Spatial Outbreak Detection

Author: 1823306 April 17, 2022

Abstract

In this paper, I have discussed the spatio-temporal effects on disease outbreak detection models, which is commonly used in the field of epidemiology. In particular, a comparison study between the spatially independent model 3.1 and spatially dependent model 3.2 was conducted to evaluate the spatial effect on predictability of an outbreak.

The MCMC methods were introduced to study the statistical characteristics of our target outbreak distributions. By constructing the corresponding posterior distribution, I used different sampling methods to sample model parameters against MCMC iterations. Then I used relevant plots and ROC curve (Section 4) to evaluate the predictability of both outbreak models.

To measure the findings in a real-life scenario, I am adopting the spatially independent model on Maryland Covid data (Section 5) (positive cases only), collected from Mar 2020 to Mar 2022.

Contents

1 Introduction **4**

1.1 Aims and Objectives . . . 4

1.1.1 Project Structure . . . 4

1.2 Challenges . . . 5

1.2.1 Measure of spatial information . . . 5

1.2.2 Computational expense of MCMC methods . . . 5

2 Topic Background **5**

2.1 Outbreak Detection . . . 5

2.2 Parametric Model Approach . . . 6

2.2.1 Frequentist Approach . . . 6

2.2.2 Bayesian Approach . . . 6

2.3 Bayesian Statistics . . . 6

2.3.1 Bayes' theorem . . . 7

2.3.2 Establishing posterior distribution . . . 7

2.3.3 Bayesian Hierarchical model . . . 8

2.4 Markov chain Monte Carlo (MCMC) . . . 8

2.4.1 Gibbs Sampling . . . 9

2.4.2 Metropolis-Hastings algorithm . . . 9

2.4.3 Adaptive Dirichlet random walk . . . 10

2.4.4 Diagnostic tools over the convergence . . . 10

2.4.5 Choosing starting values . . . 11

3 Models and Methodology **11**

3.1 Spatially Independent Model . . . 11

3.1.1 Assumptions . . . 11

3.1.2 Parameters . . . 12

3.1.3 Generate data . . . 12

3.2 Spatial Dependent Model . . . 13

3.2.1 Distance Metric . . . 13

3.2.2 Assumptions . . . 14

3.2.3 Dynamic Probability Function . . . 15

3.2.4 Parameters . . . 17

3.2.5 Generate data . . . 17

3.3 Constructing MCMC in independent model . . . 18

3.3.1 Posterior density . . . 18

3.3.2 Update α, β . . . 19

3.3.3 Update probability p . . . 19

3.3.4 Update variable x . . . 20

3.4 Constructing MCMC in spatio-temporal dependent model . . . 21

3.4.1 Posterior density . . . 21

3.4.2 Update α, β . . . 22

3.4.3 Update γ, Δ . . . 22

3.4.4 Update variable x . . . 23

4 Model Evaluation **24**
4.1 Spatially Independent Model . . . 24
4.1.1 Convergence diagnosis . . . 24
4.1.2 ROC performance . . . 24
4.2 Spatially Dependent Model . . . 26
4.2.1 Convergence diagnosis . . . 26
4.2.2 ROC performance . . . 26

5 Case Study: Coronavirus in Maryland **27**
5.1 Background . . . 27
5.1.1 Data . . . 27
5.1.2 Infection trend . . . 28
5.1.3 Time of interest . . . 29
5.2 Model formulation . . . 29
5.2.1 Outbreak Indicator . . . 29
5.2.2 Distance metric . . . 29
5.2.3 Poisson rate parameter . . . 29
5.3 Choosing startvalues . . . 30
5.3.1 Choosing α, β . . . 30
5.3.2 Choosing γ, Δ . . . 30
5.4 Evaluation on MCMC outputs . . . 31

6 Conclusion **31**
6.1 Findings . . . 31
6.1.1 Spatially independent model . . . 31
6.1.2 Spatially dependent model . . . 31
6.2 Interpretation . . . 31
6.3 Limitation . . . 31
6.3.1 Self-generating data . . . 31
6.3.2 Dependence among the parameters . . . 31
6.3.3 Measurement . . . 32
6.3.4 MCMC diagnosis . . . 32

7 Future work **32**
7.1 Hamiltonian Monte Carlo . . . 32
7.2 Model extension . . . 32
7.2.1 Spatio-temporal term . . . 33
7.2.2 Timely-temporal term . . . 33

8 Appendix **34**
8.1 Supplementary information . . . 34
8.2 Plots . . . 34
8.3 Distribution table . . . 34
8.4 R Code . . . 34

1 Introduction

1.1 Aims and Objectives

The aim of this paper is to identify outbreaks of disease from a background of sporadic cases. In particular, we aim to develop a spatial outbreak detection method that can use information from neighboring locations to improve the discrimination of an outbreak. We aim to explore:

- What is the grey area in a disease outbreak, that makes it difficult to be detection?
- By introducing spatial information to the outbreak model, what impact would it have to the risk of having an outbreak?
- How/why/when does an outbreak rise?
- How do we detect a spatial outbreak?

1.1.1 Project Structure

Timeline	Objectives
1st stage	Review relevant literature on statistical study about outbreak detection model.
	Build simple independent outbreak model.
	Construct the MCMC algorithms.
	Estimate the level of risk β for an outbreak to occur.
2nd stage	Introduce adjacent matrix to reflect spatial information.
	Construct dynamic probability function of p for the dependent model.
	Build spatial dependent outbreak model.
	Construct the modified MCMC algorithms.
	Estimate the level of risk β and neighboring effects for an outbreak to occur.
3rd stage	Implement real-life disease data.
	Evaluate the outbreak pattern across regions.
	Propose future work.

Table 1: Project Structure

In the first stage of the project, we will go through the relevant literature review on outbreak detection and the application of MCMC (Markov Chain Monte Carlo) in the field of epidemiology. Then, we will build a simple spatial independent outbreak model to estimate the level of risk (value of β in our case) for an outbreak to occur. We start to observe the relationship between risk parameter β and the grey area under simulation where it becomes difficult to detect whether an outbreak has occurred or not. Corresponding analysis will be followed to visualise the simulation.

In the second stage, we will introduce the localised information to the initial model, which would remove the independency among different regions. We would consider constructing an adjacent matrix to reflect spatial information within a given region. Using the updated spatial dependent outbreak model, we will observe the dynamic outbreak pattern, and in particular, the probability distribution of the model.

In the final stage of the project, we would further explore the open problems in our field of study. We would also consider applying our model to some real-life datasets that we find interesting. In the end,

we will give a comparison study between our models and other existing methods to propose potential research topics in the future work.

Motivation Given the context of the coronavirus, people may realise the significance and benefit of being able to detect potential outbreaks. Especially for areas that are lack of modern surveillance systems from public health agencies, researchers may find it more challenging to perform trend analysis of future infections, given limited individual health data. In such a case, by taking the spatial information into account, we may have better confidence in determining a potential outbreak event.

In this study, we want to construct a parametric outbreak detection model (Section 3.1, 3.2) in Bayesian settings (more discussion followed in Section 2.3) to represent real-life disease outbreaks within some locations.

1.2 Challenges

1.2.1 Measure of spatial information

The epidemiology can be complex with many possible transmission routes. Even when looking at a single disease such as Campylobacteriosis, we would discover a large variation between years in the timing, duration and peak incidence of the epidemic [Spe11].

We need to be cautious in constructing the adjacent metric among regions. As stated in historical research of spatial-temporal model, locations can be grouped using different selection approaches (via distance, water supply, census area [Spe11]). Any heterogeneity in selection would imply that some outbreaks would be affected less than others. This would make some detected outbreaks to be less representative, reducing the application of corresponding epidemiological inferences.

1.2.2 Computational expense of MCMC methods

In many past studies, researchers have been looking at the implications of the statistical large sample theory for computational complexity of Bayesian and quasi-Bayesian estimations [Bel12], which are derived by different random walk algorithms such as Metropolis-Hastings random walk. In particular, a polynomial complexity is established by exploiting the central limit theorem framework,giving a structural restriction on the methods (that the posterior density approaches to a normal density in large studied samples).

2 Topic Background

2.1 Outbreak Detection

In the history of applied statistics, there has been a surge in interest in early detection of infectious disease outbreaks [Unk11], using statistical algorithms based on Monte Carlo Simulation. It is essential for the National Public health [Buc08] to provide surveillance for accurate and timely outbreak detection, in order to achieve effective epidemic control.

Within the scope of epidemiology, the Poisson distribution has same practical applications in the study of disease incidence, such as analyzing counts of the number of disease outbreak in a given region or time period. In particular, Poisson distribution would become dynamic when regional spread is considered, which makes it challenging to find parameters of the model.

2.2 Parametric Model Approach

For several decades, there has been a large increase in the research activity on statistical issues that are related to prospective detection, namely detection of outbreaks as they arise.

In the modern study of statistical outbreak models, researchers are trying to incorporate the location(spatial) information to potentially enable localized outbreaks of a disease to be detected, or variations in regional patterns to be identified. In the past study, several surveillance methods require only a cut-off value that categorizes pairs of observations as either being "close/neighboring" or 'not close'(e.g. Rogerson and Kulldorff(2001)). An appropriate adjacency metric or distance metric was then defined to represent the closeness of different geographic units.

In the area of epidemiology, there are many statistical methods being studied and implemented to provide spatial disease surveillance, including Regression, cumulative sum (CUSUM) charts, Space-time scan statistics. In this paper, we will focus on Spatio-temporal regression methods. In particular, for an area level model, y_{ij} denote the number of cases at time period $i$in location j, where y_{ij} follows a Poisson distribution. Interest would then centre on estimating the Poisson mean (Lawson et al., 2003; Vidal Rodeiroand Lawson, 2006; Watkins et al., 2009; Zhou and Lawson, 2008), which varied with i and j.

2.2.1 Frequentist Approach

To estimate statistics information of our assume outbreak distribution, many historical studies were established using Spatial Scan Statistics, in which the key idea is to search for areas of maximum disease activity by computing a score of a likelihood ratio of having an outbreak in each considered cluster versus no outbreak. Many modified methods were proposed from the idea of Spatial Scan statistics, such as FSS (flexible spatial scan statistics) and ULS (upper level set scan statistics).

However, in terms of real life disease data, we might find it difficult to construct full conditional distribution of some parametric models, from the sample observations. This leads to another research approach using the Bayesian setting.

2.2.2 Bayesian Approach

Bayesian-based spatial scan statistics was proposed to overcome the lack of information on distribution, using Bayes theorem. The Bayesian approach suggests the idea to identify a rectangular sub-region, which is composed of the cells within the highest posterior probability of having an outbreak. There are also other popular methods such as Bayesian-based multilevel spatial clustering and z-score based multilevel spatial clustering.

However, the use of real-life spatial information can be computationally demanding (as mentioned in Section 1.2.2), which results in being less practical for surveillance systems that monitor hundreds of disease organisms——computer limitations will restrict the complexity of calculations that can be performed for each disease. Hence in the main body of this project, we will generalise the outbreak model under several assumptions of parameter distributions and spatial measurement.

2.3 Bayesian Statistics

In Bayesian Statistics, the prior distribution represents the uncertainty or background knowledge about a particular event or population prior to data sampling, and the posterior distribution represents updated conditional probability of a population attribute based on observed data as well as prior information [Bol07].

To compute and update conditional probabilities after sampling the data, we would construct a Markov chain Monte Carlo (MCMC) to simulate our outbreak data within a given time period and locations. this Bayesian statistical method is based on Bayes' theorem, which s first proposed by Thomas Bayes in a paper he published in 1763.

2.3.1 Bayes' theorem

The theorem is defined as below: Given two events A and B, the conditional probability of A given that B is true is expressed as follows:

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

, where $P(B) \neq 0$. $P(A)$ is the prior probability of A which expresses subject beliefs about A before any further evidence from the data is taken into account. $P(B|A)$ is the likelihood function, which expresses the probability of evidence B given that A is true. Given the theorem, we have the posterior probability:

$$P(A|B) \propto P(B|A)P(A)$$

In the implementation of Markov chain Monte Carlo on simulating outbreak data, it is difficult to compute the exact value of $P(B)$(i.e. the full evidence from the data). Therefore, the posterior $P(A|B)$(i.e. the conditional probability of model parameters given the evidence from sampled data) would be approximated without computing $P(B)$.

2.3.2 Establishing posterior distribution

Under the assumption that the number of disease Y follows a Poisson distribution ($Y \sim Poi(\lambda)$), we want to find the parameter λ for our model. By definition, posterior probability is the probability that an event will happen after all evidence or background information has been taken into account. In our case, a approximation of the posterior distribution of λ given x: $P(\lambda|Y)$ can be derived from prior distribution $P(\lambda)$ and likelihood function $P(Y|\lambda)$, using Bayes' rule.

Prior First, we need to define a prior distribution over the model parameters. From the prior distribution, we are explicitly expressing our prior uncertainty about plausible values of λ.

