Deterministic and Stochastic Modeling of Disease Propagation For Descriptive and Preventative Purposes

Long Le

January 15, 2018

Abstract

In this paper, I devised an SIR deterministic model and a stochastic model based on SIR and Markov Chain in hope of correctly describe and predict the propagation pattern of measles. To assess model performance, I fitted the data to find the reproduction number of measles disease in the 2017 season occurring in the Democratic Republic of the Congo.

Through model fitting, I determined the reproduction number of the disease ($R_0 = 3.233$), the peak of the disease (at day 120), and the cycle of the disease (280 days).

I also found there is an interesting link between the deterministic SIR model and the stochastic Markov Chain model, which allows us to transform one model to the other. This means that the two models are not only practically complementary (one is used for descriptive purpose, another for predictive purpose), but fundamentally complementary (one model could be written in a fundamentally different form).

At the end of the paper, we suggested viable improvements to the model as interesting topics for further research.

1 Introduction

Disease modeling has long been an interesting yet challenging task, due to each disease's origin and infection pattern. Due to the scope of this paper, we limit our exploration to analyzing one deterministic and one stochastic model for disease modeling, as well as fitting the model to current data sets, from which discussing the strengths and weaknesses of proposed models. Additionally, optimization is included as improvements to the models as necessary.

1.1 Problem Definition

Given a population of N members. Suppose we introduce one infected member ($I = 1$, in which I is the infected population), to the original population. For epidemiologists, the two most important questions would be:

1. Determine the possibility of endemic occurrence, and
2. What factors, or parameters, are essential for disease modeling

The following paper would aim at answering the above questions.

1.2 Intuition

For the deterministic model, we choose to look at the classic SIR model proposed by Kermack and McKendrick [1], and explore its strengths and weaknesses in disease modeling. The reason is the intuitive interpretation of the model, given that separating a society into compartments of susceptible, infected, and recovered populations is straightforward. We then assume there is a fixed rate of transferring from one compartment to another as the disease is introduced, which allows us to formulate our model as a set of differential equations.

On the other hand, for the stochastic model, we try to address one weakness of the deterministic model. That is, the deterministic model views the population to be continuous, as opposed to being discrete in the way it should be in reality. For a model to be realistic, it needs to allow a probability, however small, for extreme events to happen. That is, even in a favorable environment, a disease might not develop into an epidemic due to uncontrollable and unobserved factors, and vice versa. The stochastic model attempts to fix that shortcoming of the deterministic model.

2 Differential Equation Model: SIR model

2.1 Introduction

The SIR model is a compartmental model to describe the spread of a disease in terms of state-transferring processes in the population. By abstracting the population into three compartments, namely S, which stands for "susceptible", I, which stands for "infected", and R, which stands for "removed". Despite the convention in some other papers to call R the "recovered" compartment, we believe "removed" a more realistic word choice, as it implies either recovery from the disease, or death due to the disease. The model assumes no migration from and to the population N, or any birth or death by natural cause within the population N. From the SIR model, there are slights variations catered for specific case. But for its watershed influence in epidemic modeling, we have chosen SIR to elaborate and experiment as a benchmark for other methods.

The SIR model describes the rates of change from one compartment to another as follow:

$$\frac{dS}{dt} = -\beta \frac{SI}{N} \tag{1}$$

$$\frac{dI}{dt} = \beta \frac{SI}{N} - \gamma I \tag{2}$$

$$\frac{dR}{dt} = \gamma I \tag{3}$$

in which β denotes the effective contact rate, and γ denotes the removal rate. In other words, β determines the speed at which the population from the susceptible compartment S move to the infected compartment I, and γ determines the speed at which the population from the infected compartment I to the removed compartment R. Formally defined, β equals the contact rate times the probability of disease contraction upon contact. γ, on the other hand, is the recovery rate and is defined to be the invert of disease duration.

We cannot solve the system of equations analytically, but qualitative analysis shows that the current behavior of the epidemic depends on equation (2):

$$\frac{dI}{dt} = \beta\frac{SI}{N} - \gamma I$$
$$\frac{dI}{dt} = I(\beta\frac{S}{N} - \gamma)$$

Whether number of infected individuals is increasing or not depending on the term in parentheses: $\beta\frac{S}{N} - \gamma$. If $\beta\frac{S}{N} > \gamma$, the instantaneous per-capita growth rate of the infectious population I is positive. Otherwise, if $\beta\frac{S}{N} < \gamma$, the instantaneous per-capita growth rate of the infectious population I is negative.

