University of Sheffield

Modelling Air Pollution in Kampala

Nikolas Hadjisavvas

Supervisor: Michael Smith

A report submitted in fulfilment of the requirements
for the degree of MSc in Advanced Computer Science

in the

Department of Computer Science

May 11, 2022

Declaration

All sentences or passages quoted in this report from other people's work have been specifically acknowledged by clear cross-referencing to author, work and page(s). Any illustrations that are not the work of the author of this report have been used with the explicit permission of the originator and are specifically acknowledged. I understand that failure to do this amounts to plagiarism and will be considered grounds for failure in this project and the degree examination as a whole.

Name: Nikolas Hadjisavvas

Signature: NikolasHadjsavvas

Date:11/05/2022

Abstract

In the age of digital data, machine learning models which are driven by the availability of such resources, have the unique prospect of creating valuable solutions for major environmental problems, one of which is Air Pollution. The main objective of this project is to provide an implementation of a Gaussian Process regression model for predicting pollution levels in Kampala, Uganda. As with many other similar projects , generic pollution data such as particulate matter concentration are going to be used as the foundation of the model, but most importantly, in the sake of evolving previous work on the same field, separate modelling of the concentration of pollution spikes(very large concentrations of pollution) is going to be executed first, providing more useful information about pollution.Therefore, the results of this initial spike modelling can possibly incorporate new, beneficial knowledge into the main model, and perhaps making the final result more reliable.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Modelling air pollution in Kampala, is a project during which we want to build a machine learning(ML) model for predicting future air pollution levels expressed by the concentration of particulate matter(PM). PM is defined as the mixture of solid particles travelling through the air, including dust, dirt and smoke. The data that are going to be used for the implementation of the project are drawn from a network of low-cost sensors across Uganda, Kampala, our area of interest. The sensor network has been collecting data of air pollutant material in different key areas of Kampala through established monitoring stations, keeping track of the levels of PM2.5 and PM10(particulate matter of diameter less than 2.5 micrometers and less than 10 micrometers respectively). This collection of data will be our main and most important resource for studying the occurrence of high pollution levels(also called pollution spikes, if visualised with a plot), which we want to model separately. The results are going to be used for the implementation of the final most general model for predicting air pollution concentration [37].

1.1 Aims and Objectives

Modelling air pollution is definitely not a new task for computer scientists. Many different air pollution models have been developed up to this day. Even though all researchers and scientists try to differentiate their model from all the previous ones, many of the current methods seem to make use of emissions, meteorologic data and atmospheric characteristics(mainly concentration of particulate matter) in order to build statistical models [54, 42, 14].

As briefly mentioned in the introductory statement, in this project we try to differentiate from the work of other researchers by studying and further analysing spikes of pollution. Our main motivation is that pollution spiking(very high pollution occurring at certain times) is directly and greatly influencing general levels of pollution, therefore separately modelling these spikes can produce a set of useful information about their effect on the problem.

The project's first objective is the careful analysis of provided data and the extraction of useful insights that will assist in the implementation of the model. Analysis of data includes observations on the shape(i.e. the average of the measurements , the deviation from the mean, the positions where pollution levels seem to be higher or lower, skewness of the measurements towards higher or lower values).

Following the data analysis, we move on to possibly the most important stage of this project, which is separately modelling the rate at which pollution spikes occur. These are going to be modelled using a log Gaussian Cox process(LGCP) model. LGCPs are a class of effective and useful models for

understanding the rate at which random points enter a defined space area. In our case, this translates to modelling the rate at which pollution spikes/high measurements are observed/monitored by the sensors. By using an LGCP, we will be able to describe pollution spike occurrence using a simple and comprehensive statistical tool called a point process and then use this to model the intensity of spikes along space and time.

Using the initial dataset as well as the information received from the spikes model previously implemented, we then move on to extract suitable and meaningful features and construct the air pollution model by using Gaussian process regression(GPR). GPR models consider an infinite number of possible functions (generated by the GP's covariance matrix and mean function) which our previously observed data might follow. By using our training data, the model eventually adapts and settles down to one function, including some degree of uncertainty. With this, we will be able to draw predictions on the air pollution levels, given a specific time in the future as input. The GP's covariance matrix and mean functions are two components that need to be carefully selected as well, since with them, we are able to incorporate previous knowledge and assumptions about the data into the model.

Lastly , as required by all modelling projects, certain metrics should be used in order to evaluate the performance of our model's results.

1.2 Constraints

As with most projects involving the analysis and utilisation of datasets, it is always possible that expectations of deriving completely new useful insights from the data are not met, sometimes because specific analysis and introduction of new features might not fit right into the model.

A more specific to the project technical constraint that can affect the early steps of the project (specially the analysis of the data prior to the model construction) is the fact that the two sensors which monitor the same elements, sometimes produce conflicting measurements for the same time slot(i.e. sensor 1 records a spike where sensor 2 records a normal measurement). This will eventually lead to the need of a decision of when to consider a spike to be a legitimate measurement and when not to.

1.3 Overview of the Report

Moving on to the second chapter of this report, a variety of project related literature is going to be discussed, including information from past studies, research papers and scientific journals. The objective of this chapter is to correctly set the background of topics and mathematical/statistical tools that are related to the process of completing this project. In the third chapter of the report, the specific aims/objectives of this project are going to be discussed one by one, giving more details on each objective and what kind of knowledge is expected to be gained from each one, that will assist in the completion of the project. In chapter four, we are going to present the design for the implementation of the actual model. In the fifth chapter, we will present the implementation of the model as well as the results obtained after completing each objective set in the requirements chapter. The fifth chapter will also include sections regarding model validation, drawbacks of the implementation that lead to issues as well as future work that can be directly linked to the project.

Chapter 2

Literature Survey

Air pollution in urban areas, as a modern problem which greatly disturbs the well being of people in high population cities, requires thoroughly detailed solutions, tailored to the specifics of the problem in each area. Using data driven techniques, is one way of designing effective and accurate tools for tackling such problems. The theory and background knowledge required for these solutions can be fairly intense, thus in this section, I am carefully going through the different literature related to different aspects of the project one at a time, providing useful accompanying insights from studies in the field and possibly extending some concepts which might lack the attention ought to be brought to them.

2.1 Air pollution modelling in cities(How and why it's done.)

Modelling of a time series event/phenomenon(either natural or man made), is most of the times expected to aim to the understanding of how exactly something is generated, with the final step being the ability to predict the value of it for future time-frames. Air pollution modelling (in which we are interested), if executed right, may assist in

- Predicting/forecasting and analysing the distribution of pollution.
- Quantifying the impacts of air pollution
- Modelling the impacts of other /underlying factors on air pollution(such as pollution spikes, meteorological data, etc)[36]

Predicting/forecasting and analysing the distribution of pollution.

Predicting and forecasting pollution can prove to be useful in terms of determining the effectiveness of current mitigation policies and the management of a city's industries and population. Growing industrialisation in cities is leading to the need of constant air pollution monitoring. The study of pollution records and air quality indexes for the near future can justify the warnings given by researchers in order to expose air pollution as a major threat for both human and environmental health and stress the importance of minimising the pollution before it gets adverse[34].

Modelling the impacts of other /underlying factors on air pollution.

Air pollution is a multidimensional problem, meaning it can be affected by many different factors or interconnections of different circumstances. Work on modeling air pollution has many times shone a light to these factors and helped explain how each one of them can affect the problem on a considerable scale. An example of such piece of work is a land use regression model for estimating spatial variations in air pollution concentrations in high-rise cities with high population density. Therefore examining the potential modeling differences when the factors of city density and rise are put into the discussion[31].

Another example is the use of random forests to model regression relationships between concentrations of air pollutants (such as NO2, NOx and PM2.5) and meteorological conditions (such as wind speed, wind direction, temperature, air pressure and relative humidity) as well as traffic flow[25]. Pollution models taking traffic into account can help traffic managers to select the correct traffic management strategies[25].

Research papers studying the effects of using different choices of data (traffic vs meteorological) on the overall accuracy of the model by comparing the corresponding results are also present in the literature [30].

2.1.1 Modelling methods and which one suits better

As soon as the scope and aims of a project have been clearly defined, a team of scientists/researchers has to also consider the possible modelling methods, what each one offers and which suits better to the specific problem that needs solving. For the problem of air pollution, modelling can be either physical or mathematical based.

Physical models involve the reproduction/replication of an urban area and representation of the atmospheric flow in the wind tunnel[51]. No statistical or mathematical methods are used, and physical models become undesirable due to various technical and economical obstacles. Some of the technical obstacles being the need of abundant prior knowledge (initial states and pollution sources) which most of the times are unavailable and that physical models are unable to involve all the influence factors and apply to diverse scenarios[33].

Mathematical models are themselves divided into statistical or deterministic/dispersion models. Deterministic models are based on mathematical description (mathematical equations) of physical and chemical processes carried out in the atmosphere (mainly express laws of mass, momentum and energy). These models though, are repressed by the fact that no prior assumption can be integrated into them.

We, as computer scientists, are mainly interested in the statistical/data driven models for air pollution. Statistical models include regression, classification and re-sampling models. These models are based on analysis of previously collected data from monitoring air pollution measurements. The process of data analysis is of significant importance, due to the fact that it enables data scientists to extract useful information about the modelled event, which can later be included into the model.[22, 48, 36]

Furthermore, statistical solutions such as regression and classification can be implemented using a Bayesian approach. Using this offers remarkable advantages, two of them being the ability to import knowledge about the event we want to model into the prior, and get predictions from the corresponding posterior distribution[52].

All the above arguments regarding statistical/data driven models are exactly the reason why these kind of models are the most useful to implement.

2.1.2 Data collection and external factors.

Any data driven solution requires the availability of a sufficient amount of data for scientists to be able to extract meaning and incorporate knowledge into a model. Air pollution modelling is no exception to this fact. Thus, every air pollution modelling project starts with the collection of data, which are most of the times acquired through monitoring stations, where air pollution sensors are installed and measure pollution concentration levels and other useful underlying data. Similar work carried out in the past have established monitoring stations across the area of interest, since pollution can also be considered as a measure influenced by location. Apart from this, prior projects have also designed their data calculations, taking into account temporal factors such as year, season, month, etc. Meteorological data have also been considered, since previous research has also shown that meteorological conditions have a significant effect on pollution concentrations[22][53]. For our project, datasets include the measurements of PM travelling through the atmosphere. PM can be divided into categories according to their diameter size. The two most common in research are PM2.5 and PM2.5
.

2.2 Health effects of PM

PM2.5 and PM10 concentrations are themselves specific metrics which can express the levels of air pollution in a given area. PM2.5 stands for particulate matter of diameter less than 2.5mm travelling in the air, and PM10 for the range of particulate matter with diameter between 2.5mm and 10mm. Both PM2.5 and PM10 include particles that are small enough to be inhaled and penetrate the respiratory system, eventually leading to several health effects by both short term and long term exposure. These effects include respiratory and cardiovascular morbidity[35], as well as mortality from respiratory diseases and lung cancer[20]. Effects on children also have significant impact, as exposure to PM leads to lung development disruption[18]. Although PM generally leads to a certain domain of health impacts, it makes sense to ask how the effects can defer, depending on the duration of the exposure to high levels of PM concentration. This leads us to have a closer look at the topic of short term exposure to pollution spikes and long term exposure to ambient pollution. Especially for the nature of our project, as will be further discussed in the next section of this chapter, pollution spikes will play a major role in our modelling process, thus knowing the relevance between short term existence of pollution spikes and long term existence of lower amplitude measurements can give more meaning to our conclusions.

2.3 Short time spikes exposure vs long term low exposure

2.3.1 Exposure to long term low level pollution and its consequences

Studies have shown that exposure to ambient air pollution is directly associated with natural and cause specific mortality[49]. A further study has also added to previous knowledge, stating that even at levels below the WHO guideline values, the risks were maintained at the same level, by once again showing that even at these levels, when exposed for a long period of time, people could face problems regarding cardiovascular and respiratory disease[49].

2.3.2 Exposure to short term high level pollution and its consequences

Short term exposure to high levels of pollution, as stated above, can lead to similar magnitude of health effects. In this case, a study conducted by several researchers from Edinburgh university[47][38], has shown that both PM2.5 and PM10 in larger concentrations were positively associated with hospital admissions due to stroke or even death from stroke. As indicated by a study released by ATS (American Thoracic Society) and funded by Health And natural sciences organisations in China, other health effects such as asthma deaths are believed to be associated with short term spikes in air pollution exposure as well, but as research for this specific topic remains incomplete, conclusions are less clear.

Using the above conclusions retrieved from relevant literature on the topic of high and low pollution exposure (including PM), we can confirm that in both cases, populations are in imminent danger of several cardiovascular and respiratory related health problems.

2.4 Low cost sensors

2.4.1 What they offer?

The usage of low cost sensors for monitoring air pollution offers more advantages and functionality than one would initially expect. The use of low cost sensors enables the establishment of more monitoring stations throughout an area, following to the availability of richer variations of data. Abundance of data eventually lead to greater understanding of events and greater knowledge incorporation into the model[6]. The above statement though, manages to collide with the fact that measurements coming from low cost sensors are often of more questionable quality[12], meaning that cheaper hardware might account for more false measuring or malfunctions.This leads to noisy data, which can eventually require time and effort from the data scientist's end to cope with (and not introduce errors into the model).

On the other hand, relatively expensive sensors can provide communications and the integration of meteorological records in the measurements. Thus giving more useful context to the air pollution recordings[6].

2.5 Cox processes for modelling spatio-temporal events

As mentioned in the introduction of the report, the presence of higher pollution measurements (also called spikes) that can be found in our dataset, can be of significant importance when developing a model for air pollution, meaning that examining the occurrence of these spikes alone, can probably provide information on how and in what extend they influence pollution levels. From the modeling air pollution in cities section, we have seen that researchers have been using a variety of data to gain information on air pollution (from PM2.5 to meteorological and traffic data), while pollution spikes and separated analysis on them has not been discussed or acknowledged as a further task or improvement. Instead, pollution spikes are mostly considered as outliers resulting from sensor malfunctions and are commonly eliminated from datasets.

One of the techniques to model pollution spikes and their rate of occurrence are log Gaussian Cox processes for modelling spatio-temporal data (in our case, the pollution spikes)[16]. By using this technique, we can treat the pollution spikes as random points in space and time, whose occurrence is controlled by certain intensity values, which can be modelled using Gaussian Processes[9]. Let us first give a brief definition of a Cox process, one of the fundamental components of an LGCP. A Cox process can be defined as a collection of random points in a given space/time domain (similar to an homogeneous Poisson process, which is just a distribution of points whose intensity is constant

across space, therefore the expected number of points is the same at all locations), but unlike Poisson processes, the intensity of the occurrence of the points in a Cox process is stochastic and varies over space and time[9] resulting to an inhomogeneous point process (where the expected number of point at one location is not the same as at another).

For simple inhomogeneous Poisson processes:

$$\lambda(x) : \text{intensity function}$$
$$\lambda(x) > 0$$

For Cox processes:

$$\text{Point process characterised by } \lambda(x)$$
$$\lambda(x) \text{ is stochastic}$$
$$\lambda(x) \sim GaussianProcess(O, \Sigma)$$

Therefore, in simpler words, a Cox process is just a point process where the intensity is not known, is not stationary and is controlled by a stochastic function. This also means that the point process is inhomogeneous, i.e. the concentration of points is not the same at every location of the area, but it changes as we move from one location (or time) to another. Thus, we have heterogeneity in the distribution of the points[16, 9, 24].