Here, we define the whole parameter set as ϕ, which gives us the prior density of λ:

$$P(\lambda) = P(\lambda|\phi)$$

Likelihood Then, we construct the likelihood function for observations Y given parameter λ (,which follows a Poisson distribution $Y \sim Poi(\lambda)$).

$$P(Y|\lambda) = \sum_{i=1}^{n} P(y_i|\lambda_i)$$

and,

$$\begin{aligned} P(y_i|\lambda_i) &= F(y_i, \lambda_i) \\ &= \sum_{k=0}^{y} \frac{e^{-\lambda_i}\lambda_i^k}{k!} \end{aligned}$$

Approximated posterior As stated in the Bayes' theorem: $P(A|B) \propto P(B|A)P(A)$, we can have approximated posterior as the following:

$$\begin{aligned} P(\lambda|Y) &= \frac{P(Y|\lambda)P(\lambda)}{P(Y)} \\ &\propto P(Y|\lambda)P(\lambda) \\ &= \prod_{i=1}^{n} P(y_i|\lambda_i)P(\lambda|\phi) \end{aligned}$$

2.3.3 Bayesian Hierarchical model

Bayesian hierarchical modelling is a statistical model established in multiple levels (hierarchical form) that estimates different parameters of the posterior distribution using the Bayesian statistics. In general, hierarchical Bayesian models can be represented as a directed acyclic graph (DAG) with successive conditional probabilities flowing from a priori assumptions on the base level of model parameters.

Unlike in the regular Bayesian model where we have that posterior density is proportional to the multiplication of the likelihood and prior density.

$$\mathcal{P}(\theta|x) \propto \mathcal{P}(x|\theta) \cdot \mathcal{P}(\theta)$$

In hierarchical Bayesian modeling, we have individual parameters and population parameters. For example we can denote θ_i as the individual parameter to observation x_i and α as the population parameter. Then we can compute the posterior density as following:

$$\mathcal{P}(\alpha, \theta|x) \propto \mathcal{P}(x|\theta, \alpha) \cdot \mathcal{P}(\theta|\alpha) \cdot \mathcal{P}(\alpha)$$

2.4 Markov chain Monte Carlo (MCMC)

In Bayesian inference, the model parameters are regarded as random variables. As stated in Bayes' theorem, we can update the information about the parameters in terms of posterior density (defined as the normalised product of the prior density and the likelihood), from which one can obtain point and interval estimates (e.g. mode, mean, confidence interval).

As mentioned in the topic background, the difficulties arise when dealing with disease outbreak data. Due to the fact that many infection processes in real life is hard to be observed, the use of data imputation method is naturally considered to overcome this problem. There are two widely used data imputation methods: EM algorithm and Markov chain Monte Carlo (MCMC) methods. A main drawback with EM algorithm for epidemic inference problems (see [Bec97]) is that the evaluation of the expectation step can be rather complicated.

In this study, we introduce Markov chain Monte Carlo (MCMC) methods to help us estimate an unknown probability distribution for an outbreak process in a straightforward approach. Various simulations would be built to develop prospective surveillance on a timely basis across the locations [Ham13]. Mathematically speaking, the methods would allow us to generate various Markov chains, and each of whose stationary distributions is the distribution of our parameters of interest.

In combination with the Bayesian approach, MCMC enables analysis of the full model parameters. Posterior summaries such as means, medians, variances, intervals, etc. can all be easily obtained for individual parameters, or for joint distributions of parameters.

2.4.1 Gibbs Sampling

Gibbs Sampling (also called *alternating conditional sampling*) was first proposed by Geman and Geman(1984) and further developed by Gelfand and Smith(1990). The target distribution is the invariant distribution of the Markov chain generated by the algorithm, to which it converges through iterations.

It is an efficient way of reducing a multi-dimensional problem to a lower-dimensional problem. The model parameter vector is subdivided into smaller subvectors (e.g. vectors with a single parameter). One iteration of the algorithm results in each subvector randomly sampled using the subvector's posterior density, conditional on the other subvector's current values (Duchateau Janssen, [Duc11]). The conditional distributions used in the Gibbs sampler are often referred to as full conditionals: being conditional upon everything except the variable being sampled at each step.

Algorithm - systematic scan Given a n-dimentional variable $x : (x_1, x_2, ..., x_n)$, Gibbs sampler would start with $x^{(0)} := (x_1^{(0)}, x_2^{(0)}, ..., x_n^{(0)})$ iterate for $t = 1, 2, ...$

1. Draw $x_1^{(t)} \sim f_{x_1|x_{-1}}(\cdot|x_2^{(t-1)}, ..., x_n^{(t-1)})$
   ...
2. Draw $x_i^{(t)} \sim f_{x_1|x_{-1}}(\cdot|x_1^{(t)}, ..., x_{i-1}^{(t)}, x_{i+1}^{(t-1)}, ..., x_n^{(t-1)})$
   ...
3. Draw $x_n^{(t)} \sim f_{x_n|x_{-n}}(\cdot|x_1^{(t)}, ..., x_{n-1}^{(t)})$

The method gives an accurate representation of joint posterior densities. In order to sample from all of the conditional distributions for all model parameters, the required conditional distributions will be sampled by **Metropolis Hastings** first.

2.4.2 Metropolis-Hastings algorithm

The Metropolis-Hastings algorithm is a general Markov Chain Monte Carlo (MCMC) technique for sampling from a probability distribution that is difficult to sample from directly. The algorithm goes back to Metropolis et al. (1953) and Hastings (1970), which is based on proposing values sampled from an instrumental distribution, which are then accepted with a certain probability that reflects how likely it is that they are from the target distribution f.

Using this technique, it allows us to approximate the conditional distribution (e.g. to draw samples from the posterior) of the model. The algorithm allows for local updates, i.e. let the proposal value depend on the last accepted value. This makes it easier to come up with a suitable (conditional) proposal, at the price of yielding a Markov chain instead of a sequence of independent realisations. Since our models have multi-dimensional distributions, Metropolis-Hasting appears to be an efficient sampling technique, comparing with other sampling methods (e.g. rejection sampling, importance sampling).

Algorithm Given a n-dimensional variable $x : (x_1, x_2, ..., x_n)$, Metropolis-Hastings would start with $x^{(0)} := (x_1^{(0)}, x_2^{(0)}, ..., x_n^{(0)})$ iterate for $t = 1, 2, ...$

1. Draw $x \sim q(\cdot|x^{(t-1)})$ (Here, $q(\cdot)$ is the proposal distribution).
2. Compute the acceptance probability/ratio

$$\alpha(x|x^{(t-1)}) = min\{1, \frac{f(x) \cdot q(x^{(t-1)}|x)}{f(x^{(t-1)}) \cdot q(x|x^{(t-1)})}\}$$

3. With probability $\alpha(x|x^{(t-1)})$, set $x^t = x$, and set $x^t = x^{(t-1)}$ otherwise.

2.4.3 Adaptive Dirichlet random walk

Gelman et al. (1997) propose to adjust the proposal distributions such as the overall acceptance rate is around 1/2 for one or two dimensional target distributions, and around 1/4 for proposals in higher dimensional spaces. In our case of study, since both spatially dependent and independent outbreak model are expected to have high-dimensional target distributions, we aim to proposal the MCMC methods to achieve an acceptance rate at around 25%.

In the field of epidemiology, an adaptive Dirichlet random walk scheme was first introduced in a recent study on leptospirosis notification data in New Zealand using a Bayesian model [Ben21]. To approach the target acceptance rate for the MCMC methods used in the paper, the authors applied this adaptive Dirichlet random walk scheme during the burn-in period of the MCMC. In particular, the mixing parameter a is adjusted automatically under the following rules:

- If a proposal is accepted, then $a^* = max\{0, a - 3\}$
- If a proposal is rejected, then $a^* = a + 1$

With this adjustment, the proposal would be concentrated around the current location if the acceptance rate is too low.

2.4.4 Diagnostic tools over the convergence

After discussing over various sampling-based algorithms implemented in MCMC methods, we then introduce some practical diagnostic tools of MCMC which are widely used to examine the convergence of the Markov chains.

In the recent studies of analyzing MCMC output ([Vat19]), there are two common questions to evaluate from the outputs:

1. Deciding when the Markov chain converges enough to our target distribution? (i.e. the stationary distribution of the chain has produced a representative sample from probability distribution of the model we proposed.)
2. When the iteration is large enough for us to estimate our parameter sets using the samples?

There are currently graphical approaches led by many practitioners to approach the questions based on trace plots of the simulation components, histograms representing the posterior distribution of the parameters and other ad-hoc convergence diagnostics (Cowles and Carlin, 1996).

In terms of deciding the proper number of iterations for MCMC, there's no theoretical approach to guide assessing the convergence of sums of the MCMC samples, but there are heuristic ones to suggest reasonable choice. One of these is Effective Sample Size (ESS) [Vat19]. The key idea behind ESS is to compute a "exchange rate" between dependent and independent samples, as to measure the number of effectively independent draws from the posterior distribution that the Markov chain is equivalent to.

The definition of ESS is as followings:

$$\text{ESS} = \frac{n}{1 + 2\sum_{k=1}^{\infty} \rho(k)}$$

, where n is the number of samples and $\rho(k)$ is the correlation at lag k.

Another heuristic approach is to look at the Gelman-Rubin-Brooks statistic $\hat{R}$ [Kas98](Vats and Knudson, 2018), which is denoted as:

$$\hat{R} \approx \sqrt{1 + \frac{1}{\text{ESS}}}$$

Strictly speaking, complete convergence does not occur for any finite iterations n (Roy, 2019), but the choice of starting value (Rosebthal, 1995) would have a substantial impact on the rate of the convergence. In particular, when we start a MCMC simulation in an area of low probability of our target distribution, the convergence of the Markov chain may appear to be quite slow (Gilks et al., 1996). Therefore, choosing good starting values can save considerable time in terms of computation and increase confidence in the outputs.

2.4.5 Choosing starting values

In order to choose a proper starting value for our parameter set of interest, we decide to first use MCMC methods for a simple independent model (more discussion of the model construction will be followed in Section 3.1).

The idea of starting with a simple Independent model is to aim for a low-dimensional target distribution, in which we can use an accept-reject sampler to produce one sample from the target, and pick reasonable values in a high probability region as the starting values. Here in the Bayesian inference setting, we also rely on the prior information of our parameters given their chosen prior distributions when we are choosing the reasonable starting values.

3 Models and Methodology

3.1 Spatially Independent Model

To monitor the spread of diseases, we want to measure the number of potential infectious cases within a given time. And the probability of this event can be broken down by assuming number of cases Y follows a Poisson distribution: $Y \sim Poisson(\lambda_i)$, where λ_i represents the frequency of having an infectious case of disease at time period i.

To reflect the exponential growth of λ based on x (outbreak indicator), we start with a simple log regression model: $log(\lambda_{ij}) = \alpha + \beta x_i$.

For the parameters in our regression model, α represents the average risk of having an infectious disease and β represents the additional risk cause by regional outbreaks, identified by a binary variable x_i (in which $x_i = 1$ leads to higher risk of having infectious diseases). Under the prior assumption, x is following a Bernoulli distribution: $x \sim Bernoulli(p)$ (also written as $x \sim Binomial(1, p)$), in which an outbreak occurs when $x = 1$ and no outbreak occurs otherwise($x = 0$).

3.1.1 Assumptions

In this model, following assumptions are made:

- Each region has an independent outbreak distribution, i.e. spatial correlation are excluded in the model.
- Probability p of having higher risk of an outbreak is constant across the time scope.

3.1.2 Parameters

In order to build an outbreak model, several parameters within the distribution need to be established.

- $Y \sim Poisson(\lambda)$
- $log(\lambda_i) = \alpha + \beta x_i$
- $x_i \sim Bernoulli(p)$
- $p \sim Beta(a, b)$

Notice here probability p of having higher risk of an outbreak is assumed to be following a beta distribution with mean $\frac{a}{a+b}$, in which the beta distribution is known as the conjugate prior distribution for binomial(or Bernoulli) distribution. Here we assumed that the outbreak would occur around four times every year (52 weeks). The probability is independent among different locations and time(weeks) in the first model. That's why we refer this model as being **Spatially Independent**.

3.1.3 Generate data

Based on the assumptions that we have made for the Spatially Independent model, we can start to generate our sample data for variables X, Y based on different choices of the true values of model parameters (table 2). Discussion on the model performance (predictability on the outbreaks) is followed in section 4.1.2.

Trials	α	β	p
	ln(1)	ln(2)	4/52
	ln(1)	ln(2)	7/52
	ln(1)	ln(3)	7/52
	ln(1)	ln(4)	4/52
	ln(1)	ln(4)	7/52
	ln(2)	ln(2)	4/52
	ln(2)	ln(2)	7/52
	ln(2)	ln(3)	7/52
	ln(2)	ln(4)	4/52
	ln(2)	ln(4)	7/52
	ln(4)	ln(2)	4/52
	ln(4)	ln(2)	7/52
	ln(4)	ln(4)	4/52
	ln(4)	ln(4)	7/52
	ln(4)	ln(10)	7/52
	ln(4)	ln(10)	15/52

Table 2: Choice of true values in generated dataset

The true values under different choices are set to all model parameters $\phi = (\alpha, \beta, p)$. Here, we pick e^{α} to assume the fact that an infectious disease would normally occur e^{α} times within a given time period. And in the case of having a localised outbreak (i.e. $x = 1$), we assume that the disease would occur e^{β}

times more frequently every year, which gives:

$$\begin{aligned}\lambda_i &= e^{\alpha+\beta x_i} \\ &= e^{\alpha} \cdot e^{\beta x_i} \\ &= e^{\alpha} \cdot e^{\beta}\end{aligned}$$

Then given the total time length as n and outbreak probability as p, we use function $rbinom()$ in R to generate x_i for each time i. In particular, we first generate x_i for each time $i \in (1, 2, ..., n)$. After we successfully construct the sample data for binary variable X, we can computing the corresponding rate parameter λ for the Poisson distribution that outcome variable Y follows.

$$\lambda = e^{\alpha+\beta x}$$

, where each $log(\lambda_i) = \alpha + \beta x_i$.