This closely ties to the initial behavior of the system of equations. Suppose the initial states consist of a mostly susceptible community $S(0) = N(0) - 1$, with one infected individual introduced to the community $I(0) = 1$, which is mostly the case when a new strain of micro-parasite enters a community. At this time, the condition for an epidemic to break out is that $\frac{dI}{dt} > 0$. In other words:

$$\beta\frac{S}{N} - \gamma > 0$$

At the same time, because $N(0)$ is relatively large (assumption 1.), $S(0) \approx N(0)$, or $\frac{S}{N} \approx 1$:

$$\beta - \gamma > 0$$
$$\gamma(\frac{\beta}{\gamma} - 1) > 0$$

By convention, let $\frac{\beta}{\gamma} = R_0$, in which R_0 is the basic reproductive number of the infection. We conclude that an epidemic would only break out when $R_0 > 1$. Otherwise, the infected population dwindles, resulting in no outbreak at all. As a note, however, due to the way to derive this condition, the condition is only valid when $I(0)$ is infinitesimal (that is, any value close to 1, so $\frac{S}{N}$ approaches 1)

This reduces the problem to finding the reproduction number of the disease by fitting the model into the data. Unless the disease is relatively new, we have already understood well the length of infection of the disease, hence γ. The reproduction number, however, depends on an array of factors that would make it different in each situation (i.e. each country or region). Thus, retrospectively calculating the reproduction number helps in two ways. One, evaluating the danger of the disease through its rate of infection for further endeavors in disease prevention. Two, serving as a benchmark if the disease recurs (which is normally the case for a lot of seasonal diseases; in fact, we would look into a seasonal disease considering how well-fitted and effective this model is with regard to such diseases).

2.2 Assumptions

In this section, we discuss the assumptions made for the SIR model and their reasonability.

The following assumptions are made:

1. Because S, I, and R denote compartments of the population N, it naturally follows that S, I, and R are non-negative. They must also be continuous values that are differentiable, only when could we mathematically express their rates of change as differential equations. This assumption effectively means that we should only apply the SIR model to sufficiently large populations only. For small populations, the thermodynamic limit [2] is not satisfied, thus rendering the model unrealistic and impractical.

2. The community is closed. That is, there is no change to the total population N with regard to time. In other words,$S + I + R = N = const$. This also means that we assume no demography, e.g birth, death, or migration. This assumption is reasonable only when the model is applied to endemic diseases with short infectious period, such as measles or Ebola [Add typical infectious days before moving to R]. During those epidemics, the susceptible population S generally falls rapidly and moves to the infected I compartment. Thus birth, death, and migration rates have infinitesimal influence to the population during that insignificant period of time.

3. The population is well-mixed. That is, each person is equally likely to make contact with another person at random with anyone else in the society. This assumption is problematic in that it assumes a homogeneously mixing society without any subgroups, which is unlikely the case in any modern society. In other words, whether one person would be able to meet another person largely depends on which neighborhood the person belongs to. A person residing in San Francisco should be more likely to meet another person from the same city, instead of someone from San Diego. For this assumption to be acceptable, we must confine the scale of our model to one isolated, reasonably close-knitted community, such as a neighborhood or a city.

4. The change from one compartment to another is immediate and the infection either is fatal or would induce permanent immunity in patients. That is, patients demonstrate no exposed state upon infected by the disease, and infected patients either die, or would not contract the disease again in the time frame established by the model. This means that the population in the R compartment would only increase, and the population in the S compartment would only decrease. This assumption is true, again, only when we consider rapidly infectious diseases such as measles, whose exposed period is negligible and during whose outbreak, there is almost no chance that one recovered patient gets contracted to the disease again.

5. Last but not least, it should also be noted that this model is intended to be applied to diseases caused by micro-parasites, such as viruses, bacteria, or protozoa, as opposed to those caused by macro-parasites, such as worms, nematodes, or flukes [3]. This is because the model focuses on compartments of a population, instead of the population of the parasites themselves. This means that the best circumstances in which the SIR model could be applied to are when the status of contagiousness is independent of how many parasites are presented in the patient's body. This corresponds to diseases caused by micro-parasites only.