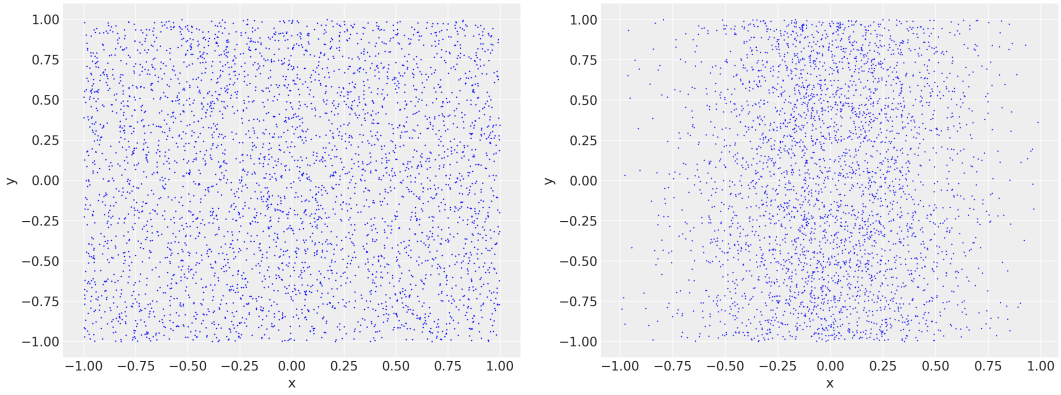


Figure 2.1: Left:A homogeneous Poisson point process. Right:An inhomogeneous Cox process with λ controlled by a stochastic function.

Cox processes are also referred to as "doubly stochastic", since they are defined as a inhomogeneous-random Poisson processes with the intensity value also being random[29].

Now that Cox processes are defined, and the significance of the point process intensity lambda is established, we can trivially define an LGCP as a Cox process whose lambda is a function that can be modelled using a Gaussian process (GP) . It's important to note that, since the intensity(λ) of a Cox process is restricted to be positive, the GP does not directly model λ, but instead models the log intensity of the Cox process $\log(\lambda)$.

$$\log S(x) \sim GaussianProcess$$

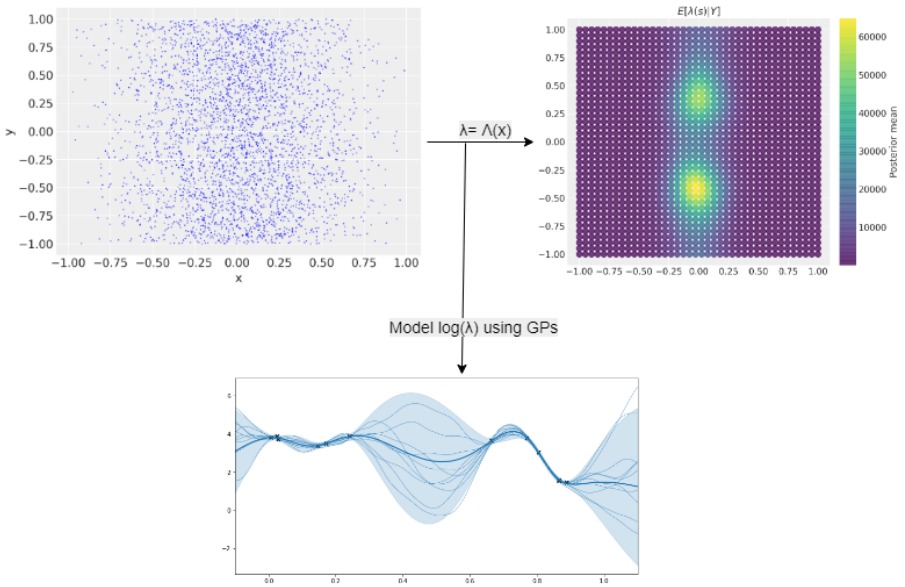


Figure 2.2: Visual representation of the process followed by an LGCP model.

2.5.1 Specific applications of LGCPs

It's common for Cox Processes to be used for modelling patterns which are influenced by meteorological settings (e.g. modelling wildfire occurrences[44]), such as pollution spikes, which we are interested in modelling in this project. Cox processes can also be used for modelling non meteorological influenced point patterns and events. This study by J.Alcala Et al. is a clear example of that[7].

2.6 Gaussian processes

Gaussian Processes (GP) , are our preferred mathematical tool for building a regression model to predict air pollution levels and were also used for previous similar projects. With Gaussian processes, we can develop Bayesian models for predicting values of interest, given past data/records of the value we want to build the model for.

In order to understand GPs, we first consider nonlinear regression models, and see how GPs differ from these. Gaussian processes are different from standard non-linear regression. The latter builds the model by fitting a function through the training data, thus our mission is to determine the values of the parameters which define our function/line that passes through the data[26]. We do that by first placing prior distributions over these parameters, we then train the model using our training data/observations which lead to the definition of a posterior, which we can sample in order to get potential values for our parameters.

On the other hand, when using Gaussian Processes, we are not trying to determine the one and only function/line that passes expresses our data, instead we consider all the possible functions that might describe our data by placing a prior over functions directly, instead of a prior over the parameters which define a function. Therefore, we don't really care about finding the best values of our function parameters, we care about finding the best function among many possible ones[41, 10] (Please note that GPs are not the only method which work like that, Bayesian linear regression models also work in a similar manner, here we just consider the differences between standard nonlinear regression with

GPs since for this project we are mainly interested in Gaussian processes).

In order to get predictions, we condition the GP given the training data and a new input for which we need a prediction for[41]. The result of conditioning of the GP gives to us another Gaussian which expresses the prediction we want to get, in other words it returns a whole distribution of what the possible value of our prediction might be. Therefore, we are returned with a prediction accompanied by confidence intervals, thus we are able to include uncertainty levels in the final output.

Gaussian processes for regression are used in a variety of applications. Some of these applications include the modelling of motion trajectories and visual analysis of moving object, gait optimisation for robotic motion as well as biological and environmental applications[27, 15].

2.6.1 Mean and Covariance functions and the GP prior.

A GP, similarly to a Gaussian Distribution which is defined by the mean and standard deviation, is implemented by defining its mean and covariance functions[41].

$$\begin{aligned} f(x) &\sim GP(m(x), k(x, x')) \\ m(x) &- meanfunction \\ k(x, x') &- covariancefunction \end{aligned} \tag{2.1}$$

The definition of covariance function (or kernel) is rightfully one of the most important components of a GP, since this is where we introduce prior knowledge about the data into the model, by picking the right choice of a kernel which itself defines the correlations between each variable in the data and forms the covariance matrix K (and therefore decides on the shape of the functions in our GP prior)[41, 19].

$$\mathrm{K}(x_1, x_2) = \begin{pmatrix} k(x_1, x_1) & k(x_1, x_2) & ... & k(x_1, x_n) \\ k(x_2, x_1) & k(x_2, x_2) & ... & k(x_2, x_n) \\ . & . & . & . \\ . & . & . & . \\ . & . & . & . \\ k(x_n, x_1) & k(x_n, x_2) & ... & k(x_n, x_n) \end{pmatrix}$$

Figure 2.3: Formation of a covariance matrix which is populated by the covariances of all the variables with each other. All variances are placed in the diagonal.

The covariance functions has its own hyperparameters, which we also want to pick carefully (either by choosing a fixed value or placing a prior over them). For most of the kernels these hyperparameters are the length scale which defines the smoothness/wiggliness of the generated functions in the prior and the vertical scale sigma, which defines the vertical span of the functions. Different choices of hyperparameters can yield different shapes of functions[41, 19].

Having decided on the mean and covariance function of the GP we then proceed on placing a GP prior over the function which we want to model, thus assuming that it is drawn from the GP we defined. Depending on the choice of kernel, the GP generates functions with certain characteristics (some choices of kernels generate smoother functions,where others generate wigglier ones, the choice of which kernel to use depends on previous knowledge about the nature of the event we want to model)[17].

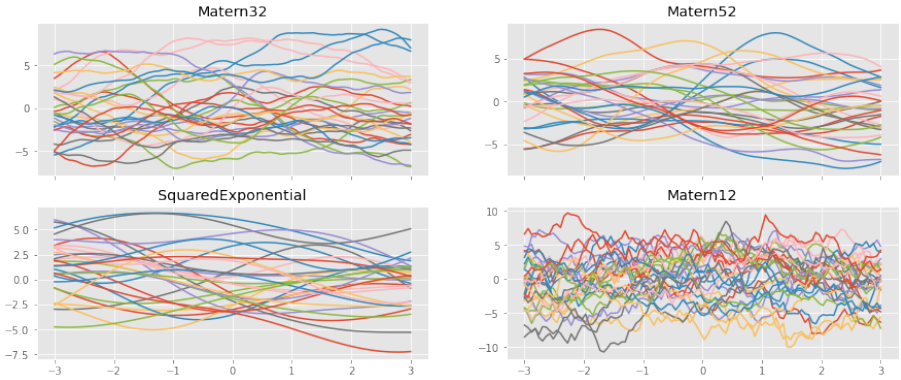


Figure 2.4: Generated prior using the Matern52 kernel with different choices of length scales.

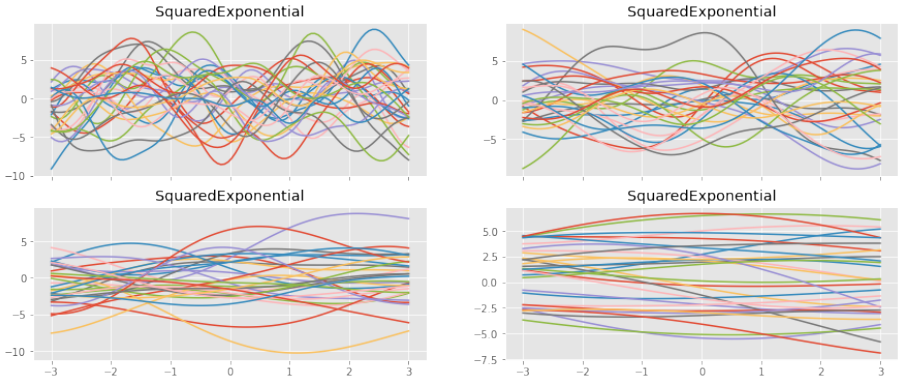


Figure 2.5: Generated prior using the squared exponential kernel with different choices of length scales.

2.6.2 GP posterior

As of now, all we have is a prior of functions that might express our data, but remember that our fundamental objective is to incorporate already acquired knowledge (training data) about the underlying function in our model. Let us consider the simplest case of a GP, where we have noise free observations, $\{(x_i, f_i)|i = 1..n\}$ - for X input space training points we know the corresponding exact output values f. We also consider X*, f* to be our test points and test output values respectively. Since working with GP is similar to working with Gaussian distributions, our first step to get to a prediction is the definition of the joint distribution of training values f and test values f* as

$$\begin{bmatrix} f \\ f_* \end{bmatrix} \sim N\left(0, \begin{bmatrix} K(X,X) & K(X,X_*) \\ K(X_*,X) & K(X_*,X_*) \end{bmatrix} \right) \tag{2.2}$$

Where K(X,X*) is the evaluated covariance between training and testing data (similar with K(X*,X)) and K(X,X) covariance between training data[41]. The posterior of a GP is then defined as the restricted version of this joint distribution, where we only consider functions that pass through our observations. This can be computed by conditioning the joint distribution on the observed data.

The conditional distribution is given by

$$f_*|X_*, X, f \sim N(K(X_*, X)K(X, X)^{-1}f, K(X_*, X_*) - K(X_*, X)K(X, X)^{-1}K(X, X_*) \tag{2.3}$$

Essentially this gives us the function values f* (predictions) given X, f training data and training output and X* test data for which we want a prediction. A visualisation of a GP posterior is included in figure 2.5. The prediction is simply the function value at point X*, this will return the value of the prediction as a whole Gaussian distribution expressing the possible values of the prediction, hence why we say that GPs incorporate uncertainty into the predictions.

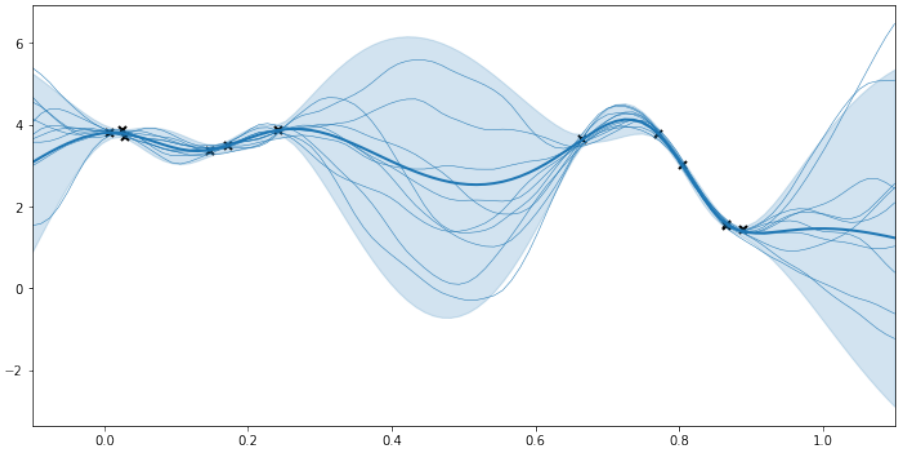


Figure 2.6: Samples from a GP posterior. Observations are visualised as black x's. We can see that the modelled function (in dark blue) has adapted to the observations. The confidence intervals are included in the shaded blue areas around the function.

In the more realistic case where we don't have access to noise free observations/function values but to noisy versions of it, so that our underlying function becomes $y = f(x) + \varepsilon$, where ε is additive Gaussian noise with variance σ^2, our prior is redefined to [41]

$$cov(y) = K(X, X) + \sigma_n^2 I \tag{2.4}$$

equation 2.2 is redefined as

$$\begin{bmatrix} f \\ f_* \end{bmatrix} \sim N\left(0, \begin{bmatrix} K(X, X) + \sigma_n^2 I & K(X, X_*) \\ K(X_*, X) & K(X_*, X_*) \end{bmatrix} \right) \tag{2.5}$$

and the conditional distribution from 2.3 is revised to

$$\begin{aligned} f_*|X_*, X, f &\sim N(\overline{f_*}, cov(f_*)) \\ &\text{where} \\ \overline{f_*} &= K(X_*, X)[K(X, X) + \sigma_n^2 I]^{-1} f \\ cov(f_*) &= K(X_*, X_*) - K(X_*, X)[K(X, X) + \sigma_n^2 I]^{-1} K(X, X_*) \end{aligned} \tag{2.6}$$

2.6.3 Approximation algorithms for GPs

The need to develop scalable Gaussian processes is certainly an increasingly important task which has attracted research from many experts. Gaussian process regression can become significantly expensive mainly because of the covariance matrix inversion in the conditional distribution equation 2.62.3, which has complexity $O(n^3)$. Rasmussen and Williams have given a range of discussions on GP approximation methods in the book Gaussian processes for machine learning, chapter 8[41]. These methods include reduced rank approximations to the covariance matrix, greedy approximations and also approximations for both regression and classification problems for fixed hyperparameters. Quinonero-Candela and Rasmussen have taken things forward by introducing other approximations such as the deterministic training conditional (DTC), subset of regressors (SoR) , partially independent training conditional (PITC) and fully independent training conditional (FITC) [39]. The FITC and PITC approximation methods achieve a complexity of $O(nm^2)$ by not forming the full covariance matrix over the whole training dataset (n training inputs). Instead, these methods rely on inducing points (m < n, m being the number of the inducing points) which are placed throughout the domain[2, 41]. These inducing points can act like a compression of the real data, meaning that by using them we can keep the structure and information included in the full dataset but this time with a smaller amount of points, therefore the covariance matrix won't have to form through the original dataset (n inputs) but from a smaller version of it.