Using the information we get from λ, we can further generate sample data for variable Y with build-in function $rpois()$ in R.

3.2 Spatial Dependent Model

Based on the build-up of the initial Spatial Independent Model, we then begin to consider adding neighbouring effects of outbreak disease for different studies locations. In particular, we want to construct a dynamic success probability p in terms of a function of both background isolated risk level and weighted spatio-temporal effects.

In the study of Spatial Scan Statistics, there are three key components being considered: the geometry of the area being scanned, the probability distribution generating events and the shapes, sizes of the scanning window. Here, we focus on constructing the dynamic probability function based on an adjacent distance metric.

3.2.1 Distance Metric

In real-life setting, designing a distance metric for the outbreak model can be very complicated (mentioned in Section 1.2.1). In particular, we need to prevent the measurement from causing heterogeneity in the selected regions.

One simple measurement on a two-dimension map would be using the Euclidean distance to compute the direct distance between the centroids of two locations. Based on the spreading pathways of different diseases, past researchers had also considered using a water supply map [Spe11] to group the nearby locations. Traffic routes among major airports, railway and bus stations could also help consider grouping two locations which are not geometrically nearby but are in fact closely connected via transportation.

In our case study, we are studying at the outbreak distribution among the inner London area, in which there are 14 boroughs involved. Inner London is smaller than Outer London both in terms of population and area, but the density of population is more than double that of Outer London. This would yield much higher risk for people living in inner London area to get exposed to various diseases, especially for those (e.g. coronavirus) can be easily spread through everyday human interactions. That's why we had our focus in this particular area.

List of boroughs

- Camden

- Greenwich
- Hackney
- Hammersmith and Fulham
- Islington
- Kensington and Chelsea
- Lambeth
- Lewisham
- Southwark
- Tower Hamlets
- Wandsworth
- Westminster
- Newham
- City of London

The figure below 1 shows the geographic allocation of 14 boroughs within the inner London area.

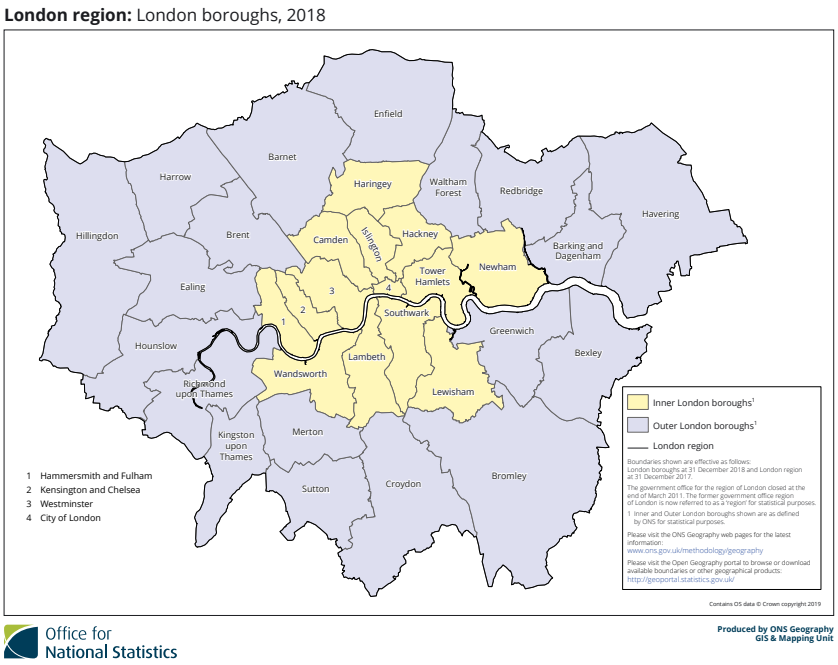

Figure 1: London region: London boroughs 2018

3.2.2 Assumptions

After considering neighbouring effects from nearby locations, two assumptions made in Spatial Independent Model have been relaxed:

- Outbreak distribution is spatially independent in each location;
- Probability p of having higher risk of an outbreak is constant;

In additional to that, in terms of measuring the number of infectious cases Y for different location at a fixed time point, we assume that:

- All locations are assumed to have the same population size

, which would make the outcome variable Y be completely dependent on the rate parameter $lambda$, without considering any variation of population sizes in different locations.

3.2.3 Dynamic Probability Function

Here in Spatial Dependent Model, we construct a dynamic probability function to reflect dynamic risk level. To consider all neighboring effects for each location j to compute x_j, we let our probability function p_j depend on the set $x_{-j} = (x_1, ..., x_k)$, where k stands for the number of locations within the study region. i.e. the Inner London Area ($k = 14$). Hence, we have our modified probability distribution of x_j as $x_j \sim Binomial(1, p(x_{-j}))$.

In our case, the probability function will be consisted of two components: the background isolated risk term and weighted spatio-temporal risk term. When considering different neighboring locations, we assign each neighboring location with inverse distance weight (i.e. d_j^{-1}) to reflect the corresponding spatio-temporal effects.

Formulation Given the inverse distance metric, the dynamic probability function can be formulated as the following:

$$p_j = \gamma + \Delta \cdot \sum_{k=1,k\neq j}^{m} \left(\frac{d_{kj}^{-1}}{\sum_{l=1,l\neq j}^{m} d_{lj}^{-1}} \cdot x_k \right)$$

,where $x_k \in 0, 1$

In particular, by considering timely variable index i, we can denote p_{ij} as the probability of having an outbreak at time i in location j. Hence, we have:

$$p_{ij} = \gamma + \Delta \cdot \sum_{k=1,k\neq j}^{m} \left(\frac{d_{kj}^{-1}}{\sum_{l=1,l\neq j}^{m} d_{lj}^{-1}} \cdot x_{ik} \right)$$

, where x_{ik} is the binary variable that indicates an outbreak or not in location k given current time i. And d_{kj} is denoted as the Euclidean distance from location k to location j.

In many previous applications and studies of Spatial Scan Statistics (such as the circular spatial scan statistics proposed by Kulldorff, and the flexible spatial scan statistic proposed by Tango and Takahashi), the authors had considered detecting possible outbreak clusters using restricted likelihood ratio (frequentist approach) or posterior probability of having an outbreak (Bayesian approach). Therefore, in real application, we may adopt Spatially scan statistics to identify sub-regions, and then assign different probabilities/risk levels for different spatial clusters.

For the purpose of reducing computational cost, we simplify the model here by assuming that γ is the background probability of having an isolated outbreak for all regions. We pick reasonable γ (i.e. background risk) from [0.1,0.2].

Δ measures the spatio-temporal effect from the neighboring locations. In the worst case of scenario where all the neighboring locations are having an outbreak, the equation becomes the following. We can compute the effect as $\Delta = max(p_j) - \gamma$.

$$\begin{aligned} max(p_j) =& \gamma + \Delta \cdot \sum_{i \neq j} \frac{d_{ji}^{-1}}{\sum_{i \neq j} d_{ji}^{-1}} \\ =& \gamma + \Delta \cdot 1 \\ =& \gamma + \Delta \end{aligned}$$

Dirichlet distribution In spatially independent model, outbreak probability p follows a beta distribution $Beta(a, b)$. However after introducing dynamic probability function, an additional hierarchical Bayesian level is added to our outbreak model, in which p is now depending on the choice of a vector of parameters:

$$\begin{pmatrix} \gamma \\ \Delta \end{pmatrix}$$

Note that p no longer follows a beta distribution $Beta(a, b)$.

Alternatively, we let the new parameter vector θ follows a multivariate beta distribution (also named as Dirichlet distribution). The Dirichlet distribution is a family of continuous multivariate probability distributions parameterized by a vector a of positive reals. We denote the distribution of θ as $Dir(a)$, where $\theta = (\gamma, \Delta, 1 - \gamma - \Delta)$.

In a Dirichlet distribution, the support over the probability simplex is defined as:

$$S_K = \{\theta : 0 \leq \theta_k \leq 1, \sum_{k=1}^{K} \theta_k = 1\}$$

The probability density function is defined as follows:

$$Dir(\theta|a) \triangleq \frac{1}{B(a)} \prod_{k=1}^{K} \theta_k^{a_k - 1} \mathbb{1}(\theta \in S_K)$$

, where $B(a_1, ..., a_K)$ is the natural generalisation of the beta function to K variables:

$$B(a) = \frac{\prod_{k=1}^{K} \Gamma(a_k)}{\Gamma(a_0)}$$

, where $a_0 = \sum_{k=1}^{K} a_k$.

We can also write as:

$$Dir(\theta|a) \triangleq \frac{\Gamma(\sum_{k=1}^{K} a_k)}{\prod_{k=1}^{K} \Gamma(a_k)} \prod_{k=1}^{K} \theta_k^{a_k - 1} \mathbb{1}(\theta \in S_K)$$

In our case, the parameter vector θ follows a degree 3 Dirichlet distribution ($K = 3$):

$$\theta \sim Dir_3(a)$$

, where $a := \{a_1, a_2, a_3\}$ correspondingly with $\theta = (\gamma, \Delta, 1 - \gamma - \Delta)$

3.2.4 Parameters

Having constructed the dynamic probability function, we can re-establish several parameters within the dependent model.

- $Y \sim Poisson(\lambda)$
- $log(\lambda_{ij}) = \alpha + \beta x_{ij}$
- $x_{ij} \sim Bernoulli(p_{ij})$
- $p_{ij} = \gamma + \Delta \cdot \sum_{k \neq j} \frac{d_{kj}^{-1}}{\sum_{k \neq j} d_{kj}^{-1}} \times x_{ij}$
- $\theta \sim Dir_3(a)$

3.2.5 Generate data

After building up the dynamic probability function, we can start to generate our sample data based on the assumptions that we have made.

As discussed previously in Section 3.1.3 on how to on generate data for spatially independent model, here I adopt the choices of those parameters α, β, p (listed in table 3), and use the corresponding findings to construct a comparison study between the two models. (Note that $\Delta = \max(p) - \gamma$, explained in 3.2.3.)

Trials	α	β	γ	Δ
	ln(1)	ln(2)	7/52	
	ln(1)	ln(3)	7/52	
	ln(1)	ln(4)	7/52	
	ln(2)	ln(2)	7/52	
	ln(2)	ln(3)	7/52	
	ln(4)	ln(2)	7/52	

Table 3: Choice of true values in generated dataset

For each choice of the model parameters, I start by picking two areas to have outbreaks, which are the City of London, and Southward because they are in the centre of the studies regions, and thus have a larger likelihood in passing the infections to their neighboring locations. Then, the localised outbreak probabilities are calculated for each locations whenever there is a location's outbreak status being updated. Using those locally updated probabilities, we use function $rbinom()$ in R to randomly generate new x in a new iteration. We keep iterating this computation for given times (e.g. 300 iterations) until the expected neighboring effect becomes stable in the region.

Next, given the total time length as n, we generate x_{ij} for each time i and location j. In particular, we first generate x_{ij} for $j \in (1, 2, ..., 14)$ using the dynamic probability function (which depends on the inverse distance matrix $\mathbf{d}^{-1}$ and parameter set (γ, Δ)), and then iterate for n times for each time i ($i \in (1, 2, ..., n)$).

After we successfully construct the sample data for binary variable X, we can computing the corresponding rate parameter λ for the Poisson distribution that outcome variable Y follows.

$$\lambda = \alpha + \beta x$$

, where each $\lambda_{ij} = \alpha + \beta x_{ij}$.

Using the information we get from λ, we can further generate sample data for variable Y with build-in function $rpois()$ in R.

3.3 Constructing MCMC in independent model

As mentioned above, we have successfully establish a MCMC simulation for our spatial independent model. There are currently only three parameters included in our model, i.e. α, β, p. We denote the full parameter set as ϕ. We also have $x := \{x_i\}$ as dependent variable, representing the existence of an outbreak disease at time i.

3.3.1 Posterior density

As discussed in Bayesian statistics, we can construct an approximation of the posterior density (denoted as $\pi(\phi|Y)$), using the likelihood and prior density. Hence, we have the following:

$$\begin{aligned}
\pi(\phi|Y) &\propto L(Y|\phi) \cdot p(\phi) \\
&= L(Y|\alpha, \beta, x) \cdot p(\alpha) \cdot p(\beta) \cdot p(x|p) \cdot p(p) \\
&= L(Y|\lambda) \cdot p(\alpha) \cdot p(\beta) \cdot p(x|p) \cdot p(p) \text{ , where } \lambda = \alpha + \beta x \\
&= \prod_{i=1}^{n} \{f_{Poisson}(Y_i = y_i, log(\lambda_i) = \alpha + \beta x_i)\} \cdot p(\alpha) \cdot p(\beta) \cdot p(x|p) \cdot p(p) \\
&= \prod_{i=1}^{n} \frac{\lambda_i{}^{y_i} e^{-\lambda_i}}{y_i!} \cdot p(\alpha) \cdot p(\beta) \cdot p(x|p) \cdot p(p)
\end{aligned}$$

We have chosen the prior density of α, β to each follow a Gamma distribution:

$$\begin{aligned}
\alpha &\sim \Gamma(\alpha, 6) \\
\beta &\sim \Gamma(\beta, 6)
\end{aligned}$$

We can also construct the prior densities of $x|p$ and p based on their prior distributions:

$$\begin{aligned}
x|p &\sim Binomial(x, 1, p) \\
p &\sim Beta(a, b)
\end{aligned}$$

, where $\frac{a}{a+b}$ is the frequency of outbreak within a given time period.