Due to the above assumptions, especially the rather conflicting assumptions 1. and 3., we decide to apply the model to measles outbreak in Congo, which stands in the top 3 most ethnically homogeneous countries in the world [4]. It is hoped that the small size of the country means its inadequate transportation facilities would not negatively affect the homogeneity in contact rate among its population. Also, Congo being a developing country might mean that disease intervention is not too prevalent to severely distort the development trajectory of the disease. On the other hand, measles is selected to model for its seasonality, infectiousness, and severity, which corresponds well to the modeling target of SIR. Indeed, historically, the disease propagation pattern of measles have been modeled using SIR [1] [1]

[1] #cs111-models:Detailed formulation and analysis of the deterministic model, as well as specifying the context in which the model is valid.

2.3 Theoretical Analysis

1. **Dimensional Analysis:** In this section, we apply dimensional analysis to determine the dimension of the parameters β, and γ.

   In equation (1), for it to be dimensionally homogeneous, the dimensions on both sides must be equal. The left side of the equation, $\frac{dS}{dt}$, represents the change in the population of susceptibles S, with regard to an infinitesimal change in the unit of time t, as thus is expressed as number of people over time. On the right side, S, I, and N have the same dimension, that is, the number of people in their respective compartment. Thus, the dimensions in the fraction $\frac{SI}{N}$ cancel out, giving us the dimension of $\frac{SI}{N}$ is also the number of people. The only parameter β, as a result, must have a dimension of T^{-1} to render the left side and right side dimensionally homogeneous.

   Similarly, for equation (3), the left side, $\frac{dI}{dt}$ represents change of the infected population over time. For the equation to be homogeneous, γ needs to have a dimension of T^{-1}. Sanity check in equation (2) shows that the dimensions γ and β are correctly identified.

   Similarly, we could as well derive a dimensionless counterpart of the above equations through dimensional analysis to facilitate analysis:

   Since we know all variables are compounds of its dimensionless form and its dimension, S, I, R and t could be expressed as:

$$S = s[S] = s \cdot N$$
$$I = i[I] = i \cdot N$$
$$R = r[R] = r \cdot N$$
$$t = \frac{\tau}{\beta} = \frac{\tau}{\gamma}$$

   in which i, s, r, and τ are the dimensionless form of S, I, and R, and t respectively.

   As such, τ, r, i and s are expressible as:

$$\tau = \beta t = \gamma t$$
$$r = \frac{R}{N}$$
$$i = \frac{I}{N}$$
$$s = \frac{S}{N}$$

   This allows us to transform the set of differential equations above into the simpler dimensionless form:

$$\frac{dS}{dt} = -\beta\frac{SI}{N} \rightarrow \frac{ds}{d\tau}\beta N = -\beta\frac{siN^2}{N} \rightarrow \frac{ds}{d\tau} = -si$$

$$\frac{dI}{dt} = \beta\frac{SI}{N} - \gamma I \rightarrow \frac{di}{d\tau}\gamma N = \beta\frac{siN^2}{N} - \gamma iN \rightarrow \frac{di}{d\tau} = R_0 si - i$$

$$\frac{dR}{dt} = \gamma I \rightarrow \frac{dr}{d\tau}\gamma N = \gamma iN \rightarrow \frac{dr}{d\tau} = i$$

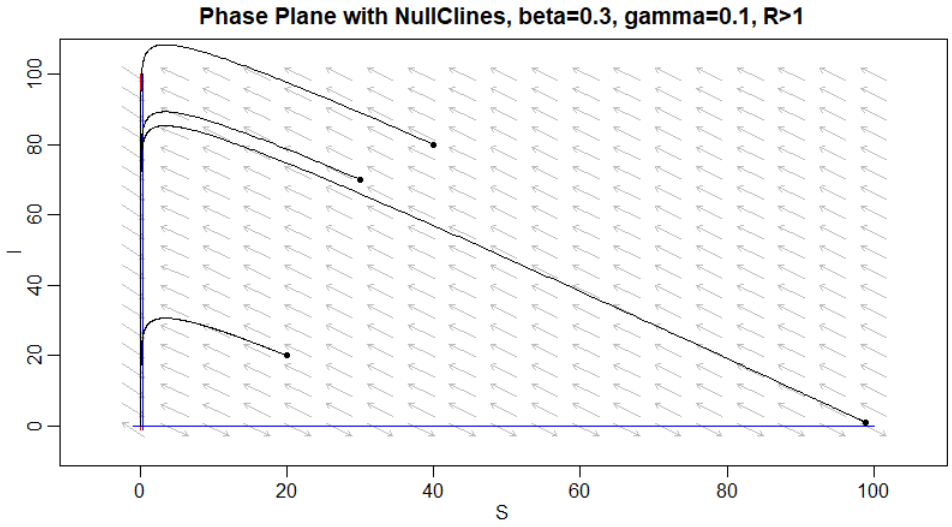
   From the dimensionless equation for $\frac{di}{d\tau}$ derived above, we found another equivalent condition for epidemic outbreak. That is, for epidemic to occur, $\frac{di}{d\tau} > 0$, or $R_0 s - 1 > 0$ (because $i > 0$). This means that $s > \frac{1}{R_0}$ for an epidemic to occur, and $s < \frac{1}{R_0}$ for the disease to die down without epidemic