2.7 Preprocessing of air pollution data

Data driven modelling, primarily depends on past observed data. Thus, a standard step for data scientists to properly prepare datasets is to go through several suitable preprocessing procedures in order to achieve missing data imputation (for more complete and accurate data) [11, 32, 50], identification and removal of outlier observations, data transformation such as the Yeo-Johnson power transformation[55] (normalisation and standardisation for more uniform data and less variability) and data smoothing(for eliminating noise). [28].

It is evident that different kinds of datasets include different patterns and shapes in their measurements, therefore the preprocessing stage is different according to the nature of the data we are dealing with. For air pollution modelling problems, where we have strong variations of data due seasonal changes, it can sometimes be difficult to indicate the trends followed by the data, hence specific smoothing functions are commonly applied to air quality data (such as moving average or Savitzky-Golay[13] filters) [28].

Another common obstacle faced with air quality data is missing values, mainly due to malfunctions of the monitoring devices and stations (especially when dealing with low cost sensors) . With the aid of different algorithms (i.e. removing or replacing missing values), the problem of missing data can be solved, but results can be accompanied with loss of data information which can affect training, especially when data are already of limited size[28].

Chapter 3

Requirements and Analysis

The aims and objectives of the project were briefly defined in the introduction of this report. In this chapter I am going to expand on the mentioned objectives one by one. The goal of modelling time series events with ML, has always been a task consisted of several stages, with each stage having its own unique importance for the derivation of the final result. Having this in mind, each one of these stages can be classified as an objective of this project, with some additional specific aims also included.

3.1 Project Requirements

3.1.1 Data acquisition

The first objective of the project, could not be anything other than the acquisition of data, as well as their careful study and analysis in order to properly understand them, extract as much meaning out of them and process them in order to use them for the construction of our models later on.

For this project, air pollution records are the main form of data upon which the model will be constructed. The project's area of interest, Kampala, has a number of installed low-cost sensors (managed by a local company called AirQo) monitoring air pollution levels in several sites across the city. By accessing the network's cloud data reserves, we can acquire chunks of time based, raw air pollution measurements from all the monitoring stations so that we can later use them by advancing to the next stage/objective of the project which is the analysis of the data.

3.1.2 Data analysis

It is vital that one of the projects early objectives should be to get as much intuition on the dataset as possible. To get that, we should first start from the careful and precise analysis of the data. This includes the visualisation of the datasets in order to identify any possible trends they follow and identification of any forms of clustering (in our case, clusters of high pollution values) . Furthermore since the pollution monitoring sites record measurements from 2 sensors, another objective is the visualisation and comparison of measurements from both of them. Studying the potential differences in measurements from both of the sensors might give away some errors due to improper function. Such errors have to be taken care of, if they appear often enough.

Specifically for our project, another objective is to identify and further analyse the presence of pollution spikes (abnormally large pollution measurements) . Since we want to separately model the spikes, it is important that during the analysis stage, we get a clear definition of what they are (what

is defined as a spike and what is a proper threshold to use in order to classify measurements) , how they look like (how large are these measurements and whether they appear in same or different time frames) , how often they appear in the given time frame and the variations between them (whether some of them are just above the threshold and some others way higher).

3.1.3 Data pre-processing

As mentioned before, it is important to always remember that data driven solutions require sufficient and meaningful preprocessing of the raw data that are going to be used. The final result of the model is always reflective of the quality of our processed data.

Therefore, the right preprocessing methods/techniques to apply to the data is another key objective of to the project. The stage of preprocessing will include the smoothing of the data, in order to eliminate excessive variations and get a clearer version of the features we are interested in, such as getting the individual measurements considered to be pollution spikes for further use. Furthermore, transformation of the data depending on their normality degree might be necessary in order to acquire normalised data, which are easier to deal with.

3.1.4 Modelling the spikes

The next major objective of the project, which itself is divided into multiple sub-objectives, is the actual utilisation of the acquired measurements representing the pollution spikes and work with them in order to build an individual model for predicting the rate at which these spikes occur (please note that this is different from the model for predicting the actual air pollution values). The significance of this stage is great, since, separately modelling the high pollution occurrences is what differentiates this project from the majority of past air pollution modelling designs, where modelling is mostly done in general on all measurements.

The choice of statistical process which is going to be used for modelling the spikes, is the log Gaussian Cox process(LGCP), since pollution spikes can be considered as individual points entering a mathematical space at specific points of time. We can therefore express this event using a point process, whose rate/intensity (the rate at which points occur/appear) varies over space and time[40].

With Cox processes, we will be able to model the rate/intensity at which the spikes enter a space by using the method of Gaussian process regression. We will be able to build this model by using the past pollution spike points (which we will have already acquired from the first stage of the project) and their rates/intensities. This means that when the modelling of this rate/intensity is complete, we will be enabled to get predictions about the value of this rate for future points in time, thus having the knowledge of what rate will pollution spikes appear in the future. It's important that we keep in mind that this separate model using GPs, will require the correct choice of its own components (mean function and covariance matrix of the GP) and the fine tuning of its components' hyper-parameters (such as length scale parameter for the covariance matrix). Hence, this is another sub-task which is going to be given the proper amount of attention.

As soon as individual modelling of the pollution spikes is completed, we will already have an important, brand new feature, which we can later use in order to build the final model which will eventually carry out the actual modelling of air pollution levels in the city of Kampala. Therefore, we need to keep in mind that the modelling of the spikes alone is not the end of the overall project, but an important tool which we can later utilise in order to execute better and more precise inference for predicting air pollution values (in terms of particulate matter).

3.1.5 Building the air pollution model

Tracking back to the first objective of the project, we can see that we still have the initial air pollution data available. These are important and relevant and will still be used for creating the model for the air pollution levels prediction.

By completing the second objective of the project, we should be able to use the pollution spikes model to get predictions about the rate at which spikes occur. Having this information can be useful, since a correlation is expected to exist between the presence of spikes and the general levels of pollution. Hence, by knowing the rate at which these spikes appear we can eventually input this information as a new dimension in the final model.

The method that is going to be used for modelling the air pollution is a Gaussian process regression (same method we will use for the previous step for modelling the occurrence rate lambda of the point process which describes the spikes) . Using GP regression, we first consider an infinite amount of functions which potentially describe the event we want to model. Initially, when no training data have been observed, uncertainty levels are very high, therefore predictions cannot be made. By including training data into the model, the candidate functions are reduced to only the ones which pass through our training data, with uncertainty levels decreasing. As soon as enough training data have been introduced in out model, we can then use the conditioning property of Gaussians to get a prediction about a specific point in time (including its confidence levels). Please refer to the literature analysis section for a more informative explanation of Gaussian process basics.

Note that as briefly mentioned in the second objective details, building a GP model requires choosing from a variety of covariance functions and the definition of a mean function. The choice of covariance function is of special importance, since by selecting the right one, we can incorporate previous knowledge and assumptions about the characteristics of the function that the modelled event follows. Hence, this will be another important sub-objective of this stage of the project.

3.1.6 Testing

Model validation

As defined in the "Modelling the spikes" section, one of the main objectives of this project is to model the intensity function of the pollution spikes. An ML model could be put to many forms of testing in order to evaluate its performance and the underlying results. A simple yet useful approach to test our final result is to check the performance of our model on modelling already known functions, so that we can later compare true and modelled function in order to assess the model's performance.

Another testing approach could be to drop observations in specific areas of the spikes point process and proceeding to develop the model without them. Later we can check how close is the modeled intensity (on the area from which observations were dropped) from the true intensity. Evaluating the mean square error over a number of different testing rounds can give us an error metric which can be used to assess the performance of the model.

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(Y_i - \widehat{Y}_i)$$

Figure 3.1: Mean squared error can be calculated as the average of the squares of the errors of each tested value. n is the number of errors Y_i and $\widehat{Y}_i$ are the true and predicted values.

3.1.7 Overview of steps

Now that all steps have been separately defined, it might be worth to put them together into a list of steps which describes the process to be followed from the first step down to the last one.

In order of execution, the list of steps is defined as follows:

1. Data acquisition (contact management authority to provide the datasets or access them through an API).
2. Data analysis: Carefully visualise and analyse the data, derive conclusions on the structure and overall behaviour (mainly the pollution spikes).
3. Data preprocessing: Execute any necessary transforms or smoothing functions to the data, design and apply a function which classifies and extracts spikes.
4. Model the spikes: Design, develop and validate an LGCP model for modeling the intensity of the spikes over space and time.
5. Proceed to build a Gaussian process regression model for air pollution having the information from the previous spikes model. Validation of the model included.

Chapter 4

Design

4.1 Data analysis and Preprocessing

As specified in the requirements and analysis chapter, a proper amount of analysis processes have to be followed in order to gain some valuable context regarding the structure of the datasets and the nature of the pollution spikes therein. Initially it would be appropriate to visualise some of the provided datasets (consisted of PM2.5 measurements) on a temporal axis, in order to examine the general behaviour of all the measurements in the available timeframe. Information about general behaviour can be the observation of timeframes with larger amounts of high pollution measurements, an initial note on the smoothness of the measurements, whether neighboring measurements are similar to each other, or whether they are they able to change rapidly within a very small time frame. These are some of the questions that can be answered after the initial visualisation.

Another simple but important task in the analysis of the the dataset will be the calculation of some basic statistic metrics. These metrics are the mean, standard deviation, maximum, minimum and skewness of the data.With these measures in hand, we will be able to measure the center and the spread of the data in terms of how close they are to the average. The skewness of the data is another useful metric, since it gives information on whether the data are concentrated towards a specific domain of values.

Visualising the whole dataset on a single axis might be able to provide us with some high level intuition about the data, but it's important that we are able to have a closer look in order to examine any possible behaviour or pattern that would not be visible when visualising large amounts of measurements at once. Increasing the time resolution of our plots while visualising small chunks of the dataset can help with that. We can also further increase the temporal resolution for visualising individual potential spikes in order to better examine their structure (shape,width,exact magnitude).

As per the requirements analysis, raw datasets should be correctly and sufficiently preprocessed in order to make them suitable for modelling in a later stage. Preprocessing of the data can deal with problems such as the extreme skewness towards lower or higher measurements, the elimination of false measurements or the imputation of missing data. These preprocessing steps are possible options for our data, since we were initially provided with the raw and unfiltered version of it. In the preprocessing stage, we will also implement and apply a smoothing function with the intention of eliminating intense "wiggliness" of the measurements. This will be done by first smoothing out the data and then subtracting this smoothed version of the data from the raw measurements. After this smoothing process, larger measurements will stand out even more against normal ones.

4.2 Spike extraction algorithm

Following the analysis and preprocessing stages, the project's next step towards fully processing the data and using it for the spikes model later is the implementation and utilisation of a spike extraction algorithm. The spike extraction algorithm is the function which will take the pre-processed pollution data and will identify and return the measurements which will be classified as pollution spikes.

Pollution spikes among the pollution measurements can also be regarded as statistical outliers. As discussed in chapter 2, outliers in a dataset can be any measurement which lies extremely far from any other normal measurement. This is very similar to the notion of pollution spikes, therefore the same methods which are used for outlier detection can also be used and adapted to form a spike detection algorithm.

Our design for the spike detection algorithm is purely statistical, and classifies measurements as pollution spikes according to how many standard deviations away from the mean a specific measurement is located. Two statistic metrics which can help us to detect spikes depending on the deviation from the mean are the standard Z-Score and modified Z-Score methods. The Z-Score can be computed for each of the measurements and can help with the identification of spikes as the measurements which statistically differ from the norm the most and later classify them as spikes. The Z-Score is proportional to the number of standard deviations the measurement is away from the mean.

$$Z = \frac{\chi - \mu}{\sigma} \tag{4.1}$$

The method we will utilise for our spike detection algorithm is a slightly updated version of the Z-Score, called Modified Z-Score. For this method, we update the denominator of the formula from just σ to MAD (Median Absolute Deviation).

$$Z = \frac{\chi - \mu}{MAD} \tag{4.2}$$

$$MAD = Median(|X_i - \tilde{X}|, \text{ for all i in dataset}) \tag{4.3}$$

MAD is a robust measure of the spread of our data just like standard deviation and variance are. The reason we prefer MAD is due to the fact that it's not heavily affected by extremely high or low values in the dataset, unlike variance and standard deviation which are mostly used for datasets that belong to a normal distribution.[45]

The spike extraction algorithm itself will then utilise the Modified Z-Score metric by taking in the whole dataset as a parameter, iterating over the measurements each time calculating the Modified Z-Score and classifying them as spikes or non-spikes depending on whether a threshold is breached.

$$\text{function extractSpikes(Dataset)} \rightarrow [spikes] \tag{4.4}$$

4.3 The LGCP spikes model

After the analysis, preprocessing and the extraction of the spike measurements from the data, comes the step of using these extracted measurements to build an LGCP for modelling the intensity of the spikes. As discussed in the Survey Analysis chapter, an LGCP model is consisted of two main components. The first one is the definition of a Cox process using our observations (the spikes). This Cox Process is essentially a Point Process which is characterised by an intensity function λ which

controls the occurrence of spikes in some given space and time domains. This stochastic function is what we will try to model using Gaussian Process Regression(the second main component of LGCPs).

The definition of a point process for representing spikes in space and time can be done in many different ways, depending on our choice of dimensions.The simplest spikes point process can just have two-dimensions; one for time (a temporal axis) and one for location. This is the one which is going to be used in this project since it's preferable to keep the model simple at first. More dimensions can certainly be added to the model as soon as this simpler version is completed.

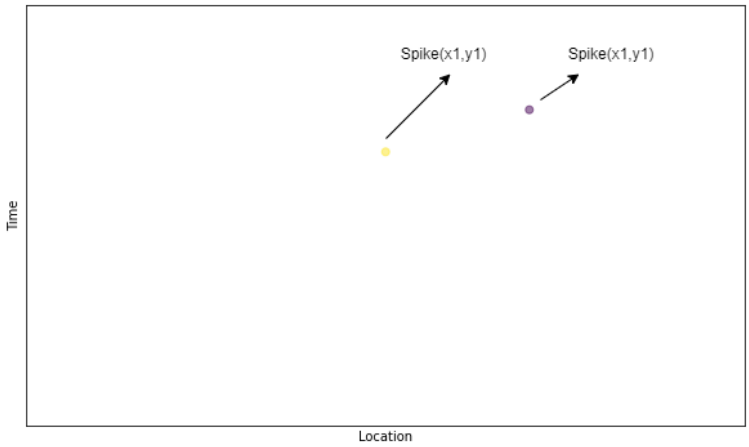


Figure 4.1: A simple pictorial definition of the format of the spike measurements and their use to define a point process.