Hence, we can continue to compute our target posterior probability function using the prior information:

$$\begin{aligned}
\pi(\phi|Y) &\propto L(Y|\phi) \cdot p(\phi) \\
&= \prod_{i=1}^{n} \frac{\lambda_i{}^{y_i} e^{-\lambda_i}}{y_i!} \cdot p(\alpha) \cdot p(\beta) \cdot p(x|p) \cdot p(p) \\
&= \prod_{i=1}^{n} \frac{\lambda_i{}^{y_i} e^{-\lambda_i}}{y_i!} \cdot f_{Gamma}(\alpha, 6) \cdot f_{Gamma}(\beta, 6) \cdot f_{Binomial}(x, 1, p) \cdot f_{Beta}(a, b) \\
&= \prod_{i=1}^{n} \frac{\lambda_i{}^{y_i} e^{-\lambda_i}}{y_i!} \cdot f_{Gamma}(\alpha, 6) \cdot f_{Gamma}(\beta, 6) \cdot \prod_{i=1}^{n} f_{Binomial}(x_i, 1, p) \cdot f_{Beta}(a, b)
\end{aligned}$$

3.3.2 Update α, β

After constructing the posterior distribution for the model, we begin our simulation by updating α,β with **Metropolis-Hasting** algorithm. We have set both proposal distributions of α, β to be Gaussian normal, i.e.

$$q(\alpha_t) \sim \mathcal{N}(\alpha_{t-1}, 0.2)$$
$$q(\beta_t) \sim \mathcal{N}(\beta_{t-1}, 0.2)$$

Note that we have added extra conditions to reject β when $\beta < ln(2)$. Since we are using a log link function in our Poisson distribution, e^{β} represents the additional risk having an outbreak (Here we set a β minimum of $ln(2)$ to ensure that model has at least twice risk during an outbreak).

Then, we can compute the acceptance ratio given the new proposed values of α, β.

$$\begin{aligned}
\text{Probability} &= min\{1, \frac{\pi(\phi^*|Y)}{\pi(\phi|Y)}\} \\
&= min\{1, \frac{L(Y|\phi^*) \cdot p(\alpha^*) \cdot p(\beta^*) \cdot p(x|p) \cdot p(p) \cdot q(\alpha^*, \beta^*|\alpha, \beta)}{L(Y|\phi) \cdot p(\alpha) \cdot p(\beta) \cdot p(x|p) \cdot p(p) \cdot q(\alpha, \beta|\alpha^*, \beta^*)}\} \\
&= min\{1, \frac{L(Y|\phi^*) \cdot p(\alpha^*) \cdot p(\beta^*) \cdot q(\alpha^*, \beta^*|\alpha, \beta)}{L(Y|\phi) \cdot p(\alpha) \cdot p(\beta) \cdot q(\alpha, \beta|\alpha^*, \beta^*)}\} \\
&\text{, where } \phi^* = (\alpha^*, \beta^*, x) \\
&= min\{1, \frac{L(Y|\alpha^*, \beta^*) \cdot p(\alpha^*) \cdot p(\beta^*)}{L(Y|\alpha, \beta) \cdot p(\alpha) \cdot p(\beta)}\} \\
&\text{, where the proposal density } q(\alpha^*, \beta^*|\alpha, \beta) = q(\alpha, \beta|\alpha^*, \beta^*) \\
&\text{given the symmetric proposal density of } (\alpha, \beta) \\
&= min\{1, \frac{\pi(\alpha^*, \beta^*|Y)}{\pi(\alpha, \beta|Y)}\}
\end{aligned}$$

After that, we generate a random number from a uniform distribution and compare it with our acceptance ratio. The idea is to accept our proposal values with probability (constructed above). Once we decide whether to accept or reject the proposal values, we move on to update p and indicator variable x, which represents a set of variables $(x_1, x_2, ..., x_n)$. n is the time length (in weeks) that we are observing using the model.

3.3.3 Update probability p

Since x has a conditional distribution on probability p, we first update the Beta distribution that p follows, from

$$p \sim Beta(a, b)$$

to

$$p|x \sim Beta(a + n_{x=1}, b + n_{x=0})$$

, given a new mean of

$$\frac{a + n_{x=1}}{a + b + n}$$

The mean here represents the average probability that has a higher risk of outbreak in a given time period. Then we sample new p, which is proposed from the new Beta distribution.

3.3.4 Update variable x

After that, we can compute the marginal posterior density given x, and use it to sample new x from Bernoulli distribution ($x \sim Bernoulli(p)$).

According to the Bayes' theorem, we can compute the posterior density of $x_i = 1$ and $x_i = 0$:

$$\mathcal{P}(x_i = 1|y_i, \phi) \propto \frac{\mathcal{L}(Y_i = y_i|x_i = 1, \alpha, \beta) \cdot p(x_i = 1|p)}{f_{Poisson}(y|log(\lambda_i) = \alpha + \beta) \cdot p} \tag{1}$$

$$\mathcal{P}(x_i = 0|y_i, \phi) \propto \frac{\mathcal{L}(Y_i = y_i|x_i = 0, \alpha, \beta) \cdot p(x_i = 0|p)}{f_{Poisson}(y|log(\lambda_i) = \alpha) \cdot (1 - p)} \tag{2}$$

Since we know that the sum of posterior density of $x_i = 1$ and $x_i = 0$ is 1, we can compute the probability easily, using the equations above:

$$\mathcal{P}(x_i = 1|y_i, \phi) = \frac{(1)}{(1) + (2)}$$

To find the marginal posterior density for x, we can compute the normalising constant c first:

$$c = 1/(\mathcal{L}(Y_i = y_i|x_i = 1, \alpha, \beta) \cdot p(x_i = 1|p) + \mathcal{L}(Y_i = y_i|x_i = 0, \alpha, \beta) \cdot p(x_i = 0|p))$$

And then we can use the normalising constant c to further compute the posterior density of $x_i = 1$.

$$\begin{aligned}\mathcal{P}(x_i = 1|y_i, \phi) &= \frac{(1)}{(1) + (2)} \\ &= c \cdot \mathcal{L}(Y_i = y_i|x_i = 1, \alpha, \beta) \cdot p(x_i = 1|p)\end{aligned}$$

Using this marginal posterior density, we sample new x_i from Bernoulli distribution ($x_i \sim Bernoulli(p^*)$, where $p^* = \mathcal{P}(x_i = 1|y_i, \phi)$).

LogSumExp trick In terms of implementing the algorithm in programming language R, we need to cautious that posterior density of x would be very small, which makes it extremely difficult for the workspace in R to store those numeric values. Alternatively, a trick using the LogSumExp function (LSE) is adopted to overcome the numeric tolerance using the LogSumExp transformation.

In this case, we want to compute $\mathcal{P}(x_i = 1|y_i, \phi)$ using (1) and (2). For the sake of simplicity, here I denote (1), (2) as p_1 and p_2. Then, we have:

$$\begin{aligned}\mathcal{P}(x_i = 1|y_i, \phi) &= \frac{p_1}{p_1 + p_2} \\ &= \frac{e^{log(p_1)}}{e^{log(p_1)} + e^{log(p_2)}}\end{aligned}$$

, where we denote $M = max(log(p_1), log(p_2))$

$$\begin{aligned}\frac{e^{log(p_1)}}{e^{log(p_1)} + e^{log(p_2)}} &= \frac{e^{log(p_1)+M-M}}{e^{log(p_1)+M-M} + e^{log(p_2)+M-M}} \\ &= \frac{e^M e^{log(p_1)-M}}{e^M(e^{log(p_1)-M} + e^{log(p_2)-M})} \\ &= \frac{e^{log(p_1)-M}}{e^{log(p_1)-M} + e^{log(p_2)-M}}\end{aligned}$$

without losing any generality, let $M = log(p_1)$. Then, we have

$$\mathcal{P}(x_i = 1|y_i, \phi) = \frac{1}{1 + e^{log(p_2)-M}}$$

Using this trick, R would be able to compute the posterior density with enough numeric tolerance.

3.4 Constructing MCMC in spatio-temporal dependent model

Comparing with the MCMC simulation we did for our spatial independent model above, we introduce two additional parameters γ, Δ in our dependent model to represent dynamic probability p. We also have $x := \{x_{ij}\}$ as dependent variable, indicating the existence of an outbreak disease in each location j at time i.

3.4.1 Posterior density

As discussed in Bayesian statistics, we can construct an approximation of the posterior density (denoted as $\pi(\phi|Y), \phi = (\alpha, \beta, \gamma, \Delta)$), using the likelihood and prior density. Hence, we have the following:

$$\begin{aligned}\pi(\phi|Y) &\propto L(Y|\phi) \cdot p(\phi) \\ &= L(Y|\alpha, \beta, x, \theta) \cdot p(\alpha) \cdot p(\beta) \cdot p(x|\theta) \cdot p(\theta)\end{aligned}$$

Likelihood Similarly to the likelihood we have for spatially independent model, we have the following:

$$\begin{aligned}L(Y|\alpha, \beta, x, \theta) &= f_{Poisson}(Y = y, \lambda = \alpha + \beta x) \\ &= \prod_{i=1}^{n}\prod_{j=1}^{m} f_{Poisson}(Y_{ij} = y_{ij}, \lambda_{ij} = \alpha + \beta x_{ij})\end{aligned}$$

Note that here we treat x as the parameter of our Bayesian Hierarchical model instead of the sampled data.

Prior density We have chosen the prior density of α, β to each follow a Gamma distribution:

$$\alpha \sim \Gamma(\alpha, 6)$$
$$\beta \sim \Gamma(\beta, 6)$$

We can also construct the prior density of θ based on the prior distribution:

$$\theta \sim Dir_3(\theta, a_\theta)$$

However, since we have modified the outbreak probability $p|\theta$ to have a strong dependence with x. In particular, we say $p_{ij}|\theta$ depends on $x_{i(-j)}|\theta$. Having such dependence would it much more difficult to compute the conditional probability $p(x|\theta)$.

Instead we proposed a **pseudo marginal likelihood** for $x|\theta$ by assuming independence among different x_{ij}. Therefore, we can write down the probability $p(x|\theta)$ based on the prior distribution $x|\theta \sim Binomial(x, 1, p(\theta))$, and $\theta = (\gamma, \Delta, 1 - \Delta)$.

$$\begin{aligned}p(x|\theta) &= f_{Binomial}(x, 1, p|\theta) \\ &= \prod_{i=1}^{n}\prod_{j=1}^{m} f_{Binomial}(x_{ij}, 1, p|\theta)\end{aligned}$$

Computation Hence, we can continue to compute our target posterior probability function using the prior information:

$$\begin{aligned}
\pi(\phi|Y) &\propto L(Y|\phi) \cdot p(\phi) \\
&= \prod_{i=1}^{n}\prod_{j=1}^{m} f_{Poisson}(Y_{ij} = y_{ij}, \lambda_{ij} = \alpha + \beta x_{ij}) \cdot p(\alpha) \cdot p(\beta) \cdot \prod_{i=1}^{n}\prod_{j=1}^{m} f_{Binomial}(x_{ij}, 1, p|\theta) \cdot p(\theta) \\
&= \prod_{i=1}^{n}\prod_{j=1}^{m} f_{Poisson}(Y_{ij} = y_{ij}, \lambda_{ij} = \alpha + \beta x_{ij}) \cdot f_{Gamma}(\alpha, 6) \cdot f_{Gamma}(\beta, 6) \\
&\cdot \prod_{i=1}^{n}\prod_{j=1}^{m} f_{Binomial}(x_{ij}, 1, p|\theta) \cdot f_{Dir}(\theta, a_\theta)
\end{aligned}$$

3.4.2 Update α, β

After constructing the posterior distribution for the model, we begin our simulation by updating α,β with **Metropolis-Hasting** algorithm. Similarly, we have added extra conditions to reject β when $\beta < ln(2)$.