outbreak. Arguably, this condition is easier to generalize than the earlier condition we found (which strictly requires $I(0)$ to be an infinitesimal value). However, due to the problem statement, we mostly analyze system where $I(0) = 1$, so both conditions are correct in the scope of this paper.
[2]

2. **Behavioral Analysis:** In this section, we find the equilibrium points, assess their stability, and analyze the general behaviors of the model.

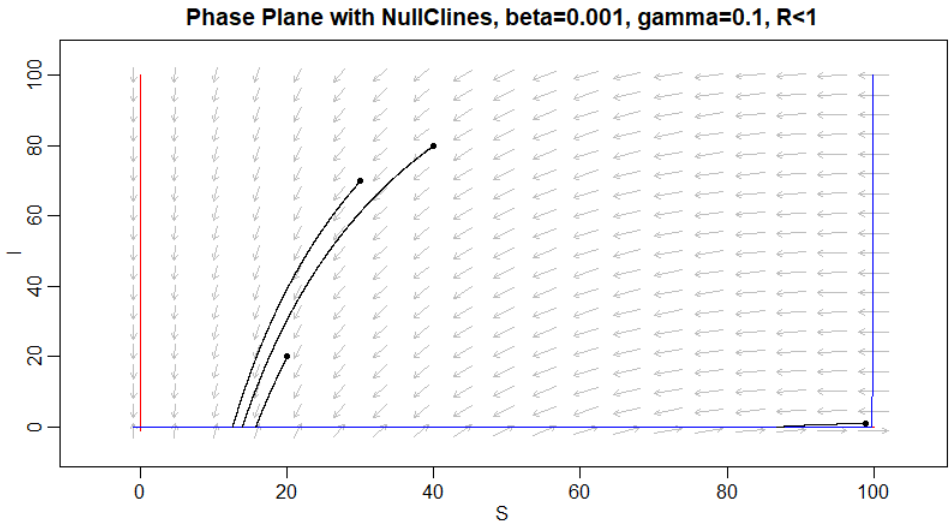
   To locate the equilibrium, we first need to locate the nullclines, or the lines at which $\frac{dS}{dt} = \frac{dI}{dt} = \frac{dR}{dt} = 0$. Because I is the only common factor in the three differential equations of the system, the only way this can happen is that there is no infected individuals in the community. In other words, an epidemic population in equilibrium can only consist of susceptible and recovered individuals. But because $S + I + R = N$ (assumption 2.), there would not be one unique equilibrium in every iteration of the model. Instead, the equilibrium could be any point in the line $S + R = N, \forall S \geq 0, R \geq 0$ (assumption 1.) when I is equal to 0, depending on the initial number of population belongs to each of the compartment, S, I, or R. It is easy to see that the equilibria are stable, as once achieved, there would be no further change in compartment populations (because the common factor $I = 0$).

   The two plots below visualize the behavior of the system under two fundamental conditions: $R_0 > 1$ and $R_0 < 1$, the former corresponds to epidemic outbreak, and the latter corresponds to the disease dying down without causing epidemic. We only analyze the phase trajectory of $\frac{dS}{dt}$ and $\frac{dI}{dt}$, the population in R depends strictly on the population of I and S.



   In this plot, for the three initial values with varied I and S ($S_1 = 99$, $I_1 = 1$, $S_2 = 40$, $I_2 = 80$, $S_3 = 30$, $I_3 = 70$, and $S_4 = 20$, $I_3 = 20$). We find that regardless of the initial condition (which is strictly in the first quadrant, as stated in our assumption), the final S and I would converge at $(0, 0)$ (where the two nullclines intersect), for a reproduction number $R_0 > 1$. This is consistent to our analysis. After the epidemic outbreak, everyone in the society would have been affected by the disease. In other words, at the end of the epidemic, $R = N$.

