Title

Modeling COVID-19 Spread in Taipei Using the Hawkes Point Process Model: A Spatial-Temporal Analysis

Author

Yen, Hao Ting

Publication Date

2024

Peer reviewed|Thesis/dissertation

UNIVERSITY OF CALIFORNIA
Los Angeles

Modeling COVID-19 Spread in Taipei Using the Hawkes Point Process Model: A Spatial-Temporal Analysis

A thesis submitted in partial satisfaction
of the requirements for the degree
Master of Applied Statistics and Data Science

by

Hao Ting Yen

2024

ABSTRACT OF THE THESIS

Modeling COVID-19 Spread in Taipei
Using the Hawkes Point Process Model:
A Spatial-Temporal Analysis

by

Hao Ting Yen
Master of Applied Statistics and Data Science
University of California, Los Angeles, 2024
Professor Frederic R. Paik Schoenberg, Chair

This paper examines the transmission properties of the COVID-19 virus in Taipei from May 2021 to April 2022. Taipei, being a densely populated city, is an easy target for a highly transmissible virus such as COVID-19. However, the Taiwanese government set up strict public health policies that include a monitoring system and quarantine to prevent the outbreak that was effective in controlling the virus during the peak of infection. Incorporating data collected and maintained by the g0v community, this study dive into the temporal and spatial properties of the virus in Taipei by utilizing the Hawkes Process model. Three variations of the models are fitted in an attempt to find out the most fitting model for the virus data. In conclusion, we found out that using a normal distribution in modeling time and exponential distribution in distance of event occurrence is a suitable way of modeling the COVID-19 spread in Taipei.

The thesis of Hao Ting Yen is approved.

Hongquan Xu

Yingnian Wu

Frederic R. Paik Schoenberg, Committee Chair

University of California, Los Angeles

2024

To my parents and brother for their unending love and support

To everyone whose lives have been affected by the COVID-19 pandemic

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1

Introduction

The COVID-19 pandemic has been a wide spread virus ever since its initial outbreak from Wuhan in 2019.[WHO20]. Its highly transmissible property and high mortality rate of the virus underscores the need for advancement in statistical modeling to better understand and predict the spread of the disease[BQN20] . This thesis focuses on using the Hawkes Point Process Model to analyze the temporal and spatial dynamics of the disease in Taipei. Hawkes Process's ability to handle clustering of events makes it suitable for studying the complex dynamics of disease transmission within environments of high population density such as Taipei[CLM22].

Taiwan, being one of the closest region to Mainland China, was estimated to have one of the highest infection number with its high number of Taiwanese workers and visitors who fly back and forth from China to Taiwan[WNB20]. During the uprising of COVID-19 pandemic, the Taiwanese government was quick to set up a sophisticated monitoring system called "Contact Information Registration" that includes recording one's identity whenever they enter any sort of public space, such as convenience stores and restaurants[Wik23]. It also notified when a person might be exposed to the virus. This system records information such as the exact time and location of an individual as they enter a perimeter. Strict public health policies such as mask mandates, social distancing, and quarantine rules were also hard enforced throughout the peak of the pandemic[Wik23]. All these efforts allowed Taiwanese government to suppress the number of infected individuals from the surge of 8924 total cases in May 2021 to under 100 daily new case in June 2021, and eventually under single digit of eight new cases in July 17, 2021[LLH23].

The Hawkes Model has been shown to be a more accurate forecasting model than the

traditional SEIR model in modeling epidemic diseases such as COVID-19 [KSM22a]. Variations of Hawkes Model such as the non-parametric Hawkes model, latent Hawkes model, and Multivariate Recursive Hawkes has also been shown to be a suitable model for epidemiological modelling for COVID-19 and other virus such as Ebola in order to extract different properties of the virus transmission[LGM23][CSB22][JS22].

With the flexibility of its triggering density, this paper attempts to fit the Hawkes Process model with varying formation of triggering density in an attempt to extract addition information of the COVID-19 transmission property. With the special nature of the dataset for Taiwanese COVID-19, meaning the inclusion of longitude, latitude and the exact time of event occurrence, this study wants to take advantage of the detailed inclusion of such data. In particular, this study looks at the spatial properties of the virus through three model compositions. All three models utilize normal distribution for time variable. For the spatial variable, two of the models use bivariate normal distribution to fit the longitude and latitude parameters, and also fitting the correlation ρ between the two parameters. The third model uses a decaying exponential distribution to fit an α parameter value to determine how quickly the triggering density decreases as the distance from an event increase.

By utilizing R, this study uses the maximum likelihood estimation method to find the best fitting parameter values. In particular, we use the pseudo negative log-likelihood of the conditional intensity of the specific model, and run optimization algorithm using R library[R C23] to find the parameter values that yields the least negative log-likelihood of the conditional intensity. This will in turn give us the parameter values that returns the maximum likelihood estimation of the conditional intensity. We will then examine the fitted parameter values and interpret them given the context of the dataset.

The three models each have different parameter values to fit. Therefore, this study is not a side by side comparison to see which model fits the dataset the best. Rather, this study simply examine the fitted values and attempts to interpret them with the given context of the dataset. We will still, however, try to give a general conclusion of which model seems to be the most useful for the given dataset in term of the extract of information.

CHAPTER 2

Data Preparation

2.1 Data Loading:

The data used in this study comes from Taiwan g0v COVID-19 Traces, which is a database maintained by the g0v community, licensed under CC BY-NC-SA 4.0. The online google sheet includes all traceable covid cases in Taiwan from May 2021 to April 2022. To make the analysis more effective, only events observed in the city of Taipei were kept from the original dataset. The original database contains many spatial information such as the actual address of the event occurrence, longitude, and latitude. There are also several temporal information such as duration (time frame of the event occurrence), begin (initial sighting of the event occurrence), and end (when the person leaves the location indicated). For the simplicity of the analysis, this study only considers the begin column, which will indicate when the person, who is later diagnosed with covid virus, first appears at the exact location. Each person with a case of covid-19 will be given an ID number. However, the ID is not unique. If the person reported to travel to several locations, then each location will be recorded separately as a new occurrence with the same id number. For the simplicity of this study, we group the dataset by the ID and only consider the first occurrence of the covid-19 event location and time with the particular ID.

2.2 Data Cleaning:

As mentioned in the previous section, we take in three main columns from the dataset for our point process model. Namely, longitude and latitude for our location, and the begin column

for the event time. A group by statement is employed to only keep the first occurrence of a person who contracted covid-19. Longitude and Latitude are jittered and scaled. Jittering is necessary due to the way that data was recorded. Assuming two friends with covid go to a restaurant together, then their data would be recorded such that the two cases have the exact same longitude, latitude, and time. This would cause the point process to be no longer a simple point process, and therefore caused a problem during model fitting. Each point was jittered by adding an uniform random variable between -1e-3 and 1e-3 to ensure that each point is still in relatively close proximity to the original location. Longitude was scaled by the process:

$$\frac{Longitude - min(longitude) - 0.01}{max(longitude) + 0.01 - min(longitude) - 0.01}$$

The ± 0.01 ensures that none of the points lie on the very edge of the observed region. Latitude was scaled in similar fashion.

To convert 'begin' to the desired format, as.numeric() function was used to convert date into time. With 0.0 being the time where the first event in our dataset occurred. Integer represents the day passed, while the fraction represents the fraction of the day when the event occurred. The time variable was jittered by adding a uniform random variable ranging between 0 to 600 seconds, which is ten minutes.

Appendix A.1 shows the code for grouping, jittering, and scaling the time and location of the data.

CHAPTER 3

Preliminary Analysis

3.1 Data Visualization:

To effectively visualize the data, the leaflet library was employed to overlay the points onto Taipei's map, shown in figure 3.1 [CSK24]..

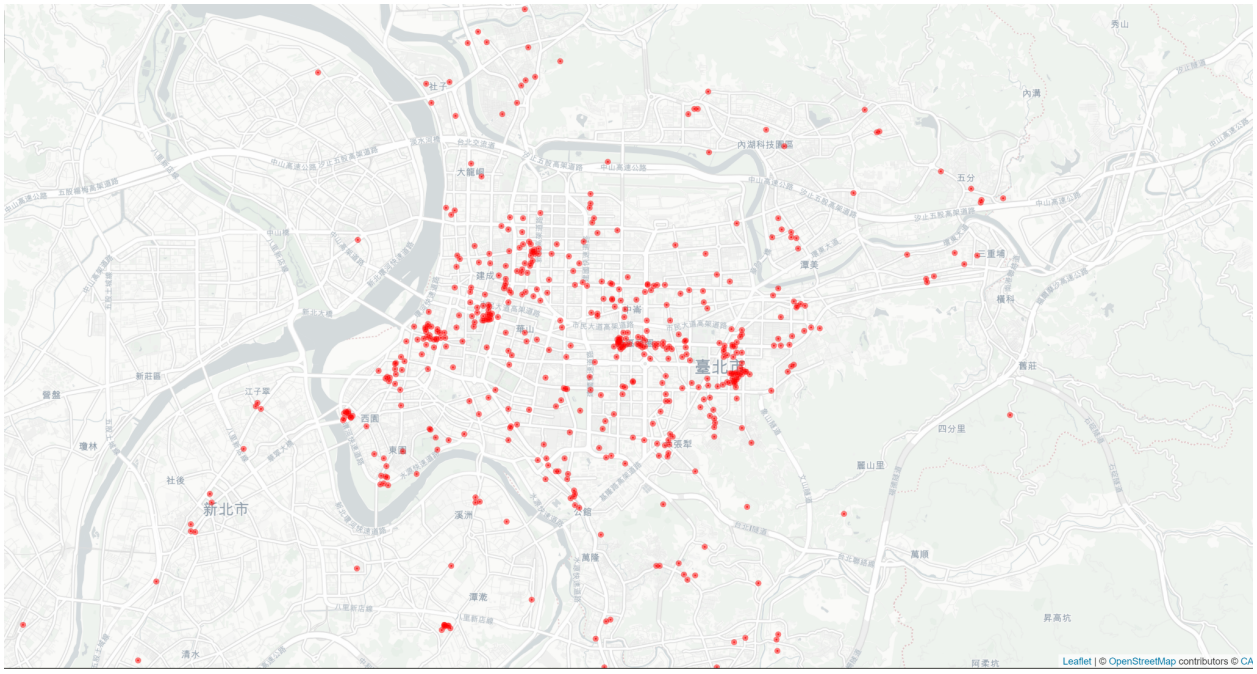

Figure 3.1: Overlaying data points onto city map

This visualization gives a better understanding of the covid spread in the Taipei city area. One can see that most points are clustered in the middle area, with very few points on the outskirt of the city. For obvious reasons, there are no points in the river, and almost zero points inside the mountain area, which is indicated by the darker region of the map.