As seen above, points representing spikes will be of the form spike=$(timeOfOccurrence, location)$. Time of occurrence is a date or timestamp measurement which will be directly retrieved from the dataset; as every pollution measurement in the datasets is always accompanied with a timestamp representing the time of creation of the record. The location of the spike can take several values. Alongside the datasets which contain the measurements of pollution for many different sites in Kampala, we were also provided with metadata for each monitoring site. These data include information on the latitude, longitude, distance from city centre, distance from nearest road, etc. of each site. Therefore, the location component of a spike tuple can be any of those information. The ones most suitable to our problem would most probably be distance measures such as distance from city centre or distance from nearest road. These can be regarded as very informative features since roads and the city centre are places with high vehicle congestion, something that can highly affect pollution. This time, we will use distance from city centre as the location measure of choice.

Eventually, the dimensions of our 2-D point process will come down to the time dimension represented by date values and the location dimension represented by the distance from city centre of the monitoring site corresponding to each spike measurement.

4.3.1 The python implementation

The definition of the point process representing the spikes will then lead us to another vital stage of the spikes model, which is the actual implementation of the LGCP. Recall from the survey analysis chapter that there several available Python libraries for implementing Gaussian processes (and eventually other models involving GPs such as LGCPs). The choice of tools/library used needs to be based on several important aspects which can affect the course of the project. Specifically, we ideally need

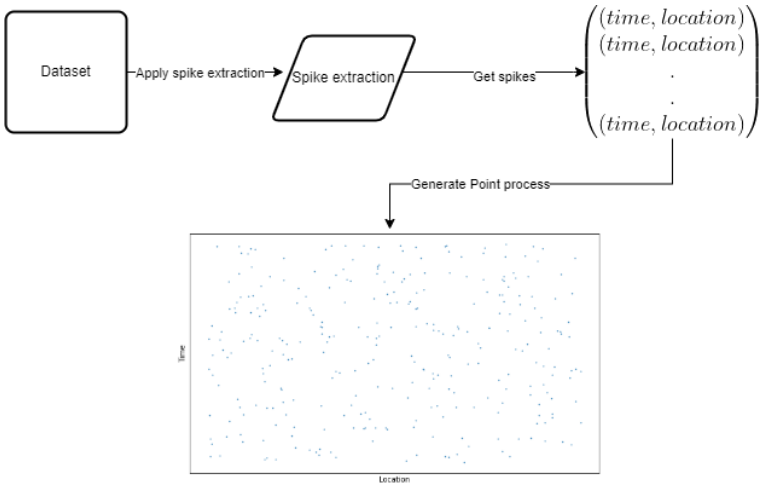


Figure 4.2: Work process for extracting the spike measurements and using them for defining a point process.

a library which provides GP implementation using MCMC sampling, provides easy integration of approximation algorithms like FITC into the model (because our model can become computationally complex), provides enough choices of covariance functions and also includes plotting functionality, in case we want to visualise any graphs,distributions of model parameters and posterior distributions.

Having the above criteria in mind, it's indicated that the best option is the use of PyMC3 for implementing the LGCP model. PyMC3 ticks all the checkboxes defined above, with an additional advantage of having easy syntax rules, meaning it can be easily written and understood. Other libraries such as GPflow, GPy or Stan either don't offer MCMC sampling with the use of FITC approximation algorithm or plotting functionalities.

The documentation of PyMC3 provides several example models including an LGCP one which was the one that is followed for this project. In the course of developing a model for point data, this example of the official documentation suggests a simple way conducting inference which is dividing/slicing our domain X into many small pieces $A_1, A_2, .., A_M$, then treat the number of points in each of these small subsets as distinct Poisson random variables, such that each one is a Poisson distribution characterised by a random λ(intensity). We then go on to model the $log\lambda_1, log\lambda_2, log\lambda_3, .., log\lambda_m$ of all the subsets using a Gaussian process. We will also compute the centre of each sub-domain which we will refer to as centroids. These are just the central coordinates of each sub-domain.

Once the splitting will be completed, we will then proceed to the implementation of our model. The first step in the implementation process, as for any probabilistic Bayesian model, is placing prior distributions or assigning fixed values to the hyperparameters of the GP. These are the length scale (ρ or rho) for the covariance function and the mean (μ or mu). In terms of PyMC3, this should be possible to implement in a few lines of code similar to the following.

```
with pm.Model() as lgcp_model:
    example_hyperparameter = <A choice of probability distribution>
    cov_function = pm.gp.cov.ChoiceOfKernel(example_hyperparameter)
    mean_func = pm.gp.mean.ChoiceOfMeanFunction()
```

In PyMC3, we have a variety of available kernels to use, via *pm.gp.cov*. Recall that the choice of kernel depends on the nature of the function we want to model, therefore specific kernels should be more useful and better suited to our model. The choice of kernel will therefore be determined based

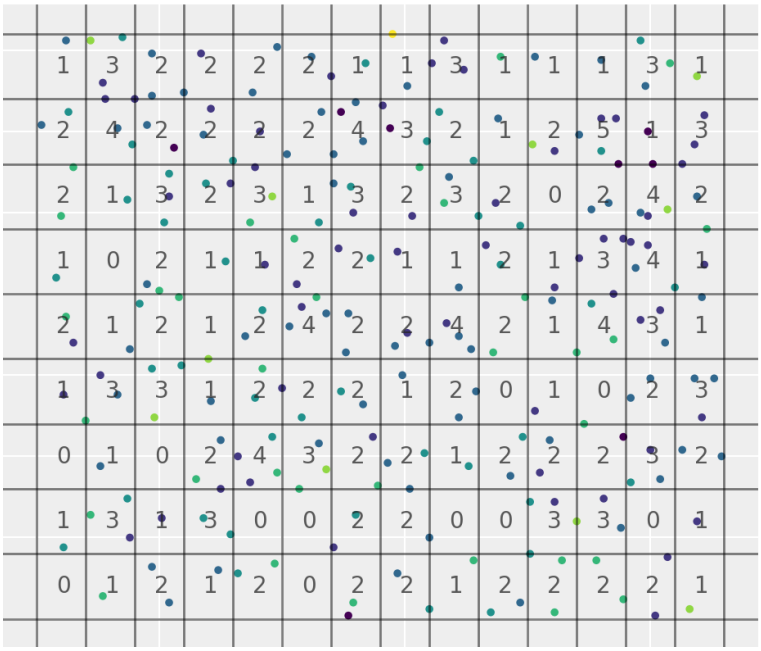


Figure 4.3: An example on the division of the domain into many sub-domains. Each of the cell counts (number of points within) is treated as a random Poisson variable driven by an intensity λ which will be modelled using a GP. The number annotations represent the count of points in corresponding sub-domains.

on the observations we make when we visualise and analyse the distribution and overall behaviour of spikes (right after we define the spike extraction algorithm). After all the hyperparameters of the model are defined, we can go forward and create the GP object using *pm.gp*

```
gp = pm.gp.Latent(mean_func=mean_func, cov_func=cov_function)
```

Note that in PyMC3, this is where we specify whether we want to use a more efficient implementation of a GP if we want to improve the complexity of our model. This will be achieved by using *pm.gp.MarginalSparse* instead of *pm.gp.Latent*. Using *pm.gp.MarginalSparse* we will be able to pass the approximation algorithm of our choice(the FITC) as a parameter.

```
gp = pm.gp.MarginalSparse(mean_func=mean_func, cov_func=cov_function,
approx="FITC")
```

Remember that we are actually modelling the log intensity of the point process and not the intensity, thus we need to place the prior on the log intensity and then transform the GP to a positive-valued process. The placement of the prior in PyMC3 is carried out by calling the *.prior* method on our GP object. The transformation to a positive process will be possible by using the *mp.math.exp* method. The Poisson likelihood for our model will then be specified using pm.Poisson.

```
log_intensity = gp.prior()
intensity = pm.math.exp(log_intensity)
poisson_likelihood = pm.Poisson()
```

By now the model will be fully specified, so we will be able to proceed to sampling from the posterior by calling the pm.sample() method.

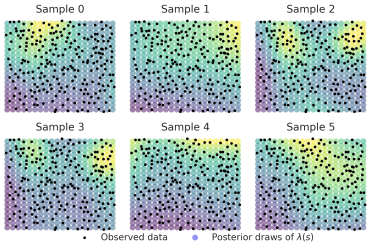
```
trace = pm.sample(target_accept=0.95, chains=4, return_inferencedata=True)
```

In order to evaluate the intensity at new test points in space, we can first use the gp.conditional to get the new log intensity values given our new test points and then use these values to *pm.samplePosteriorPredictive* method to sample from the new posterior.

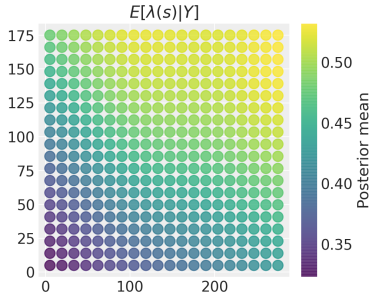
```
intensity_new_values = gp.conditional("log_intensity_new", Xnew=new test points)

spp_trace = pm.sample_posterior_predictive(
        trace, var_names=["log_intensity_new"], keep_size=True
    )
```

After evaluating the posterior of our model, we will then be able to visualise the 2D intensity field of our spike point process. The intensity field will be color graded to describe the intensity value in all the regions of the point process.



(a) An example of visualising some samples from the posterior of the model.Brighter colors signify higher intensity and darker colors signify lower intensity values.



(b) An example of a visualisation of the posterior mean of our model.

Figure 4.4

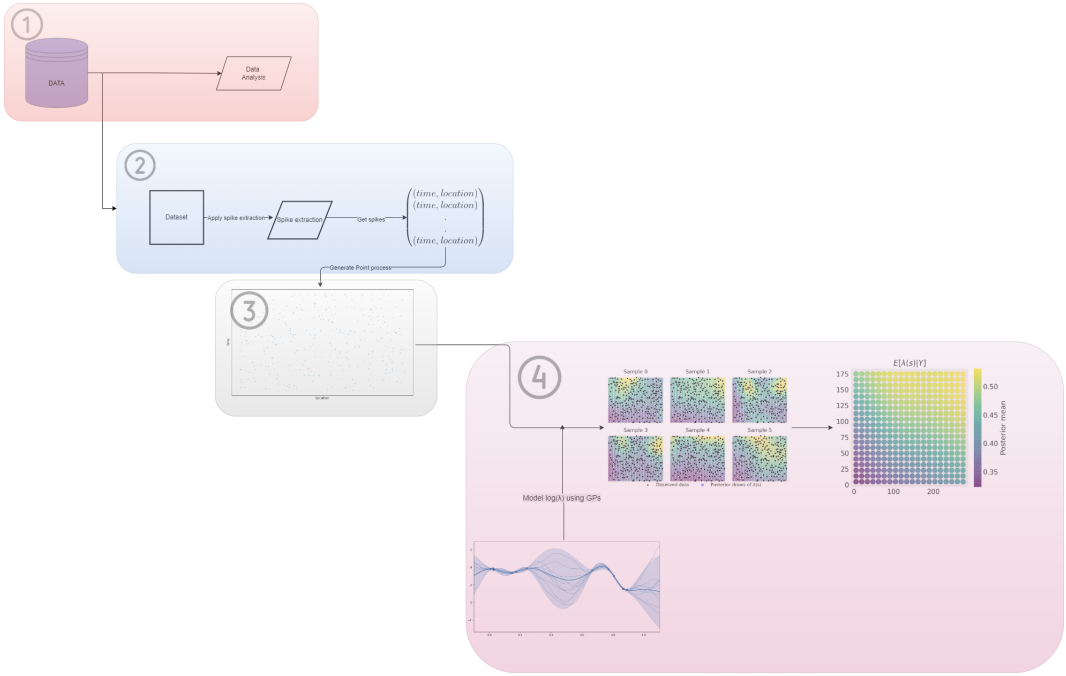


Figure 4.5: A complete diagram of the project's workflow, with step 1 being the collection of the datasets from a data storage and the analysis of the data, step 2 being the development and application of the spike extraction algorithm on the data in order to obtain spike measurements, step 3 being the definition of the point process which represents the pollution spikes and the last step being the implementation of the LGCP model for modelling the intensity of the point process.

Chapter 5

Implementation and results

5.1 Data acquisition and analysis

As specified in the project aims and objectives chapter, the project was started with the acquisition of the air pollution measurements dataset. Initially, the dataset was consisted of two years worth of measurements for just seven monitoring sites in Kampala. This chunk of data was sufficient and was used for the completion of the following task, the analysis of pollution measurements records and the design of the preprocessing schemes that will be applied to the entire dataset before proceeding to the next stage. The full dataset consisting of measurements for more than seventy monitoring sites was not used for this section of the project. Please note that only the findings of the analysis on just one monitoring site are discussed and visualised here, since including plots and discussion for all initial seven sites would occupy too many pages.

The analysis of the provided air pollution data, was carried out using several Python tools and libraries for visualising and studying plots and distributions (e.g. Matplotlib). Further libraries (e.g. Scipy[43]) were used for the utilisation of specific signal/data preprocessing algorithms.

Moving on to the results of the analysis, the process was initiated by observing some simple statistical metrics of the dataset, such as the mean, maximum, minimum and standard deviation of the dataset[8]. These metrics provided some basic intuition about the measurements and gave some early information about the differences of the recorded values in the two sensors used in each site.

Table 5.1: General statistic metrics for PM2.5 and PM10 measurements in the Nakasero A site dataset.

	Sensor 1	Sensor 2		Sensor 1	Sensor 2
PM2.5 mean	33.3	44.6	PM10 mean	38.5	51.0
PM2.5 std	21.0	30.3	PM10 std	24.5	32.8
PM2.5 max	0.0	0.0	PM10 max	0.0	0.0
PM2.5 min	901.7	854.5	PM10 min	904.6	863.4
PM2.5 skew	2.8	2.7	PM10 skew	2.1	2.3
PM2.5 0.25 percentile[21]	18.5	26.4	PM10 0.25 percentile	19.8	28.1
PM2.5 0.75 percentile	43.2	52.3	PM10 0.75 percentile	52.9	65.3

As we can see (table 5.1), compared to the first sensor, measurements coming form the second one have an upwards shift in their values, as the mean climbs from 33.3 to 44.6 (for PM2.5) . By plotting the entire Nakasero A dataset, we see that sensor 2 measurements are higher than the ones of sensor 1, while the positions of the highest spikes remain the same (although by observing more, we can see

that at certain points, spikes in one of the sensors differ a lot from the same measurement at the other sensor, these are probably individual sensor malfunctions/anomalies).

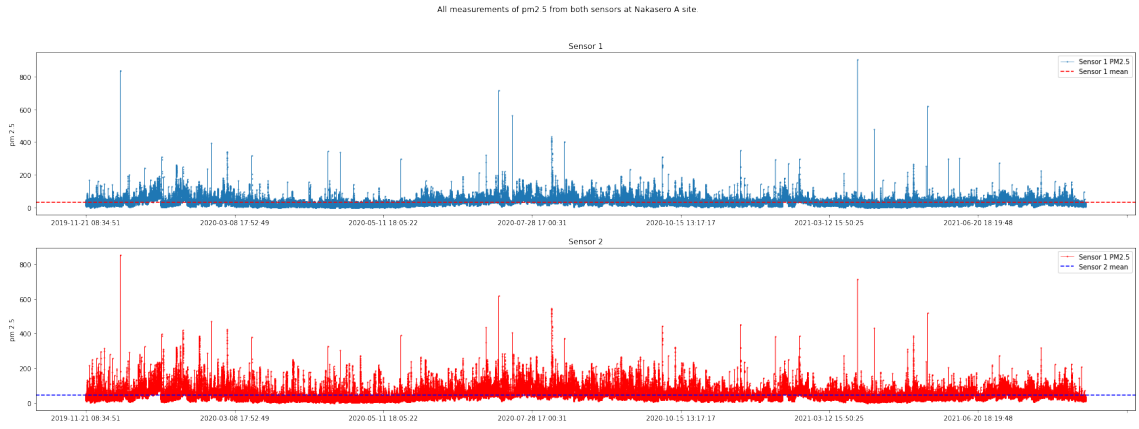


Figure 5.1: Plots of PM2.5 from both sensors in Nakasero A monitoring site from 11/2019 to 06/2021.