[2] #cs111-approxandscaling: Used dimensional analysis to verify the formulation of the system of equations and derived dimensionless versions of the equations.



This plot features the second case where there is no epidemic outbreak. In this case, regardless of the initial condition, the system would tend towards a point on the line $I = 0$, but not a unique equilibrium. This is consistent with our previous analysis in that the disease dies down after some time without affecting the whole population.

Through qualitative analysis, we reaffirm the importance of the parameter R_0 in determining the propagation of the disease in absence of prevention efforts.[3]

3. **Numerical Solution:** In this section, we elaborate on the numerical solution of SIR model and discuss its various behavior.

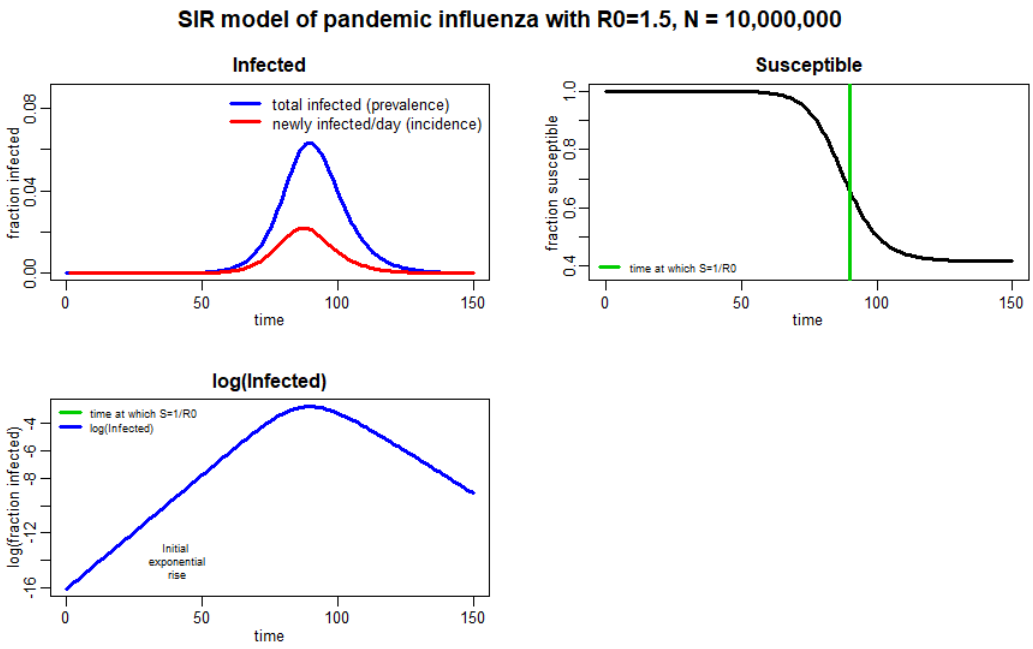
   Because analytical solution is a fairly advance topic, we resort to the Runge-Kutta 4th order method to find the numerical solution. The method, in turns, in preferred over the classic Euler's method for its more accurate result. Intuitively, while Euler's method gets the first derivative correct for the ODE, the Runge-Kutta gets the first four derivatives correct, hence a superior accuracy, albeit with a (expendable) computation requirement. The solutions are derived and plotted in R.