3.1.1 Kernel Smoothing:

To better visualize the clustering of the points, kernel smoothing was employed in figure 3.2

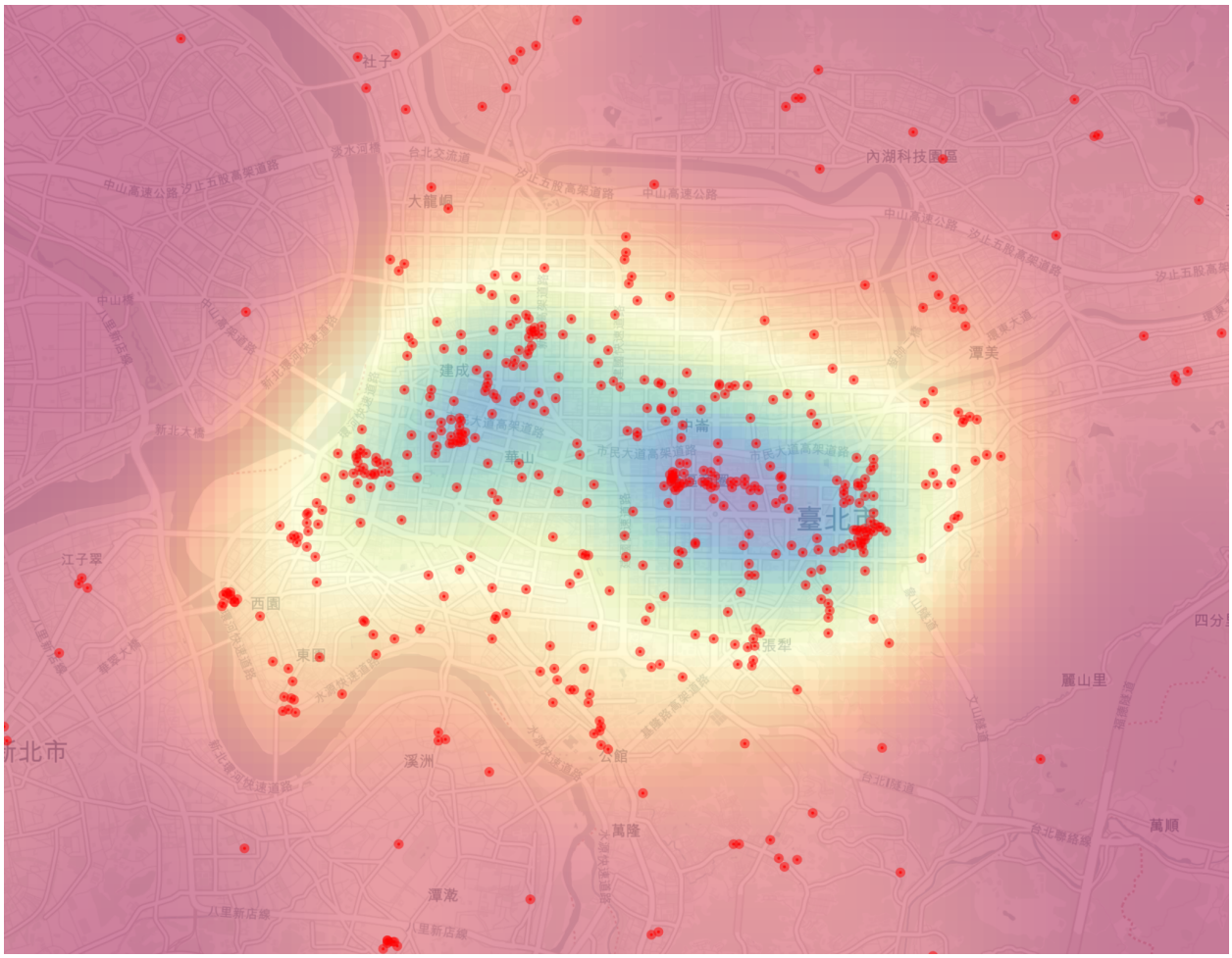

Figure 3.2: Overlaying kernel smoothing onto city map with data points

With this visualization, we can see that there are two main clusterings in the observed area. For context, the clustering on the left is centered at Taipei Main Station, the central station for Taipei's underground railway station, with numerous shops and activities around the area. The right clustering is centered at Taipei Eastern District, which gathers most of Taipei's high-end department stores and is close proximity to Taipei's financial district.

Code in Appendix A.2 shows how to overlay the points onto the city map with kernel smoothing.

3.2 Standard Functions and Interpretation of the Dataset:

We use standard functions to scrutinize the various aspects of our dataset.

3.2.1 F function:

The F function is also called the "empty space function". It is the cumulative distribution of the distances from a location to the nearest event. Here, a location is not necessarily an occurrence of an event, but rather an arbitrary location on the observational region. This location is typically an empty space, hence the name of the function. When there is more empty space in a region, the distance from a point to the nearest neighboring event is typically longer[PR21]. The F function is formally defined as:

$$\hat{F}(r) = \frac{(d_{ik} \leq r, \forall i)}{n} \tag{3.1}$$

where d_{ik} is the distance from location i to its nearest neighboring event at location k, and and n is the number of events in the observational region. The numerator with a parentheses is the number of location in i whose distance from k is less than or equal to r. Therefore, $F(r)$ is the probability that the distance from a randomly chosen location to its nearest point of the process is $\leq r$.

We would like to compare the F function of our point process to those of a spatially random Poisson Process:

$$F_{pois}(x) = 1 - exp(-\lambda \pi x^2) \tag{3.2}$$

The F-function in Figure 3.3 of the data looks to be under the theoretical line of the Poisson Process. This indicates that there is an over-dispersion in our dataset. In the context of our study, an over-dispersion of the points indicate that Covid-19 cases in Taipei are more spread out than expected by chance. This could indicate that the government's policy with social distancing, mask wearing, quarantine measures, and other numerous interventions were effective in letting the virus spread freely. This interpretation, however, likely only

applies to the outskirts of the city, given the obvious clustering around the Taipei Main Station and the Taipei Eastern District.

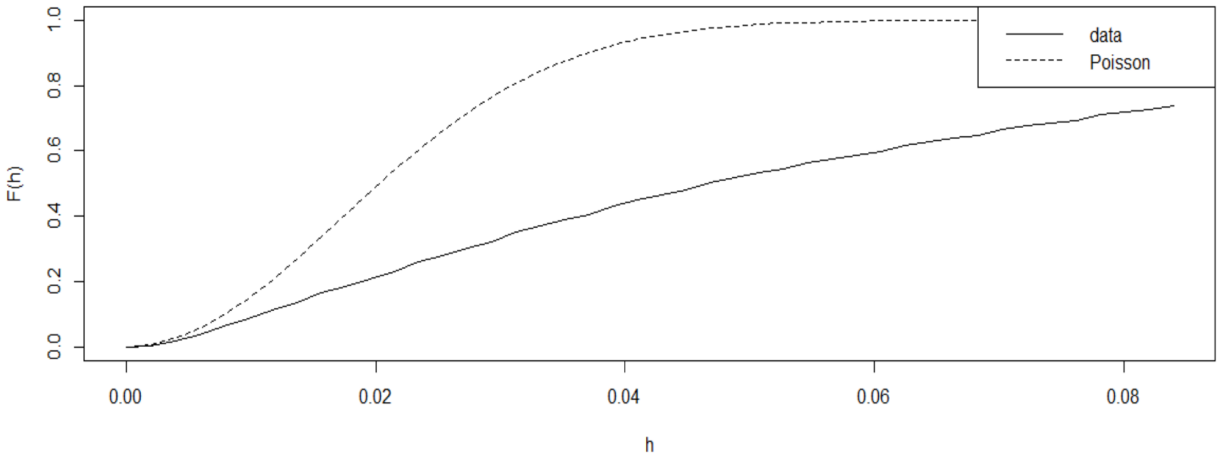

Figure 3.3: F function of the dataset compared to the theoretical Poisson Process

3.2.2 G function:

The G function is also known as the "nearest neighbor distribution function". It is the cumulative distribution function of distances from event to nearest neighbor[PR21]. The G function is defined the same way as the F function where:

$$\hat{G}(r) = \frac{(d_{ik} \leq r, \forall i)}{n} \tag{3.3}$$

The only difference being that i and k are both events in our data. The G function of a Poisson Process is also:

$$G_{pois}(x) = 1 - exp(-\lambda \pi x^2) \tag{3.4}$$

When the empirical $\hat{G}(x)$ is greater than the theoretical Poisson Process G function, that means the data is clustered.

The G function in Figure 3.4 lies above the theoretical line of the Poisson Process. This suggests that the points are more clustered than would be expected than a randomly distributed set of points. The nearest neighbor distances are shorter than expected, meaning that there is a tendency for cases to occur in close proximity to each other. This interpretation would support what was observed in the Kernel Smoothing graph, and likely only applies to the increased local density around the Taipei Main Station and Taipei Eastern District.

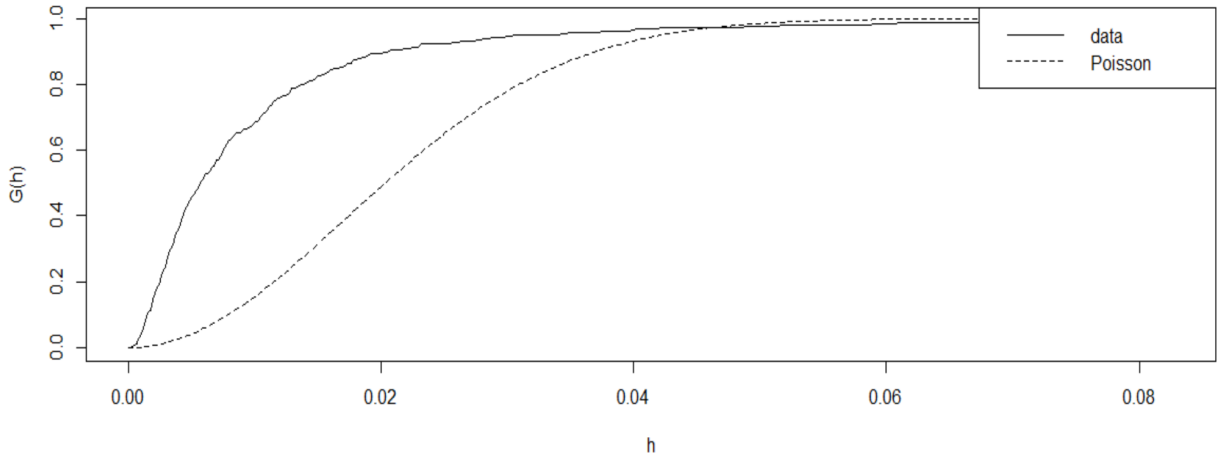

Figure 3.4: G function of the dataset compared to the theoretical Poisson Process

3.2.3 J function:

The J function utilizes the fact that both G and F function of a random Poisson Process being $1 - exp(-\lambda\pi r^2)$. The J function is formally defined as:

$$J(r) = \frac{(1 - G(r))}{(1 - F(r))} \tag{3.5}$$

By taking the ratio between the two functions, a random Poisson Process should have its J function value to be around 1 throughout r. If $J > 1$, then that means the data is dispersed, and if $J < 1$, that means the data is clustered.

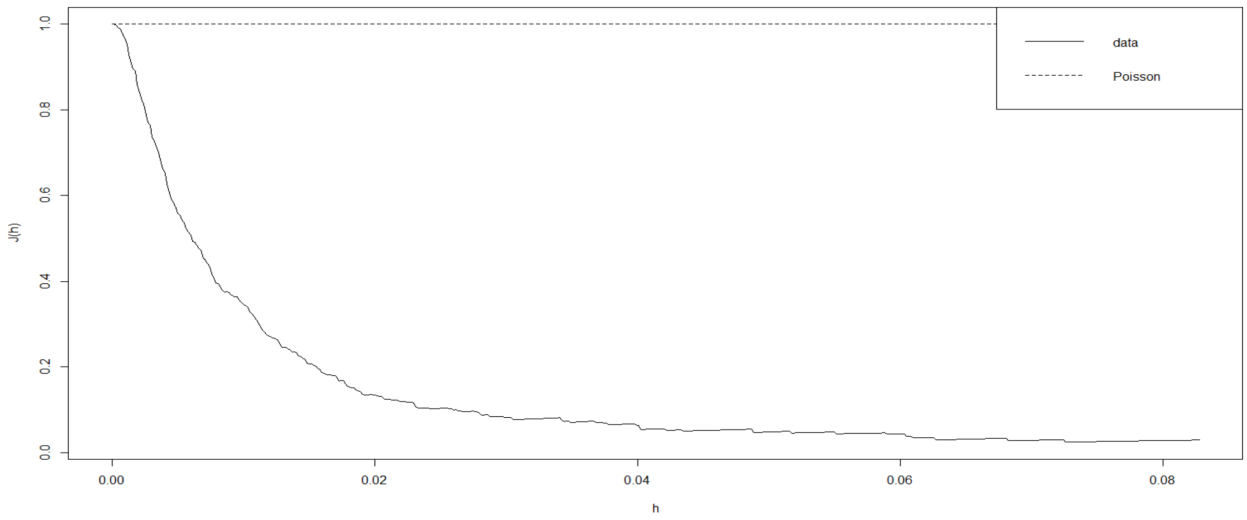

Figure 3.5: J function of the dataset compared to the theoretical Poisson Process

The J function in Figure 3.5 compares the G and F function. With the J function lying below the theoretical line of the Poisson Process, it suggests an over-dispersion of the points despite local clustering. One can interpret this as the effectiveness of government protocols that were in effect during the time.