Both proposal density of α, β are following a Gaussian normal distribution, i.e.

$$\begin{aligned}
q(\alpha_t) &\sim \mathcal{N}(\alpha_{t-1}, 0.2) \\
q(\beta_t) &\sim \mathcal{N}(\beta_{t-1}, 0.2)
\end{aligned}$$

We can compute the acceptance ratio given the new proposed values of α, β.

$$\begin{aligned}
\text{Probability} &= min\{1, \frac{\pi(\phi^*|Y)}{\pi(\phi|Y)}\} \\
&= min\{1, \frac{L(Y|\phi^*) \cdot p(\alpha^*) \cdot p(\beta^*) \cdot p(x|\theta) \cdot p(\theta) \cdot q(\alpha^*, \beta^*|\alpha, \beta)}{L(Y|\phi) \cdot p(\alpha) \cdot p(\beta) \cdot p(x|\theta) \cdot p(\theta) \cdot q(\alpha, \beta|\alpha^*, \beta^*)}\} \\
&= min\{1, \frac{L(Y|\phi^*) \cdot p(\alpha^*) \cdot p(\beta^*) \cdot q(\alpha^*, \beta^*|\alpha, \beta)}{L(Y|\phi) \cdot p(\alpha) \cdot p(\beta) \cdot q(\alpha, \beta|\alpha^*, \beta^*)}\} \\
&\text{, where } \phi^* = (\alpha^*, \beta^*, \theta) \text{ and } \theta = (\gamma, \Delta, 1-\gamma-\Delta) \\
&= min\{1, \frac{L(Y|\alpha^*, \beta^*) \cdot p(\alpha^*) \cdot p(\beta^*)}{L(Y|\alpha, \beta) \cdot p(\alpha) \cdot p(\beta)}\} \\
&\text{, where the proposal density } q(\alpha^*, \beta^*|\alpha, \beta) = q(\alpha, \beta|\alpha^*, \beta^*) \\
&\text{given the symmetric proposal density of } (\alpha, \beta) \\
&= min\{1, \frac{\pi(\alpha^*, \beta^*|Y)}{\pi(\alpha, \beta|Y)}\}
\end{aligned}$$

Once we decide whether to accept or reject the proposal values, we move on to update γ, Δ and indicator variable $x := \{x_{ij}, i \in (1, ..., n) \text{ and } j \in (1, ..., m)\}$. n is the time length that we are observing using the model and m is the number of locations we are sampling from.

3.4.3 Update γ, Δ

The Dirichlet distribution is a conjugate prior for the multinomial distribution. This means that if the prior distribution of the multinomial parameters is Dirichlet then the posterior distribution is also a Dirichlet distribution (with parameters different from those of the prior). The benefit of this is that (a) the posterior distribution is easy to compute and (b) it in some sense is possible to quantify how much our beliefs have changed after collecting the data.

(Dirichlet distribution, nice properties of prior and posterior, relationship with beta distribution. And why as $\lambda \to \infty$, the distribution becomes approximately the Gaussian distribution, based on the theory of central limit theorem)

With Dirichlet distribution, we can then represent the probabilities for the multinoomial probability vector $\theta = (\gamma, \Delta, 1-\gamma-\Delta)$. And we have the following properties:

- $\sum \gamma, \Delta, 1-\gamma-\Delta = 1$
- $\gamma, \Delta, 1-\gamma-\Delta \geq 0$

Recall that a conjugate priors has the same form as the posterior distribution, we can compute an approximation (proportional solution) of posterior density using likelihood function from multinomial distribution and prior density derived from Dirichlet distribution.

$$\begin{aligned} p(\theta|X) &\propto p(X|\theta)p(\theta) \\ &\propto \prod_{k=1}^{K} \theta^{n_k} \frac{\Gamma(\sum_{k=1}^{K} a_k)}{\prod_{k=1}^{K} \Gamma(a_k)} \prod_{k=1}^{K} \theta_k^{a_k - 1} \\ &\propto \frac{\Gamma(\sum_{k=1}^{K} a_k)}{\prod_{k=1}^{K} \Gamma(a_k)} \prod_{k=1}^{K} \theta_k^{a_k + n_k - 1} \\ &\propto Dir(a+n) \end{aligned}$$

, where $n := \{n_1, ..., n_K\}$ is a vector of the observed counts.

The density of the proposal θ^* can be written as $\theta^* \sim Dir(a \cdot \theta + a_\theta)$. To update θ (i.e. γ, Δ), we apply Metropolis-Hasting algorithm to compute the acceptance probability:

$$\begin{aligned} \text{Probability} &= \frac{\pi(\theta^*|x) \cdot q(\theta|a \cdot \theta^* + a_\theta)}{\pi(\theta|x) \cdot q(\theta^*|a \cdot \theta + a_\theta)} \\ &= \frac{\mathcal{L}(x|\theta^*) \cdot p(\theta^*|a_\theta) \cdot q(\theta|a \cdot \theta^* + a_\theta)}{\mathcal{L}(x|\theta) \cdot p(\theta|a_\theta) \cdot q(\theta^*|a \cdot \theta + a_\theta)} \\ &= \frac{\mathcal{L}(x|\theta^*) \cdot Dir(\theta^*|a_\theta) \cdot Dir(\theta|a \cdot \theta^* + a_\theta)}{\mathcal{L}(x|\theta) \cdot Dir(\theta|a_\theta) \cdot Dir(\theta^*|a \cdot \theta + a_\theta)} \end{aligned}$$

, where $\mathcal{L}(\cdot), p(\cdot), q(\cdot)$ represent the likelihood, prior density and proposal density respectively. Then, we can update the new proposal of θ with the acceptance probability.

3.4.4 Update variable x

After that, we can compute the marginal posterior density given x, and use it to sample new x from Bernoulli distribution ($x \sim Bernoulli(p)$).

According to the Bayes' theorem, we can compute the posterior density of $x_{ij} = 1$ and $x_{ij} = 0$:

$$\mathcal{P}(x_{ij} = 1|y_{ij}, \phi) \propto \frac{\mathcal{L}(Y_{ij} = y_{ij}|x_{ij} = 1, \alpha, \beta) \cdot p(x_{ij} = 1|p_{ij})}{f_{Poisson}(y|log(\lambda_{ij}) = \alpha + \beta) \cdot p_{ij}} \tag{3}$$

$$\mathcal{P}(x_{ij} = 0|y_{ij}, \phi) \propto \frac{\mathcal{L}(Y_{ij} = y_{ij}|x_{ij} = 0, \alpha, \beta) \cdot p(x_{ij} = 0|p_{ij})}{f_{Poisson}(y|log(\lambda_{ij}) = \alpha) \cdot (1 - p_{ij})} \tag{4}$$

Since we know that the sum of posterior density of $x_{ij} = 1$ and $x_{ij} = 0$ is 1, we can compute the probability easily, using the equations above:

$$\mathcal{P}(x_{ij} = 1|y_{ij}, \phi) = \frac{(3)}{(3) + (4)}$$

In terms of implementing the algorithm in programming language R, we can compute the normialising constant c first:

$$c = 1/(\mathcal{L}(Y_{ij} = y_{ij}|x_{ij} = 1, \alpha, \beta) \cdot p(x_{ij} = 1|p_{ij}) + \mathcal{L}(Y_{ij} = y_{ij}|x_{ij} = 0, \alpha, \beta) \cdot p(x_{ij} = 0|p_{ij}))$$

And then we can use the normalising constant c to further compute the posterior density of $x_{ij} = 1$.

$$\begin{aligned}\mathcal{P}(x_{ij} = 1|y_{ij}, \phi) &= \frac{(3)}{(3) + (4)} \\ &= c \cdot \mathcal{L}(Y_{ij} = y_{ij}|x_{ij} = 1, \alpha, \beta) \cdot p(x_{ij} = 1|p_{ij})\end{aligned}$$

Using this marginal posterior density, we sample new x_{ij} from Bernoulli distribution ($x_{ij} \sim Bernoulli(p^*)$, where $p^* = \mathcal{P}(x_{ij} = 1|y_{ij}, \phi)$). Note here I adopt the same LogSumExp trick used in the simple model 3.3.4

4 Model Evaluation

4.1 Spatially Independent Model

4.1.1 Convergence diagnosis

We investigate the convergence of the Markov chain using different number of iterations and burn-In period. Eventually, we have set our MCMC to have 30000 iterations and 5000 burn-In period. The trace plots 2 (histograms and Markov chains) are plotted to visualise the convergence.

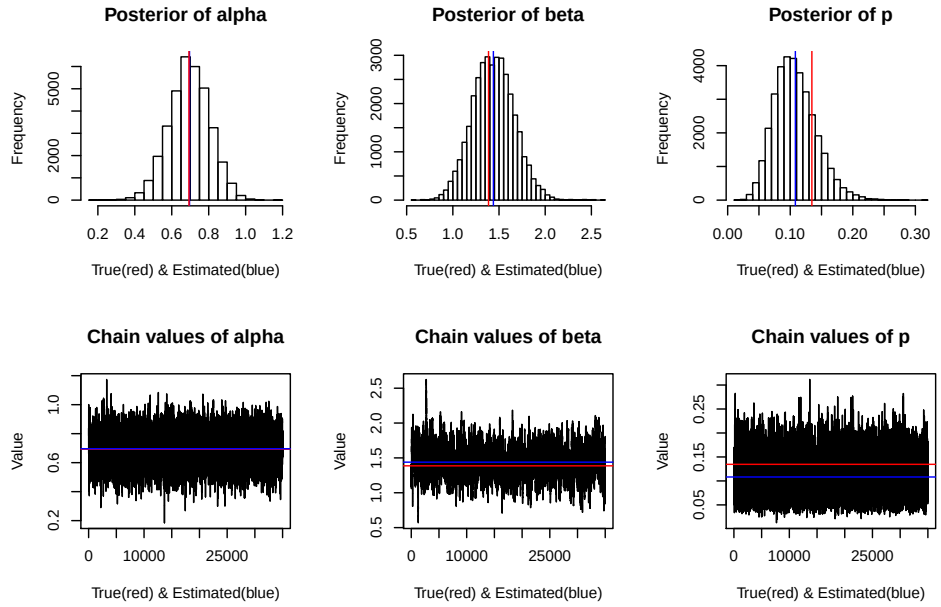

Figure 2: Trace Plot of Parameters for Spatial Independent Model

4.1.2 ROC performance

By looking at the estimated outbreak probability plot for each x_i 3, we can find that a higher value of true β would give a much better prediction on x_i. Note in the plot, the red and blue lines represent for the (0.25%,99.75%) quantile value of outbreak probability, and the vertical green lines represent the week

that is having an outbreak. For those weeks that have an outbreak (with true x_i being 1), sampled x_i would tend to have a very large posterior density indicating an outbreak. In the case of higher β, the ROC curve 4 that we plot would receive a higher AUC (area under curve) in between (0.98,1.00).

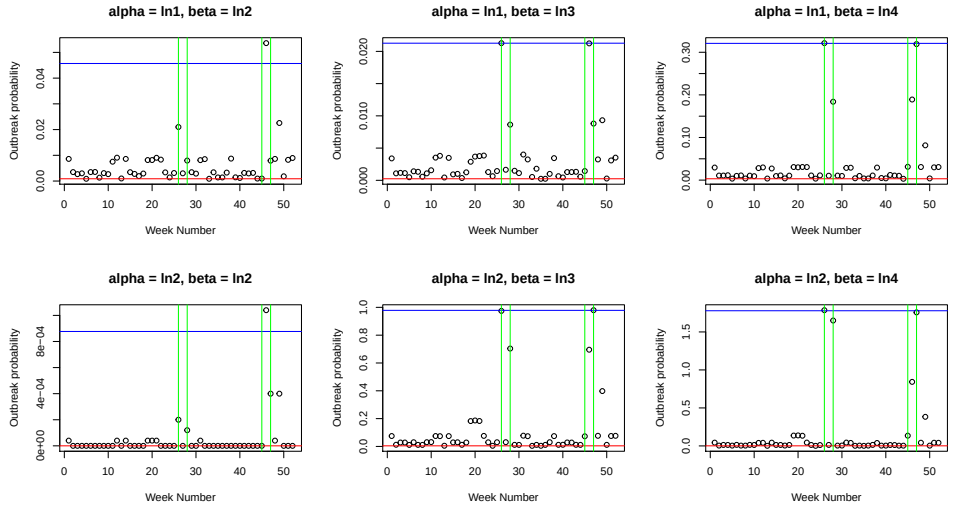

Figure 3: Estimated outbreak probability using Spatial Independent Model

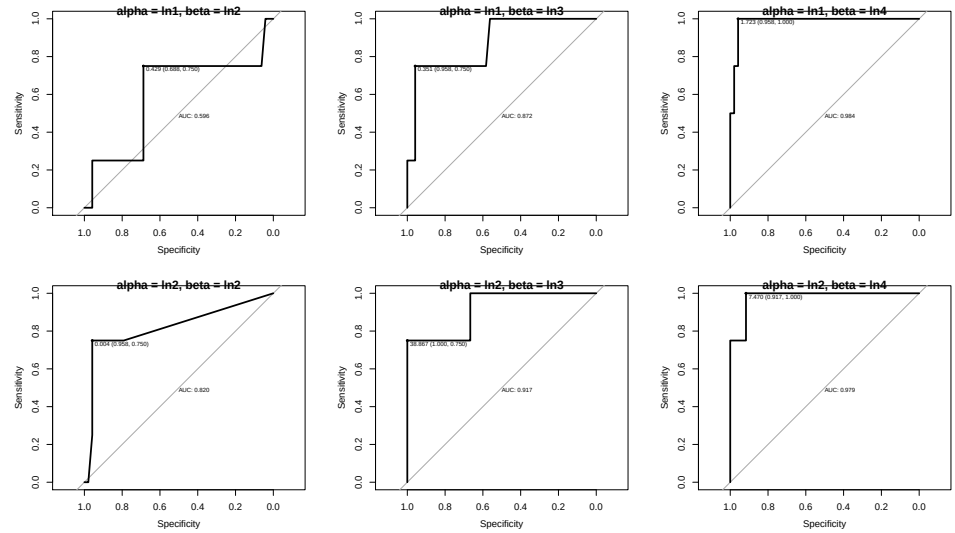

Figure 4: ROC curves for Spatial Independent Model

The summary table 4 shows a comparison of the AUC (area under ROC curve) for different generated data (in trials). Given the table, we can see that the model would have a better predictability on outbreaks with greater value of β in (ln(4),ln(10)) (i.e. greater discrimination between whether having an outbreak or not), resulting in a higher AUC.