In figure 5.1, hundreds of thousands of measurements are plotted into the same plots, therefore observing if the data follow a similar shape in both of the sensors cannot be fully or easily determined. To get a better resolution, plots of the first 500 measurements in Nakasero A site are plotted side by side in 5.2 and it is now evident that data from both sensors follow similar paths, but once again, the upwards shift of the sensor 2 measurements is confirmed.

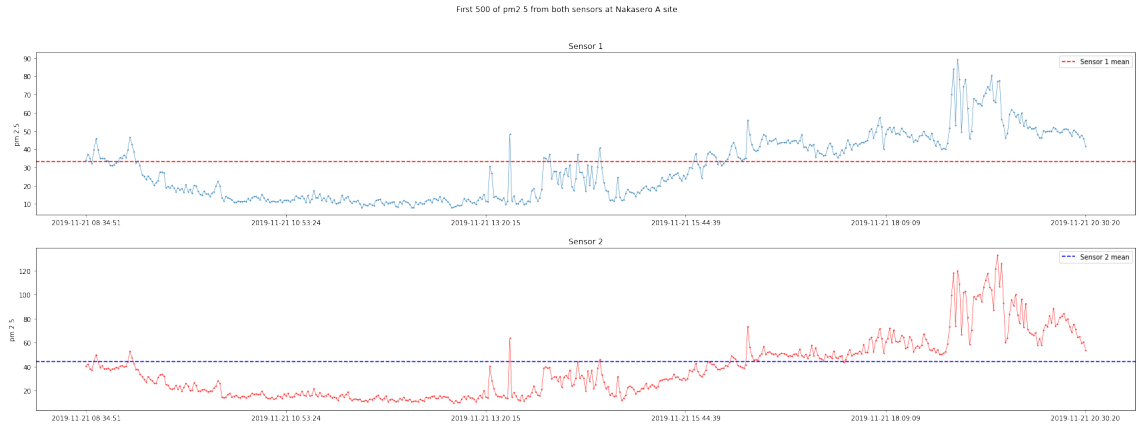


Figure 5.2: First 500 measurements of PM2.5 from both sensors in Nakasero A monitoring site.

After these more general observations on the dataset, we then moved forward to an initial examination of the most obvious pollution spikes, the measurements whose volume clearly stands out among others.We can now have another look at a plot of a subset of the measurements (for simplicity) from both sensors. In the plots of 5.3, several measurements of higher values can be manually detected, with more of them being concentrated in the middle of the x-temporal axis (if these measurements are legitimate, this can possibly give us a hint of spikes cluster formations and more intense occurrence of spikes in specific time frames, indicating variability of the occurrence rate of the spikes) . One further thing to discuss, is the fact that not all the spikes appear to both of the sensors. Highlighted with red, are some individual spike measurements which appear in the exact same place at both sensor 1

and sensor 2 measurements, and highlighted with green is the position/time where a spike appears in sensor 2 but not in sensor 1. This confirms the possibility of the two sensors detecting highly conflicting records and it might be a hint that a malfunction was present at that point of time in one of the two sensors.

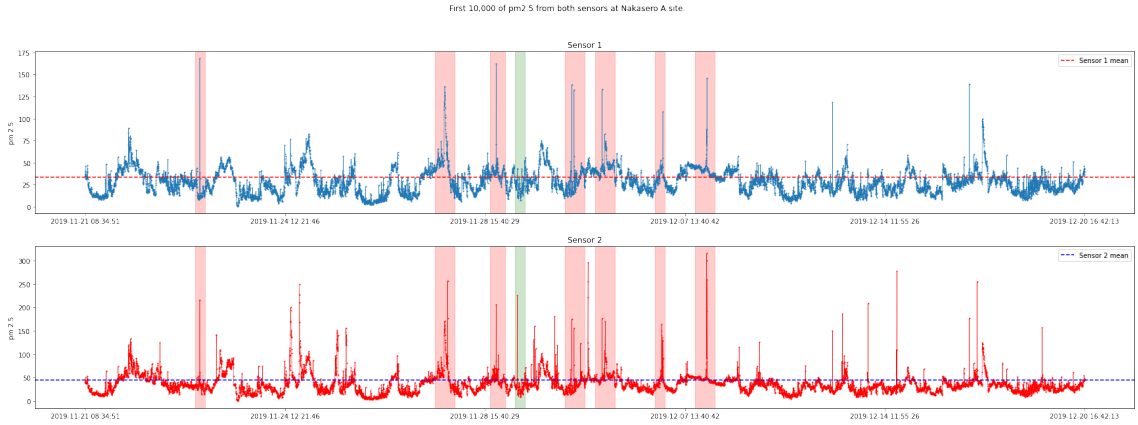


Figure 5.3: First 10,000 measurements of PM2.5 from both sensors. A subset of high measurements spikes from both sensors highlighted with red, contradicting spike measurements in green. Note that obviously not all spikes are selected here, just a some of them in order to give an example of conflicting measurements.

Until now, we only considered looking at the possible spike measurements in a more general manner by looking at the plots of many measurements and just picking the ones that stood out. Following that, it was decided to have closer looks at random individual spikes, in order to better observe the magnitude, how wide, smooth, steep they are and in what degree do both sensors agree with each other. In figure 4.4 spike 1, we visualise an individual spike which appeared between measurements 1144-1149. It is clear that both sensors detected similar values, thus, there is an increased possibility that the specific measurement is indeed a spike. Shapes in regards to width and slope are also maintained in both sensors. The same conclusions can be inferred from Spike 2 and 3 subplots in the same figure, but not from the Spike 4 subplot, where extreme variations are observed at specific measurements in both sensors.

Whether the conflicting measurements in the sensors is to be considered as sensor malfunctioning is a matter which needs to be discussed with the manufacturers of the sensors and the engineers responsible for them. This is a topic outside of the scope of this project, therefore we are presented with the choice to only consider one of the sensors or only examine spikes which are detected from both sensors.

When analysing datasets for ML projects, it is important to examine the kind of distribution the data follow. Usually, it is preferred for datasets to be a part of a normal distribution, since normal distributed data can be more understandable and outlier detection algorithms (which we use later) perform better when applied to Gaussian-like distributions. By visualising our data (sensor 1 PM2.5 data) using Python's histogram function, we get the distribution in figure 4.5. It is obvious that the extremely high pollution measurements have forced the distribution to be severely skewed towards the left, lower values. Furthermore, the data don't seem to follow the normal distribution, being fairly uneven, especially in the middle where 2 peaks are observed. In the same figure we also visualise the corresponding box-plot[46], confirming the great influence that outliers (black outlined circles on the right) have in the overall dataset. The observations from the distribution study, lead to the important

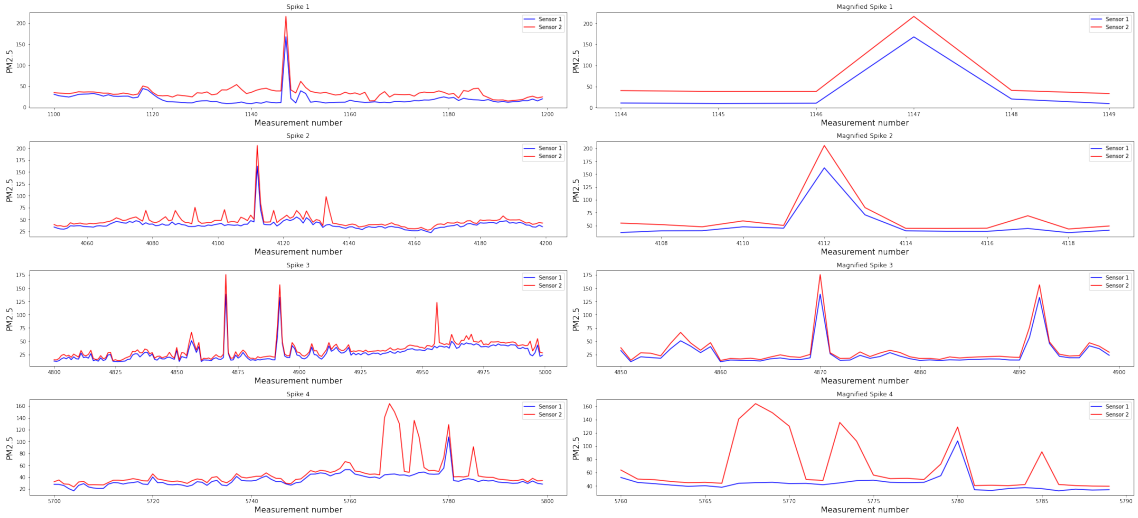


Figure 5.4: Individual spikes at left side, magnified in the right hand side subplots.

decision that suitable preprocessing methods have to be applied to the dataset in order to acquire manageable normalised data.

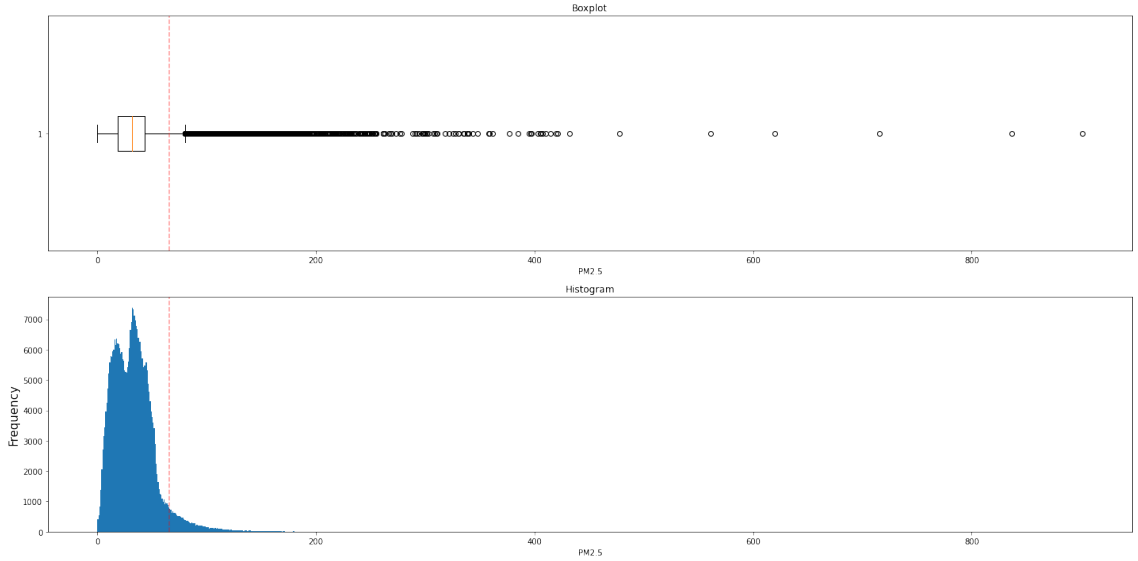


Figure 5.5: Data boxplot and histogram(1000 bins)

5.2 Preprocessing

As mentioned above, certain preprocessing of the data is needed in order to continue to the outlier detection process. From the basic plots of the raw data, we can certainly identify that the numerous high measurement records are the main reason that make our data look unstable and noisy (see fig 4.1, where in certain x-axis regions, measurements are much higher than others in different frames, shaping some hill-looking peaks, which heavily influence the mean) . Applying a smoothing function

to the data now becomes an important task, in order to remove as much variability as possible. The smoothing function of choice is the convolutional Savitsky-Golay filter[13], which is available to use through Python's Scipy library. Specifying window size and polynomial order as parameters, the filter was applied to the raw datasets of all the monitoring sites. In figure 5.6, as an example we can see a subset of the raw data for the Nakasero A monitoring site in blue and the same subset after applying the filter as the red dotted line, plotted in the same axes.

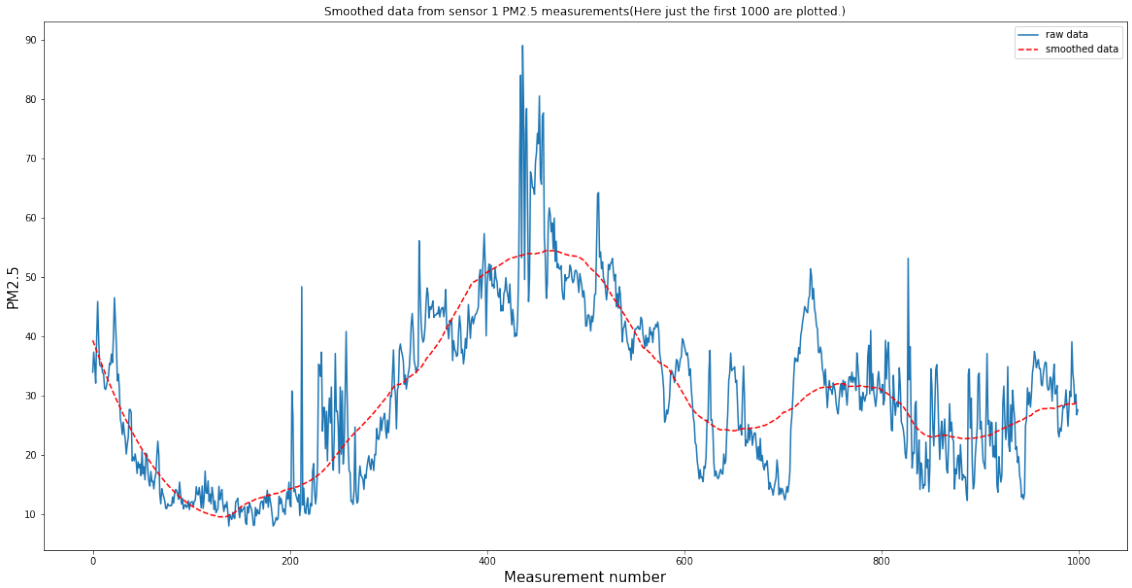


Figure 5.6: First 1000 PM2.5 sensor 1 measurements and the corresponding filtered signal.

The new smoothed data could then be subtracted from the raw 5.7, in order to obtain a new distribution, where large scale variability is eliminated, but enough resolution is still present so that we can apply a spike detection algorithm and get the individual spikes.

