quantities defined at the end of the previous section, we can also obtain their theoretical expressions.

- Coefficient of variation:

$$\nu(h) := \frac{\mathbb{E}\left[(S_{p,i}(h) - \mathbb{E}[S_{p,i}(h)])^2\right]^{\frac{1}{2}}}{\mathbb{E}[S_{p,i}(h)]} = \frac{\sqrt{Var(S_{p,i}(h))}}{\mathbb{E}[S_{p,i}(h)]} = \frac{\phi_p^{1/2}(h, 0)}{\mu_p(h)};$$

- lag 1 autocorrelation:

$$\rho(h) := Corr[S_{p,i}(h), S_{p,i+1}(h)] = \frac{\phi_p(h, 1)}{\phi_p(h, 0)};$$

- skewness:

$$\kappa(h) := \frac{\mathbb{E}\left[(S_{p,i}(h) - \mathbb{E}[S_{p,i}(h)])^3\right]}{\mathbb{E}\left[(S_{p,i}(h) - \mathbb{E}[S_{p,i}(h)])^2\right]^{\frac{3}{2}}} = \frac{\psi_p(h)}{\phi_p^{3/2}(h, 0)},$$

where we have used relations (3.5)-(3.8). Note that all the functions do not depend on the site: for any h, they take the same value for all sites $p \in R^2$. From equations (3.5)-(3.8), the dependence on the model parameters it is also clear.
The radii parameters γ_s and γ_c are instead estimated using the cross correlations for each pair of sites $p, q \in \mathbb{R}^2$

$$\rho_{p,q}(h) := \frac{Cov[S_{p,i}(h), S_{q,i+l}(h)]}{\sqrt{Var(S_{p,i}(h))Var(S_{q,i}(h))}} = \frac{\phi_{p,q}(h, 0)}{\phi_p^{1/2}(h, 0)\phi_q^{1/2}(h, 0)}.$$

Let us give the detailed procedure. Firstly, for each site-month, we divide the 10 minutes values by the hourly rainfall mean. In this way, we obtain a new series with unit mean at the 1 h aggregation level, and more generally, with mean h at the h-th aggregation level. So, the coefficient of variation $\nu(h)$ becomes $\frac{\phi_p^{1/2}(h,0)}{h}$. Secondly, for each month $m = 1, \ldots, 12$, the site data are pooled and the sample coefficient of variations $\tilde{\nu}_m(h)$, the lag 1 autocorrelation $\tilde{\rho}_m(h)$ and the sample skewness $\tilde{\kappa}_m(h)$ are evaluated at different aggregations level $h = 1/6$ h, 1 h and 24 h. Then, the first group of parameters is estimated separately for each month by minimizing the following cost function:

$$\sum_{h=\frac{1}{6},1,24} \left\{ \left(1 - \frac{\tilde{\nu}_m(h)}{\nu_m(h)}\right)^2 + \left(1 - \frac{\nu_m(h)}{\tilde{\nu}_m(h)}\right)^2 + \left(1 - \frac{\tilde{\rho}_m(h)}{\rho_m(h)}\right)^2 + \left(1 - \frac{\rho_m(h)}{\tilde{\rho}_m(h)}\right)^2 + \left(1 - \frac{\tilde{\kappa}_m(h)}{\kappa_m(h)}\right)^2 + \left(1 - \frac{\kappa_m(h)}{\tilde{\kappa}_m(h)}\right)^2 \right\}. \tag{3.9}$$

The sum contains two terms per model function, one of which contains the model function divided by the sample value and the other contains the reciprocal term. This helps to ensure that the optimal solution is not biased from above or below the sample values, when an exact fit to the sample value is not obtained. The sum of squared terms (3.9) is minimized using the simplex method by Nelder and Mead ([20]), described herein in Section 3.2.
Using the estimates of $\tilde{\lambda}_m, \tilde{v}_{0,m}, \tilde{v}_{1,m}$ and $\tilde{\eta}_m$ the scale parameter $\varphi_{p_j,m}$ is estimated for each site p_j and each month $m = 1, \ldots, 12$ using the next equation, which follows directly from (3.5),

$$\tilde{\varphi}_{p_j,m} = \frac{6\tilde{\mu}_{p_j,m}(1)\tilde{\eta_m}}{\tilde{\lambda}_m(\tilde{v}_{0,m} + 2\tilde{v}_{1,m})},$$

where $\tilde{\mu}_{p_j,m}(1)$ is the 1 h sample mean.
In the same way, the radii parameters are estimated minimizing, by the simplex method, the cost function

$$\sum_{j \neq k} \sum_{h=\frac{1}{6},1,24} \left\{ \left(1 - \frac{\rho_{p_j,p_k}(h)}{\tilde{\rho}_{p_j,p_k}(h)}\right)^2 + \left(1 - \frac{\tilde{\rho}_{p_j,p_k}(h)}{\rho_{p_j,p_k}(h)}\right)^2 \right\},$$

where $\tilde{\rho}_{p_j,p_k}(h)$ is the sample cross correlations for the pair of sites p_j, p_k for $j, k = 1, \ldots, 23$.
The above strategy replace the more popular strategy of maximum likelihood estimation, which reveals to be infeasible in this case, as likelihood functions are not usually available for point process rainfall models.

Chapter 4

A Hidden Markov model for rainfall

Our goal is now to formulate a new model for rainfall based on the Cowpertwait's one, in which parameters depend on a hidden Markov chain not directly observed. In formulating the model we define a random process $\{I_n(x)\}_{n\geq 1}$ for the intensity of rain in x in discrete time, whose parameters depend on a hidden Markov chain; spatial points can be thought as locations among a certain geographic area of interest whilst discrete time usually represents days. The intensity process will then be the observed process of a Hidden Markov model.
In order to fit the model, we will use a time series of observations of daily rainfall intensity from $L = 30$ rain gauges (site) across Germany during a period of T days (corresponding to a period of 70 years, from January 1931 to December 2000). The fitting procedure will make use of Hidden Markov models' properties as well as statistical tools, like Monte Carlo simulation and Approximate Bayesian Computation.

4.1 Model formulation

In formulating the model, in the first place, we emulate Cowpertwait's discs structure and, at the same time, we try to simplify it since the double discs system is abolished. Correlations which were originally induced by bigger storm discs are now intended to be enclosed in the hidden Markov chain mechanism.
We suppose there is a random number of rain cells, with random location and radius, that gives rise to a random rain intensity over the disc's area. The main difference from Cowpertwait's is that, here, we assume parameters

depend on a hidden Markov chain.
More precisely, given a point $x \in \mathbb{R}^2$ (or, better, in a certain region of interest $A \subset \mathbb{R}^2$) and an integer time $n \geq 1$, we want to write an expression which can give the intensity of rain in x during the day n. So, we define the intensity process $\{I_n(x)\}_{n\in\mathbb{N}}$ such that for $n \geq 1$ and $x \in \mathbb{R}^2$,

$$I_n(x) := \sum_{i=1}^{N^{(n)}} \sigma_i^{(n)} \mathbb{I}_{D(x_i^{(n)}, R_i^{(n)})}(x), \tag{4.1}$$

where

- $D(x_i^{(n)}, R_i^{(n)})$ is a rain disc with centre in $x_i^{(n)}$ and radius $R_i^{(n)}$, with $R_i^{(n)} \sim \exp(\xi)$ and $x_i^{(n)} \sim \mathrm{U}(A)$;
- $\sigma_i^{(n)} \sim \exp(\tau)$ is a random variable that gives the rain intensity;
- $N^{(n)} \sim \mathrm{Pois}(\lambda)$ represents the number of rain discs generated.

In this way, $\{I_n(x)\}_{n\in\mathbb{N}}$ is the observable process of a Hidden Markov model; the parameters $\lambda, \xi, \tau \in \mathbb{R}^+$ are taken to depend on a hidden Markov chain $(J_n)_{n\in\mathbb{N}}$, with state space $\{1, \ldots, m\}$, where $m \in \mathbb{N}$ is a fixed parameter.
So, for each $j = 1, \ldots, m$, given the chain is in state j at time n, $\{J_n = j\}$, we have certain values for the distribution parameters of N, σ and R; let us denote them as λ_j, ξ_j and τ_j. In fact, λ, ξ and τ are vectors in $\mathbb{R}_+^m$, such that $\lambda = (\lambda_1, \ldots, \lambda_m)$, $\xi = (\xi_1, \ldots, \xi_m)$ and $\tau = (\tau_1, \ldots, \tau_m)$.
Besides, let us denote by $\delta = (\delta_1, \ldots, \delta_m)$ the initial distribution of the chain and $\Gamma = (\gamma_{ij})_{i,j=1,\ldots,m}$ its transition probability matrix.
Similarly to what stated in Chapter 1, let $p_j(i_n)$ denote the state-dependent density function of I_n associated with state j evaluated in i_n, where i_n denotes the rainfall intensity observed in one of L sites available, for $n \geq 1$, $j = 1, \ldots, m$.
As we have seen in Chapter 1, these densities have a central role in likelihood evaluation and, thus, in solving the learning problem. However, in this case, the complex model formulation (4.1) do not led to an explicit formula to express the state-dependent density function. The procedure of finding the parameters that best adapt to observations can not follow exactly methods described for general Hidden Markov model. In the next sections, we explore possible solutions to this problem.