3.2.4 K Function:

K function is also called "Ripley's K-function". G and F function are only able to detect pattern at a single scale. The K function, however, is able to detect patterns at multiple scales [Rip76]. The theoretical values for the K function for a Poisson Process is:

$$K_{pois}(x) = \pi x^2 \tag{3.6}$$

When the empirical function is greater than the theoretical function, then that will indicate clustering at that scale. On the other hand, if the empirical function is less than the theoretical function, that will suggest dispersion.

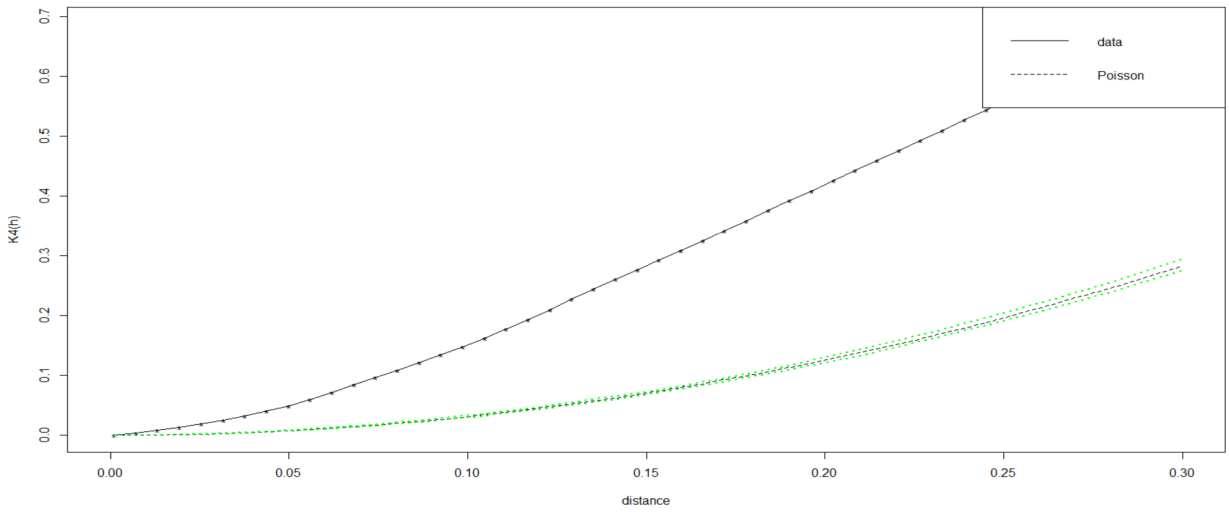

Figure 3.6: K function of the dataset compared to the theoretical Poisson Process

The K Function in Figure 3.6 lies above the theoretical line of the Poisson Process. This would indicate there are significant clustering of cases. This would support the clustering that was observed in the two areas of Taipei Main Station and Taipei Eastern District.

3.2.5 L Function

The L function is an alternative to the K function with edge correction. With the K function, a Poisson Process in R^2 is:

$$K(r) = \pi r^2 \tag{3.7}$$

The L function alters the function by:

$$L(r) = \sqrt{(K(r)/\pi} \tag{3.8}$$

The resulting L function will simply be:

$$L(r) = r \tag{3.9}$$

Then we can graph $L(r) - r$, which should be approximately 0 if the empirical L function is roughly stationary Poisson Process.

The L function in figure 3.7 shows a strong clustering at a short distance. We can see that by the rapid increase in the L-h function, there are many cases occurring in close proximity, which is likely within a few blocks. The function levels off as the distance gets larger, this indicates that the clustering is not as obvious as the distance between cases increases. This makes intuitive sense as the virus is difficult to transmit without close contact, and a person carrying the virus can only travel so far to spread the virus within a certain time frame.

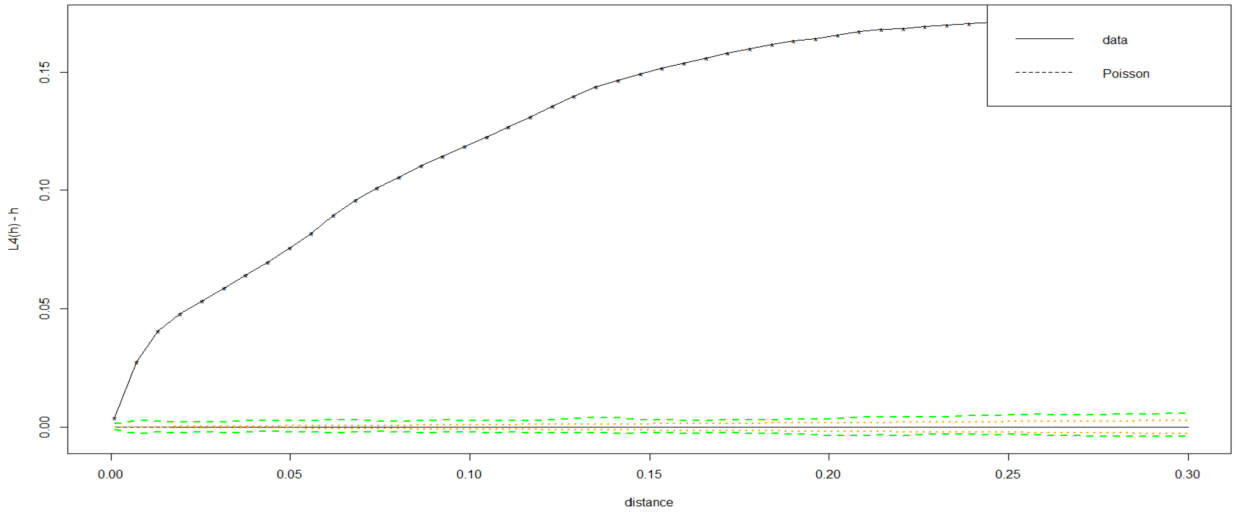

Figure 3.7: L-h function of the dataset compared to the theoretical Poisson Process

Appendix A.3 shows the code for how to draw all the standard functions mentioned in this chapter.

CHAPTER 4

Methodology

4.1 Hawkes Models:

Hawkes Process is a self-exciting point process model that is used to model clustered point patters in applications such as seismology, finance, crime, and infectious diseases. [spr03][Rei18][Oga88] [CBD12][KSM22b] with the conditional intensity (lambda function):

$$\lambda(t,s) = \mu(s) + K \int_{t'=0}^{t} \int_{s'} g(t-t^{'}, s-s^{'})\ dN(t^{'}, s^{'}) \tag{4.1}$$

$$= \mu(s) + K \sum_{\{t',s':t'<t\}} g(t-t^{'}, s-s^{'}) \tag{4.2}$$

where $s \in X \subseteq \mathbb{R}^2$ and $t \in [0,T)$, where $\lambda(s,t|\mathcal{H}_t)$ is the conditional rate at which points or events are expected to accumulate around a spatial-temporal location (s,t), given information $\mathcal{H}_t$ on all previous events. [KSM22b] By integrating all points t that occur before $t^{'}$

g: triggering function or triggering density that takes into account the past event and its effect on the occurrence of new events. The triggering density is usually a decaying function, with more recent events having more effect on the triggering of new events. This function is typically assumed to be a nonnegative and to integrate to 1 over all time and space[KSM22b].

K: productivity of the function. Provided that g is a density function, K is the expected number of points triggered directly by each point[KSM22b].

μ: background intensity or baseline rate of the process. This rate is independent of any previous events, and it sets a lower bound for the process's minimum intensity to ensure that

there is always some chance of the event occurring. Each background point associated with $\mu(s)$, is expected to generate $K + K^2 + K^3 + ... = 1/(1-K) - 1 = K/(1-K)$ triggered points. As a result, in the Hawkes process, the expected fraction of background points is $1 - K$[KSM22b].

With a dataset consisting of n points within a space-time observation region B, maximum likelihood estimation (MLE) is commonly used to fit a Hawkes Process. One can obtain the parameters estimates $\hat{\Theta}$ of the Hawkes process by maximizing the loglikelihood:

$$L(\Theta) = \sum_{i=1}^{n} log(\lambda_\theta(s_i, t_i,)) - \int_B \lambda_\theta(s,t)dtds \tag{4.3}$$

The integral term in the loglikelihood for Hawkes Process can be estimated:

$$\int_0^T \int\int \lambda(t,x,y)dxdydt = \int_0^T \int\int \mu\rho(x,y)dxdydt \tag{4.4}$$

$$+ \int_0^T \int\int K \sum_{i:t_i<t} g(t-t_i, x-x_i, y-y_i)dxdydt \tag{4.5}$$

$$= \mu T + \int_0^T \int\int K \int_B 1_{\{t'<t\}} g(t-t', x-x', y-y')dN(t',x',y')dxdydt \tag{4.6}$$

Next we interchange the integrals to get

$$= \mu T + K \int_B \int_0^T \int\int 1_{\{t'<t\}} g(t-t', x-x', y-y')dxdydtdN(t',x',y') \tag{4.7}$$

next we change the coordinates, letting $u = t - t'$, $v = x - x'$, $w = y - y'$,

$$= \mu T + K \int_B \int_0^{T-t'} \int\int_{S-(x',y')} g(u,v,w)dudvdwdN(t',x',y') \tag{4.8}$$

$$\sim \mu T + K \int_B (1)dN(t',x',y') \tag{4.9}$$

$$= \mu T + KN(B) \tag{4.10}$$

This nice simplification of the integral term allow us shown in equation 4.10 simplify the calculation for us when dealing with calculating the log-likelihood of the lambda function.

4.1.1 Properties of Hawkes Model:

There are a few nice properties of the estimated $\hat{\Theta}$. It had been shown that the MLE $\hat{\Theta}$ is asymptotically unbiased, consistent, normal, and efficient. The standard errors can also be easily constructed using the diagonal elements of the inverse of the Hessian of log-likelihood evaluated at $\hat{\Theta}$ [Oga78].

4.2 Model Configuration:

For the triggering function, there are the time portion and location portion of the function.

4.2.1 Temporal Distribution of Triggering Function:

Common choices for the time portion of the triggering function g are exponential and Pareto distribution[Rei18]. In the case of Hawkes Model triggering density, the exponential time will take the form:

$$g_1(t - t') = \alpha e^{-\alpha(t-t')} \tag{4.11}$$

The exponential distribution in time inherently assumes that the probability of event occurrence strictly increases as the time difference between the current event and previous event decreases. This narrative, however, contradict with the idea of incubation time of a virus.

The Pareto distribution [Tre96] has the PDF form:

$$g_1(t - t') = \frac{\alpha(t - t')_m^{\alpha}}{(t - t')^{\alpha+1}} \tag{4.12}$$

One characteristic of the Pareto Distribution is its heavy tail. This can be a problem for estimating incubation period and overestimate the probability of the virus's infection after a long period of time.

For this paper, we only consider single variate normal distribution, or Gaussian distribution for time[spr03]. In our study, the distribution takes the form:

$$g_1(t-t') = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{1}{2}(\frac{(t-t')-\mu}{\sigma})^2} \tag{4.13}$$

Intuitively, the Univariate Normal distribution in time can offer a few advantage in epidemiology modeling over exponential and Pareto distribution. Firstly, Normal distribution has a central peak, indicating that most infections cluster around a central value with fewer infections the further away we move from the mean. This could potentially mimic the natural variability of the incubation time, providing that a virus variant has a unimodal incubation time. Without a heavy tail, normal distribution would not suggest a significant probability of extremely long incubation period. Thirdly, the normal distribution is easy to work with and interpret.