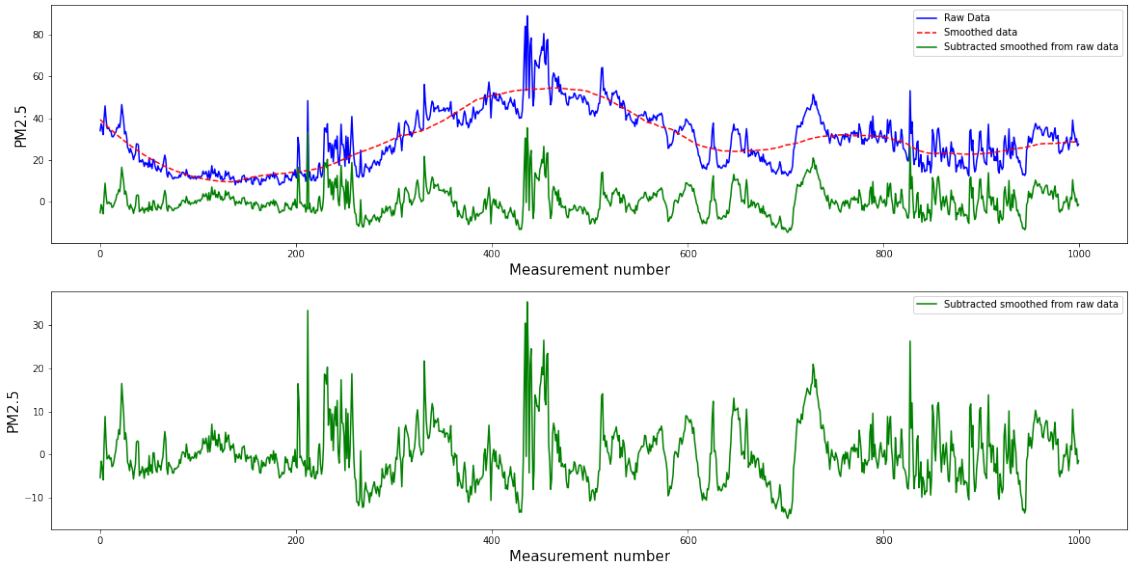


Figure 5.7: Raw data as blue, filtered data in red and the subtraction between the two in green.

5.3 Spike detection algorithm

Now that the preprocessing is done, the dataset is ready for us to apply a function for classifying whether a measurement is a spike or not.Different outlier detection methods have been examined and were considered for this task, all of them statistical. Among them, the Z-Scores and the Interquartile range methods. As indicated in the design chapter, the one that was picked was the modified Z-Scores (4.2), a simple algorithm which classifies measurements as spikes according to the number of standard deviations the measurement is placed away from the mean[1].

Iglewicz and Hoaglin[23] recommend that scores above 3.5 would be potential outliers/spikes, although this value is subject to change, depending on the nature of the problem. For this project, the threshold was initially set to 3.5 with the possibility of increasing it remaining open.

Below, you can see a simple Python implementation of the modified z-scores algorithm, taking the data as input and returning the list with the corresponding scores. Subsequently the same method is used in another method called get spikes which given the data, it chooses the measurements with a score greater than 3.5 and stores the timestamp and the location of each of these measurements in two data structures called X and Y. The values in these structures will later be used to create the spatio-temporal point process by plotting them using matplotlib.scatter() method. A structure called outliers which contains all spikes as tuples of the form (time,location) is also returned.

```
from statsmodels import robust

def modified_z_scores(data):
  z_scores=[]
  median = np.median(data)
  mad = robust.mad(data)
  for i in data:
    score = 0.6745*(i-median)/mad
    z_scores.append(score)
  return z_scores
```

```
def get_spikes2(location,data):
    smoothed = get_smoothed_data(data)
    timestamps = data["created_at"]
    z_scores=modified_z_scores(data)
    outliers = []
    X=[]
    Y=[]

    for i in range(len(z_scores)):
        if z_scores[i]> 3.5 :
            outliers.append((locations[i],timestamps[i]))
            X.append(locations[i])
            Y.append(timestamps[i])

    return X,Y,outliers
```

The above implementation was then applied on the datasets of all measurement sites (after smoothing), and returned the measurements that were successfully classified as spikes. In figure 5.8 the raw sensor 1 PM2.5 data (for Nakasero A site) plot is included, and underneath it a plot only including the measurements that were classified as spikes. We can see that many of the higher initial measurements are indeed returned as spikes, as well as many smaller values that didn't initially stand out as much.

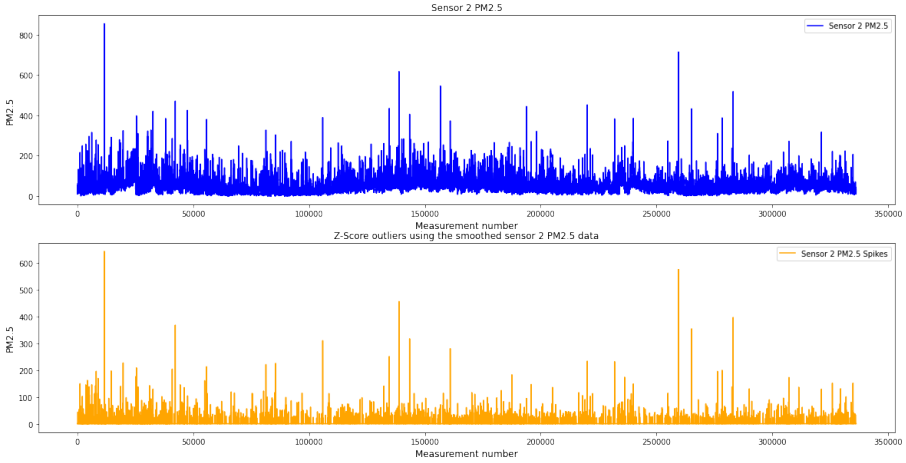


Figure 5.8: Raw sensor 1 PM2.5 data in blue and their corresponding spikes in orange.

For better resolution, scatterplots in figure 5.9 depict the spikes in the first 12000 measurements for both sensors. Here, we can see the void present in the lower part of the plots, probably the space where normal measurements would be placed, and then measurements above a specific point/threshold start to appear as spikes.

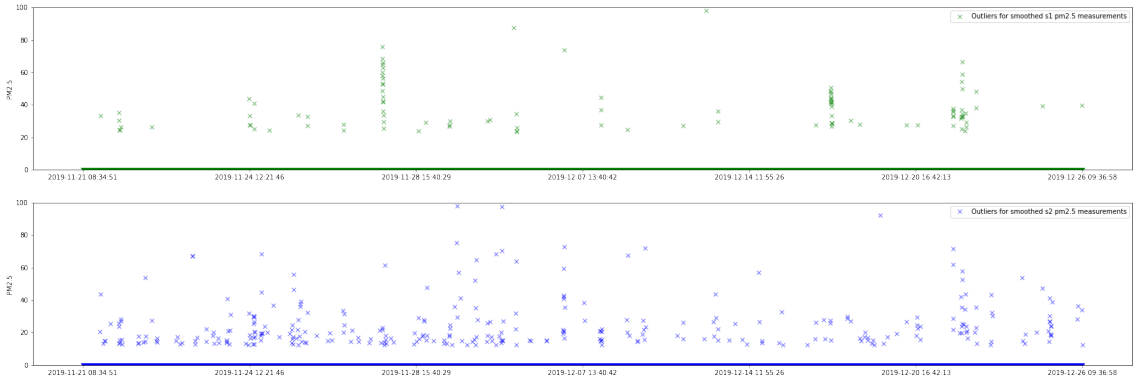


Figure 5.9: Spike occurrences in the first 12000. Temporal x-axis and the amplitude of the spikes in y-axis.

5.4 LGCP spikes model

5.4.1 Definition of the spikes point process

The application of the spike extraction algorithm has now equipped us with the isolated spike measurements, which we can now use in order to define the point process representing spike occurrence in space and time. As described in the design chapter, the location component (values in array X) can be anything from latitude, longitude,distance from city centre, distance from nearest main road or distance from nearest secondary road. During the point process definition process, a few versions of it were implemented using the distance from city centre, nearest main and secondary road. This was done in order to examine the differences between each generated point process and possibly determine any relations between the intensity of spike occurrences and the corresponding location metrics used. The different versions of the point process were obtained by simply changing the source of the location information in the spike extraction algorithm. Initially, different data structures were created with each one including different location information for each monitoring site. Therefore, for defining a point process using distance from city centre as location, we simply draw the corresponding location metric for a specific site by indexing a dictionary of the form dictionary[monitoringSite:locationMetric].

It is important to note that these location metrics, as well as other relevant features could all be used together to define a more complex, multi-dimensional point process and configure the model to consider multiple factors when modelling the intensity of the spikes. In this project we fix the dimensionality of our point process to just 2 features in order to simplify an already difficult task.

Version 1

One of the point process versions was defined using distance from city centre (Kampala) as the location component for the spikes. After applying the spike extraction algorithm to all datasets we were returned with spikes of the form (distance from city centre, dateOfOccurrence). Using these we created the point process represented in figure 5.10.

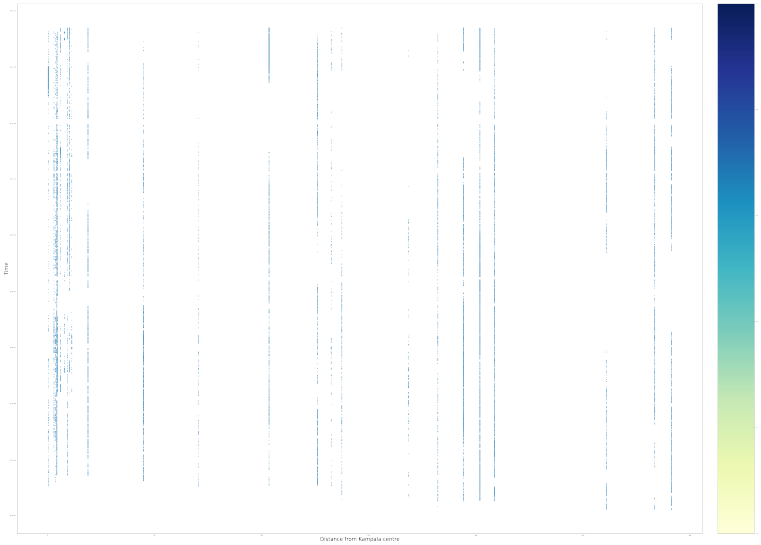


Figure 5.10

Before we proceed to examining the point process, let us first discuss its structure. Since the X-axis is set as the location axis and Y axis is set as the temporal axis , we are therefore presented with a point process whose points form some straight line shaped structures along the location dimension. Remember that that the location component of the point process is discrete, resulting to the separation of the 'lines' of each location as seen in the plot. The large number of spike points at each location along the time axis is causing them to overlap and form these solid looking lines (especially at locations when spikes occurrence is present along the whole temporal axis).

A brief observation of the resulting point process reveals some important information about the concentration of spikes in relation to the distance of each monitoring site from the centre of Kampala. Specifically, it's evident that spikes are more frequent at locations not so far from the city centre, since at the left most side of the point process the population of points is noticeably denser along the temporal axis than any other region. Density of points over the time dimension of the point process is what indicates the presence more pollution spikes. This can most probably be attributed to the higher volumes of traffic within the centre of the city, which can be a reasonable cause of higher concentration of air pollution. A closer inspection of the point process(5.11) also revealed a potential issue regarding a collective lack of measurements during a small timeframe approximately in May 2021. The cause for the void in measurements during that period of time is not clear but could possibly be anything from internet outages to malfunction of the whole network of sensors. Any long periods of missing data is certainly a factor that can influence the performance of our model, especially when building GP regression models where a lack of training data will lead to greater amounts of uncertainty in the result. The time frame of missing data appears to be small, but the temporal axis covers almost 2 years, therefore that small time frame translates to several days of missing observations.

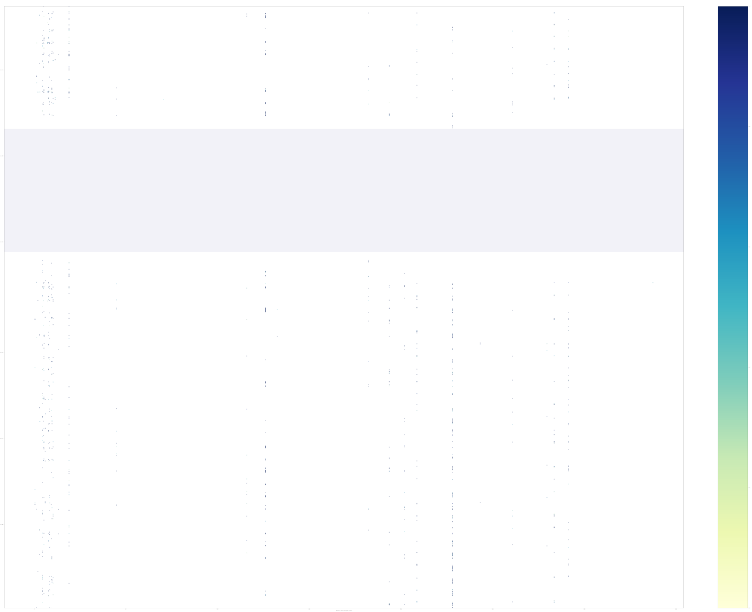


Figure 5.11: Shaded region indicates the time period during which no data are available.

Version 2

Another version of the spikes point process(5.12) was defined using distance to nearest road instead of distance to city centre. Note that $log(distance)$ was used because of the wide range of distance to nearest road values. Using logarithmic scale helped us visualise the data in a compact way. In this version, we can see that locations at the rightmost side have lower density of spike concentration, while locations with smaller distance to nearest road appear to be more dense in spike points along the time axis. The small time frame of missing observations is also visible in this version.

Figure 5.12

Both versions of the point process were used for the implementation of two different models. In the following subsection we will first go through the version using the distance from city centre and then present the results for the version using the distance to nearest road.

5.4.2 Implementation of the model in PyMC3

Using a subset of the dataset and applying the grid

After the definition of the point process representing the pollution spikes, we could move on to placing the grid over the point process as specified in the corresponding design chapter. Using the entirety of the spike observations for modelling proved to have a bad effect on later stages of the model implementation (mainly errors that could not be resolved). Using subsets of the whole training dataset and therefore simplifying the model enabled us to get rid of any similar issues so it was decided to carry on with the building of the model by using just a subset of the data.

By applying the spike extraction algorithm to air pollution records only between Jan 2019 and Oct 2019 we were able to form a reduced and more manageable point process, to which we would then place grid. Recall that the grid is placed on the point process in order to divide the area into many small pieces which themselves constitute individual point processes, afterwards we go on and model the intensity values of these small point processes using a GP.

Figure 5.13 visualises our updated, reduced point process (with the location axis being the distance to city centre). Recall from the design chapter that along along placing the grid over the point process, we also compute the coordinates of the centres of each cell as well as the count of points (spikes) in each of them. These will serve as our training data and observations respectively which will be used to train the GP model.

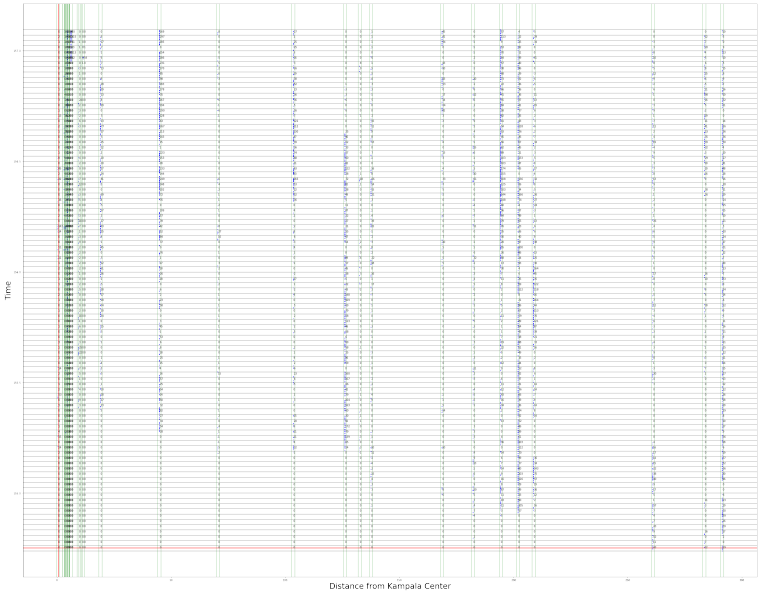


Figure 5.13

Defining the GP object and the challenge of using an approximation method.