4.2 Fitting procedure

The objective of the fitting procedure is to find an estimation of all the model parameters which better adapt to the data. That is obviously of great

interest in this context, as it subsequently allows to use the model for simulations. The parameters required in this context are the initial distribution δ, the transition matrix probability Γ of the Markov chain and the distribution parameters λ, ξ, τ specific of this process.
As mentioned before, we use $L = 30$ spatial points, where each point corresponds to the location of a rain gauge from which observed daily data are available for a period of 70 years. The sites, which are here identified as points $x_1, \ldots, x_L \in \mathbb{R}^2$, are located in Germany and have data in the period $1931 - 2000$.
As we have seen in Chapter 1, the most natural way to find the best set of parameters in Hidden Markov model, i.e., to solve the learning problem, it is trough a maximum likelihood estimation procedure. In particular, the Expectation Maximization algorithm usually results the best way to do it.
The likelihood of T consecutive observations $(i_1, \ldots, i_T)$ in a fixed site assumed to be generated by the model presented herein can be written, by equation 1.2, as

$$\begin{aligned} & L_T(i_1, \ldots, i_T) \\ = & \sum_{j_1, \ldots, j_T = 1}^{m} \delta_{j_1} p_{j_1}(i_1) \gamma_{j_1 j_2} \ldots \gamma_{j_{T-1} j_T} p_{j_T}(i_T), \end{aligned}$$

where, as usual, j_n denotes the state of the chain at time n, for $n = 1, \ldots, T$.
In a general context, from this expression it would be possible to estimate the required parameters either by direct likelihood maximization or by the Expectation Maximization algorithm. However, the main problem we have to face is that we do not have an explicit expression for the state-dependent density functions, so adopting these techniques could be no-trivial.
In order to solve this problem we present and test two different approaches: the first one, in the most natural way, attempts to estimating the state-dependent distribution by a Monte Carlo simulation and, subsequently, applying procedures described in Chapter 1. As this procedure results to be computationally expensive, an alternative approach is considered. Instead of approximating the density functions, we bypass the likelihood evaluation and we use the Approximating Bayesian Computation, described in Chapter 2, to directly estimate the parameters.

4.2.1 Maximum likelihood estimation

As illustrated in the previous section, the main task to face in maximum likelihood estimation for the model defined by (4.1) is the absence of an expression for the state-dependent density functions p_j of I_n associated with

state $j \in \{1, \dots, m\}$. In this section, we present results obtained by approximating them trough a Monte Carlo simulation.
In this procedure, we firstly fix values for the model parameters and a state j for $j \in \{1, \dots m\}$ for which computing the corresponding density. According to Monte Carlo heuristic, we then compute a large number T_{mc} of possible intensities, at a fixed time n, given the chain is in state j, by simulating the model. Now, as we want to approximate a continuous-valued function some precautions need to be taken. In the following discussion, let us consider a fixed intensity $\bar{i}$. If the intensity process assumed discrete values, the state-dependent density would be a probability, say $\bar{p}_j(\bar{i})$. So, it would be approximated by the quotient of the number of occurrences of $\bar{i}$ and the total number of simulation. Namely, it would be

$$\bar{p}_j(\bar{i}) \approx \frac{1}{T_{mc}} \sum_{k=1}^{T_{mc}} \mathbb{I}_{\{I_n^{(k)}=\bar{i}\}}, \tag{4.2}$$

where $I_n^{(k)}$ denotes the k-th simulated intensity for $k = 1, \dots, T_{mc}$. In this case, as it is well known, by the law of large numbers it holds

$$\frac{1}{T_{mc}} \sum_{k=1}^{T_{mc}} \mathbb{I}_{\{I_n^{(k)}=\bar{i}\}} \quad \underset{T_{mc}\to\infty}{\longrightarrow} \quad \mathbb{E}[\bar{p}_j(\bar{i})],$$

so for large value of T_{mc} we obtain a good approximation of the mean value of $\bar{p}_j(\bar{i})$.
In our case, as the intensity process is continuous-valued, in first instance, we approximate it by a piecewise constant function. Specifically, we set a large enough interval in which intensity may fall and we divide it in T_{int} equispaced sub-intervals, for a chosen integer $T_{int} > 0$. Then, for every sub-interval we compute the numbers of simulated intensities which fall in it and we divide it by T_{mc}; we thus obtain the piecewise approximation required.
In order to have a smoother approximation of the density p_j, we subsequently perform a linear interpolation between each midpoint of the sub-intervals and the corresponding value for p_j computed by (4.2).
In practice, in order to have a singular real observation and following Cowpertwait's system, the site data are pooled. Once we have established these operations, the maximum likelihood estimation can be performed.
At first, direct maximization of likelihood is performed by numerical algorithms. In doing so, the first step is formulating a function for likelihood evaluation: it receives as input the model parameters $\delta, \Gamma, \lambda, \xi$ and τ, then trough the Monte Carlo simulation described above, the state-dependent densities are computed and, finally, the likelihood is computed by mean of the

scaling algorithm described at the end of Section 1.3, in order to avoid numerical underflow. The second step is the optimization. It is a constrained optimization, as all the parameters components should be non negative and, additionally, Γ should be a stochastic matrix. Moreover, nothing is known about the smoothness of the likelihood function. We then choose to adopt the Nelder-Mead method presented in Section 3.2. It also has the great advantage of requiring only few evaluations of the likelihood function at each step.
These operations are clearly computationally expensive. So, in order to reduce the overall cost, some simplifications are made. At first, to reduce the dimension of the problem, the initial distribution δ and the transition matrix probability Γ of the Markov chain are taken to be fixed and they are not estimated. Furthermore, only observations for three of the 70 years are considered. Subsequently, the opposite procedure is considered: some values are established for λ, ξ and τ, while Γ and δ are estimated. This second practice has the advantage of requiring only one initial Monte Carlo simulation, as the state-dependent density functions do not depend on the transition matrix probability and the initial distribution of the hidden Markov chain. Besides, in implementing it, there is not need to use the Nelder-Mead algorithm: the likelihood function is smooth with respect to Γ's and δ's entries and transformations outlined in Section 1.4.3 can be used. An unconstrained optimizer based on Newton-type algorithm is instead used.
A second approach to achieve maximum likelihood estimation is the Expectation Maximization algorithm for Hidden Markov model, described in Section 1.4.2. After setting an initial random choice for the model parameters $\delta, \Gamma, \lambda, \xi, \tau$, the forward and backward probabilities need to be computed. In particular, their expression, given in Definition1.2.3 and in Definition 1.2.4, require the state-dependent density functions to be evaluated. They are again estimated by mean of a Monte Carlo simulation. Further, the density functions appear at each iteration of the M step of the algorithm and they have to be maximized with respect to the distribution parameters λ, ξ, τ. Even in this case, this task is achieved by mean of the Nelder-Mead method, due to the unknown properties of the expression under consideration. More precisely, after the initialization, the following steps are repeated until convergence:

1. the state-dependent density function p_j are computed by a Monte Carlo simulation for each $j = 1, \ldots, m$;

2. the forward and backward probabilities, (and so the auxiliary variables $\hat{u}_j(n)$ and $\hat{v}_{jk}(n)$ for $j, k = 1, \ldots, m$ and $n = 1, \ldots, T$, are computed us-

ing functions obtained in (1)). In other words, the E step is performed using the density functions from (1);

3. the M step is accomplished: in particular, new values for δ e Γ are calculated according to equation (1.6), whilst λ, ξ, τ are obtained by maximizing the third term of Equation (1.5), substituting p_j with their approximations of step (1), this optimization is made trough the Nelder-Mead method.

The algorithm terminates when the difference between the resulting parameters is smaller than a fixed threshold.
It is easy to conclude that this second approach do not improve the computational effort needed to produce a reasonable estimates of the values needed and, in practice, it turns out to be relatively inefficient.
In conclusion, adopting a Monte Carlo simulation seems not to be in practice the best way to solve the problem of the missing density due to its high computational cost. Thus, in the next section, an alternative strategy, which avoids completely the likelihood evaluation, is presented and analysed.