However, with some of the previous studies conducted specifically on the distribution period of incubation time, it had been shown that most infectious diseases follow a right-skewed distribution [LRB09]. This would mean that using a symmetric distribution such as the normal distribution will potentially yield underestimate the right-hand tail of the incubation period[XWM21].

With these information in mind, this study will still use the univariate normal distribution for the time portion of the triggering density. For Normal time, there are two parameters to consider, namely, the mean and standard deviation of the normal distribution.

4.2.2 Spatial Distribution of Triggering Function:

For this paper, we tried implementing Gaussian distribution and Exponential distribution for space. With Gaussian distribution, since we have longitude and latitude, we can use a bivariate normal distribution that takes the form:

$$g_2(x_d, y_d) = \frac{1}{2\pi\sigma_{xd}\sigma_{yd}\sqrt{1-\rho^2}} e^{\left(-\frac{1}{2(1-\rho^2)}\left(\frac{(x_d-\mu_{x_d})^2}{\sigma_{x_d}^2}+\frac{(y_d-\mu_{y_d})^2}{\sigma_{y_d}^2}-\frac{2\rho(x_d-\mu_{x_d})(y_d-\mu_{y_d})}{\sigma_{x_d}\sigma_{y_d}}\right)\right)} \tag{4.14}$$

- $x_d = x - x'$ and $y_d = y - y'$ represents the difference between longitude and latitude of two points
- μ_{x_d} and μ_{y_d} are the means of x_d and y_d respectively,
- σ_{x_d} and σ_{y_d} are the standard deviations of x_d and y_d respectively,
- ρ is the correlation coefficient between x_d and y_d.

The Gaussian Distribution is still relatively easy to understand. With the five parameters:μ_x, μ_y, sd_x, sd_y, and ρ, we can hopefully uncover some interesting property of transmission pattern of the virus in Taipei.

This study also uses an exponential distribution in distance for the spatial part of the triggering density. The exponential distribution here takes the form:

$$g(x_d, y_d) = \frac{\alpha}{\pi} \exp(-\alpha r^2), \text{ where } r^2 = x_d^2 + y_d^2 \tag{4.15}$$

The π is added in the denominator of the coefficient to ensure that the density integrate to 1.

Using the exponential distribution of the distance between the two points, we try to fit the α parameter to see essentially how fast the triggering density decay as we move away from a previously occuring point.

With these information in mind, let's take a look at the three models that we are going to fit.

4.2.3 Model 1: Normal Time * Bivariate Normal Location (norm2d)

The first model to consider is the normal time with dnorm function [R C23] paired with Bivariate Normal Location using the dnorm2d function from fCopulae R library [WSC23]. The dnorm2d takes the x, y, and rho and calculates the probability density of the two points given the correlation. In our case, we input the difference between an event an previous occurrence of a point, namely:

$$dnorm2d(x - x', y - y', \rho)$$

And try to fit ρ to find the spatial correlation between two events. Therefore the conditional intensity being fitted here looks like:

$$\lambda(t, x, y) = \mu(x, y) + K \sum g_1(t - t') * g_2(x - x', y - y') \tag{4.16}$$

where $g_1(t - t')$ and $g_2(x - x', y - y')$ are equation 4.13 and 4.14, respectively.

4.2.4 Model 2: Normal Time * Bivariate Normal Location (bvnorm.pdf)

The second model is essentially the same as the first, with a slight variation with the function being used. Here we use the bvnorm.pdf function from the nnspat library [Cey23]. Bvnorm.pdf also takes into account the μ_x, μ_y, sd_x, sd_y in addition to ρ. This model is used in an attempt to see if we can derive more spatial information of the COVID-19 transmission.

The full conditional intensity takes the same formation as equation 4.16.

4.2.5 Model 3: Normal Time * Exponential Decay Distance

The third and last model uses exponential decaying distance for the spatial portion of the triggering function. This is a more traditional approach for a Hawkes process. We will have an alpha parameter that represents the spatial decaying parameter.

The full conditional intensity have the form:

$$\lambda(t, x, y) = \mu(x, y) + K \sum g_1(t - t') * g_2(x - x', y - y') \tag{4.17}$$

$$= \mu(x, y) + K \sum \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{1}{2}(\frac{(t-t')-\mu}{\sigma})^2} \tag{4.18}$$

$$* \frac{\alpha}{\pi} \exp(-\alpha r^2) \tag{4.19}$$

where $r^2 = (x - x')^2 + (y - y')^2$

Appendix A.4 shows the code for calculator of negative log-likelihood of the three functions

CHAPTER 5

Model Fitting and Evaluation

5.1 Model fitting:

For the model fitting process, we first set the initial values for all the parameters that need to be fitted for each model. The number of parameters varies depending on the triggering functions. The optim() function[R C23] is run iteratively for three or eight times (depending on the model) to fit the parameters with maxit set to 200. Each time the optim function ends, we take the end value of the previous iteration and use it as the initial parameters for the next iteration.

We use the pseudo-Maximum likelihood estimation for Hawkes process with the form mentioned by equation 4.3:

Where for a Hawkes process, the integral part is estimated with:

$$\mu T + KN(B)$$

which is shown to be a good approximation in equations 4.4-4.10. Therefore the actual pseudo log-likelihood estimation takes the form:

$$L(\Theta) = \sum_{i=1}^{n} log(\lambda_\theta(s_i, t_i,)) - (\mu T + KN(B)) \tag{5.1}$$

Here are the steps for model fitting:

1. Set initial values for all parameters

2. Setup the f function to calculate negative pseudo log-likelihood by incorporating lambda function in equation 4.16 and 4.19 into the log-likelihood in equation 5.1

3. Setup the optim() function in R to optimize the f function from step 2

4. Run the optimization for n times, with n being an arbitrary number of iteration to see if the parameters have been properly optimized.

5. Obtain the parameters from optim()

The code in appendix A.5 shows how to find the parameters for the pseudo log-likelihood using the optim function for the first model. The code for the other two models are similar, where the only difference is the actual function called and the parameters being fitted.

5.2 Model Evaluation:

5.2.1 Superthinning:

Superthinning is a way of evaluating the fitted conditional intensity, and seeing whether the fitted model accurately represents the dataset. By combining thinning and superposition techniques, it had been shown that the new process is homogeneous Poisson process if and only if the fitted model is correct[CSV12]. This technique had also been applied to assessment of parametric space-time point process model for California Earthquakes [CSV12]. The exact process of superthinning is as followed:

1. Obtain a conditional intensity λ of a model through a fitting process.

2. Setting a tuning parameter k to control the magnitude of thinning and superposition.

3. Superposing points where the intensity is lower than some constant $c \sim mean(\hat{\lambda})$ with the rate $c - \hat{\lambda}(t, x, y)$.

4. Thinning points with the probability $c/\hat{\lambda}(t_i, x_i, y_i)$ where the intensity is higher than c.

The end result of the superthinned points of sufficiently fitted model parameters should look roughly uniformly distributed throughout the observed area.

Code in Appendix A.6 shows the function for superthinning.

5.2.2 Residual G and F functions:

Using the Standard functions that were used in the previous chapter, we can again assess whether the residual resembles a theoretical Poisson Process by seeing how the G function and F function deviates from the theoretical Poisson Process. Significant deviation will indicate whether our superthinned points are clustered or dispersed. By comparing the G and F function, we can assess which model fits the dataset the best.

5.2.3 Model evaluation results

One intuitive way of assessing whether or not the models are properly fitted is by looking at the deviation of the fitted values from the initial parameter values. If the values does not look to deviate significantly from the initial values, then we should be cautious with interpretation of the final fitted values, as there might be other factors that effect the fitting process. In fact, this phenomenon showed up during the fitting of our second model, where a few of the parameters are very close to their initial values, and those values are significantly different from parameters of similar qualities from the other two models.

To assess which model fits the best, this study will compare the results of the superthinning points and the G/F functions. With the superthinning points, we can visualize how well the models are fitted. With two models that are both well fitted, we can then compare their G/F function and compare how closely the superthinned points' G/F function adhere to that of a Theoretical Poisson Point Process.

CHAPTER 6

Results

6.1 Model 1:Normal Time * Bivariate Normal Location (norm2d)

Looking at the results of the parameters in table 6.1, we can see that all five parameters are significantly different from their initial value after three iterations of optimization. Had the process only taken place over one iteration, the mean of time difference c would not have been that far from the initial value. Also note that most of these parameters are not optimized in a strictly increasing or decreasing way. For example, k was set to 0.35 during the first iteration, but decreased to 0.34 on the second iteration. On the third iteration it was again increased to 0.67. This phenomenon is apparent in almost all the parameters in the model fitting process.

	μ	k	c	p	ρ
Initial Parameter values	1.000000	0.3000000	15.000000	8.0000000	0.9000000
n=1	1.169119	0.3538265	12.711482	11.466967	0.9999553
n=2	1.173273	0.3494587	1.513158	0.6056333	0.9998395
n=3	0.503452	0.6724181	1.623128	0.7116683	0.9996001

Table 6.1: Parameter result for model 1

6.2 Interpretation:

To interpret these parameters. First see that μ=0.5, meaning that per unit time (day) we would expect roughly 0.5 event occurrence independent of any affect from previous events. Given that Covid-19 is strictly a transmitted disease, one might assume that the occurrence

is due to someone traveling from outside of the observation region, from another city or country, or was simply transmitted by someone who was not previously recorded in our dataset.

K = 0.67 indicates that for every event in our dataset, there are on average 0.67 new events directly triggered. This means that the process is not explosive and is rather stable.

C = 1.623 and P=0.71 indicates that for every previous event, the peak influence on subsequent event occurrence on average is around 1.623 days after the previous event with an standard deviation of 0.7 days. One might be tempted to interpret this value as the incubation period of the virus. However, for the COVID-19 virus, the incubation period is around 2-14 days[BKW20] with a mean incubation period of 4.5 days. This actually fits what was discussed earlier in chapter 4, where we talked about the possibility of underestimating the incubation period when using normal distribution for time.

ρ = 0.99 means there is almost a perfect correlation between longitude and latitude change. In the context of this model, this could mean that there is a very specific pattern for the way in which the disease spreads. However, such a high correlation should be carefully scrutinized before making a conclusive statement.

	Pseudo negative log-likelihood
Initial	368.61452
n=1	242.21108
n=2	13.85291
n=3	-65.05902

Table 6.2: Pseudo negative log-likelihood for each iteration in model 1

Notice here that the pseudo negative log-likelihood in table 6.2 of each iteration is strictly decreasing. This means that each time, our optim function makes progress in optimizing the parameters.

6.3 Residual Analysis:

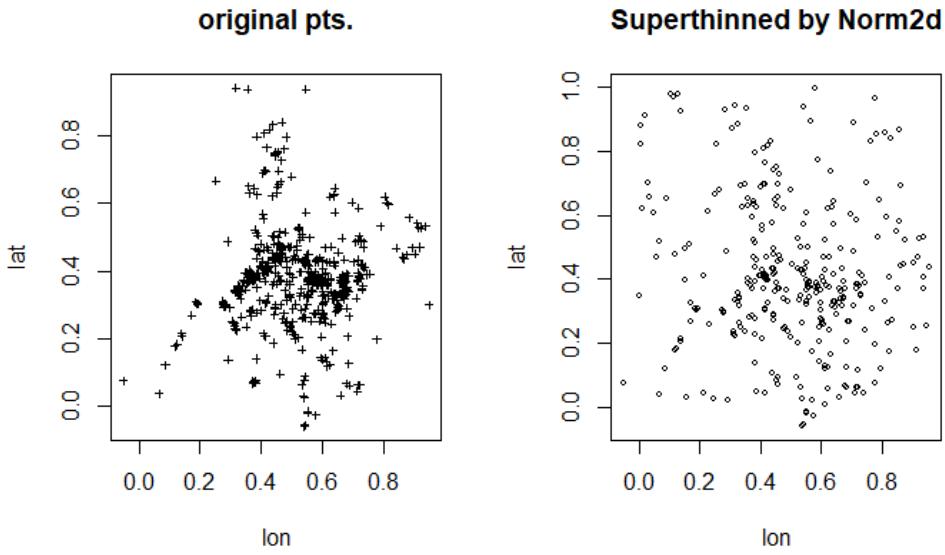

Figure 6.1: Original points (left) vs. Superthinned points of model 1 (right)

Looking at the superthinned points on the right of figure 6.1, the points seem roughly uniformly distributed, with some slight clustering in the middle that were not completely thinned out.