However, when we use a smaller value of true β in (ln(1),ln(3)), we begin to find much more samples lying in the grey area, which is indicated by the quantile range that I set (0.25%,99.75%) based on posterior density of sampled x_t. In such scenarios, it becomes much more difficult for the model to capture those outbreaks (in the grey area), resulting in a poorer outbreak predictability.

Also, we may find that the variation in the value of α doesn't appear to have a large effect to the overall outbreak predictability. Intuitionally, the reason is that e^{α} only represents as the number of infectious cases that would occur within a given time period (e.g. in this case, a week)

Trials	α	β	p	ROC performance
	ln(1)	ln(2)	4/52	0.90
	ln(1)	ln(2)	7/52	0.61
	ln(1)	ln(3)	7/52	0.83
	ln(1)	ln(4)	4/52	0.99
	ln(1)	ln(4)	7/52	0.92
	ln(2)	ln(2)	4/52	0.50
	ln(2)	ln(2)	7/52	0.83
	ln(2)	ln(3)	7/52	0.96
	ln(2)	ln(4)	4/52	1.00
	ln(2)	ln(4)	7/52	0.98
	ln(4)	ln(2)	4/52	0.98
	ln(4)	ln(2)	7/52	0.86
	ln(4)	ln(4)	4/52	1.00
	ln(4)	ln(4)	7/52	1.00
	ln(4)	ln(10)	7/52	1.00
	ln(4)	ln(10)	15/52	1.00

Table 4: ROC Performance for Spatially Independent Model

In order to study the additional effect that spatial information would have on predicting outbreaks, here we want to focus our interest in those scenarios where it makes the model more difficult to detect an outbreak. In the next stage, by removing the independency among the regions, we want to enhance our detection using extra information from localised neighbors, where a region that is having an outbreak ($x_{i,j} = 1$) would increase the probability of its neighboring region to also have an outbreak ($x_{i,j+1}, x_{i,j-1} = 1$).

4.2 Spatially Dependent Model

4.2.1 Convergence diagnosis

In the case of implementing MCMC methods on Spatially dependent model, we have a much heavier burden in terms of the total computational cost (or time). After investigating the convergence of the Markov chain using different number of iterations and burn-In period, we have set our MCMC to have 3000 iterations and 500 burn-In period.

The trace plots 5 (Markov chains after the burn-in period) are plotted to visualise the convergence. It appears that all four parameters are mixing nicely after the burn-in period.

4.2.2 ROC performance

By looking at the cumulative sum plot for x 9, we can find that a higher value of true β in (ln(20),ln(200)) would give a much better sampling for x_t. In particular for those weeks with true x_t being 1, sampled x_t would tend to have a very large posterior density (close to 1) indicating an outbreak. In such cases, the ROC curve **??** that we plot received a perfect AUC (area under curve) in between (0.99,1.00).

However, when we use a smaller value of true β in (ln(1),ln(3)), we begin to find much more samples lying in the grey area in which we set a quantile range (0.25%,99.75%) based on posterior density of

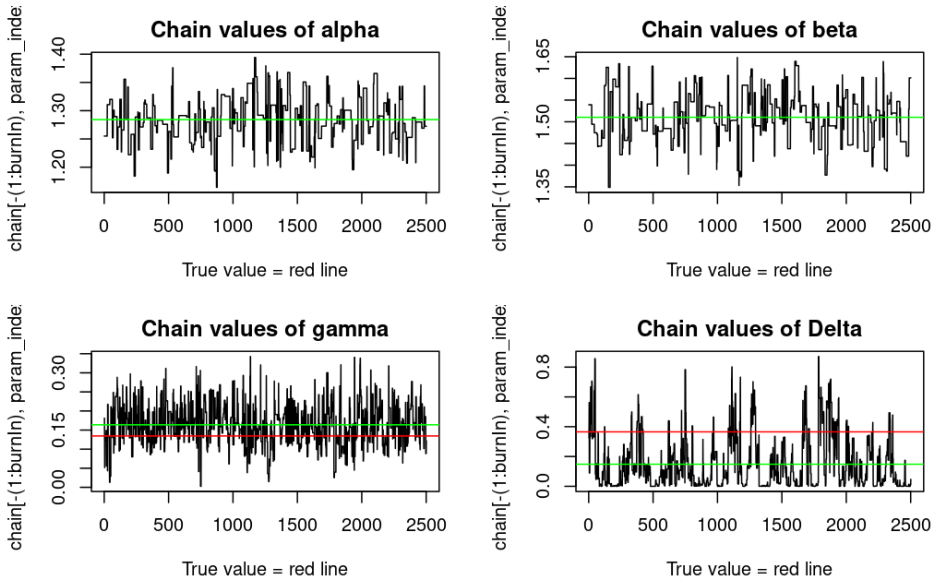

Figure 5: Trace Plot of Parameters for Spatial Dependent Model

sampled x_t. In such scenarios, the model tends to have a poorer outbreak predictability, resulting in a smaller AUC.

From the table 5, we may have similar findings regarding the changes in the value of β.

Trials	α	β	γ	Δ	ROC (spatial)	ROC (simple)
	ln(1)	ln(2)	7/52		0.701	0.668
	ln(1)	ln(3)	7/52		0.525	0.809
	ln(1)	ln(4)	7/52		0.509	0.922
	ln(2)	ln(2)	7/52			0.802
	ln(2)	ln(3)	7/52		0.513	0.919
	ln(4)	ln(2)	7/52			0.867

Table 5: Choice of true values in generated dataset

5 Case Study: Coronavirus in Maryland

5.1 Background

Given the context of the coronavirus, outbreak data in recent years has once again draw our attention to the infectious disease. In the public health sector, the state of Maryland, US has been monitoring a collection of positive COVID-19 test results that have been reported each day by the local health department via the ESSENCE system. The data is recorded for each individual counties in Maryland, from Mar 15th, 2020 til present.

5.1.1 Data

As mentioned above, the dataset was published and maintained by `opendata.maryland.gov`. It contains the cumulative number of positive COVID-19 cases among 24 Maryland counties within the Maryland jurisdiction (fig 6). New cases were updated in the system at 10:00am every day.

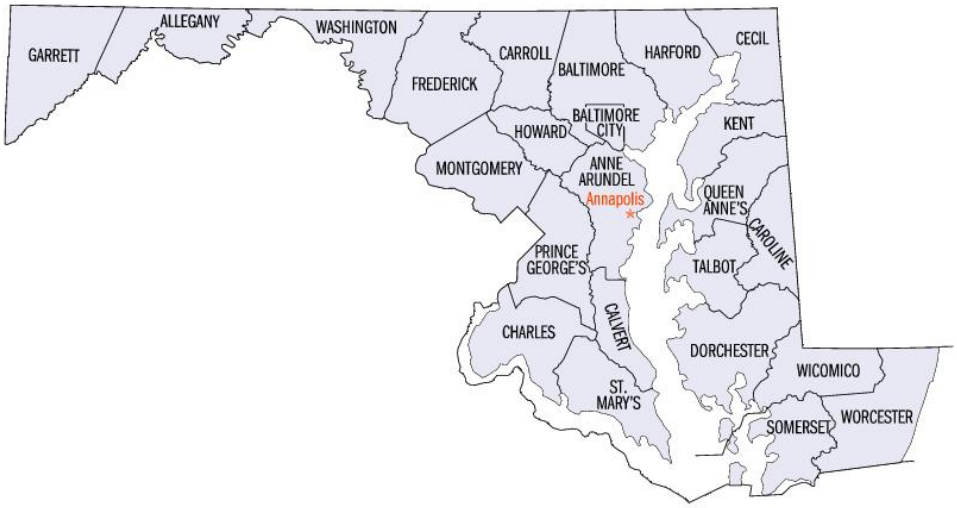

Figure 6: Map of Maryland Counties

There are some missing data (less than 1% of the whole dataset), in which I have replaced them with 0. To study the distribution of the Covid disease in Maryland, I converted the cumulative amounts of cases into a weekly frequency format, in which I computed the number of new weekly cases for each county. By doing so, any extreme daily updates would not be too sensitive to the underlying model, and it also helps reduce the computational expense for MCMC methods.

5.1.2 Infection trend

When looking at the infection trends (fig 7,8) for different counties in Maryland (from year 2020 to 2022), a similar trend is found for all counties. In particular, there are two extreme peaks with large amount of positive cases being tested, which lies in around 2021 January to February and 2021 December to 2022 January. This phenomenon is also shared commonly with other western countries across the world. The red line in each plots represents the average number of cases for each county.

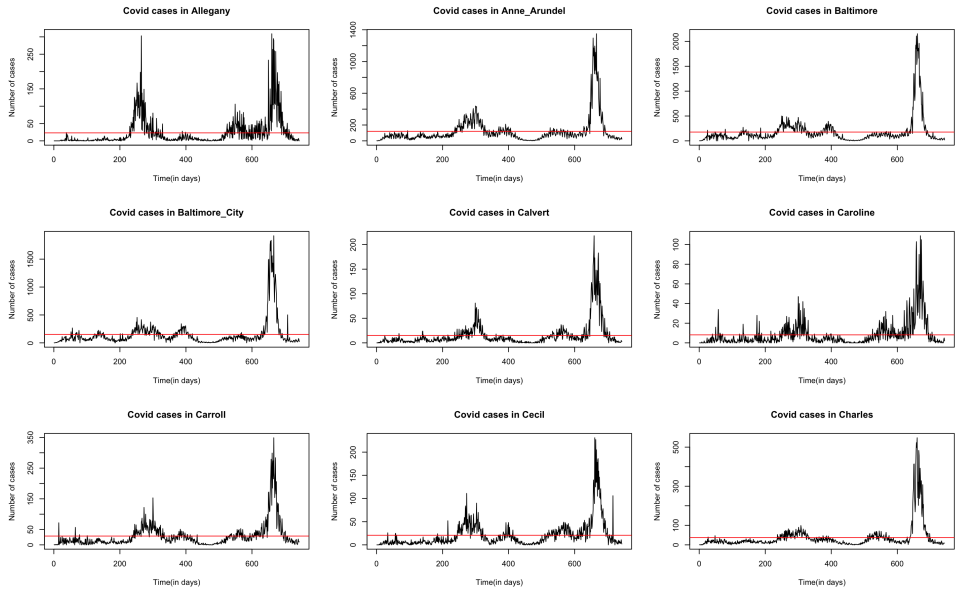

Figure 7: Infection trend for different counties (1)

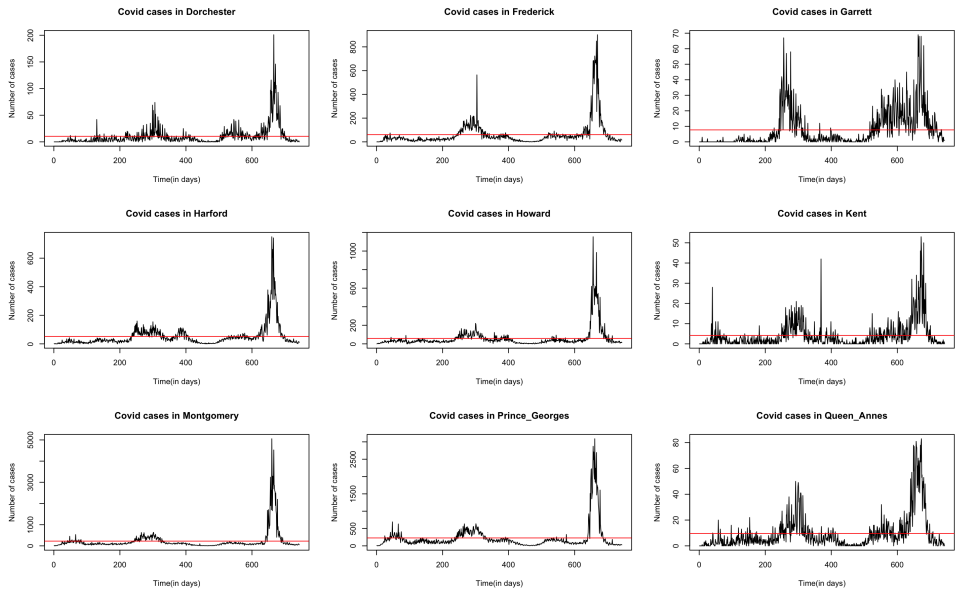

Figure 8: Infection trend for different counties (2)

5.1.3 Time of interest

In order to conduct an explicit study on the spatial-temporal effect of the outbreak model, I chose a fixed time period (2021 November to 2022 March) to characterise the distribution. Given the possibility that the population size could get dynamic over the years (2020 to 2022), a fixed time period would restrict such variation. And MCMC methods are implemented over the chosen data.

5.2 Model formulation

5.2.1 Outbreak Indicator

Here, we are making an assumption that an outbreak is occurred only when it's larger than 86% quantile value of the number of positive cases for each county. Take Baltimore county for an example, if 86% quantile value is 200 cases per week, than any week that is having a larger number of cases would be marked as a temporal outbreak. The value of this threshold comes from the estimated outbreak probability p in spatially independent model, or γ in spatially dependent model, which equals to $\frac{7}{52} = 0.134$. Therefore, I took a threshold at around $1 - \frac{7}{52} \simeq 86\%$.