4.2.2 Parameters estimation via Approximate Bayesian Computation

We describe here how the Approximate Bayesian Computation can be adopted to achieve parameters estimation in the Hidden Markov model for rainfall. In this context, we refer to Approximate Bayesian Computation as the ABC rejection sampling algorithm described in Section 2.2 with kernel function chosen to be the uniform kernel. Even if the literature is lacking of examples of application of Approximate Bayesian Computation to Hidden Markov models, some theoretical results can be found in [14].
As illustrated in Chapter 2, this method has, in general, the advantage of being likelihood-free; so, it lets, at least in theory, to circumvent the main problem of our rainfall model. Nevertheless, the high dimension of the problem, together with the usual choices to be made in the algorithm (i.e., distance, summary statistics) affect the performance of the Bayesian method.
The number of parameters to be estimated is $(m^2 + 4m)$, where m is the number of states of the chain; estimating all of them may be too expensive or even infeasible for the method. For this reason, at first, we assume to have a fixed transition probability matrix Γ and an initial distribution δ and we look for an estimate of the remaining parameters $\lambda, \xi, \tau \in \mathbb{R}_+^m$.
Very little is known about these parameters and finding an appropriate prior distribution seems to be a difficult task to accomplish. Moreover, even for established Γ and δ, the number of parameters remains significant. To overcome

these problems, at least partially, we then adopt the following strategy, which also furnishes a remarkable way to test the Approximate Bayesian Computation's performance for the model considered. The main idea is choosing some initial values for the model parameters, which can give rain intensities *close* (in a sense to be specified) to the real ones, then simulating the model with these parameters. Rainfall intensities thus obtained are then used as real data to test Approximate Bayesian Computation's capability to recover the initial *real* parameters. This technique is purely heuristic and it is only aimed to explore the problem, as decisions made are completely arbitrary.
We begin by picking possible values for each component of λ, ξ and τ, then we simulate our model, i.e., the intensity process, using the parameters chosen, the initial distribution for the chain and the transition probability matrix fixed initially. At the same time, we also choose a summary statistic which will be used in applying Approximate Bayesian Computation. In selecting the parameters, we also compute the summary statistic both for the real data and the simulated ones and we eventually pick the parameters whose corresponding statistics are closer to the real data's one, according to an established metric.
Specifically, we randomly generate a probability vector $\delta \in \mathbb{R}^m$ and a stochastic matrix $\Gamma \in \mathbb{R}^{m \times m}$. We choose as summary statistic the mean rain intensity in each site in the period under consideration. Now, we choose some possible values for λ, ξ and τ randomly; in doing so, an effort is made in order to have a coherent setting: for $j = 1, \dots, m$, the rain intensity is intended to increase as j gets higher. Therefore, the mean rain intensity is computed for both real data and simulated ones and they are compared. We eventually choose as parameters the ones that gives the lowest distance between the two summary statistics.
In order to test the method's performances, the real data are temporarily set apart and the selected parameters together with the corresponding simulated rain intensity series are assumed to be the *real data* we will use in the estimation procedure. In the following, we refer to them as the *ad hoc* data and parameters. The aim is applying the Approximate Bayesian Computation to these, choosing as prior for the parameters an uniform distribution in a small neighbourhood of the *ad hoc* parameters. This eventually allows to evaluate the gap between results obtained by the Bayesian method and the *ad hoc* ones.
In applying Approximate Bayesian Computation, further decisions have to be made: we keep the summary statistic previously set and we consider the metric to be the Euclidean distance in $\mathbb{R}^L$. At first instance, as nothing is known about the distance between the mean of simulated data and the mean of *ad hoc* data, we do not fix any threshold ϵ. We simply simulate the model

a great number of times with parameters chosen according to the prior distribution; for each time, we save the Euclidean distance between the mean of simulated data and the mean of *ad hoc* data as well as the parameters which have given rise to the simulated intensities. If the method is truly efficient, lower distance values will correspond to parameters estimates closer to the *ad hoc* ones and vice versa. In other words, if we plot each component of the parameters values against the corresponding distances, we will expect a concentration of points around the true parameter values for small distances which become more sparse as distance increases, a sort of conic-shaped set of points.
In practice, when we attempt to estimate all the $3m$ components of the parameters λ, ξ and τ, the plot obtained do not present a clear "conic" structure, as results presented in Section 5.2 show. Further exploration are then made: in particular, following the same strategy described above, we test Approximate Bayesian Computation's capability of estimating a subset of the complete set of parameters, i.e., at first m components of the total $3m$ elements of λ, ξ, τ are considered and, then, each component singularly.
This last trial finally gives satisfactory outcomes. This allows to perform Approximate Bayesian Computation on real data in order to estimate, in first instance, the component parameters for which adequate estimates are obtained in the test procedure. The outputs achieved are collected in Chapter 5. Moreover, further tests can be made aimed to analyse the performances of an iterative method that can be developed from these.

Chapter 5

Numerical results

This chapter is committed to numerical results obtained in the fitting procedure for the model described in Chapter 4.
Data used to calibrate the model are daily time series for rainfall intensity (expressed in 0.1 mm of rain). Statistics are available for 30 sites across Germany from 1st January 1931 to 31st December 2000. [1]

Table 5.1: Spatial location of 30 rainfall gauges across Germany. Percentage of missing values for each site.

Site	$\mathbf{x_1}$ (m)	$\mathbf{x_2}$ (m)	**Missing (%)**	**Site**	$\mathbf{x_1}$ (m)	$\mathbf{x_2}$ (m)	**Missing (%)**
1	538354	5455159	1.43	16	503847	5363207	1.43
2	491036	5738802	2.86	17	506649	6009451	0.13
3	503044	5477483	0	18	504130	5983190	2.86
4	649852	5329931	3.21	19	381345	5514918	2.86
5	607972	5523257	0	20	561051	5781991	2.86
6	664833	5415885	0.72	21	350269	5450610	0.13
7	519984	5502867	2.86	22	544660	5904963	2.86
8	588063	5441942	0.61	23	667321	5571289	2.86
9	494323	5455251	1.43	24	704817	5702654	0
10	647976	5501735	0	25	623624	6011886	1.43
11	605168	6000630	2.86	26	387943	5821494	0
12	426656	5537020	0	27	645452	5875581	2.86
13	503652	5645378	1.43	28	682269	5538726	2.86
14	532874	5969508	1.43	29	379112	5581374	3.09
15	668862	5515364	1.08	30	456839	5787142	0

[1] Klein Tank, A.M.G. and Coauthors, 2002. Daily dataset of 20th-century surface air temperature and precipitation series for the European Climate Assessment. Int. J. of Climatol., 22, 1441-1453. Data and metadata available at https://www.ecad.eu

As shown in Table 5.1, sites are identified by their first two coordinates (Easting x_1 and Northing x_2). In Table 5.1 the percentage of missing values is also indicated; in the fitting procedure missing values are excluded.
In all the numerical simulations, the number of states of the hidden Markov chain is fixed to be $m = 3$. Ideally, the three states of the chain are thought to model different intensities of precipitation (respectively low, medium and high rain).
Finally, the region in which data are simulated is rectangular and it is chosen from site's coordinates such that each site is in the region, not in a peculiar position (e.g., on a edge).
The experiments have been run on the personal laptop (Intel(R) Core(TM) i5-4210U CPU with 2 cores running at 1.70 GHz) and on the cluster of the University of Sussex (Intel(R) Xeon(R) E5-2640 CPU with 8 cores running at 2.6 GHz) using R (R version 3.6.2 and 3.5.1 respectively).

5.1 Maximum Likelihood Estimation

This section is dedicated to results concerning parameters estimation trough maximum likelihood estimation, both via direct maximization and via the Expectation Maximization algorithm. The general scheme adopted is the one illustrated in Section 4.2.1. In particular, steps required for Monte Carlo simulations are easily carried out in R by mean of the command *hist*. It furnishes the interval extent in which simulated intensities fall as well as an equispaced partition of it (which can also be refined). Moreover, the R function *approxfun* is used to perform the linear interpolation required to obtain density functions. As discussed above, estimating all the $m^2 + 4m$ parameters is computationally prohibitive, so only a subset of the complete set of parameters is considered. At first, a random stochastic matrix and a probability vector are set as fixed values for Γ and δ, in order to calibrate the remaining λ, ξ and τ. Then, vice versa, an attempt is made to estimate the Markov chain parameters for established values of λ, ξ and τ. Eventually, the model is simulated according to the parameters found and compared with real data. Specifically, some quantities of interest, such as the mean annual intensity for each site, are computed.

5.1.1 Direct Maximization

We present here the outcomes from direct maximization of likelihood trough numerical algorithms. In first instance, results derived from estimating only the distribution parameters are shown. Secondly, the analogous

outcomes are displayed for the case of Γ and δ estimation.
We begin by generating randomly a probability vector δ and a stochastic matrix Γ, by normalizing properly what obtained by mean of the R function *runif*. Then, as it is computationally infeasible to consider all the available data, we set the number of days to be $T := 365 \times 3$, so that only the first three years of the time series are effectively examined. In establishing the number of Monte Carlo's iterations two issues have to be contemplated: the state-dependent density functions must have a meaningful support and performing all the simulations must be computationally feasible. By doing some proofs, setting $T_{mc} = 6 \times 10^3$ seems to be a good choice. The R function *Nelder_Mead* (from the R package *lme4*) is used in the optimization procedure; this version allows optimization subject to box constraints. The initial choice for the parameters values, to be given as an input for the function, is made randomly as indicated below:

- $\lambda_1 \sim U([4, 12])$, $\lambda_2 \sim U([25, 35])$ and $\lambda_3 \sim U([38, 45])$;
- $\xi_1 \sim U([0.04, 0.1])$, $\xi_2 \sim U([0.005, 0.1])$ and $\xi_3 \sim U([0.0008, 0.004])$;
- $\tau_1 \sim U([0.8, 1.4])$, $\tau_2 \sim U([0.09, 0.3])$ and $\tau_3 \sim U([0.04, 0.1])$.

The procedure took 16.51 hours to be completed on the personal laptop. We show the outcomes from simulating the model under the parameters found by the scheme illustrated above. In particular, we focus on the mean rainfall intensity in each site, since it represents a first significant measure in rainfall estimates. In order to evaluate the method's performance, results are shown for simulated data obtained both with the initial choice for parameters (pre-optimization) and with the ones resulting from the minimization (post-optimization).