Now we were finally able to define our the model. First of all came the initialisation of the GP's main components, the mean and the covariance function. For simplicity, the mean function was set

to be 0 and the length scale for the different choice of kernels was set to fixed values. The model was developed with different experimental values of length scales for two different kernel choices. These are the Exponential and rational quadratic kernels. The choice of kernel and the corresponding length scale values will be stated when the posterior results of each model are presented.

After defining the all the essential elements of the GP, we could then use PyMC's dedicated method for initialising a GP object (*pm.gp*). PyMC offers both a regular implementation of GPs as well as a marginal sparse implementation which can use approximation methods in order to optimise the complexity of the model. Initially , no approximation algorithms were used in the implementation of the model since most of the documentation and examples used the basic, latent implementation. Of course, our model has to deal with enough data to make the sampling and inference process at later stages painfully time consuming. The inversion of the covariance matrix with size NxN when computing the posterior increases the need of a scalable Gaussian process that can afford to work with larger datasets.

Long and extremely time consuming running times then became the reasons to look for ways to use approximation methods when implementing the GP. Even though PyMC3 does offer a syntactically simple to use marginal sparse implementation, it took a considerable amount of time to switch from our initial implementation to the marginal sparse one (marginal sparse implementations are the ones which use approximation methods).

In order to understand the transition from the first implementation to the other, it was decided to work on the task to switch from Latent to marginal sparse in a separate toy example of a model. Most of the alterations that had to be made were simple. First of all the definition of the GP object was revised by switching from pm.gp.Latent() to pm.gp.MarginalSparse(). As described in the design chapter of this report, the approximation method of choice is the FITC. Fortunately, marginal sparse implementation provides the ability to use that.

The use of the marginal sparse implementation also requires the definition of a number of inducing points to use in order for the FITC approximation to be properly utilised. At the beginning, the inducing points were initialised as random points, not in the domain of our training data. This was not proved to be helpful, since after some runs the model seemed to have trouble generating sensible results. Following specific parts of the documentation[4], the inducing points were then initialised using PyMC's *pm.gp.util.kmeansInducingPoints* method which helps us initialise the inducing points as points picked from our training data. Lastly, the method placing the prior over the log intensity also had to be revised, changing from *pm.gp.Prior* to *pm.gp.MarginalLikelihood* which needs needs the training data (centroids), inducing points and observations (the cell counts) as parameters.

Finally, the process that needed to be followed in order to switch from the basic latent to marginal sparse implementation was clear. All we had to do was to use the changed implementation in that toy model in our actual pollution spikes LGCP model.

```
with lgcp_model:
    gp = pm.gp.MarginalSparse(cov_func=cov_func_RatQuad, mean_func = mean_func,
        approx='FITC') # Marginal sparse
    Xu = pm.gp.util.kmeans_inducing_points(30, final_centroids)

       = pm.Normal("  ", mu=10, sigma=2)
    log_intensity = gp.marginal_likelihood("log_intensity", X=final_centroids, Xu=Xu,
        y=observations, noise=  ) # Sparse

    intensity = pm.math.exp(log_intensity)

    rates = intensity * area_per_cell
    # Poisson likelihood
```

```
    counts = pm.Poisson("counts", mu=rates, observed=observations)
```

As described above, this code snippet initialised the GP with the essential parameters already defined, it created the inducing points(Xu), created the noise added to the model(σ) and then called the *gp.marginalLikelihood* method to place a prior over the log intensity of the spike point process which is the value we want to model. This value was later transformed to be positive only using the *pm.math.exp* method and then used to define the model's Poisson likelihood (full version of this code with comments included in appendix).

We could then use the *pm.Trace* method to sample from the resulting posterior, in case we need to have a look at the distributions of any model parameters we set a prior on. The first parameter of the pm.trace method indicates the number of posterior draws[].

```
with lgcp_model:
    trace = pm.sample(5000, chains=4, tune=2000, target_accept=0.85,
    return_inferencedata=True)
```

Since we are only using a subset of the data as well as using the FITC approximation algorithm and fixing some of the models hyperparameters, the running time for this initial sampling from the posterior is very short, with most runs taking even under a minute.

Predictions were now ready to be made. All that had to be done was to define new testing points in the same domain as the axes on our spike point process and then feeding these points to the *gp.conditional* method which will extend the model by adding the conditional distribution (log intensity new) so we can predict at new testing locations. We can then sample from this newly added distribution to get the predictions by using the sample_posterior _predictive function (full version of below code with comments included in appendix).

```
x_new = x_new = np.linspace(0, 6, 40)
y_new = np.linspace(15.45, 15.73, 40)
xs, ys = np.asarray(np.meshgrid(x_new, y_new))
xy_new = np.asarray([xs.ravel(), ys.ravel()]).T

with lgcp_model:
    intensity_new = gp.conditional("log_intensity_new", Xnew=xy_new)

spp_trace = pm.sample_posterior_predictive(
    trace, var_names=["log_intensity_new"], keep_size=True
)
```

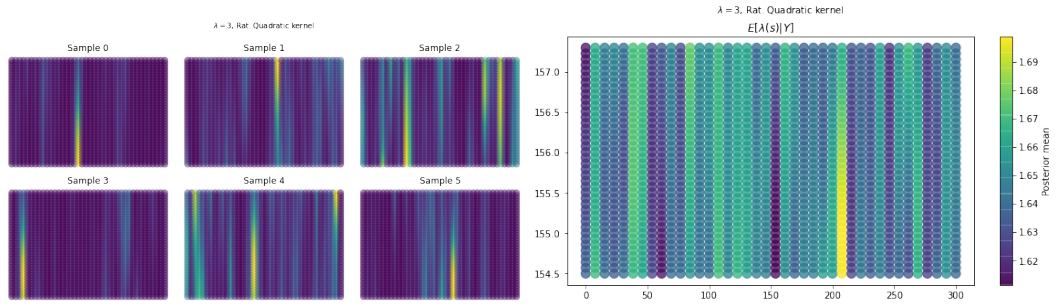
Visualising Predictions

After sampling the predictive distribution for predictions, we are now able to visualise the samples. Testing points were set at random areas over the domains of our point process, therefore by sampling the predictive posterior distribution we will be able to visualise the intensity over the whole point process. Recall from the design chapter that since we are modeling the intensity of a 2 dimensional area, the visualisation of the resulting posterior draws is a 2D surface which has been color graded to signify the different intensity levels across the point process. The model results were checked on different choices of fixed length scales over 2 different choices of kernels for the GP's covariance function. The kernels/covariance functions used were the Exponential (eq. 5.2) and Rational quadratic (eq. 5.1) [17].

$$k_{RQ}(x, x') = \sigma^2(1 + \frac{(x - x')^2}{2al^2})^{-a} \qquad (5.1) \qquad k_E(x, x') = \exp[-\frac{||x - x'||}{2l^2}] \qquad (5.2)$$

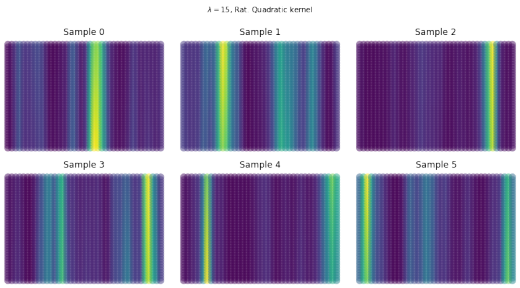
Some of the posterior draws as well as the corresponding posterior mean surfaces are included below.

Model with Rational Quadratic kernel

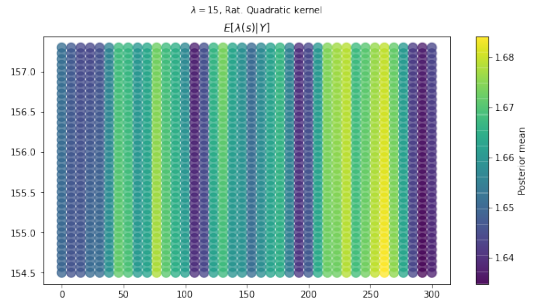


(a) length scale = 3

(b) length scale = 3



(c) length scale = 15

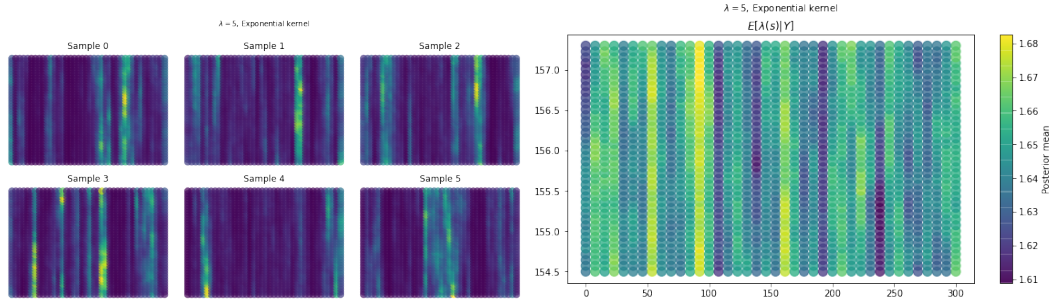


(d) length scale = 15

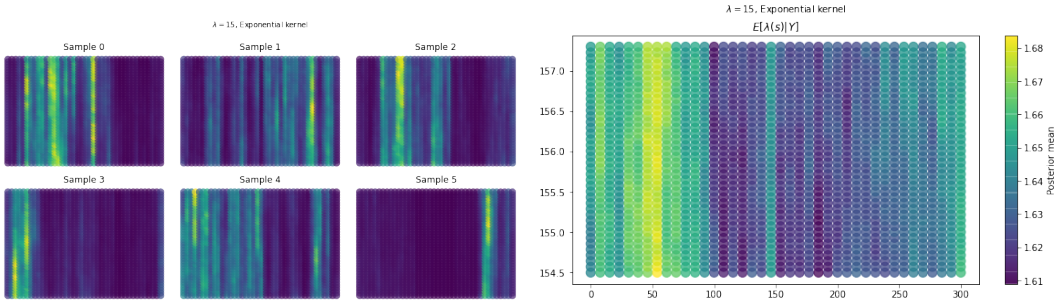
Figure 5.14: On the left: realisations of 6 posterior samples visualising the modeled intensity of the spikes point process. Lighter colored areas represent higher values of intensity and areas with darker colors indicate lower intensity values. On the right: Mean posterior surface depicting the modeled intensity over the spikes point process (X axis is distance from city centre and Y axis is time-timestamps). Color grading indicates higher intensity in areas of lighter colors and lower intensity in darker areas.

The impact of the length scale values is visible in both cases. Specifically, a length scale equal to 3 results to less smooth and swiftly changing predictions and mean posterior surface. Increasing the length scale from 3 to 15 resulted to smoother, not so rapidly changing predictions. When larger length scales are used, one needs to travel larger distances in input space in order to observe a considerable change in the modeled function.

Model with exponential kernel



(a) length scale = 5 (b) length scale = 5



(c) length scale = 15 (d) length scale = 15

Figure 5.15: On the left: realisations of 6 posterior samples visualising the modeled intensity of the spikes point process. Lighter colored areas represent higher values of intensity and areas with darker colors indicate lower intensity values. On the right: Mean posterior surface depicting the modeled intensity over the spikes point process (X axis is distance from city centre and Y axis is time-timestamps). Color grading indicates higher intensity in areas of lighter colors and lower intensity in darker areas.

The same observations regarding the effect of different values of length scales can be made from **??**. A difference in the overall structure of the resulting surfaces is apparent as well, since the covariance function was switched from rational quadratic to exponential.

5.5 Unit testing and model validation

The validation of the model has been carried out by intentionally removing the observations in certain regions of the spikes point process (for which the spike count per cell area is already known). We later proceed to model the intensity using the remaining observations and comparing whether the modeled intensity for the region for which the observations were dropped corresponds to the actual number of spike counts in the same area. In order to get an overall metric for measuring the error, we use the mean squared error over 4 selected regions (removing observations from 4 different regions and calculating the error over these given the true and predicted spike counts).

Let us consider the model using the rational quadratic kernel with length scale equal to 5. In the figure below, we visualise the 4 areas from which observations were dropped.

Figure 5.16: Circled with red are the areas for which observations were dropped for the validation process.

Each round , the model is developed so that we can get a prediction of the intensity at the regions for which the observations were dropped. The true spike counts for the regions are known and then used to determine the accuracy of the modeled intensity.

Below, we document the true spike occurrence counts of the selected regions for each round as well as the predicted intensity values translated into spike counts.

	True count	Predicted count	Mean squared error
Region 1	185	44	19881
Region 2	54	44	100
Region 3	792	47	555,025
Region 4	14	57	1849
Avg			144,213

Figure 5.17: Table containing the true and predicted counts of the dropped observation regions. Mean squared error also included.

5.5.1 Discussion

Passing the model through some form of validation has proven that the results are not the ones we desired from the very beginning of the project. The error metric is very large, therefore signifying the inability of the model to produce accurate results. The lack of accuracy in the results was also evident from the visualisations of the predictive posterior samples in the results section. By visualising the samples as well as the mean posterior intensity surface of the point process we could clearly observe that the posterior did not manage to sufficiently adapt to the observations/training data, therefore leading us to the stage of being unable to use the posterior for quality predictions.

There are many reasons that lead to the poor results of the model, spanning from data to implementation related issues. All these potential factors that affected our result are listed and discussed below.

Not using the whole dataset.

It can be said that one the most basic principles of machine learning has not been fully put into practise for this project, since we ended up not using the whole training dataset in order to develop our model. Using a sufficient amount of training data to train a ML model cannot itself guarantee successful and accurate results, but it at least provides a certain amount of confidence that the model will have enough data and true values to adapt to, thus increasing the odds of getting meaningful results. For this project, using the full training dataset to form the spike point process yielded errors that were both hard to understand and to solve and was causing serious delays in the course of developing the model. This was the reason for using a more manageable subset of the whole training dataset to train our model on.

Suboptimal parameterisation

In Bayesian statistics, models are developed so that later the parameters within that model can be estimated. Whether this estimation is successful or not greatly depends on the proper parameterisation of the parameters. A poor choice of model parameters increases the risk of slow and biased MCMC sampling (Markov chain Monte Carlo - a class of algorithms for systematic random sampling from high-dimensional probability distributions)[3]. This is another problem that can be either hard or relatively easier to spot depending on the circumstances and it can often be hard to solve. This issue was unfortunately present in our project as well. Specifically, using fixed hyperparameter values might have spared us of other problems that were limiting our progress, but it immediately became the main cause for biased inference which we were able to spot due to the appearance of divergence warnings while we were sampling using the *.sample* method. The problem of biased inference and divergences is itself another relatively complex topic which I will not explain in this report, but a reasonable and brief introduction to it can be found in this official PyMC3 tutorial called "Diagnosing Biased Inference with Divergences"[5].

Other data deficiencies

Dealing with raw data is certainly one the many challenges one will need to face when trying to develop a model, especially when new incentives are in place Chunks of missing data across all sensors/monitoring sites could also be an issue that potentially affected the training of the model. A brief discussion of this problem was included in the spikes point process definition.