5.2.2 Distance metric

Similar to the distance metric of the inner London area 3.2.1, I constructed a matrix to store the geographical distance between each two counties in Maryland. Given the fact that the most common way of transportation in Maryland, US is to drive in private vehicles, I am using the driving distance (in miles) from Google API database as the measurement.

5.2.3 Poisson rate parameter

In terms of the model parameters, one slight modification that I made is the initial rate parameter λ in the Poisson distribution which outcome variable Y follows. In the Maryland dataset, since the number of cases is highly correlated to the overall population size of each county, the rate parameter I use for the

Poisson distribution is now represented as $\lambda_j n_j$, in which n_j denotes as the population size for county j and λ_j is representing the rate parameter of having positive cases per each person.

Note here that each county is assumed to have the same λ_j for each individual. In other words, the population size n is treated as an offset in the rate parameter, and the underlying Poisson distribution varies with the population size in the county. Hence, by considering the difference in the population size, the threshold of having an outbreak would be different for each county.

5.3 Choosing startvalues

Given the high computational expense of implementing the MCMC methods in Bayesian hierarchical models, here I aim to find estimations of the model parameters based on the principle of Maximum Likelihood estimation, and use those estimates as the starting values in the MCMC simulation.

5.3.1 Choosing α, β

Based on the setting of our outbreak models, the value of λ given x is assumed to be constant, which means in the case of Maryland data, both $\lambda_{per,x=0}$ (when there is no outbreak) and $\lambda_{per,x=1}$ (when there is an outbreak) are fixed. And we know that:

$$\begin{aligned}\lambda_{per,x=0} &= e^{\alpha} \\ \lambda_{per,x=1} &= e^{\alpha} \cdot e^{\beta}\end{aligned}$$

Note that here λ_{per} refers to the number of cases that each person would have within a week. This gives:

$$\begin{aligned}\alpha &= \log(\lambda_{per,x=0}) \\ \beta &= \log(\frac{\lambda_{per,x=1}}{\lambda_{per,x=0}}) \\ &= \log(\lambda_{per,x=1}) - \log(\lambda_{per,x=0})\end{aligned}$$

Using the principle of Maximum Likelihood estimation, we can estimate α, β as the following:

$$\begin{aligned}\alpha &= \log(\text{mean}(\lambda_{per,x=0})) \\ &= \log(0.52) \\ \beta &= \log(\text{mean}(\lambda_{per,x=1})) - \alpha \\ &= \log(4.36)\end{aligned}$$

Because our estimation on α is approximately negative, we need to set an additional lower bound $alpha.min$ to restrict the proposal of α, so that the prior distribution ($\alpha \sim \Gamma(\alpha, 6)$) still holds.

5.3.2 Choosing γ, Δ

Since our outbreak probability is consist of γ, Δ, where γ is representing the independent/background probability that an isolated outbreak occurs in one of the counties. Therefore, I took the average of the outbreak frequency in each county (,i.e. the isolated outbreak frequency) as the starting value for γ:

$$\gamma = \frac{1}{m} \sum_{j=1}^{m} \frac{\sum_{i=1}^{n} x_{ij}}{n}$$

, where m is the number of counties, and n is the length of weeks. The nominator $\sum_{i=1}^{n} x_{ij}$ represents the number of outbreaks in county j within the given time period $(t_1, t_2, \ldots, t_n)$.

Rearrange the equation above, we would have:

$$\begin{aligned}\gamma &= \frac{\sum_{j=1}^{m}\sum_{i=1}^{t} x_{ij}}{t \cdot m} \\ &= 0.22\end{aligned}$$

, which is the total outbreak frequency given the entire dataset.

After the estimation on γ, we can further derive Δ based on the relationship (mentioned in Section 3.2.3: formulation of dynamic probability function) that:

$$\max(p_j) = \gamma + \Delta$$

, where $\max(p_j)$ stands for the maximum outbreak probability of location j given all the other neighboring locations are having outbreaks.

Here, we set this maximum probability to be 0.8, which gives us $\Delta = \max(p_j) - \gamma = 0.58$.

5.4 Evaluation on MCMC outputs

some MCMC outputs

6 Conclusion

6.1 Findings

6.1.1 Spatially independent model

In the setting of a spatially independent model, we have found that

6.1.2 Spatially dependent model

6.2 Interpretation

6.3 Limitation

6.3.1 Self-generating data

In terms of implementing MCMC methods for both spatially independent (Section 3.1) and dependent model (Section 3.2), I am using self-generating data based on the assumed parameters' true values and distributions. A comparison study on the spatio-temporal effects is established explicitly by excluding any randomised noise in real-life disease data.

The main drawback of using self-generating data is that the corresponding models would be less robust when implementing in real-life datasets.

6.3.2 Dependence among the parameters

The performance of Gibbs sampler is negatively effected by the fact that x is highly dependent in the posterior. When constructing the likelihood of x_j, it is difficult to compute the full conditional probability of it given other $x_{k|k\neq j}$ (the neighboring information).

To cope with this issue, a pseudo likelihood computation is adopted by assuming independence among x. This eases the overall computation of MCMC algorithms, at the costs of possible slow convergence of the Markov chains to our target distributions.

6.3.3 Measurement

Dynamic probability function In our spatial function $p_i(\gamma, \Delta, x_i)$, the probability of having an outbreak for each location is depending on all the other locations, in which the effect is weighted based on the Euclidean distance.

However in real life scenarios, infection is most likely spread in regions that have highly dense population. In such cases, using distance between locations to measure the neighboring effects might no longer be reliable in terms of model prediction and interpretation.

Dynamic population For large modern metropolis, the population could get quite mobile from time to time, which makes the population estimation for each county (or location) less reliable.

By considering the epidemic nature of most infectious disease like coronavirus, it could be reasonable to use population size of people not staying at home, which in the case of Maryland, relevant public data can be found at the U.S. Bureau of Transportation Statistics. The data source also records the dynamic population size for each month.

6.3.4 MCMC diagnosis

There are so far many heuristic approaches (discussed in Section 2.4.4) to diagnose and monitor the performance of a MCMC simulation, which makes it quite model-dependent without any theoretical criterion.

7 Future work

7.1 Hamiltonian Monte Carlo

Recently, convenient implementations of a powerful MCMC technique called Hamiltonian Monte Carlo (HMC: also called hybrid MC) have become available [LiM18]. HMC uses the concept of Hamiltonian dynamics to create a proposal distribution for the Metropolis-Hastings algorithm, together with the leap-frog algorithm and the No U-Turn sampler. HMC requires more computational effort per sample step compared to other MCMC techniques, but because subsequent steps are less strongly correlated it also produces more effective samples per sample step.

To cope with additional computational requirement, we can consider the software platforms that implement the automatic construction of MCMC samplers for user-defined models. Some of the widely used platforms [LiM18] include JAGS(Just Another Gibbs Sampler), NIMBLE (Numerical Inference for Statistical Models for Bayesian and Likelihood Estimation), Stan (,which provides full Bayesian inference for continuous-variable models based on the No-U-Turn sampler, an adaptive form of HMC).

7.2 Model extension

In our current models (both spatially independent and dependent), the construction of our rate parameter λ only relies on a two-dimensional parameter set:

$$\begin{bmatrix} \alpha \\ \beta \end{bmatrix}$$

,where α is the intercept coefficient which stands for the averaged sporadic frequency of an infectious disease, and β is the additional infectious effect cause by an external risk factor $\mathcal{X}$ (which leads to higher risk of having an outbreak).

$$\begin{bmatrix} \alpha \\ \beta_1 \\ \vdots \\ \beta_k \end{bmatrix}$$

In the following future studies, more advanced models can be constructed by considering a higher dimensional parameter set. Using those models, we can aim to increase the overall model flexibility as well as interpretability of observing disease outbreaks. Possible parameter extensions are listed below:

7.2.1 Spatio-temporal term

In the studied models, the outcome variable λ or the individual λ_per for people living in Maryland, U.S. (Section 5) is assumed to be the same for all locations or counties. In a more complex setting, we can adopt the methodology used in the study of *leptospirosis notification data in New Zealand* [Ben21], by introducing the spatio-temporal term u_j , which follows a normal distribution $u_j \sim \mathcal{N}(0, \sigma_u^2)$. This would give us:

$$log(\lambda_j) = \alpha + \beta x + u_j$$

, where j is the location index.

7.2.2 Timely-temporal term

In our study, we have only considered the spatial effects of disease outbreak. In spatial dependent model where we have constructed a dynamic probability function $p_i(\gamma, \Delta, x_i)$, the probability p_i is only considering the spatial effect at time i, indicated by $x_i := \{x_{ij}, j \text{ is the location index}\}$.

In spatially and timely dependent model, we can the expand the spatial effect to a timely dimension, where a location that is having an outbreak would last for a given time period. This means other neighboring locations would likely be effected by this outbreak location over a time period, instead just at time i.

References

[Bec97] Britton.T Becker.N.G. "Uses of the EM algorithm in the analysis of data on HIV/AIDS and other infectious diseases". In: *Stat Methods Med Res* 6.1 (Mar. 1997), pp. 24–37. DOI: 10.1177/096228029700600103. URL: https://pubmed.ncbi.nlm.nih.gov/9185288/.

[Bel12] Chernozhukov.V Belloni.A. "On the Computational Complexity of MCMC-based Estimators in Large Samples". In: *arXiv preprint* 37.4 (Jan. 2012). DOI: 10.1214/08-AOS634. URL: https://arxiv.org/abs/0704.2167v3.

[Ben21] Spencer.SEF Benschop.J Nisa.S. "Still 'dairy farm fever'? A Bayesian model for leptospirosis notification data in New Zealand". In: *J. R. Soc. Interface* 18 (Jan. 2021). DOI: 10.1098/rsif.2020.0964. URL: https://doi.org/10.1098/rsif.2020.0964.

[Bol07] W. Bolstad. *The Frailty Model*. 2007.

[Buc08] David L. Buckeridge. *Predicting Outbreak Detection in Public Health Surveillance: Quantitative Analysis to Enable Evidence-Based Method Selection*. 2008. URL: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC2656053/.

[Duc11] P. Duchateau L. Janssen. *Understanding Computational Bayesian Statistics*. 2011.

[Ham13] Richardson D. Hamra G MacLehose R. *Markov chain Monte Carlo: an introduction for epidemiologists*. 2013. URL: https://academic.oup.com/ije/article/42/2/627/738896.

[Kas98] Gelman.A Kass.E.Robert Carlin.P.B. "Markov Chain Monte Carlo in Practice: A Roundtable Discussion". In: *The American Statistician* 52.2 (May 1998), pp. 93–100. DOI: 10.2307/2685466. URL: https://www.jstor.org/stable/2685466.

[LiM18] Bolker.M.B Li.M Dushoff.J. "Fitting mechanistic epidemic models to data: A comparison of simple Markov chain Monte Carlo approaches". In: *Stat Methods Med Res* 27.7 (July 2018). DOI: 10.1177/0962280217747054. URL: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC6027774/.

[Spe11] Simon E.F. Spencer. *The detection of spatially localised outbreaks in campylobacteriosis notification data*. July 2011.

[Unk11] Steffen Unkel. *Statistical methods for the prospective detection of infectious disease outbreaks: a review*. July 2011. URL: https://rss.onlinelibrary.wiley.com/doi/10.1111/j.1467-985X.2011.00714..

[Vat19] Flegal.M.J Vats.D Robertson.N. "Analyzing MCMC output". In: *arXiv preprint* 2 (Dec. 2019). URL: https://arxiv.org/pdf/1907.11680.pdf.

8 Appendix

8.1 Supplementary information

The relevant coding for data processing and MCMC methods can be found on **GitHub** repository: spatial outbreak detection.

Lastly, I want to thank my supervisor Dr Simon Spencer, who has offered me valuable advice and has shared his research experience with me during the project. Some of the methodology (Section 3) used in this paper is adpoted from Simon's work in disease studies (Leptospirosis [Ben21], Campylobacteriosis [Spe11]). Also, I want to thank Professor Martyn Plummer, who has been offering me academic as well as mental guidance as my person tutor and also as the lead author of JAGS (Just Another Gibbs Sampler), a program for Bayesian modelling using Markov Chain Monte Carlo.