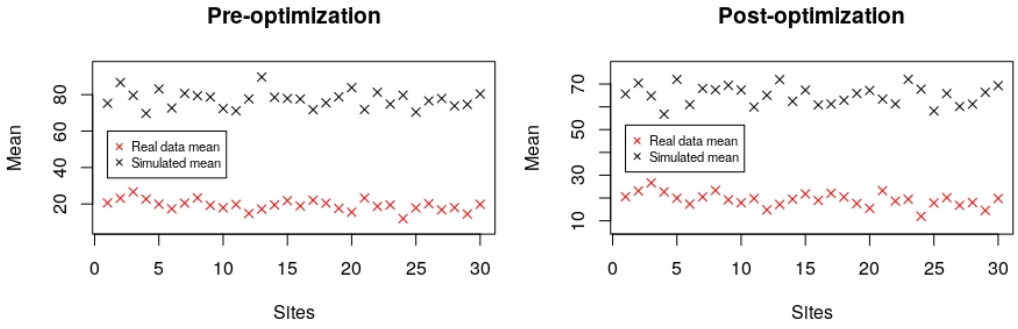


Figure 5.1: Mean rainfall intensity in each site for a period of 3 years.

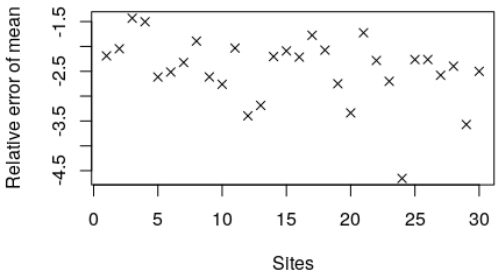


Figure 5.2: Relative error of the mean rainfall intensity for a period of 3 years.

The mean rainfall intensity is certainly not well estimated by the simulated model with the estimated parameters. Moreover, we notice that there is only a slight improvement on the results from the case where parameters are chosen randomly and the one with optimized parameters. In fact, if we compute the difference between initial and final parameters, we find out they are unnaturally close to zero, as Table 5.2 shows.

Parameter	Difference	**Parameter**	Difference	**Parameter**	Difference
λ_1	0.0001077	ξ_1	0.0002789	τ_1	0.0004849
λ_2	0.0004849	ξ_2	0.0016649	τ_2	0.0004849
λ_3	0.0104849	ξ_3	0.0004849	τ_3	0.0051763

Table 5.2: Differences between initial parameters and parameters obtained after the optimization procedure (with Nelder-Mead method).

The causes of this behaviour may be found in the highly irregular form of the likelihood function; for example, it may have many local minima in which the Nelder-Mead algorithm prematurely stops its iterations. The small dimension of real data adopted may affect the outcomes as well. Therefore, this procedure needs some refinements to be used. One possible choice could be reducing the number of elements to be estimated.

Let us consider the second case mentioned. As previously discussed, the overall computational cost is inferior, thus it allows to consider a greater number of observations than before. We set $T = 365 \times 10$ days. At the same time, the number of Monte Carlo's iterations T_{mc} must be higher, let it be $T_{mc} = 10^5$. Besides, to achieve the optimization step, transformations illustrated in Section 1.4.3 are made (choosing as transformation function $g(x) = e^x$) and the R function *nlm* is used; it carries out a minimization using a Newton-type algorithm.
Other than the outcomes presented in the previous case, we also show here

the box plot of the annual daily mean rainfall for each site, compared with the real data, as well as results concerning the standard deviation. This is interesting in this context as it gives an indication of how values are far from their mean in real data and simulated ones and so we can evaluate the variation in both the rainfall patterns.
The choice of λ, ξ and τ is made randomly: each component is chosen to be uniformly distributed in the same intervals indicated for the previous case. We then simulate the model for the same period with the new values for Γ and δ. Even in this case, we report outcomes obtained after the optimization procedure as well as the ones we had before the minimization, in order to show the improvements achieved. The R programmes used needed 7.15 hours to generate all the following data on the personal laptop.

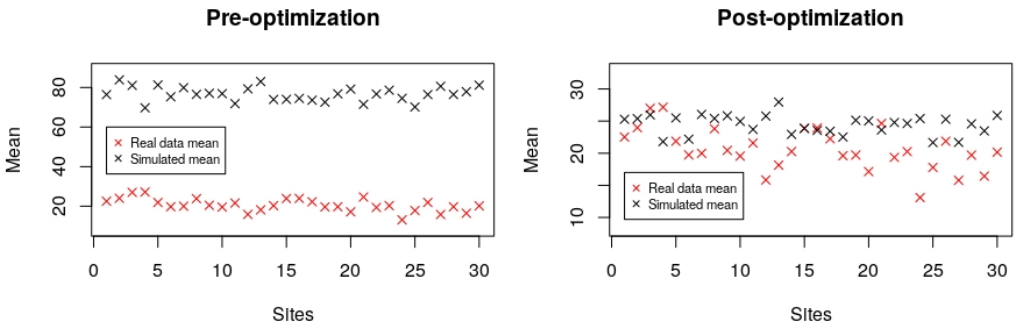


Figure 5.3: Mean rainfall intensity in each site for a period of 10 years.

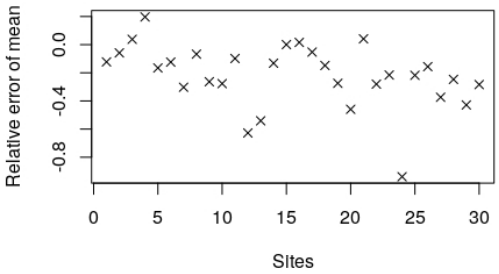


Figure 5.4: Relative error of the mean rainfall intensity for a period of 10 years.

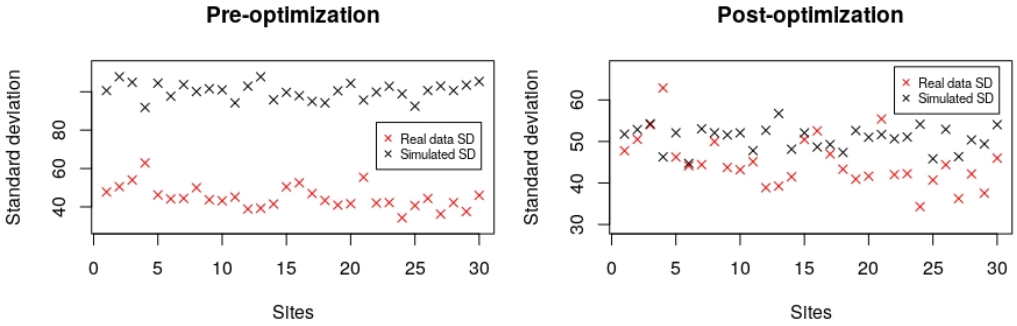


Figure 5.5: Standard deviation of rainfall intensity in each site for a period of 10 years.

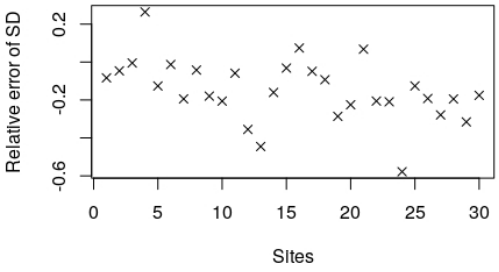


Figure 5.6: Relative error of the standard deviation of rainfall intensity for a period of 10 years.

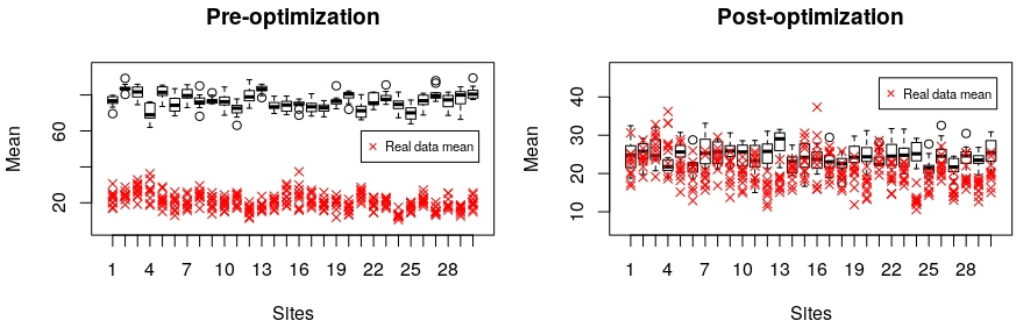


Figure 5.7: Box plot of annual daily mean rainfall for a period of 10 years for each site. Red crosses are real data.

Above results are quite satisfactory. In first instance, they show that the optimization procedure effectively performs well and allows to have data-adapted initial distribution and transition probability matrix. Moreover, even if the parameters λ, ξ and τ are chosen randomly (yet with some precautions), the

simulated model achieves mean rainfall intensities closed to the real ones. Even the real standard deviation's behaviour is captured by the simulated data. The mean relative error between sites for the mean and the standard deviation are quite close to zero, respectively

$$\epsilon_{mean} = 0.2382977 \qquad \text{and} \qquad \epsilon_{sd} = 0.1761288.$$

So, the Hidden Markov model simulated with the parameters given by the minimization procedure, together with the specific parameters λ, ξ and τ chosen initially, accomplish the task of emulate the real time series. In particular, the hidden Markov chain really has a key role in the rainfall mechanism. This is a good starting point to use the model for the purpose stated initially, even if some improvements may be done. In order to have stronger and more general results from a statistical point of view, these proofs may be repeated for a large number of values for the parameters; however, in our cases, the high computational costs have allowed to do so only few times (and analogous results are obtained).