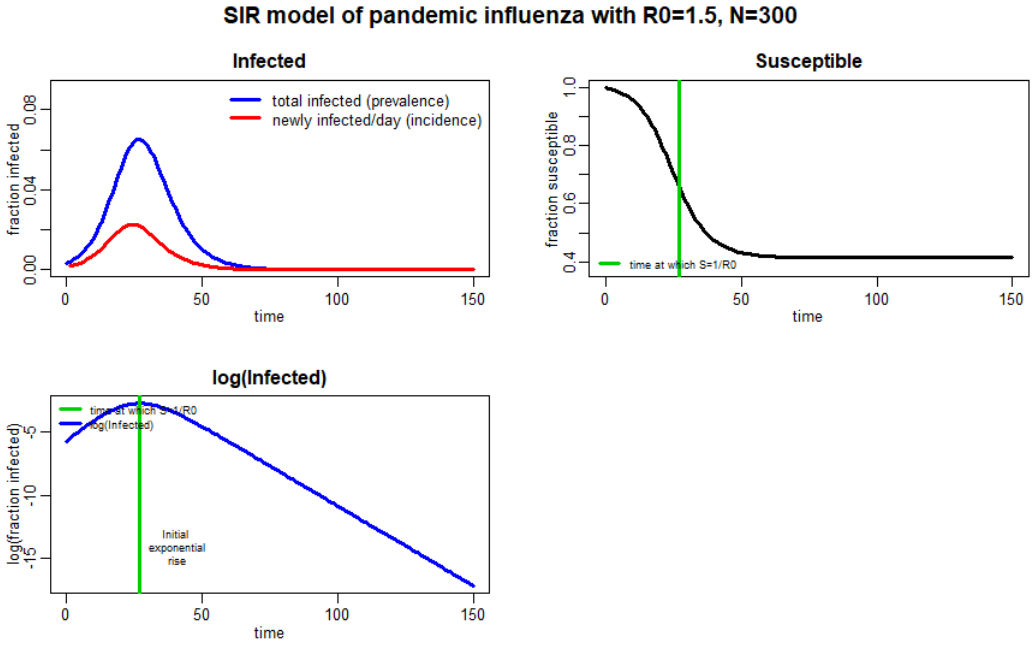
   For experimental purpose, we start with a variety of different values of $N(0)$ and $I(0)$. We will also vary the parameters β and γ, hence the reproductive number R_0. By analyzing the result, we prove the importance of the reproductive number R_0 regardless of the initial population N in determining the spread of the disease.

   As we follow the problem statement of adding 1 infected individual to the population, we only need to analyze two situations as follow:

   (a) **Case 1:** $R_0 = 1.5 > 1$

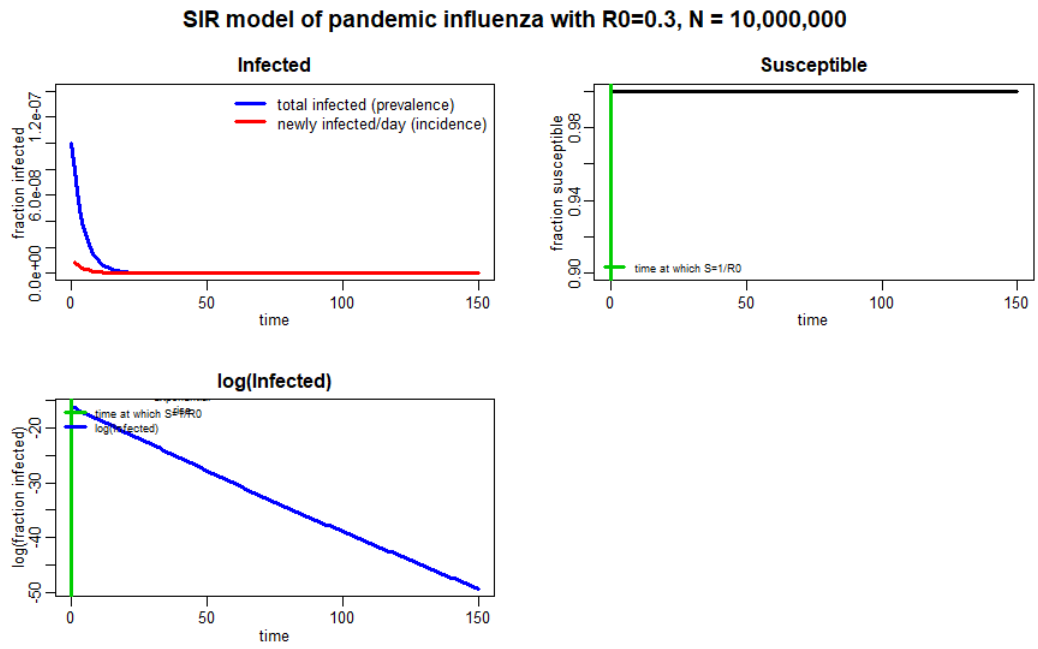
[3] #cs111-odes: formulate and analyze a system of differential equations qualitatively and quantitatively, with connection to real world behavior of disease.