The F function of the superthinned points in figure 6.2 has a way better adherence to the theoretical poisson process. Being slightly under the theoretical line means that there are still dispersions in our dataset. This means that the lambda function is overestimating the rate of the events in places where there should be very little to no event occurrence.

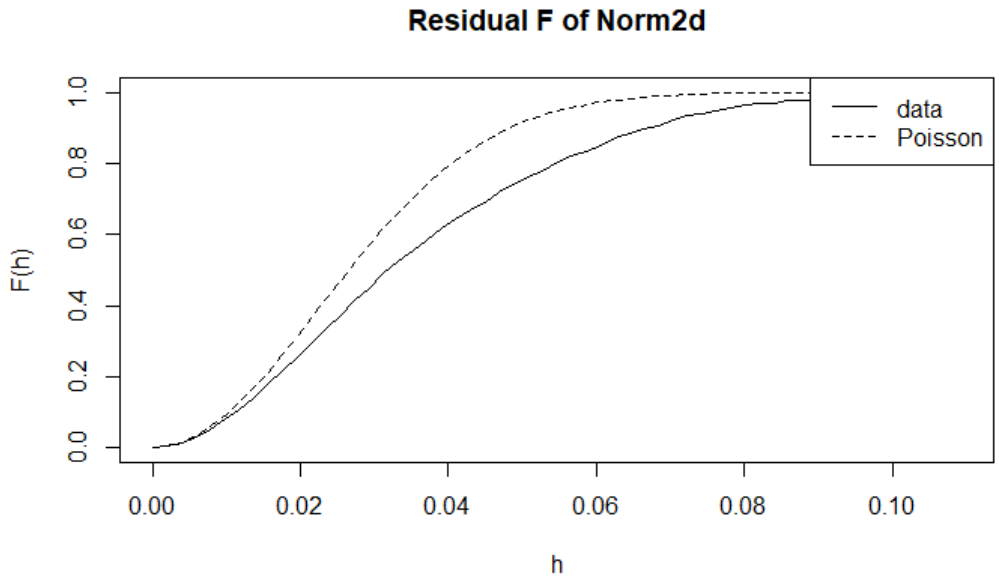

Figure 6.2: F function of superthinned points for model 1

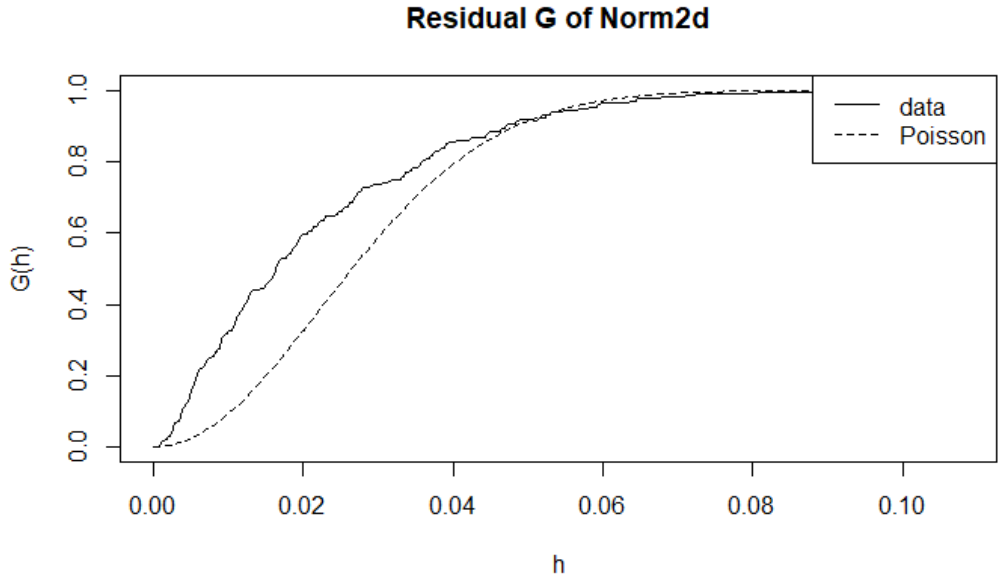

Figure 6.3: G function of superthinned points of model 1

The G function in figure 6.3 is a little above the theoretical poisson process at h=0 to h=0.04. This means that the lambda function is underestimating how cluster the points are at very close proximities. This could be due to the fact that we jittered the points by a very small amount during the data cleaning process.

6.4 Model 2: Normal Time * Bivariate Normal Location (bvnorm.pdf)

Table 6.3 shows the parameters values after the optimization. The code intentionally ran the optimization process for eight iterations, which is five more than what was run on the other two models. This is to ensure that all parameters are properly optimized. Looking at the pseudo negative log-likelihood in table 6.4, we can see that the negative log-likelihood does not further decrease after the third iteration. But if we look at the actual fitted values, we can see that many of the parameters value does not deviate much from the initial setting.

	μ	**k**	**c**	**p**	μ_x	μ_y	**sd_x**	**sd_y**	ρ
Initial Values	0.500000	0.500000	15.00000	5.000000	1.0000000	1.000000	1.000000	1.000000	0.5000000
n=1	1.522447	0.00002316394	15.04875	5.474872	0.9253758	1.043929	0.7840481	1.113817	0.6545615
n=2	1.522024	0.00000001768	15.10169	6.354985	0.8819344	1.030111	0.9249959	1.119730	0.6546220
n=3	1.522024	0.00000000203	15.15563	6.408920	0.9358691	1.084046	0.9789306	1.173665	0.6546220
n=4	1.522024	0.00000000203	15.15563	6.787811	0.9358691	1.084046	0.9789306	1.173665	0.6546220
n=5	1.522024	0.00000000203	15.15563	6.787811	0.9358691	1.084046	0.9789306	1.173665	0.6546220
n=6	1.522024	0.00000000203	15.15563	6.787811	0.9358691	1.084046	0.9789306	1.173665	0.6546220
n=7	1.522024	0.00000000203	15.15563	6.787811	0.9358691	1.084046	0.9789306	1.173665	0.6546220
n=8	1.522024	0.00000000203	15.15563	6.787811	0.9358691	1.084046	0.9789306	1.173665	0.6546220

Table 6.3: Parameter results for model 2

6.5 Interpretation:

μ = 1.52 suggests that everyday there are around 1.5 new cases that are independently occurring without any effect from other previous events. With productivity k being close to zero, this model also suggests that almost all COVID-19 cases are results of transmission from outside of the city. C =15 and p = 6.8 suggests that the virus has a peak influence on average of two weeks after an event, with a standard deviation of a week time. μ_x and μ_y being close to one should be interpreted with caution. With the longitude and latitude scaled during the data cleaning process, μ_x and μ_y being so close to 1 would suggests that one event would influence spaces outside of our observational region. This phenomenon is likely due to the fact that our values were not properly fitted. sd_x and sd_y also being close

to one would also suggest that there is a huge variation in the influence of each event. ρ = 0.65 indicates that there some positive correlation between shifts in longitude and shift in latitude. This could reflect that there is a directional trend in how events influence each other spatially.

	Pseudo Negative Log-likelihood
Initial	736.3455
n=1	308.5196
n=2	308.5103
n=3	308.5103
n=4	308.5103
n=5	308.5103
n=6	308.5103
n=7	308.5103
n=8	308.5103

Table 6.4: Pseudo negative log-likelihood values of model 2

6.6 Residual Analysis:

Right side of figure 6.4 shows the plotted superthinned points by the model. It is clear that there is still significant clustering in the middle of the plot. Other than the clustering, the rest of the plot looks roughly uniformly distributed.

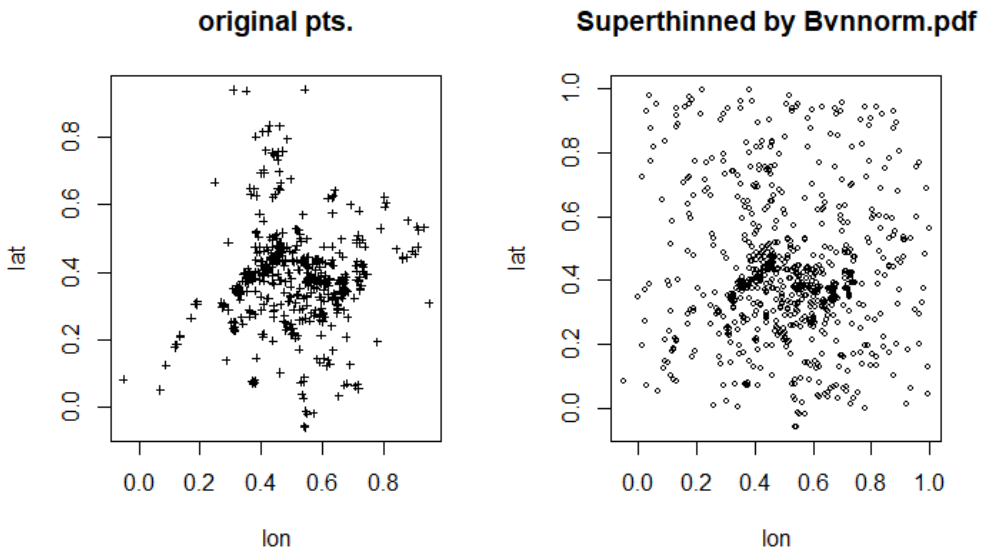

Figure 6.4: Original Points (left) vs. Plotted Superthinned points of model 2 (right)

Looking at the F function of the superthinned points in figure 6.5, we can see a significant difference between the data and the theoretical Poisson Process. The lambda function is overestimating the rate of event occurrence, therefore not thinning enough points out of the dataset.

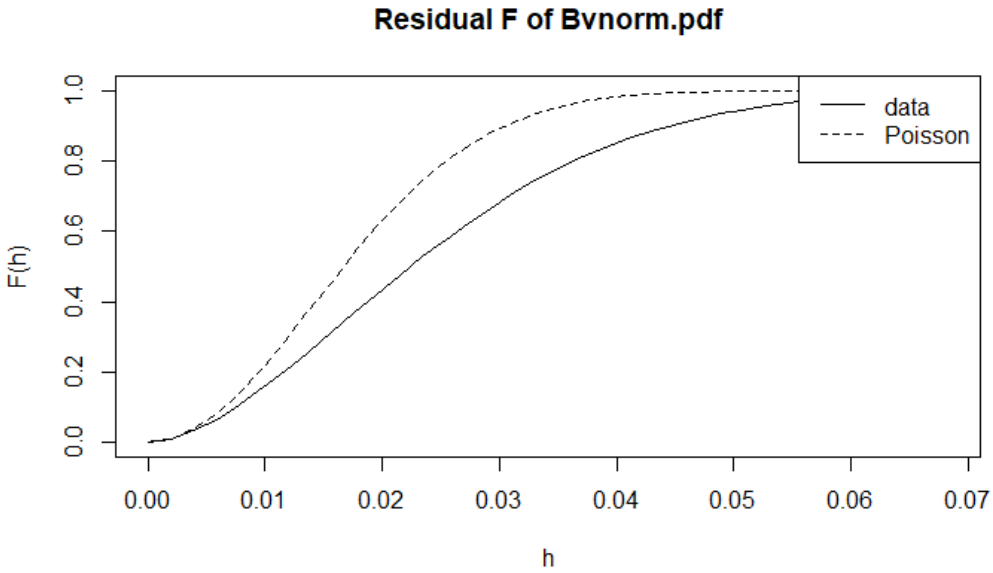

Figure 6.5: F Function of superthinned points of Model 2

With the G function in figure 6.6, we see a different problem. Here the G function of the

superthinned points is underestimating the rate of events at close proximity, meaning there are significant clustering happening at close proximity. As the distance gets beyond 0.028, the G function dips below the theoretical poisson process line, and showing dispersion at larger distance.