5.1.2 Expectation Maximization algorithm

By the Expectation Maximization algorithm we attempt to estimate all the model parameters. Due to the necessity of using Monte Carlo simulation, it results to be an expensive procedure, not as efficient as in the standard case. Moreover, the high computational costs do not allow to perform completely the iterations of the Expectation Maximization algorithm. Indeed, computing only one iteration for a period of $T = 365$ days with the number of Monte Carlo iterations fixed to $T_{mc} = 5 \times 10^4$ required 18.69 hours.
We anyway simulate the model for the parameters obtained; results are better than what we expected. We display below, the same statistics considered for the case of direct maximization.

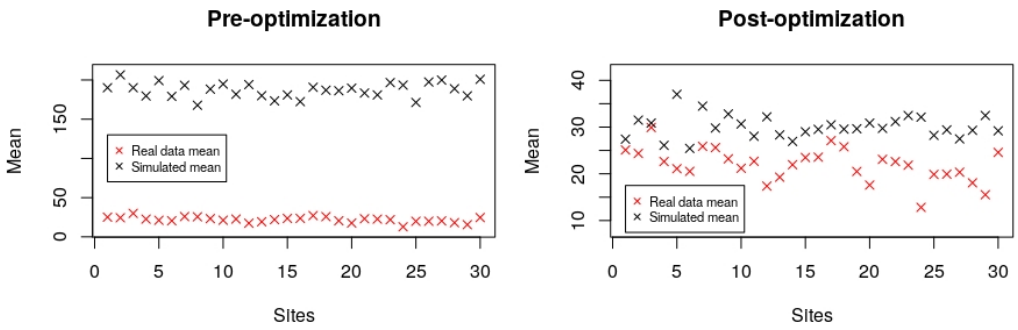


Figure 5.8: Mean rainfall intensity in each site for a period of one year.

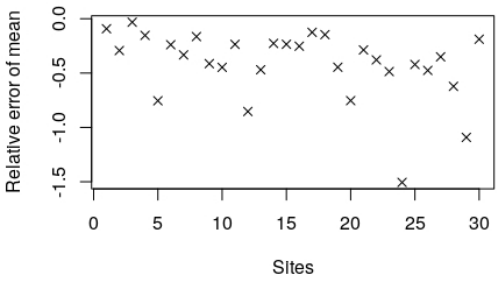


Figure 5.9: Relative error of the mean rainfall intensity for a period of one year.

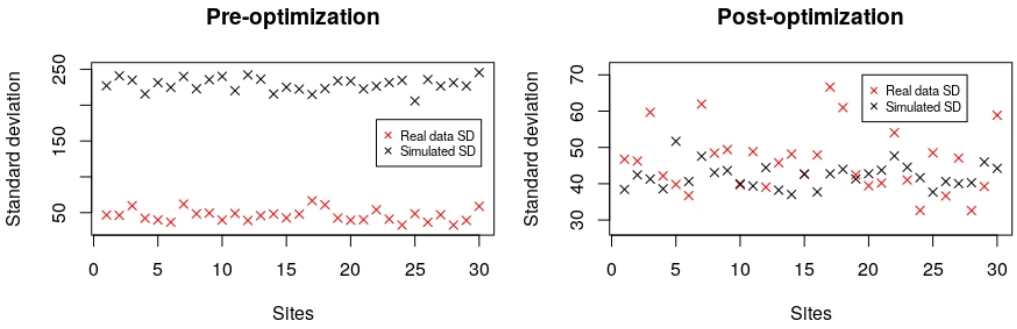


Figure 5.10: Standard deviation of rainfall intensity in each site for a period of one year.

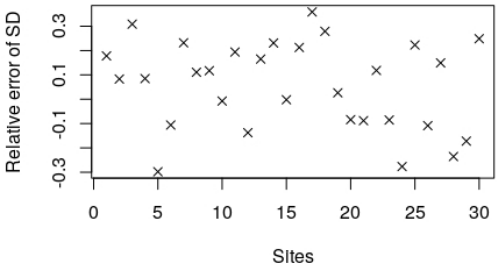


Figure 5.11: Relative error of the standard deviation of rainfall intensity for a period of one year.

From further analysis, we find that the unexpected behaviour of the Expectation Maximization algorithm, even after only one iteration, is due to its capability of estimating the initial distribution δ and the transition probability matrix Γ of the Markov chain, rather than the distribution parameters λ, ξ and τ. Indeed, as Table 5.3 displays, the difference between the initial values and the optimized ones is low. Even in this case, the complexity of

the expression to be minimized badly affects the Nelder-Mead method's performances.

Parameter	Difference	**Parameter**	Difference	**Parameter**	Difference
λ_1	0.0038722	ξ_1	0.0089056	τ_1	0.0027747
λ_2	0.0021145	ξ_2	0.0003589	τ_2	0.0065559
λ_3	0.0028909	ξ_3	0.0058480	τ_3	0.0025211

Table 5.3: Differences between initial parameters and parameters obtained after the optimization procedure in the EM algorithm.

Results presented confirms conclusions made in previous sections. The Nelder-Mead method is again proved not to perform well in minimizing expressions involving Monte Carlo simulations. At the same time, the hidden Markov chain is further affirmed to have an important role in determining rainfall intensities.

5.2 Approximate Bayesian Computation

In this section, we present results obtained in applying the Approximate Bayesian Computation to our model. As described in Section 4.2.2, we first test the method's performance in estimating the parameters of simulated fake data. We begin by trying to estimate all the components of λ, ξ and τ, and we then reduce progressively the number of parameters to be estimated in order to obtain more accurate outcomes.
At first, we randomly generate the transition matrix probability Γ and the initial distribution δ. Then, we choose, according to the heuristic criteria presented above, the following values for the parameters, which will constitute, in this step, the so-called *ad hoc* parameters:

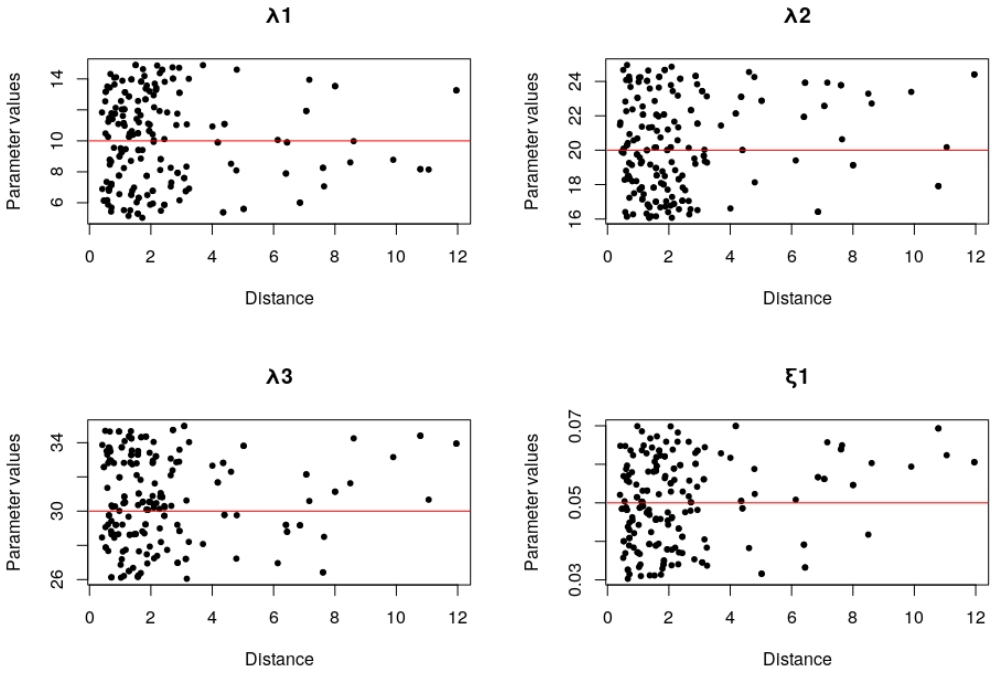
- $\lambda = (\lambda_1, \lambda_2, \lambda_3) = (10, 20, 30)$;
- $\xi = (\xi_1, \xi_2, \xi_3) = (0.05, 0.01, 0.005)$;
- $\tau = (\tau_1, \tau_2, \tau_3) = (1.1, 0.5, 0.1)$.

This selection provides mean values for each site close enough to the real mean. Moreover, one can note that the first component of each vector corresponds to, in order, lower mean of number of discs generated (λ), lower mean radius of the discs ($1/\xi$) and lower mean intensity ($1/\tau$). Each of them increases as the index of the component gets higher. Using these values, we

simulate the model in order to obtain a time series of rain intensity in the L sites: our *ad hoc* data. We generate data for a period of $T = 365 \times 5$ days. We then apply Approximate Bayesian Computation for $N = 200$ iterations, or better, we apply a slightly modified version in which we do not fix a threshold, but we save the distance together with the corresponding parameters. In doing so, we set the prior distribution of the parameters to be uniform in a small neighbourhood of the *ad hoc* ones, that is:

- $\lambda_1 \sim U([5, 15])$, $\lambda_2 \sim U([16, 25])$ and $\lambda_3 \sim U([26, 35])$;
- $\xi_1 \sim U([0.03, 0.07])$, $\xi_2 \sim U([0.008, 0.02])$ and $\xi_3 \sim U([0.001, 0.007])$;
- $\tau_1 \sim U([0.8, 1.4])$, $\tau_2 \sim U([0.3, 0.7])$ and $\tau_3 \sim U([0.07, 0.2])$.