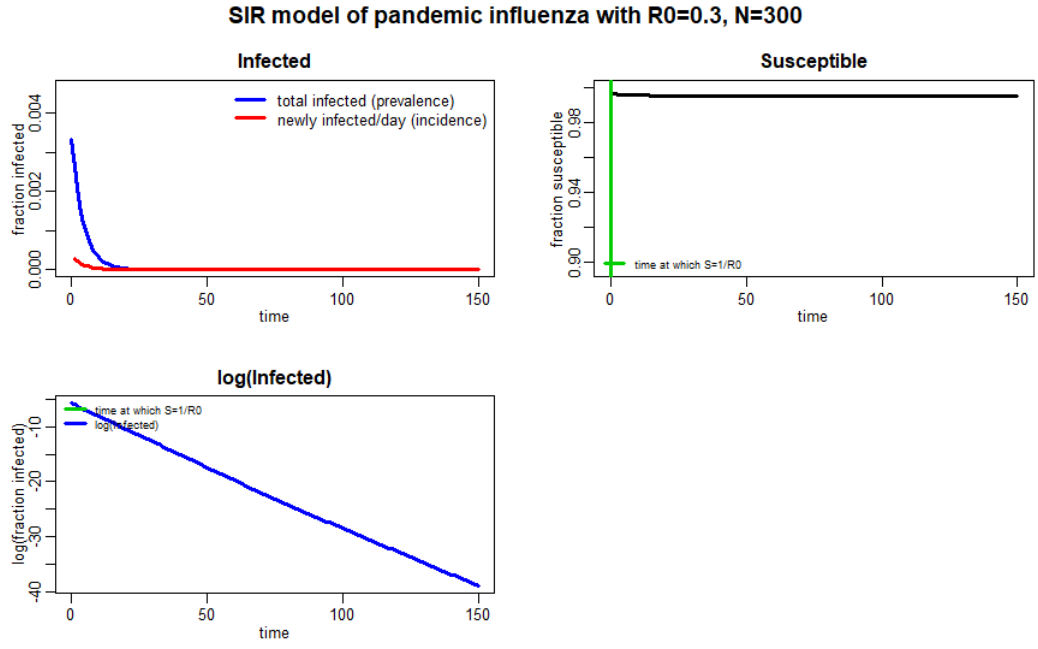




From the six charts above, we find that for the above set-up so that the reproduction number $R_0 > 1$, the infected population rises to a maximum before dropping down, indicating the occurrence of an epidemic, regardless of the starting population. The log plot shows steady rate of infection until a peak is reached, at which point, the rate of infection steadily drops down (as there is less susceptibles for infection).