5.6 Further work

5.6.1 Pollution spikes model

Throughout the course of developing the spikes model, we tried simplifying the model in several possible ways so that we minimise the complexity of the project which proved to be challenging from the very beginning. This simplification included limiting the model (and consequently the point process) to two input dimensions/ two features (space and time). More sophisticated models use multiple dimensions/features in order to derive more useful results. A pollution spikes model could use many different features apart from location and time. As stated in the literature review, research has shown that the air pollution problem is certainly a multidimensional issue which can be influenced from factors like altitude, population density, meteorological factors such as wind speed and precipitation as well as traffic and vehicle congestion. Our model only used a time and location feature, therefore an obvious addition to this could be the incorporation of new features like the ones mentioned above.

5.6.2 Modelling air pollution

As stated in the requirements analysis chapter, the full scope of this project included the separate modeling of the pollution spikes as well as a general air pollution model which would utilise the spikes model in order to acquire new useful information and possibly new features/input dimensions.

We ended up not including the second more general air pollution model, but as soon as a reliable and accurate model is built on the pollution spikes, a model predicting air pollution levels in terms of particulate matter would be the next most obvious step.

Chapter 6

Conclusion

Researching, designing and implementing this project, even in the absence of accurate results has helped us identify several challenges that can emerge in the course of developing a model for pollution spikes and use it for further air pollution modelling in the future. Data visualisation and analysis have aided to the cause of understanding the structure and basic behaviour of an event such as pollution spike occurrence. Furthermore, the identification of some data deficiencies have made it clear that complete and stable availability of data is crucial when building a statistical model and aiming for accurate results.

The use of certain techniques has highlighted the importance of having specialised tools for solving specific problems (the availability of a dedicated library such as PyMC3 which helps develop probabilistic models with specific sets of algorithms such as MCMC), but has also exposed difficulties and issues that might arise when managing larger amounts of data with these tools. Possible factors that influenced the outcome have also been identified and discussed towards the end and gave hints on what could have been done differently in order to get better results.

Even though the accuracy and completeness of the pollution spikes model is questionable, I really hope that the findings of this project can lay the groundwork for further study of pollution spikes as a major influencing factor on air pollution, and possibly inspire new, even more helpful research on the topic.

Appendix A

PyMC3 model implementation scripts

This appendix includes the full versions of the code snippets included in chapter 5 for the implementation of the model in Python's PyMC3.

```
# Build model
with pm.Model() as lgcp_model:
    mu = 0
    rho = 8
    variance = 1
    cov_func_RatQuad = variance * pm.gp.cov.RatQuad(2,1,ls=rho)
    mean_func = pm.gp.mean.Constant(mu)

with lgcp_model:
    # Create the GP object using MarginalSparse instead of Latent.
    # Approximation algorithm used is FITC.
    gp = pm.gp.MarginalSparse(cov_func=cov_func_RatQuad, mean_func = mean_func,
        approx='FITC') # Marginal sparse

    # initialize 10 inducing points with K-means
    # gp.util
    Xu = pm.gp.util.kmeans_inducing_points(10, centroids)

    # Noise to be passed as parameter to the marginal_likelihood method(
    # version of .prior method for marginal
    # sparse implementation)
       = pm.Normal("  ", mu=10, sigma=2)

    # Place the prior over the function we want
    # to model(the log intencity of the point process)
    # using the .marginal_likelihood method instead of
    # the .prior method we used in the latent
    # implementation. Parameters are the X training data,
    # Xu inducing points, y observed data and noise sigma.
    log_intensity = gp.marginal_likelihood("log_intensity", X=centroids, Xu=Xu,
        y=cell_counts, noise=  ) # Sparse

    intensity = pm.math.exp(log_intensity)

    rates = intensity * area_per_cell
    # Poisson likelihood
```

```
    counts = pm.Poisson("counts", mu=rates, observed=cell_counts)

with lgcp_model:
    trace = pm.sample(5000, chains=4, tune=2000, target_accept=0.85,
        return_inferencedata=True)
```

```
# Create new entry points to apply the conditional and get prediction
x_new = x_new = np.linspace(0, 300, 40)
y_new = np.linspace(154.5, 157.3, 40)
xs, ys = np.asarray(np.meshgrid(x_new, y_new))
xy_new = np.asarray([xs.ravel(), ys.ravel()]).T

with lgcp_model:
    # Perform inference
    # The conditional method creates the conditional, or predictive distribution
    # over the latent function at arbitrary X* input points(xy_new here), f(x*).
    intensity_new = gp.conditional("log_intensity_new", Xnew=xy_new)

    spp_trace = pm.sample_posterior_predictive(
        trace, var_names=["log_intensity_new"], keep_size=True
    )
    # Extend trace to include spp_trace
    trace.extend(
        az.from_dict(posterior_predictive=spp_trace, dims={"log_intensity_new": ["sample"]})

    )
    # The predicted intensity values we get after applying conditioning stored in a var.
    intensity_samples = np.exp(trace.posterior_predictive["log_intensity_new"])
```

Bibliography

[1]

[2] Fitc and vfe.

[3] A gentle introduction to markov chain monte carlo for probability.

[4] Modeling spatial point patterns with a marked log-gaussian cox process.

[5] Diagnosing biased inference with divergences.

[6] Air, U. Aqeg advice on the use of 'low-cost' pollution sensors- Defra, UK. `https://uk-air.defra.gov.uk/research/aqeg/pollution-sensors.php`. [Online; accessed 13-November-2021].

[7] Alcalá, J., Parson, O., and Rogers, A. Detecting anomalies in activities of daily living of elderly residents via energy disaggregation and cox processes.

[8] Alizadeh, E. A Guide to Metrics (Estimates) in Exploratory Data Analysis. `https://ealizadeh.com/blog/guide-to-estimates-in-exploratory-data-analysis`. [Online; accessed 13-November-2021].

[9] Brix, A., and Diggle, P. J. Spatiotemporal prediction for log-gaussian cox processes. *Journal of the Royal Statistical Society: Series B (Statistical Methodology) 63*, 4 (2001), 823–841.

[10] Carl Edward Rasmussen, H. N. Gaussian processes for machine learning (gpml) toolbox. p. 3011.

[11] Castelli, M., Clemente, F., Popovič, A., Silva, S., and Vanneschi, L. A machine learning approach to predict air quality in california. *Complexity 2020* (08 2020), 5–6.

[12] Commission, E. Measuring Air Pollution with low cost sensors. `https://ec.europa.eu/environment/air/pdf/Brochure%20lower-cost%20sensors.pdf`. [Online; accessed 13-November-2021].

[13] D, W. Savitzky-golay filters, Jan 2019.

[14] Daly, A., and Zannetti, P. Air pollution modeling–an overview.

[15] Daniel Lizotte, Tao Wang, M. B. D. S. Automatic gait optimization with gaussian process regression.

[16] Diggle, P. J., Moraga, P., Rowlingson, B., and Taylor, B. M. Spatial and Spatio-Temporal Log-Gaussian Cox Processes: Extending the Geostatistical Paradigm. *Statistical Science 28*, 4 (2013), 542 – 563.

[17] Duvenaud, D.

[18] Europe, W. Health effects of particulate matter. policy implications for countries in Eastern Europe, Caucasus and Central Asia (2013). https://www.euro.who.int/en/health-topics/environment-and-health/air-quality/publications/2013/health-effects-of-particulate-matter.-policy-implications-for-countries-in-eastern-europe,-caucasus-and-central-asia-2013. [Online; accessed 10-November-2021].

[19] Gabasova, E.

[20] Gerard Hoek, Ranjini M Krishnan, R. B. A. P. B. O. B. B. . J. D. K. Long-term air pollution exposure and cardio- respiratory mortality: a review.

[21] Glen, S. Percentiles, Percentile Rank Percentile Range: Definition Examples. https://www.statisticshowto.com/probability-and-statistics/percentiles-rank-range/. [Online; accessed 13-November-2021].

[22] Goyal, P., and Kumar, A. *Mathematical Modeling of Air Pollutants: An Application to Indian Urban City*. 06 2011, pp. 101–103.

[23] Iglewicz, B., and Hoaglin, D. C. How to detect and handle outliers. *American Society for Quality Control , Statistics Division 16* (1993), 9–13.

[24] Jesper Møller, Anne Randi Syversveen, a. R. P. W. Log Gaussian Cox processes. [Online; accessed 13-November-2021].

[25] Kamińska, J. A. The use of random forests in modelling short-term air pollution effects based on traffic and meteorological conditions: A case study in wrocław. *Journal of Environmental Management 217* (2018), 164–174.

[26] Kenton, W. Defining Nonlinear Regression. https://www.investopedia.com/terms/n/nonlinear-regression.asp#:~:text=Nonlinear%20regression%20is%20a%20form,a%20nonlinear%20(curved)%20relationship. [Online; accessed 13-November-2021].

[27] Kim, K., Lee, D., and Essa, I. Gaussian process regression flow for analysis of motion trajectories. In *2011 International Conference on Computer Vision* (2011), pp. 1164–1171.

[28] Kyriakidis, I., Karatzas, K., and Papadourakis, G. Using preprocessing techniques in air quality forecasting with artificial neural networks. pp. 363–365.

[29] Lang, J.-W. DOUBLY STOCHASTIC POISSON PROCESS AND THE PRICING OF CATASTROPHE REINSURANCE CONTRACT. [Online; accessed 13-November-2021].

[30] Laña, I., Del Ser, J., Padró, A., Vélez, M., and Casanova-Mateo, C. The role of local urban traffic and meteorological conditions in air pollution: A data-based case study in madrid, spain. *Atmospheric Environment 145* (2016), 424–438.

[31] Lee, M., Brauer, M., Wong, P., Tang, R., Tsui, T. H., Choi, C., Cheng, W., Lai, P.-C., Tian, L., Thach, T.-Q., Allen, R., and Barratt, B. Land use regression modelling of air pollution in high density high rise cities: A case study in hong kong. *Science of The Total Environment 592* (2017), 306–315.

[32] Liang, Y.-C., Maimury, Y., Chen, A., and Juarez, J. Machine learning-based prediction of air quality. *Applied Sciences 10* (12 2020), 9151.

[33] Ma, R., Xu, X., Wang, Y., Noh, H. Y., Zhang, P., and Zhang, L. Guiding the data learning process with physical model in air pollution inference. In *2018 IEEE International Conference on Big Data (Big Data)* (2018), pp. 4475–4483.

[34] Mahalingam, U., Elangovan, K., Dobhal, H., Valliappa, C., Shrestha, S., and Kedam, G. A machine learning model for air quality prediction for smart cities. In *2019 International Conference on Wireless Communications Signal Processing and Networking (WiSPNET)* (2019), pp. 452–457.

[35] Michael Brauer, Gerard Hoek, P. V. V. K. M. P. H. F. A. W. L. P. K. H. J. N. J. G. M. K. J. H. T. B., and Brunekreef, B. Air pollution from traffic and the development of respiratory infections and asthmatic and allergic symptoms in children.

[36] Munir, S., Mayfield, M., Coca, D., Mihaylova, L., and Osammor, O. Analysis of air pollution in urban areas with airviro dispersion model—a case study in the city of sheffield, united kingdom. *Atmosphere 11* (03 2020), 1–2.

[37] Munir, S., Mayfield, M., Coca, D., Mihaylova, L. S., and Osammor, O. Analysis of air pollution in urban areas with airviro dispersion model—a case study in the city of sheffield, united kingdom. *Atmosphere 11*, 3 (2020).

[38] Pinault, L., Tjepkema, M., Crouse, D., Weichenthal, S., Donkelaar, A., Martin, R., Brauer, M., Chen, H., and Burnett, R. Risk estimates of mortality attributed to low concentrations of ambient fine particulate matter in the canadian community health survey cohort. *Environmental Health 15* (02 2016), 18.

[39] Quiñonero-Candela, J., and Rasmussen, C. E. A unifying view of sparse approximate gaussian process regression. *Journal of Machine Learning Research 6*, 65 (2005), 1939–1959.

[40] Rasmussen, C. E., Williams, C. K. I., and undefined, u. u. MIT Press, 2008.

[41] Rasmussen, C. E., Williams, C. K. I., and undefined, u. u. *Chapter 2: Regression*. MIT Press, 2008.

[42] Rybarczyk, Y., and Zalakeviciute, R. Machine learning approaches for outdoor air quality modelling: A systematic review. *Applied Sciences 8*, 12 (2018).

[43] Scipy.

[44] Serra, L., Saez, M., Varga, D., Tobias, A., Juan, P., and Mateu, J. Spatio-temporal modelling of wildfires in catalonia, spain, 1994-2008, through log gaussian cox processes. pp. 39–49.

[45] S.Glen. Median Absolute Deviation. `https://www.statisticshowto.com/median-absolute-deviation/`, 2019. [Online; accessed 01-December-2021].

[46] S.Glen. Box Plot (Box and Whiskers): How to Read One How to Make One in Excel, TI-83, SPSS. `https://www.statisticshowto.com/probability-and-statistics/descriptive-statistics/box-plot/`, 2021. [Online; accessed 01-December-2021].

[47] Shah, A., Lee, K., Mcallister, D., Hunter, A., Nair, H., Whiteley, W., Langrish, J., Newby, D., and Mills, N. Short term exposure to air pollution and stroke: Systematic review and meta-analysis. *BMJ (Clinical research ed.) 350* (03 2015), h1295.

[48] Srivastava, A., and Rao, B. *Urban Air Pollution Modeling.* 06 2011, p. 16.

[49] Strak, M., Weinmayr, G., Rodopoulou, S., Chen, J., Hoogh, K. d., Andersen, Z., Atkinson, R., Bauwelinck, M., Bekkevold, T., Bellander, T., Boutron-Ruault, M.-C., Brandt, J., Cesaroni, G., Concin, H., Fecht, D., Forastiere, F., Gulliver, J., Hertel, O., Hoffmann, B., and Samoli, E. Long term exposure to low level air pollution and mortality in eight european cohorts within the elapse project: Pooled analysis. *BMJ 374* (09 2021), n1904.

[50] Sun, X., Xu, W., and Jiang, H. Spatial-temporal prediction of air quality based on recurrent neural networks. p. 1269.

[51] Vardoulakis, S., Fisher, B. E., Pericleous, K., and Gonzalez-Flesca, N. Modelling air quality in street canyons: a review. *Atmospheric Environment 37*, 2 (2003), 155–182.

[52] Vitolo, C., Scutari, M., Ghalaieny, M., Tucker, A., and Russell, A. Modeling air pollution, climate, and health data using bayesian networks: A case study of the english regions. *Earth and Space Science 5* (01 2018), 77–78.

[53] Vitolo, C., Scutari, M., Ghalaieny, M., Tucker, A., and Russell, A. Modeling air pollution, climate, and health data using bayesian networks: A case study of the english regions. *Earth and Space Science 5*, 4 (2018), 76–88.

[54] Xi, X., Wei, Z., Xiaoguang, R., Yijie, W., Xinxin, B., Wenjun, Y., and Jin, D. A comprehensive evaluation of air pollution prediction improvement by a machine learning method. In *2015 IEEE International Conference on Service Operations And Logistics, And Informatics (SOLI)* (2015), pp. 176–181.

[55] Yeo, I.-K., and Johnson, R. A new family of power transformations to improve normality or symmetry. *Biometrika 87* (12 2000).