At each iteration, values for the parameters are chosen according to these distributions, the rainfall intensity is computed according to the Hidden Markov model and the Euclidean distance between the *ad hoc* mean rainfall intensity and the simulated one is saved.
The following graphics show the results: for each component of the parameters, the simulated values are plotted against the Euclidean distance between the two summary statistics for all the 200 iterations, the red line represents the *ad hoc* value of the parameter component under consideration. Completing all the 200 iterations on the personal laptop required 40.63 hours.



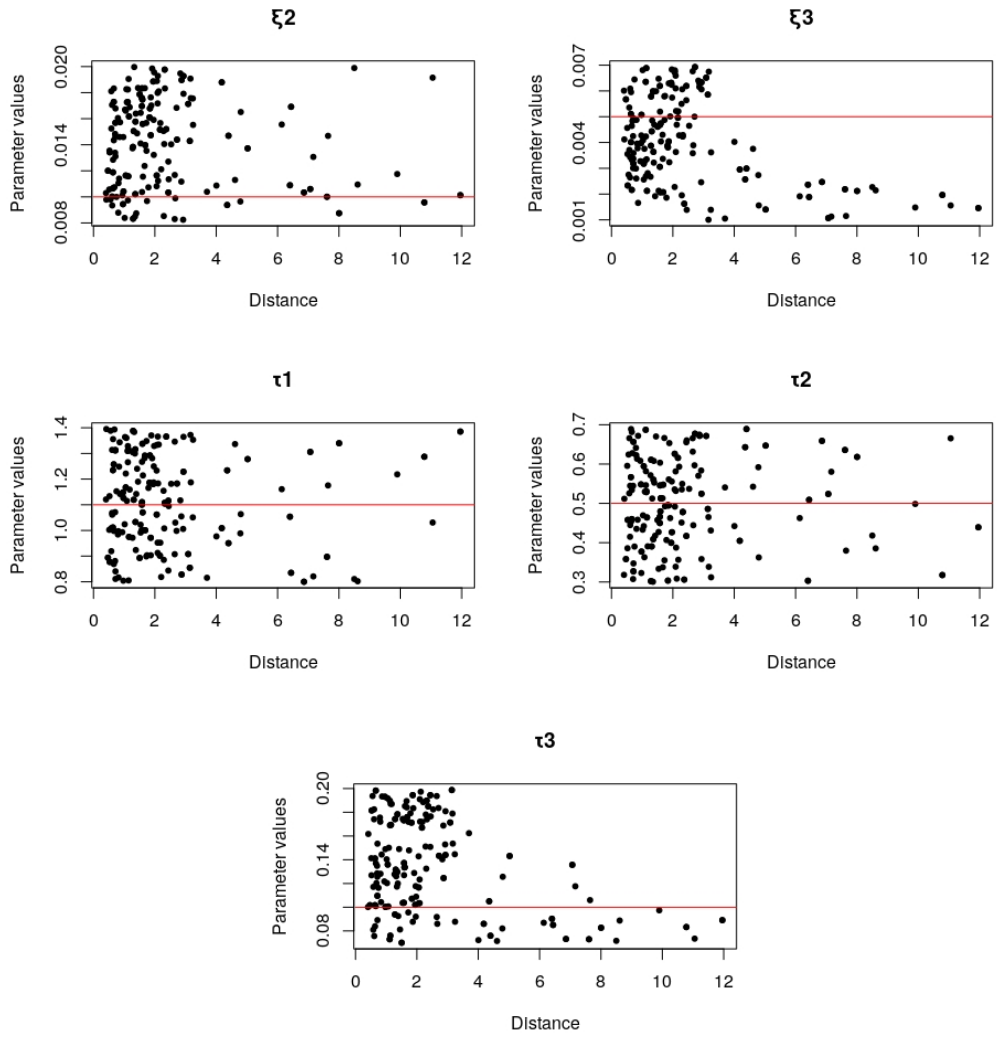


Figure 5.12: Parameters values plotted against the corresponding distance between summary statistics. The red line represents the true value.

Approximate Bayesian Computation does not to perform properly when estimating all the vector components. We expected that for small values of the distance, parameters would be more concentrated around the red line, i.e., their real value, whilst for higher values of the distance they were supposed to fall far from the line. In other words, if the method succeeded, we would observe points distributed according to their posterior distribution, which is concentrated on the *ad hoc* parameter values. Though, above graphics show that the posterior distribution is still uniform in a neighbourhood of the true parameters, so no improvements have been made by the method.
For the sake of completeness, in order to give also quantitative results, we

report in the Table 5.4 below, errors obtained for each parameter both for low and high values of the distance. In particular, an indicative threshold is fixed for the distance (i.e., $dist = 2$). Then, the standard deviation and the relative error of each parameter are computed in both the cases the distance is lower (case I) or higher (case II) than the threshold. In Table 5.4 differences between relative errors obtained are also shown.
These results give a further clue of the poor Approximate Bayesian Computation's capability of estimating all the parameters: even if errors achieved in the first case are (mostly) lower than the ones in the second case, there is a slight difference between them, not significant for our purpose.

Parameter	**SD I**	**Relative error I**	**SD II**	**Relative error II**	**Difference**
λ_1	2.82	0.282	2.97	0.297	0.015
λ_2	2.61	0.130	2.78	0.139	0.009
λ_3	2.74	0.091	2.53	0.084	0.007
ξ_1	0.010	0.210	0.011	0.226	0.016
ξ_2	0.005	0.531	0.005	0.534	0.003
ξ_3	0.001	0.315	0.002	0.471	0.156
τ_1	0.170	0.155	0.185	0.168	0.013
τ_2	0.104	0.209	0.121	0.242	0.033
τ_3	0.051	0.519	0.052	0.522	0.003

Table 5.4: Absolute and relative error for each parameter component for small distances (I) and for higher distances (II).

The next step is thus testing, according to the same procedure, Approximate Bayesian Computation's capability of recovering a further subset of the model parameters.
Choosing some subsets of three components (i.e., $\{\lambda_1, \lambda_2, \lambda_3\}$, $\{\lambda_1, \xi_1, \tau_1\}$ and $\{\lambda_3, \xi_3, \tau_3\}$) do not yield to better outcomes. However, if we finally attempt to estimate the parameters singularly, better outcomes are obtained. In particular, graphics below show how results for the third component of each parameter are close to what we expected.
In this case, due to the high computational costs, only $N = 100$ iterations are repeated. The first two graphics have been generated on the personal laptop; the last one has been produced on the cluster of the University of Sussex. The R programmes involved required, respectively, 30.09, 27.68 and 25.79 hours.

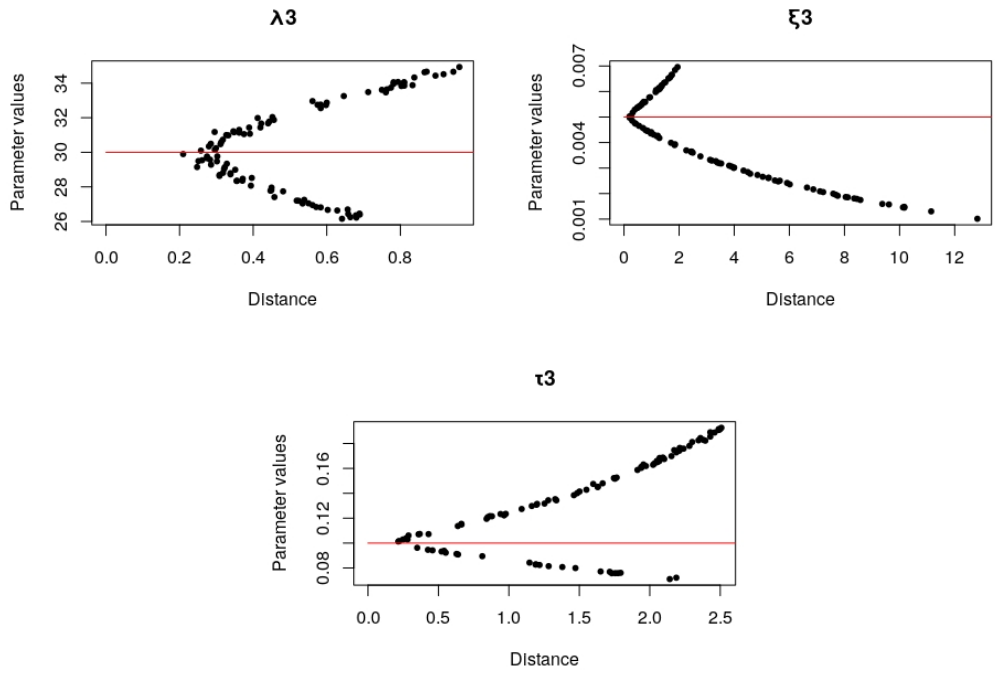


Figure 5.13: Parameters values, estimated singularly, plotted against the corresponding distance between summary statistics. The red line represents the true value.