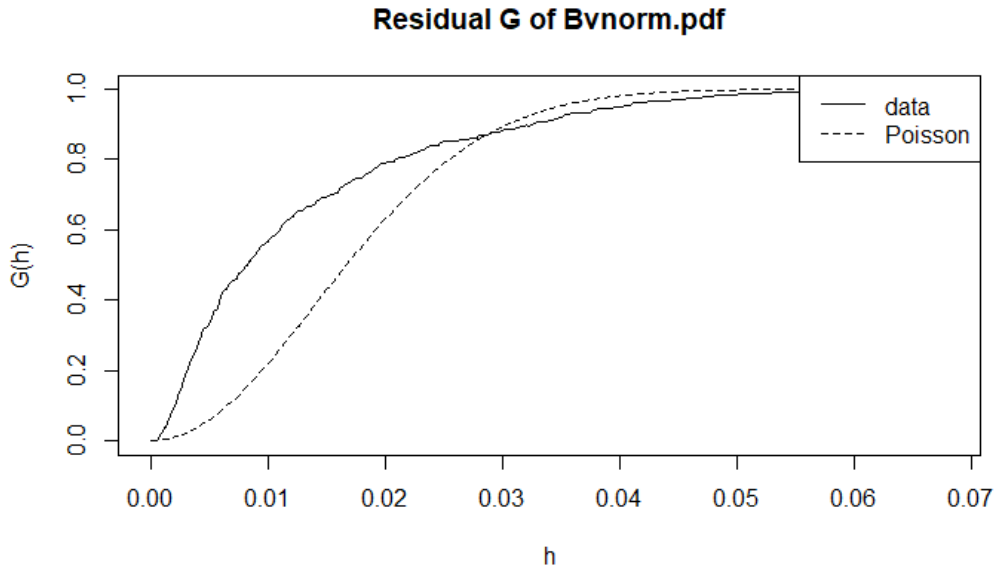

Figure 6.6: G function of Superthinned points of model 2

6.7 Model 3: Normal Time * Exponential Distance

Table 6.5 shows the parameters of the parameters values after optimization. Here we see a similar phenomenon where the values will quickly increase or decrease to a value after just one iteration, and slowly fine tuned to optimized value for the next two iterations.

	μ	k	c	p	α
Initial value	1.00000000	0.3000000	15.000000	8.000000	0.90000
n=1	0.11881655	0.9998139	3.895947	1.805284	32.41264
n=2	0.06041692	0.9821356	2.136727	1.119980	46.98395
n=3	0.14057316	0.9598938	2.298605	1.078206	66.64522

Table 6.5: Parameters values of model 3

6.8 Interpretations of Model Values:

μ is tuned to 0.14, meaning that the background rate of covid is merely 0.14 new cases per day. This is quite a different result from our first model, which implied that there are almost 0.5 cases per day. On the other hand, the productivity (k) is 0.96, close to 1. This suggests that each case of covid directly contributes to almost 1 new case in Taipei. c=2.29 and p = 1.07 suggests that for every previous event, the peak influence on subsequent event occurrence on average is around 2.3 days after the previous event with a standard deviation of 1 day. Lastly, it is important to look at the α value, which represents the exponential decay of distance between event occurrences. $\alpha = 66.65$ suggests a very rapid decrease in spatial influence with distance. This makes sense in that covid transmission happens only when two people are in close proximity, and that effective social distancing can greatly reduce the risk of transmission.

	Pseudo Negative Log-likelihood
Initial	383.8642
n=1	-827.0045
n=2	-894.4675
n=3	-923.4462

Table 6.6: Pseudo log-likelihood values of model 3
shows a continuously decreasing value in negative pseudo log-likelihood.

6.9 Residual Analysis:

The plot on the right side of figure 6.7 shows the superthinned points with the Lambda Function for Model 3. The superthinned points show a roughly uniform distribution throughout the observed region. There are some slight overlapping points in very few spots.

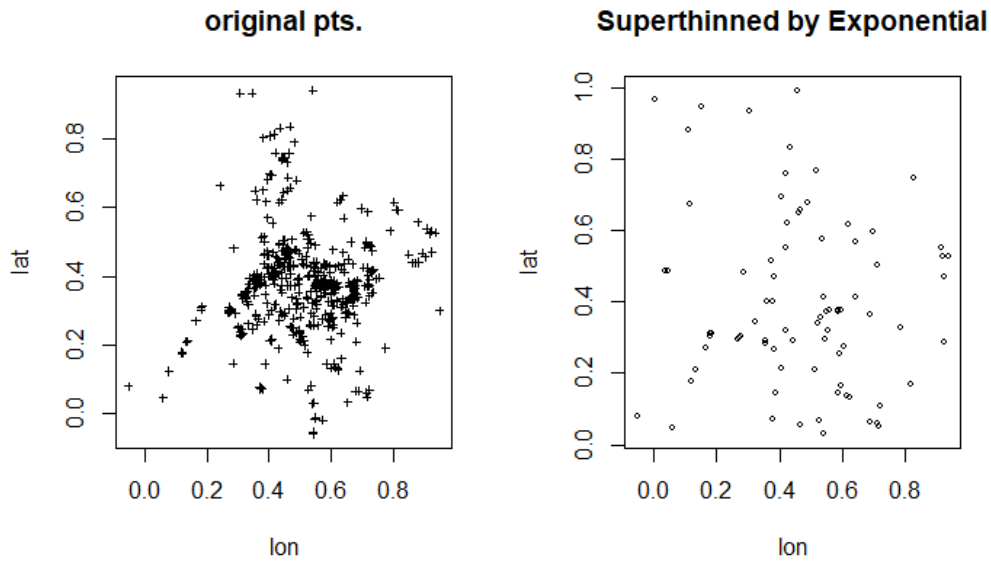

Figure 6.7: Original points (left) vs. Plotted Superthinned points of model 3 (right)

The F function in figure 6.8 shows that the superthinned points are very close to a Poisson Process, with a slight dispersion/overestimation in the lambda function around h = 0.07.

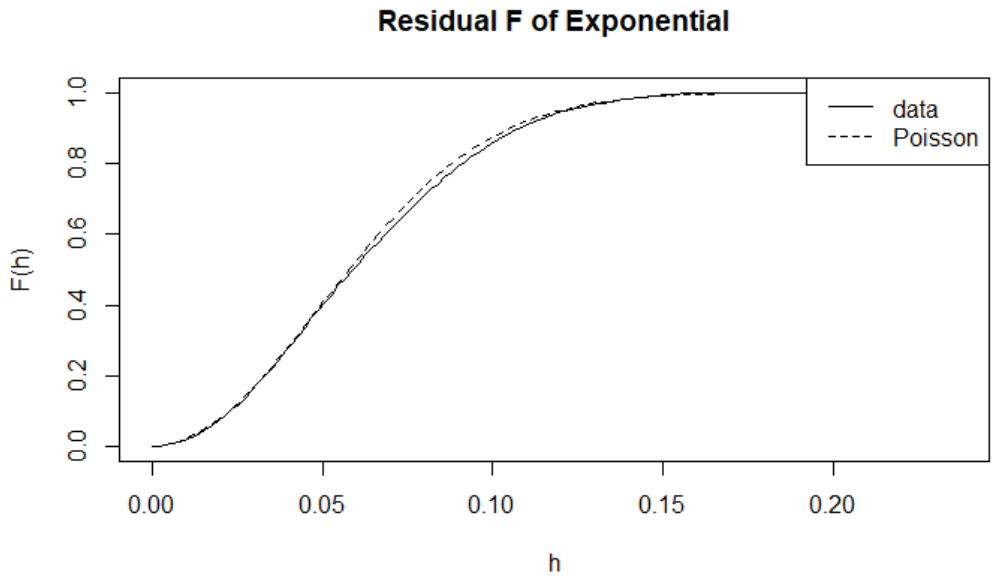

Figure 6.8: F function of superthinned points of model 3

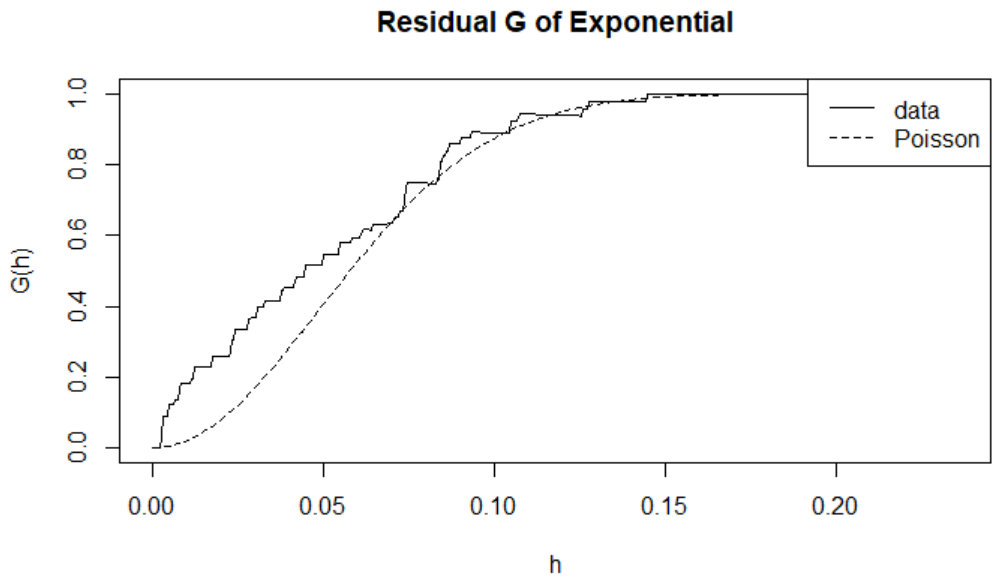

Figure 6.9: G function of superthinned points of model 3

The G function of the superthinned points adheres to the Poisson Process for h $>$ 0.7. However, There seems to be a large deviation from h=0 to h=0.06. This suggests that there is still clustering happening at very close proximity, and the lambda function is underestimating the conditional intensity in that range. This might be due to the fact that points are jittered slightly during the data cleaning process.

CHAPTER 7

Discussion and Conclusion

7.1 Summary of Findings:

In this study, we used three different models to try to better understand various properties of COVID-19 transmission in Taipei from May 2021 to April 2022. The first two models that were fitted utilize normal distribution for temporal variables of the dataset. Two variations of the bivariate normal distribution were used to try to extract the spatial properties of the data. From residual analysis, it seems that the normal distributed time paired with exponential distance returns the best results in modeling the COVID-19 transmission in Taipei. On the other hand, using bivariate normal spatial distribution with the bvnorm.pdf function in R shows inadequate results as the final fitted parameters values do not move significantly away from the initial set values, this might be due to the excessive number of parameters need, but the actual cause should be further scrutinized.