8.2 Plots

8.3 Distribution table

8.4 R Code

Construct dynamic function of p

```
# set parameters for gamma and Delta
gamma = 7/52
max_p = 0.5 # maximum p when all neighboring locations are having an outbreak
Delta = max_p - gamma

a_theta <- c(gamma,Delta,1-gamma-Delta)
```

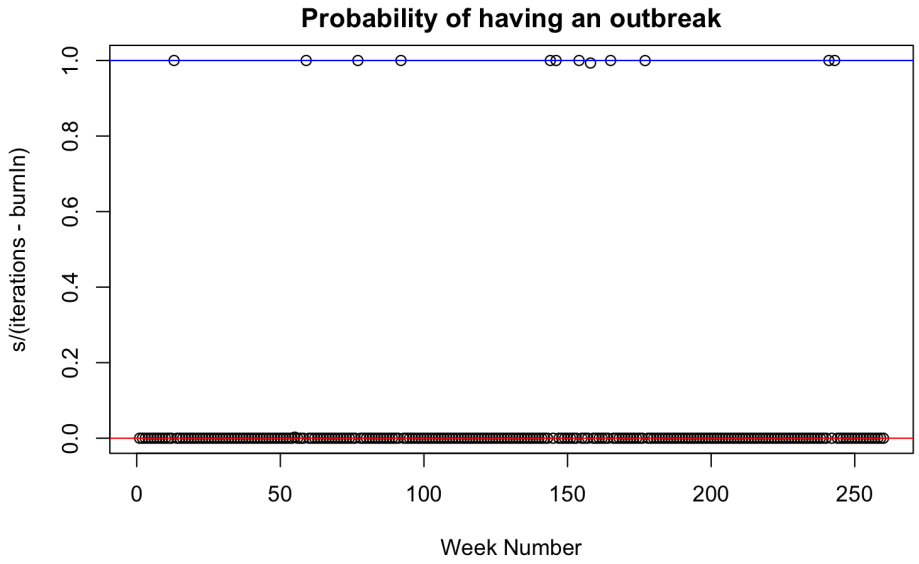

Figure 9: Probability of having an outbreak per week

Distribution	parameters	support	PDF
Poisson	$\lambda \in (0, \infty)$	$k \in \mathbb{N}_0$	$f(k) = \frac{\lambda^k e^{-\lambda}}{k!}$
Binomial	n, p	$k \in 0, 1, \ldots, n$	$f(k) = \binom{n}{k} p^k (1-p)^{n-k}$
Gamma	$\alpha > 0, \beta > 0$	x$\in (0, \infty)$	$f(x) = \frac{\beta^\alpha}{\Gamma(\alpha)} x^{\alpha-1} e^{-\beta x}$
Beta	$\alpha > 0, \beta > 0$	x$\in [0, 1]$	$f(x) = \frac{x^{\alpha-1}(1-x)^{\beta-1}}{B(\alpha,\beta)}$ where $B(\alpha, \beta) = \frac{\Gamma(\alpha)\Gamma(\beta)}{\Gamma(\alpha+\beta)}$
Dirichlet	$K \geq 2$	$x_i, \ldots, x_K \in (0, 1)$	$f(x) = \frac{1}{B(\alpha)} \prod_{i=1}^K x_i^{\alpha_i - 1}$ where $B(\alpha) \frac{\prod_{i=1}^K \Gamma\alpha_i}{\Gamma(\sum_{i=1}^K \alpha_i)}$
	$\alpha_1, \ldots, \alpha_K > 0$	$\sum_{i=1}^K x_i = 1$	

Table 6: Probability distribution table for different distributions

```
# define probability function at each time t
local_prob <- function(x_t,d_inv,theta){
  # x_t: a vector of Xs indicating an outbreak or not for each location at time t
  # d_inv: an inverse distance metrix among all locations within the region
  # j: the index of the location
  # theta: a vector of gamma, Delta and 1-gamma-Delta
  gamma = theta[1]
  Delta = theta[2]
  m <- dim(d_inv)[1] # number of locations
  prob <- rep(0,m)

  if (length(x_t) != 1) {
    for (j in 1:m){
      x_t = x_t[-j] # removing x_j
      d_j = d_inv[j,-j] # the jth row excluding the jth location, giving a vector
          where j!=k
      weighted_d_j = d_j/sum(d_j) # compute the weight to each entry
```

```
      prob[j] <- gamma+Delta*sum(weighted_d_j*x_t)
    }
  }
  else {prob = gamma+Delta*x_t} # if x_t is single value
  return(prob)
}
```

Generate data

```
# generate data
set.seed(1012)
n = 5 # number of weeks

p_i <- local_prob(x_0,d_inv,a_theta)
#x <- matrix(data = NA, nrow = n, ncol = 1)
x <- x_0
p <- p_i
for (i in 1:n){
  x_i <- rbinom(1,size = 1,prob = p_i[1]) #p in location 1 at time i
  for (j in 2:m){
    x_i[j] <- rbinom(1,size = 1,prob = p_0[j]) #p in location j at time i
  }
  x <- rbind(x,x_i)
  # update p_i using x_i
  p_i <- local_prob(x_i,d_inv,a_theta)
  p <- rbind(p,p_i)
}



x_2 = x
x <- x[-1,] # a matrix of x_it with time and location, remove the 1st empty column
p_2 = p
p <- p[-(n+1),] # a matrix of p_it with time and location, remove the 1st empty
    column


# set parameters
alpha = ln(3) # = ln(3)
beta = log(6) # = ln(6)
# noise = rnorm(n,mean = 0,sd = 1)
# lambda = exp(alpha + beta*x+noise)
lambda = exp(alpha + beta*x) # excluding the noise, returns a matrix
# simulated probability
cat('Simulated probability is:', mean(lambda)/(n*m))


# construct y (observations)
y <- matrix(data = NA, nrow = n, ncol = 1)
for (j in 1:m){
  y_j = rpois(n,lambda = lambda[j]) # sampled observations for one location
  y_j <- array(y_j,dim = c(n,1)) # convert numeric into a column
  y <- cbind(y,y_j)
```

```
}
y <- y[,-1] # a matrix of y_it with time and location, remove the 1st empty column
```

Construct Prior and Likelihood from the model

```
# log density function of the Dirichlet distribution
lddirichlet <- function(x,a) {
  wh<-which(a!=0) # allow a for having 0s for some a_i
  return(lgamma(sum(a[wh]))+sum((a[wh]-1)*log(x[wh])-lgamma(a[wh])))
}

# Log Prior density
prior <- function(param,x){
  # x: a matrix of x_it, outbreak indicator in location i at time t
  alpha = param[1]
  beta = param[2]
  gamma = param[3]
  Delta = param[4]
  theta = c(gamma,Delta,1-gamma-Delta)

  alpha_prior = dgamma(alpha, shape = 6, log = T)
  beta_prior = dgamma(beta, shape=6, log = T)

  x <- as.matrix(x)
  x_prior <- matrix(data = NA, nrow = 1, ncol = dim(x)[2])
  for (t in 1:dim(x)[1]){
    p_t = local_prob(x[t,],d_inv,theta) # compute localized probability at time i
    x_t_prior <- dbinom(x[t,],1,p_t,log = T) # a vector of density for each x_t
    x_prior <- rbind(x_prior,x_t_prior)
  }
  x_prior <- x_prior[-1,] # remove the 1st empty row

  gamma_Delta_prior = lddirichlet(theta,a_theta)

  return(alpha_prior+beta_prior+x_prior+gamma_Delta_prior) # a matrix
}


# Log Likelihood
# likelihood of y given x
likelihood <- function(param,x){
    alpha = param[1]
    beta = param[2]
    pred = exp(alpha + beta*x)

    # each week has a lambda based on x_i
    singlelikelihoods = dpois(y, lambda = pred, log = TRUE) # predicted lambda, y
        comes from above
    # sum_all = sum(singlelikelihoods)
    return(singlelikelihoods) # an matrix of likelihoods
}
```

```
# likelihood of x given p(theta)
likelihood_x <- function(x,theta){
  likelihood_x <- matrix(data = NA, nrow = 1, ncol = dim(x)[2])
  for (i in 1:dim(x)[1]){
    res = dbinom(x[i,], 1, local_prob(x[i,],d_inv,theta), log = T)
    likelihood_x <- rbind(likelihood_x,res)
  }
  likelihood_x <- likelihood_x[-1,] # remove the 1st empty row
  return(sum(likelihood_x)) # likelihood for x given theta
}


# Posterior
posterior <- function(param,x){
  return (likelihood(param,x) + prior(param,x)) # matrix + matrix
}
```

Parameter Estimation using MCMC

```
### Metropolis Hasting algorithm ###

proposalfunction <- function(param){
  # param: alpha,beta,gamma,Delta
  tmp = rnorm(2, mean = param[c(1,2)], sd= rep(.2,2)) # proposal distribution of
      alpha,beta is Normal here
  param[1] = tmp[1]
  param[2] = tmp[2]
  return(param)
}

run_metropolis_MCMC <- function(startvalue,x,iterations,burnIn){
  # create chain to store alpha,beta,gamma,Delta
  chain = matrix(NA,iterations+1,length(startvalue)) # each row:
      c(alpha,beta,gamma,Delta)
  chain[1,] = startvalue # 1st row

  # create chain_x to store x
  t <- dim(x)[1] # total time
  m <- dim(x)[2] # total locations
  chain_x <- array(NA, c(iterations+1,t,m)) # create a 3 dimensional array
  chain_x[1,,] <- x # startvalue for x at time 0
  a <- 1 # scale parameter in Dirichlet distribution Dir(theta+a*theta_)
  prob_alpha_beta = 0
  prob_theta = 0

  # iterations
  for (i in 1:iterations){
    proposal = proposalfunction(chain[i,]) # propose new alpha, beta only

     # MH for alpha, beta
    probab = exp(sum(posterior(proposal,x)) - sum(posterior(chain[i,],x))) # ratio
    prob_alpha_beta[i] = probab # record probability ratio
```

```
  if (is.na(probab) == F & runif(1) < probab & proposal[2]>alpha) { # reject
      beta if it's smaller than alpha
    chain[i+1,] = proposal # accept alpha,beta

  }else{
    chain[i+1,] = chain[i,] # reject
  }

  # Dirichlet random walk
  theta <- c(chain[i,3],chain[i,4],1-chain[i,3]-chain[i,4]) # current theta
  theta_ = rdirichlet(1,a*theta+a_theta) # proposal for new theta
  # ratio of likelihoods
  prob1 <- likelihood_x(x,theta_) - likelihood_x(x, theta)
  #
  prob2 <- lddirichlet(theta_, a_theta)-lddirichlet(theta, a_theta)
  # proposal ratio
  prob3 <- lddirichlet(theta,a_theta+a*theta_) -
      lddirichlet(theta_,a_theta+a*theta)
  full_prob <- exp(prob1+prob2+prob3)
  prob_theta[i] = full_prob # record probability ratio
   if (runif(1) < full_prob){
    chain[i+1,3] = theta_[1] # accept gamma
    chain[i+1,4] = theta_[2] # accept Delta
    a = max(0,a-3) # adaptive approach on choosing scaling parameter a, to get
        close to 25% ratio
   } else{a = a+1}


  # n_case <- sum(chain_x[i,,]) # sum of x in ith iteration

   # update x, Gibbs sampler
  param <- chain[i+1,]
  normalized_c <- 1/(exp(posterior(param,1))+exp(posterior(param,0)))
  for (j in 1:m){
    chain_x[i+1,,j] = rbinom(t,1,exp(posterior(param,1))*normalized_c) # vector
        of proposed x for one location
  }
}

# return acceptance for alpha,beta
acceptance = 1-mean(duplicated(chain[-(1:burnIn),c(1,2,3,4)]))

# return(chain)
setClass(Class="res",
       representation(
          chain="matrix",
          chain_x="array",
          prob_alpha_beta="numeric",
          prob_theta="numeric",
          acceptance="numeric"
        )
```

```
    )
    return(new('res',
               chain = chain,
               chain_x = chain_x,
               prob_alpha_beta = prob_alpha_beta,
               prob_theta = prob_theta,
               acceptance = acceptance))
  }
```

Produce Trace plot for α, β, p from MCMC:

```
# visualization for model parameters

# plot the histogram of the parameter distributions
plot_hist <- function(burnIn,param_index,param_name,param_true){
  hist(chain[-(1:burnIn),param_index],nclass=30, main=paste("Posterior
      of",param_name[param_index]), xlab="True value = red line" )
  abline(v = mean(chain[-(1:burnIn),param_index]),col='green')
  abline(v = param_true[param_index], col="red" )
}
# plot the MCMC chains for the parameter
plot_chain <- function(burnIn,param_index,param_name,param_true){
  plot(chain[-(1:burnIn),param_index], type = "l", xlab="True value = red line" ,
      main = paste("Chain values of",param_name[param_index]))
  abline(h = param_true[param_index], col="red" )
  abline(h = mean(chain[-(1:burnIn),param_index]),col='green')
}

par(mfrow = c(3,3))
for (i in 1:length(param_name)){
  plot_hist(burnIn = burnIn, param_index = i, param_name = param_name, param_true =
      param_true)
}
for (i in 1:length(param_name)){
  plot_chain(burnIn = burnIn, param_index = i, param_name = param_name, param_true =
      param_true)
}
```

Visualing x with cumulative probability plot and ROC curve:

```
# visualizing x
par(mfrow = c(2,2))

s1 = apply(chain_x,2,sum) # full chain
s = apply(chain_x[-(1:burnIn),,],2,sum) # accumulating column sum for each week of x
quantile.no = 400
segments <- quantile(s/(iterations-burnIn),probs = seq(0,1,1/quantile.no))
point <- unname(segments[c(1,quantile.no-1)])

plot(s/(iterations-burnIn), xlab="Week Number" , main = "Probability of having an
    outbreak")
abline(h = point[1], col="red")
abline(h = point[2], col='blue')
```

```
# ROC curve
df <- data.frame(s/(iterations-burnIn+1)*100,apply(x,1,sum))
names(df) <- c('prediction','label')

library('pROC')
roc(df$label,df$prediction,plot=TRUE, print.thres=TRUE, print.auc=TRUE)
```