These last results indicate that Approximate Bayesian Computation may perform well if applied to estimate the third components of λ, ξ and τ separately, maintaining fixed the remaining parameters (i.e., other components of λ, ξ and τ, the transition probability matrix and the initial distribution). According to choices previously made, the third status of the chain is linked to high rainfall intensity and so, determining this component would correspond to data-adapted simulation for heavy rain.

Thus, the next step is achieving Approximate Bayesian Computation to the real rainfall time series available. In line with the analysis made on *ad hoc* data, only the third components of the distribution parameters are estimated. A probability vector δ, a stochastic matrix Γ are generated randomly; then, in turn, the third component of each of the distribution parameters is chosen to be uniform distributed in a neighbourhood of the *ad hoc* parameters while the remaining elements are taken to be fixed. As before, we do not fix an *a priori* threshold as we do not have any hints of where it should be placed to obtain significant results. So, for each iteration, both distance and parameters values are saved; we finally plot the parameters values against the distance in order to verify their general behaviour. Eventually, if adequate

results are achieved, we will choose the parameter values corresponding to the lowest distance, in order to simulate the model.

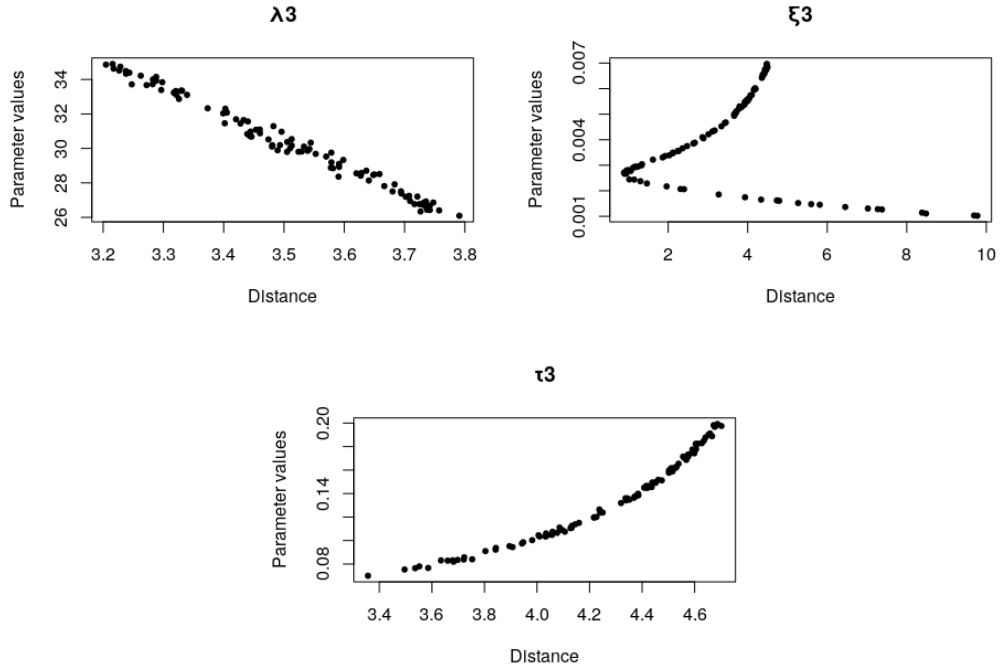


Figure 5.14: Parameters values, estimated singularly, plotted against the corresponding distance between summary statistics for real data.

These experiments have been run on three different cores on the cluster of the University of Sussex for, respectively, 24.71, 22.79 and 23.44 hours.
Above graphs are encouraging: points are not uniformly distributed, but they seem to have the peculiar trend we expected. The most interesting result concerns the third component of ξ since it lets to establish clearly which is the value that corresponds to the minimum distance.
For λ_3 and τ_3 further analysis have to be made: previous proofs are repeated for these two components choosing a different interval for the uniform distribution, so that to include (or at least, try to) the value corresponding to the lowest distance achieved.
The following graphics are obtained by choosing

- $\lambda_3 \sim U([50, 100])$;
- $\tau_3 \sim U([0.008, 0.1])$;

which finally give the expected results.

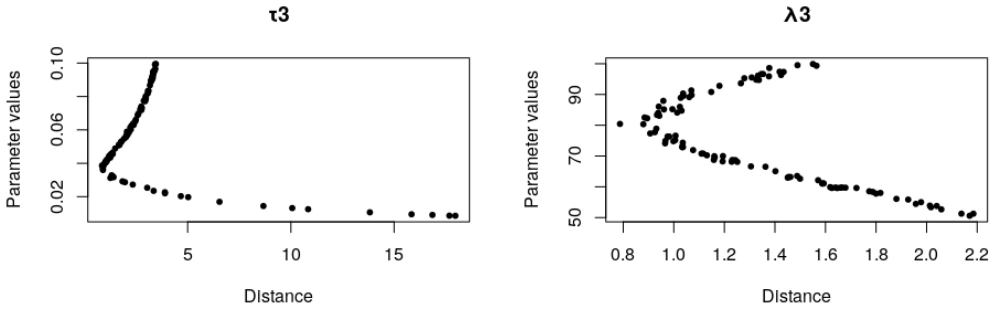


Figure 5.15: Parameters values, estimated singularly, plotted against the corresponding distance between summary statistics for real data.

We now want to simulate the rainfall model using the parameter values corresponding to the lowest distance achieved. These values are indicated in table 5.5 below.

Parameter	**Value**	**Min. Dist.**
λ_3	80.436	0.787
ξ_3	0.0027	0.887
τ_3	0.0386	0.849

Table 5.5: Parameter estimates achieved by ABC on real data.

Simulating the Hidden Markov model with all the third components fixed to be the ones of Table 5.5 do not yield to solid results: even if, a progress is made with respect to the data obtain with random parameters, the mean and the standard deviation of rainfall intensity are still far from the real ones. This is not surprising: first of all, most of the choices are still made (almost) randomly. Besides, each of the parameter components has been estimated given a fixed different value of other components; so, it is highly unlikely to have an adequate fit.
In practice, if we simulate the model setting each parameter singularly we obtain better results. As an example, graphics below show the same outcomes illustrated in previous sections when the Hidden Markov model is simulated with the value obtain for ξ_3 via Approximate Bayesian Computation (i.e., value of Table 5.5).

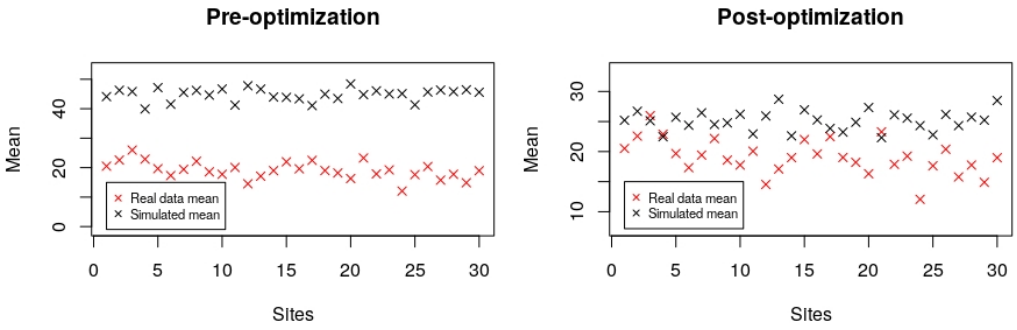


Figure 5.16: Mean rainfall intensity in each site for a period of 5 years.

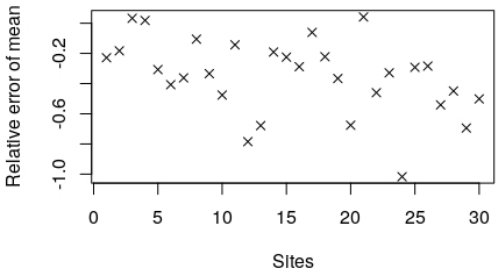


Figure 5.17: Relative error of the mean rainfall intensity for a period of 5 years.

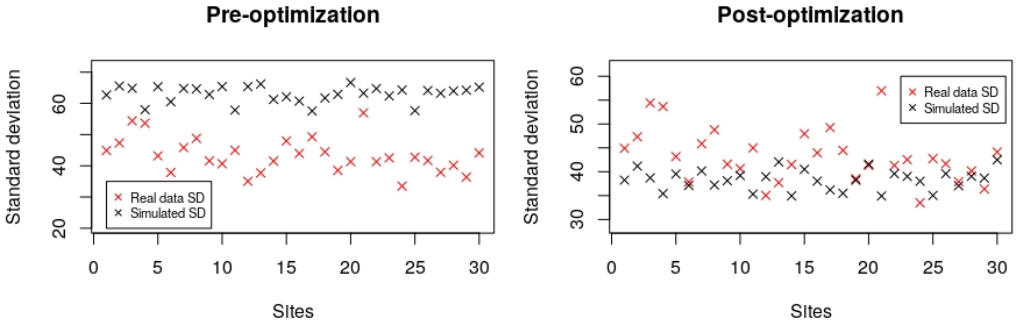


Figure 5.18: Standard deviation of rainfall intensity in each site for a period of 5 years.