It seems that with the first model of normal time * bivariate normal space, there are on average two cases of COVID-19 that are not caused by previous events in our observational region, and 0.67 new events that are directly caused by each of the previous events. The time shift with mean of 1.6 days and standard deviation of 0.7 days suggests that each event has a rather short peak influence on subsequent event occurrence. It should be kept in mind that it had been shown in previous study that using a normal distribution in modeling incubation time can underestimate the actual length[XWM21]. Although it is tempting to interpret this time delay as the incubation period of the virus, one should take this result with caution, as the average incubation period of COVID-19 virus is about six to seven days, with 2 days of incubation period being on the low end of the range. The correlation of space being close

to 1 should also be interpreted with caution. This value suggests that there is an extremely strong pattern to how the virus is transmitted throughout the city.

The third model of pairing normal time * exponential distance suggests different properties of the COVID-19 virus. Compared to our first model, the third model suggests that only 0.14 of the new events appear independently of any other previous events from the observational region. Each of our events directly contributes to 0.95 new events. This model would suggest that the COVID-19 virus during that time frame is highly contagious and transmissible. For this model, it also suggests that the virus has a slightly longer incubation period of on average 2.3 days with a day of standard deviation. The alpha value also suggests that the virus is very contagious, but only at close distance. As the distance increases, the effectiveness of the virus quickly decreases. This could also suggest that there is high effectiveness in the social distancing policy that was employed.

7.2 Implications of the Study:

This type of modeling is only possible because of the way that Taiwan's COVID-19 cases were recorded. In many countries, only the number of COVID-19 infections are recorded. In Taiwan, however, the government employed a monitoring system that is able to track down and record where each person went. And when a COVID case is diagnosed, they are able to pinpoint the exact location of the event. This type of monitoring network enables the use of more sophisticated statistical models that also utilize spatial information such as longitude and latitude. From this study, not only can we extract more information about the virus, but also the effectiveness of public health policy that the government and the people were enforcing. We are able to strategize better public health policies in the future when the next virus inevitably occurs.

7.3 Limitations:

There are many limitations to this study. Although many of the COVID cases were recorded, it is not guaranteed that all COVID cases that occur in Taiwan were recorded by the dataset. Although sophisticated tracking system were set up in almost every stores in Taipei, it still relies on human effort to make sure the recording of one's information is properly enforced. There could also be cases where a tracking system could not be implemented. For example, assume that someone from the outskirts of Taipei city had the virus, but was never diagnosed, due to the lack of medical facilities in the area, then that person would have not been recorded into the dataset. However, that specific person could still transmit the virus to other people. It is also likely that this type of phenomenon has a specific pattern in areas around the city, which would cause discrepancies between the reflected model and the real conditional density.

Another limitation would be the lack of specific information about the virus. The COVID-19 virus has gone through many variants in the past few years. This specific dataset did not include the type of variant of the virus. Each variant of the virus has different properties, which means this model would be of little value for someone who wishes to use the model to study a specific variant of the virus.

7.4 Future Improvements:

One obvious improvement to make is within the data cleaning process. One thing that I suggest is treating points that occur at the exact same location at the exact same time as one unique event. Merely jittering the points could have effect the model in suggesting that the point that occurred slightly earlier had caused the second point. However, in the case of epidemiology, it is obvious that it is not the case, and that the two points could have been two people going to the same restaurant together.

Given the current data, another possible improvement would be to look at the change in clustering and properties of the virus spread throughout time. This study only looks at the

virus as a whole. However, government policy during the time was quick to change and could have had a significant impact on the spread of the virus. A month to month analysis could help to identify whether a policy was either effective, or enforced thoroughly. Such analysis could also help epidemiologists to see how the virus transmission rate varies throughout the seasons. This could potentially lead to a change in public health policy in the future.

This dataset is also unique in that it tracks where each person goes after being diagnosed with the virus. Given this property, another improvement could be to use the travel distance, or the number of locations that the person has been to as the magnitude of the event. With the magnitude, one could even employ the ETA point process model to conduct more sophisticated research on the virus.

APPENDIX A

Additional Data

A.1

```
covid_pp <- covid %>%
  group_by(id) %>%
  mutate(date = as.Date(first(begin))) %>%
  ungroup() %>% # Ensure jittering applies across the whole dataset, not within groups
  mutate(
    longitude = longitude + runif(n(), min = -1e-3, max = 1e-3), # Jitter longitude
    latitude = latitude + runif(n(), min = -1e-3, max = 1e-3)  # Jitter latitude
  ) %>%
  group_by(id) %>%
  slice(1) %>%
  arrange(begin)

covid_pp$latitude_scaled <- (covid_pp$latitude - min(covid_pp$latitude)-0.01) / (max(covid_pp$latitude)+0.01 -
min(covid_pp$latitude)-0.01)
covid_pp$longitude_scaled <- (covid_pp$longitude - min(covid_pp$longitude)-0.01) / (max(covid_pp$longitude)+0.01 -
min(covid_pp$longitude)-0.01)

covid_pp$timestamp = as.POSIXct(covid_pp$date)
# View the updated dataframe
covid_pp$year <- format(covid_pp$timestamp, "%Y") # Extract year
covid_pp$month <- format(covid_pp$timestamp, "%m") # Extract month
```

```
# Find the earliest time in the 'begin' column
start_time <- min(covid_pp$begin)

# Calculate the 't' value
covid_pp$t <- as.numeric(difftime(covid_pp$begin, start_time, units = "days"))

# Adjust 't' to include the time of day as a decimal
covid_pp$t <- covid_pp$t + (as.numeric(format(covid_pp$begin, "%H")) * 3600 + as.numeric(format(covid_pp$begin, "%M")) * 60 +
as.numeric(format(covid_pp$begin, "%S"))) / (24 * 3600)

covid_pp$t <- covid_pp$t + runif(length(covid_pp$t), min = 0, max = 600) / (24*60*60)
covid_pp <- covid_pp %>% arrange(t)
```

A.2

```
W <- obs_window
points_ppp <- ppp_cases
kde <- density.ppp(points_ppp, sigma=0.01)
kdeRaster <- raster(kde)
# Assuming your raster is named 'kdeRaster'
crs(kdeRaster) <- CRS("+init=epsg:4326")

kdeRaster
# Compute the kernel density estimate
# Adjust the sigma parameter as needed for your data's scale and density

map <- leaflet(data = locations) %>%
  # addTiles() %>%
  addProviderTiles(providers$CartoDB.Positron) %>%
  addCircleMarkers(~longitude, ~latitude,
                   radius = 5,
                   # color = ~colorPalette(timestamp_factor_ym),
                   color = 'red',
                   fillOpacity = 0.8,
                   popup = ~as.character(timestamp))%>%  # Keep the popup with full timestamp
  addRasterImage(kdeRaster, opacity=0.5)

map
```

A.3

```
#
# # here b1 is being use as the data point to detect clustering by

## par(mfrow=c(2,1)) ## if you want to make a 2x1 grid of plots
s = seq(.001,.3,length=50)
k4 = khat(b1,bdry,s)
plot(s,k4,xlab="distance",ylab="K4(h)",lty=1)
lines(s,k4)
lines(s,pi*s^2,lty=2)
legend("topright",lty=c(1,2),legend=c("data","Poisson"))

L4 = sqrt(k4/pi)-s
plot(c(0,.3),range(L4),type="n",xlab="lag, h",ylab="L4(h) - h")
points(s,L4,pch="*")
lines(s,L4)
lines(s,rep(0,50),lty=2)
```

```
### CONFIDENCE BOUNDS FOR K-FUNCTION via simulation
k4conf = Kenv.csr(npts(b1), bdry, 1000, s)
plot(c(0,max(s)),c(0,max(k4conf$upper,k4)), type="n",xlab="distance",ylab="K4(h)")
points(s,k4,pch="*")
lines(s,k4)
lines(s,pi*s^2,lty=2)
lines(s,k4conf$upper,lty=3,col="green",lwd=2)
lines(s,k4conf$lower,lty=3,col="green",lwd=2)
legend("topright",lty=c(1,2),legend=c("data","Poisson"))

L4upper = sqrt(k4conf$upper/pi) - s
L4lower = sqrt(k4conf$lower/pi) - s

plot(c(0,max(s)),c(min(L4lower,L4),max(L4upper,L4)),
type="n",xlab="distance",ylab="L4(h) - h")
points(s,L4,pch="*")
lines(s,L4)
lines(s,L4upper,lty=2,col="green",lwd=2)
lines(s,L4lower,lty=2,col="green",lwd=2)
lines(s,rep(0,length(s)))
```

```
### THEORETICAL BOUNDS for L-function
## bounds = 1.96 * sqrt(2*pi*A) * h / E(N), where
## A = area of space, and
## E(N) = expected # of pts in the space (approximated here using
## the observed # of pts
L4upper = 1.96 * sqrt(2*pi*1*1) * s / n
L4lower = -1.0 * L4upper
lines(s,L4upper,lty=3,col="orange",lwd=2)
lines(s,L4lower,lty=3,col="orange",lwd=2)

legend("topright",lty=c(1,2),legend=c("data","Poisson"))
```

```
### F-function (empty-space function):
# The cumulative distribution function (cdf), F,
# of the distance from a fixed location to the nearest point of X.
# Lower F indicates clustering.
# If F(0.2) = 0.4, for instance, then
# 40% of locations are within distance 0.2 of a point of the process.
b2 = as.ppp(b1, W = c(0,1,0,1))
## the above convert the points into a "ppp" object,
## using as a window [0,1] x [0,1]
par(mfrow=c(1,1))
# f4 = Fest(b2)
# plot(f4)
## or to control the plot yourself, try:
plot(f4$r[f4$r<.5],f4$theo[f4$r<.5],xlab="h",ylab="F(h)",type="l",lty=2)
lines(f4$r[f4$r<.5],f4$rs[f4$r<.5],lty=1)
legend("topright",lty=c(1,2),legend=c("data","Poisson"))
```

```
### G-function:
# the cdf, G, of the distance from a typical point to its nearest neighbor.
# Higher G indicates clustering.
# If G(0.2) = 0.9, then 90% of points have another point within 0.2 of them.
g4 = Gest(b2)
plot(g4)
## or to control the plot yourself, try:
plot(g4$r[g4$r<.3],g4$rs[g4$r<.3],xlab="h",ylab="G(h)",type="l",lty=1)
lines(g4$r[g4$r<.3],g4$theo[g4$r<.3],lty=2)
legend("topright",lty=c(1,2),legend=c("data","Poisson"))
```

```
### J-function:
# J(r) = (1-G(r))/(1-F(r)).
# J = 1 corresponds to a stationary Poisson process.
# J < 1 indicates clustering. J > 1 indicates inhibition.
j4 = Jest(b2)
plot(j4)
## or to control the plot yourself, try:
plot(j4$r[j4$r<.1],j4$rs[j4$r<.1],xlab="h",ylab="J(h)",type="l",lty=1)
lines(j4$r[j4$r<.1],j4$theo[j4$r<.1],lty=2)
legend("topright",lty=c(1,2),legend=c("data","Poisson"))
```