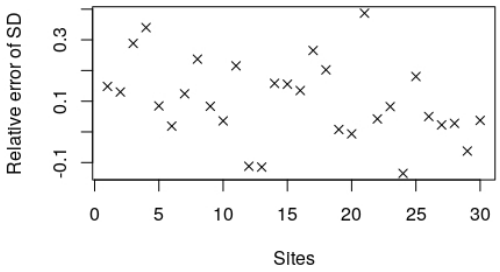


Figure 5.19: Relative error of the standard deviation of rainfall intensity for a period of 5 years.

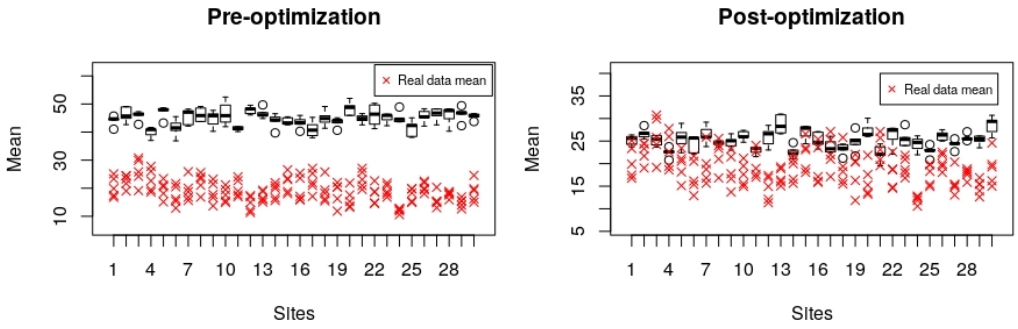


Figure 5.20: Box plot of annual daily mean rainfall for a period of 5 years for each site. Red crosses are real data.

These results are quite satisfactory and encouraging. Even if only one parameter is estimated from data, we obtain a good overall fit. This may open the way to a first development of this procedure. Each third component of the distribution parameters may be estimated singularly in a sequential way. That is, at first, one component, say τ_3, is estimated given the usual fixed values for the remaining parameters; then, another component, e.g., ξ_3, is estimated using the value previously found for τ_3; at the end, this step is repeated for λ_3, using the estimation for ξ_3, too. As this operations have to be done sequentially, the overall time needed to obtain an estimate for all the third components increases noticeably; however, it may give better estimates, that allow to have simulated data closer to the real ones.

Chapter 6

Conclusions and further developments

In this thesis we committed to analysing and developing a space-time model for rainfall, based on a double discs structure that may occur in the $\mathbb{R}^2$ plane. Out first aim was to generalize it by mean of Hidden Markov models. In doing so, different tasks was examined. At first, for what concerns model formulation, a simplification is made with respect to the original model, since only one disc system is considered. Simultaneously, according to Hidden Markov models' fashion, parameters of the distributions are meant to depend on a Markov chain not directly observed.
A remarkable issue we dealt with is the model parameter estimation in the fitting procedure. Two main different techniques were studied in order to solve problems derived from the absence, in the model formulated, of an explicit formula for the state-dependent probability density functions. The first approach considered attempted to estimate the density functions via a Monte Carlo simulation. Despite the high computational costs, promising results are obtained in trying to estimate the initial distribution of the chain and its transition probability matrix using numerical algorithms for maximum likelihood estimation. This has a non-trivial meaning: the initial distribution and the transition matrix probability of the Markov chain are essential for the model, so the hidden Markov structure is really capable of capturing and describing rainfall patterns. On the other hand, experiments produced poorer outcomes when numerical methods (i.e., the Nelder-Mead method) are used to estimate the parameters specific of this model. This results may open the way to further analysis. At first, the capability of numerical methods to estimate a lower number of parameters may be tested. Then, possibly, the model can be simplified in order to reduce the total number of parameters to be estimated.

The second strategy adopted to overcome the problem is the Approximate Bayesian method, which allowed to directly bypass the evaluation of the state-dependent density functions. A preliminary research on the method's performance revealed its capability in estimating a specific subset of the model parameters (i.e., the last component of the three distribution parameters). Thus, this Bayesian process is applied to real data; valid outcomes are achieved.

Even in this case, some improvements and further studies may be done. In order to improve the results and find an estimate of the remaining components of the distribution parameters, we may perform an iterative procedure. Once the values for the last components are found, they can be taken to be established and we can repeat the analogous operations to find an estimation for another component. At this point, this last step is also achieved for all the remaining components. At the end, data-adapted values for the parameters will be obtained. An important precaution in applying it would be to first test the strategy on *ad hoc* simulated data, as we did for the previous part, in order to have a warranty on the results reliability.

In the work presented herein, we set initially a fixed number of possible states for the hidden Markov chain ($m = 3$) and all the experiments are made according to this established value. An additional task that may worth accomplishing is repeating all the numerical proofs for different number of states, in order to compare various models obtained. The comparison may be made through standard model selection criteria, such as the Akaike Information Criterion or the Bayesian Information Criterion.

Bibliography

[1] Ailliot, P., Allard, D., Monbet, V., and Naveau, P. Stochastic weather generators: an overview of weather type models. *Journal de la Société Française de Statistique 156*, 1 (2015), 101–113.

[2] Cappé, O., Moulines, E., and Rydén, T. *Inference in hidden Markov models*. Springer Science & Business Media, 2006.

[3] Cowpertwait, P. A neyman-scott model with continuous distributions of storm types. *ANZIAM Journal 51* (2009), 97–108.

[4] Cowpertwait, P., Isham, V., and Onof, C. Point process models of rainfall: developments for fine-scale structure. *Proc. R. Soc. Lond. Ser. A Math. Phys. Eng. Sci. 463*, 2086 (2007), 2569–2587.

[5] Cowpertwait, P., Kilsby, C., and O'Connell, P. A space-time neyman-scott model of rainfall: Empirical analysis of extremes. *Water Resources Research 38*, 8 (2002), 6–1.

[6] Cowpertwait, P., Salinger, J., and Mullan, B. A spatial-temporal stochastic rainfall model for auckland city: Scenarios for current and future climates. *Journal of Hydrology (New Zealand)* (2009), 95–109.

[7] Cowpertwait, P. S. A generalized spatial-temporal model of rainfall based on a clustered point process. *Proceedings of the Royal Society of London. Series A: Mathematical and Physical Sciences 450*, 1938 (1995), 163–175.

[8] Cowpertwait, P. S. A poisson-cluster model of rainfall: some high-order moments and extreme values. *Proceedings of the Royal Society of London. Series A: Mathematical, Physical and Engineering Sciences 454*, 1971 (1998), 885–898.

[9] Cowpertwait, P. S. A spatial–temporal point process model of rainfall for the thames catchment, uk. *Journal of Hydrology 330*, 3-4 (2006), 586–595.

[10] Cowpertwait, P. S. A spatial-temporal point process model with a continuous distribution of storm types. *Water Resources Research 46*, 12 (2010).

[11] Cowpertwait, P. S. P. A generalized point process model for rainfall. *Proc. Roy. Soc. London Ser. A 447*, 1929 (1994), 23–37.

[12] Daley, D., and Vere-Jones, D. *An Introduction to the Theory of Point Processes Volume I: Elementary Theory and Methods*. Springer.

[13] Daley, D. J., and Vere-Jones, D. *An introduction to the theory of point processes: volume II: general theory and structure*. Springer Science & Business Media, 2007.

[14] Dean, T. A., Singh, S. S., Jasra, A., and Peters, G. W. Parameter estimation for hidden markov models with intractable likelihoods. *Scandinavian Journal of Statistics 41*, 4 (2014), 970–987.

[15] Dempster, A. P., Laird, N. M., and Rubin, D. B. Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society: Series B (Methodological) 39*, 1 (1977), 1–22.

[16] Forney, G. D. The viterbi algorithm. *Proceedings of the IEEE 61*, 3 (1973), 268–278.

[17] Lagarias, J. C., Reeds, J. A., Wright, M. H., and Wright, P. E. Convergence properties of the nelder–mead simplex method in low dimensions. *SIAM Journal on optimization 9*, 1 (1998), 112–147.

[18] Leroux, B. G. Maximum-likelihood estimation for hidden markov models. *Stochastic processes and their applications 40*, 1 (1992), 127–143.

[19] Little, R. J. A., and Rubin, D. B. *Statistical analysis with missing data*, second ed. Wiley Series in Probability and Statistics. Wiley-Interscience [John Wiley & Sons], Hoboken, NJ, 2002.

[20] Nelder, J. A., and Mead, R. A simplex method for function minimization. *The computer journal 7*, 4 (1965), 308–313.

[21] Sisson, S. A., Fan, Y., and Beaumont, M. *Handbook of approximate Bayesian computation*. Chapman and Hall/CRC, 2018.

[22] Vere-Jones, D. Some models and procedures for space-time point processes. *Environmental and Ecological Statistics 16*, 2 (2009), 173–195.

[23] Viterbi, A. Error bounds for convolutional codes and an asymptotically optimum decoding algorithm. *IEEE transactions on Information Theory 13*, 2 (1967), 260–269.

[24] Zucchini, W., MacDonald, I. L., and Langrock, R. *Hidden Markov models for time series: an introduction using R*. Chapman and Hall/CRC, 2017.