A.4

```
### Fitting a Pseudo-Likelihood model. With the Hawkes Model

n1 = z$n
X1 = 1
Y1 = 1
T = ceiling(max(z$t))
## Note that this is for a window of [0,1] x [0,1]
f2 = function(theta){
## theta input the initial values
## the function include the use of z as out datapoints to be fitted the model to
## returns the negative pseudo log-likelihood
## p = (mu, K, c, p, mu_x, mu_y, sd_x, sd_y, rho)
  mu = theta[1]; K = theta[2]; c = theta[3]; p = theta[4]; rho = theta[5]
  if(min(mu, K, c, p, rho)<0.000000001) return(99999)
  if(K>.99999) return(99999)
  if(rho >.99999) return (99999)
    # change this code so we don't sum up over the bins, but rather put all of lambdaj into a list and sum across all of them
  const = K
  lam = rep(0,n1)
   for(j in 1:n1){ ## j is the index of the point
     gkj = 0
     if(j>1) for(k in 1:(j-1)){ ## k are indices of previous points.
        gkj = gkj + dnorm(z$t[j]-z$t[k],c,p,0)*dnorm2d(z$lon[j]-z$lon[k],z$lat[j]-z$lat[k],rho)
     }
     lam[j] = mu/X1/Y1 + const*gkj
     if(lam[j] < 0){
         cat("lambda ",j," is less than 0.")
         return(99999)
     }
     # print(gkj)
   }
  if (min(lam) < 0) return (99999)
  int2 = mu*T + K*n1
  # print(lam)
  return(int2 - sum(log(lam)))
}
```

```
n1 = z$n
X1 = 1
Y1 = 1
T = ceiling(max(z$t))
## Note that this is for a window of [0,1] x [0,1]
f = function(theta){
## theta input the initial values
## the function include the use of z as out datapoints to be fitted the model to
## returns the negative pseudo log-likelihood
## p = (mu, K, c, p, mu_x, mu_y, sd_x, sd_y, rho)
  mu = theta[1]; K = theta[2]; c = theta[3]; p = theta[4]; mu_x = theta[5]; mu_y = theta[6]; sd_x = theta[7]; sd_y = theta[8]; rho = theta[9]
  if(min(mu, K, c, p, mu_x, mu_y, sd_x, sd_y, rho)<0.000000001) return(99999)
  if(K>.99999) return(99999)
  if(rho >.99999) return (99999)
    # change this code so we don't sum up over the bins, but rather put all of lambdaj into a list and sum across all of them
  const = K
  lam = rep(0,n1)
   for(j in 1:n1){ ## j is the index of the point
     gkj = 0
     if(j>1) for(k in 1:(j-1)){ ## k are indices of previous points.
        Xp <- cbind(z$lon[j]-z$lon[k],z$lat[j]-z$lat[k])
        mu_xy <- c(mu_x,mu_y)
        gkj = gkj + dnorm(z$t[j]-z$t[k],c,p,0)*bvnorm.pdf(Xp,mu_xy, sd_x,sd_y,rho)
     }
     lam[j] = mu/X1/Y1 + const*gkj
     if(lam[j] < 0){
         cat("lambda ",j," is less than 0.")
         return(99999)
     }
     # print(gkj)
   }
  if (min(lam) < 0) return (99999)
  int2 = mu*T + K*n1
  # print(lam)
  return(int2 - sum(log(lam)))
}
```

```r
n1 = z$n

X1 = 1
Y1 = 1
T = ceiling(max(z$t))
## Note that this is for a window of [0,1] x [0,1]
f3 = function(theta){
## theta input the initial values
## the function include the use of z as out datapoints to be fitted the model to
## returns the negative pseudo log-likelihood
## p = (mu, K, c, p, mu_x, mu_y, sd_x, sd_y, rho)
  mu = theta[1]; K = theta[2]; c = theta[3]; p = theta[4]; alpha = theta[5]
  if(min(mu, K, c, p, alpha)<0.000000001) return(99999)
  if(K>.99999) return(99999)
    # change this code so we don't sum up over the bins, but rather put all of lambdaj into a list and sum across all of them
  const = K*alpha/pi
  lam = rep(0,n1)
   for(j in 1:n1){ ## j is the index of the point
     gkj = 0
     if(j>1) for(k in 1:(j-1)){ ## k are indices of previous points.
        r2 = (z$lon[j]-z$lon[k])^2+(z$lat[j]-z$lat[k])^2
        gkj = gkj + dnorm(z$t[j]-z$t[k],c,p,0)*exp(-alpha*r2)
     }
     lam[j] = mu/X1/Y1 + const*gkj
     if(lam[j] < 0){
         cat("lambda ",j," is less than 0.")
         return(99999)
     }
     # print(gkj)
   }
  if (min(lam) < 0) return (99999)
  int2 = mu*T + K*n1
  # print(lam)
  return(int2 - sum(log(lam)))
}
```

A.5

```r
n <- 3
pstart.points.norm2d <- matrix(nrow = n+1, ncol = 5)  # Create a matrix to hold starting points for each iteration
plikelihood.norm2d <- numeric(n+1)  # Store likelihood values

# Initialize the first set of parameters
pstart.points.norm2d[1, ] <- c(1, 0.3, 15, 8, 0.9)
plikelihood.norm2d[1] <- f2(pstart.points.norm2d[1, ])


for (i in 1:n) {
  pstart <- pstart.points.norm2d[i, ]
  fit1 <- optim(pstart, f2, control = list(maxit = 200), hessian = TRUE)
  # if (fit1$convergence == 0) {
  pend <- fit1$par
  pstart.points.norm2d[i+1, ] <- pend
  plikelihood.norm2d[i+1] <- f2(pend)
  # } else {
  #   print(paste("Optimization failed at iteration", i))
  #   break
  # }
}
```

A.6

```
supthin = function(z,lambda,f,b=mean(lambda)){
## z = data, lambda = conditional intensity at pts, f = function to compute lambda,
## and b = resulting rate.
## First thin, then superpose
keepz = list()
for(i in 1:z$n){
  if(runif(1) < b/lambda[i]){
  keepz$t = c(keepz$t,z$t[i])
  keepz$lon = c(keepz$lon,z$lon[i])
  keepz$lat = c(keepz$lat,z$lat[i])
  }
}
candn = rpois(1,b*X1*Y1*T)
candt = sort(runif(candn)*T)
candx = runif(candn)*X1
candy = runif(candn)*Y1
print(candn)
for(i in 1:candn){
v = f(candt[i],candx[i],candy[i],z)
if(v < b){
if(runif(1) < (b-v)/b){
keepz$t = c(keepz$t,candt[i])
keepz$lon = c(keepz$lon,candx[i])
keepz$lat = c(keepz$lat,candy[i])
}}
}
keepz$lon = keepz$lon[order(keepz$t)]
keepz$lat = keepz$lat[order(keepz$t)]
keepz$t = sort(keepz$t)
keepz$n = length(keepz$t)
keepz
}
```

REFERENCES

[BKW20] Jantien A Backer, Don Klinkenberg, and Jacco Wallinga. "Incubation period of 2019 novel coronavirus (2019-nCoV) infections among travellers from Wuhan, China, 20–28 January 2020." *Eurosurveillance*, **25**(5), 2020.

[BQN20] David Baud, Xiaolong Qi, Karin Nielsen-Saines, Didier Musso, Léo Pomar, and Guillaume Favre. "Real estimates of mortality following covid-19 infection." *The Lancet Infectious Diseases*, **20**(7):773, Jul 2020.

[CBD12] Simon Cauchemez, Pierre-Yves Boëlle, Christl A. Donnelly, Neil M Ferguson, Guy Thomas, Gabriel M. Leung, Anthony J Hedley, Roy M. Anderson, and Alain-Jacques Valleron. "Real-time Estimates in Early Detection of SARS." *Emerging Infectious Diseases*, **12**(1):110–113, January 2012.

[Cey23] Elvan Ceyhan. *nnspat: Nearest Neighbor Methods for Spatial Patterns*, 2023. R package version 0.1.2.

[CLM22] Wen-Hao Chiang, Xueying Liu, and George Mohler. "Hawkes process modeling of COVID-19 with mobility leading indicators and spatial covariates." *International Journal of Forecasting*, **38**(2):505–520, 2022.

[CSB22] Bohan Chen, Pujan Shrestha, Andrea L. Bertozzi, George Mohler, and Frederic Schoenberg. *A Novel Point Process Model for COVID-19: Multivariate Recursive Hawkes Process*, pp. 141–182. Springer International Publishing, Cham, 2022.

[CSK24] Joe Cheng, Barret Schloerke, Bhaskar Karambelkar, and Yihui Xie. *leaflet: Create Interactive Web Maps with the JavaScript 'Leaflet' Library*, 2024. R package version 2.2.2.9000, https://github.com/rstudio/leaflet.

[CSV12] Robert Alan Clements, Frederic Paik Schoenberg, and Alejandro Veen. "Evaluation of space–time point process models using super-thinning." *Environmetrics*, **23**(7):606–616, September 2012.

[JS22] Ryan J. Harrigan Junhyung Park, Adam W. Chaffee and Frederic Paik Schoenberg. "A non-parametric Hawkes model of the spread of Ebola in west Africa." *Journal of Applied Statistics*, **49**(3):621–637, 2022. PMID: 35706773.

[KSM22a] Conor Kresin, Frederic Schoenberg, and George Mohler. "COMPARISON OF HAWKES AND SEIR MODELS FOR THE SPREAD OF COVID-19." *Advances and Applications in Statistics*, **74**:83–106, 03 2022.

[KSM22b] Conor Kresin, Frederic Paik Schoenberg, and George Mohler. "COMPARISON OF HAWKES AND SEIR MODELS FOR THE SPREAD OF COVID-19." *Advances and Applications in Statistics*, **74**:83–106, March 2022.

[LGM23] Stamatina Lamprinakou, Axel Gandy, and Emma McCoy. "Using a latent Hawkes process for epidemiological modelling." *Plos one*, **18**(3):e0281370, 2023.

[LLH23] Chih-Cheng Lai, Ping-Ing Lee, and Po-Ren Hsueh. "How Taiwan has responded to COVID-19 and how COVID-19 has affected Taiwan, 2020–2022." *Journal of Microbiology, Immunology and Infection*, **56**(3):433–441, 2023.

[LRB09] Justin Lessler, Nicholas G Reich, Ron Brookmeyer, Trish M Perl, Kenrad E Nelson, and Derek AT Cummings. "Incubation periods of acute respiratory viral infections: a systematic review." *The Lancet Infectious Diseases*, **9**(5):291–300, May 2009.

[Oga78] Yoshiko Ogata. "The asymptotic behaviour of maximum likelihood estimators for stationary point processes." *Annals of the Institute of Statistical Mathematics*, **30**(2):243–261, December 1978.

[Oga88] Yosihiko Ogata. "Statistical Models for Earthquake Occurrences and Residual Analysis for Point Processes." *Journal of the American Statistical Association*, **83**(401):9–27, 1988.

[PR21] Antonio Paez and , Robin. "paezha/spatial-analysis-R: Pre-release to obtain doi from Zenodo.", 2021.

[R C23] R Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria, 2023.

[Rei18] Alex Reinhart. "A Review of Self-Exciting Spatio-Temporal Point Processes and Their Applications." *Statistical Science*, **33**(3), August 2018.

[Rip76] B. D. Ripley. "The second-order analysis of stationary point processes." *Journal of Applied Probability*, **13**(2):255–266, June 1976.

[spr03] Springer-Verlag, 2003.

[Tre96] Gotz Trenkler. "Continuous univariate distributions: N.L. Johnson, S. Kotz N. Balakrishnan (1995): Vol. 2, 2nd Edition. John Wiley, New York, ISBN: [pound sign] 70.00, pp xix + 719." *Computational Statistics Data Analysis*, **22**(5):568–569, 1996.

[WHO20] WHO. "WHO statement regarding cluster of pneumonia cases in Wuhan, China." *Beijing: WHO*, **9**, 2020.

[Wik23] Wikipedia contributors. "COVID-19 contact tracing in Taiwan — Wikipedia, The Free Encyclopedia.", 2023. [Online; accessed 24-May-2024].

[WNB20] C. Jason Wang, Chun Y. Ng, and Robert H. Brook. "Response to COVID-19 in Taiwan: Big Data Analytics, New Technology, and Proactive Testing." *JAMA*, **323**(14):1341–1342, 04 2020.

[WSC23] Diethelm Wuertz, Tobias Setz, Yohan Chalabi, and Paul Smith. *fCopulae: Rmetrics - Bivariate Dependence Structures with Copulae*, 2023. R package version 4022.85.

[XWM21] Hualei Xin, Jessica Y Wong, Caitriona Murphy, Amy Yeung, Sheikh Taslim Ali, Peng Wu, and Benjamin J Cowling. "The Incubation Period Distribution of Coronavirus Disease 2019: A Systematic Review and Meta-analysis." *Clinical Infectious Diseases*, **73**(12):2344–2352, June 2021.