(b) **Case 2:** $R_0 = 0.3 < 1$

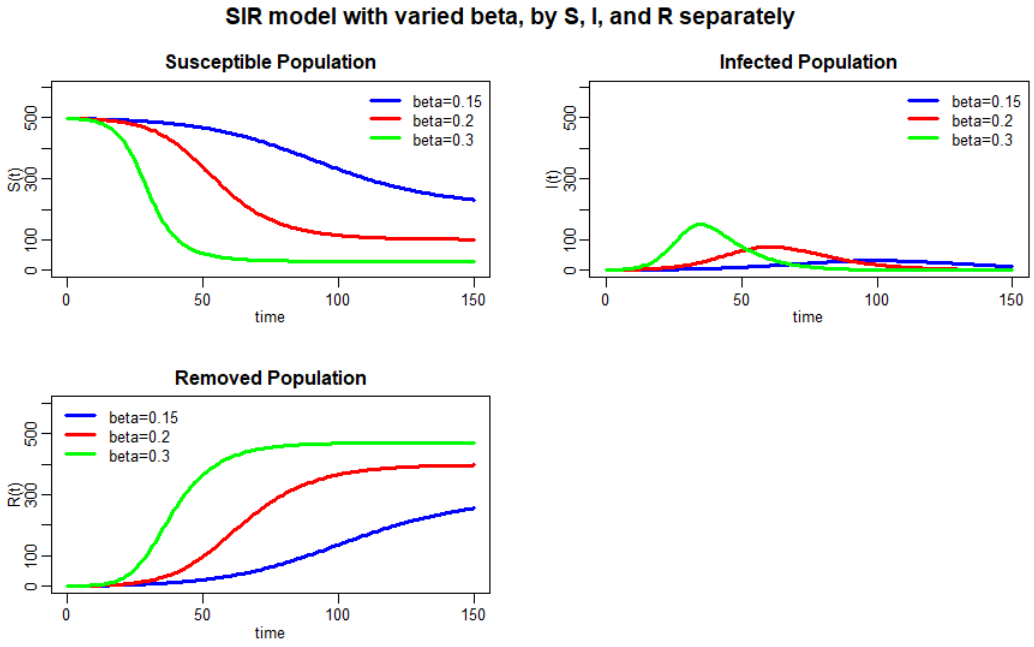




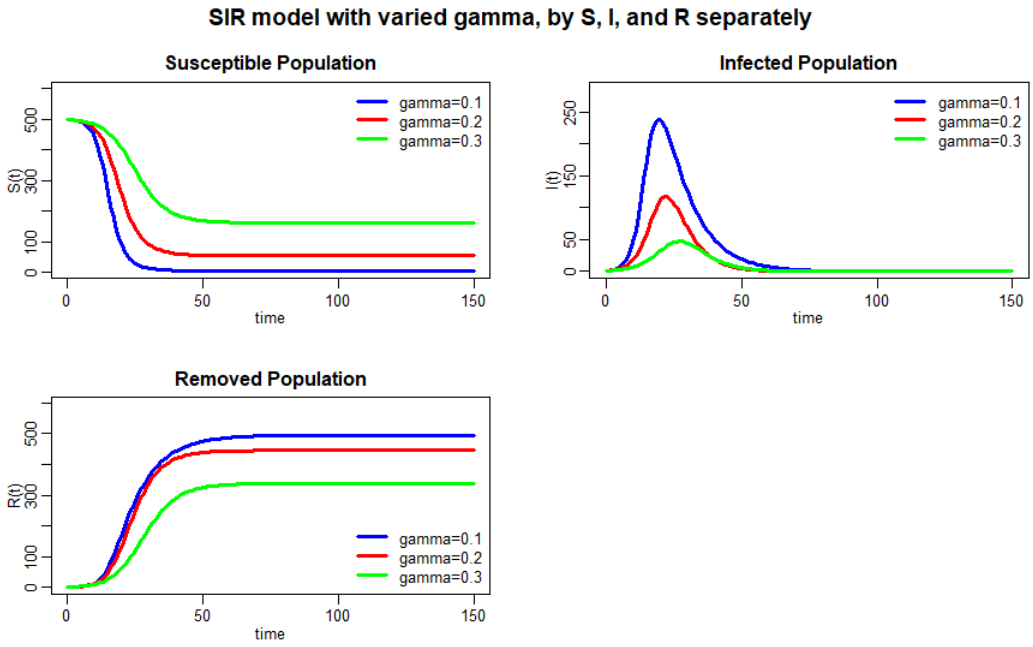
From the six charts above, the opposite result could be found. As long as the the reproduction number $R_0 < 1$, the number of infected people lowers over time, indicating that the disease dies down without growing into an epidemic. The log plot shows almost no increase in the infected population but a steady drop to 0.

The result above makes sense, as the occurrence of an epidemic depends solely on the parameter $R_0 = \frac{\beta}{\gamma}$. Intuitively, when there are more people getting infected than people recovering ($\beta > \gamma$), the disease would keep spreading until, at one point, it becomes an epidemic. Vice versa, if $\beta < \gamma$, the disease would not spread fast enough to become an outbreak, but rather dies down.

Also, from a graphical perspective, we confirm the role of β and γ in modeling the disease. β, which denotes the rate at which someone from the susceptible S compartment moves to the infected I compartment, is proportionate to how quickly and gravely the disease would turn into an epidemic. As shown in the following chart comparing different β while keeping the other values fixed, we could see that the higher β gets, the faster people are infected, and the larger the total infected population gets at the peak of the epidemic. This corresponds well with the logic of the model. That is, a more infectious disease would find it way around the population much more quickly and violently than a less infectious disease. With the same rate of recovery γ, it easily follows that a more infectious disease would infect more people in the same time frame as a less infectious one, thus resulting in a larger infected population up until it reaches its peak. After that, as we assume no addition to the population, an infectious disease find no more victims and thus would dies down more quickly compared to its less infectious counterpart (thus the steeper slope in the chart).



On the other hands, γ, the rate at which someone from the infected compartment moves to the removed compartment determines the threshold of the susceptible compartment to epidemic occurrence. That is, with a higher "recovery rate" or higher γ, people would recover, or be removed, from the population before infecting as many people. Thus the lower epidemic equilibrium with higher γ with other parameters fixed. Interestingly, we also find that the higher γ, the lower $R(t)$ gets (which could be seen in the last chart). This makes sense as a higher γ correlates with a smaller infected population. And less infected people means less people recovering, thus a lower $R(t)$.



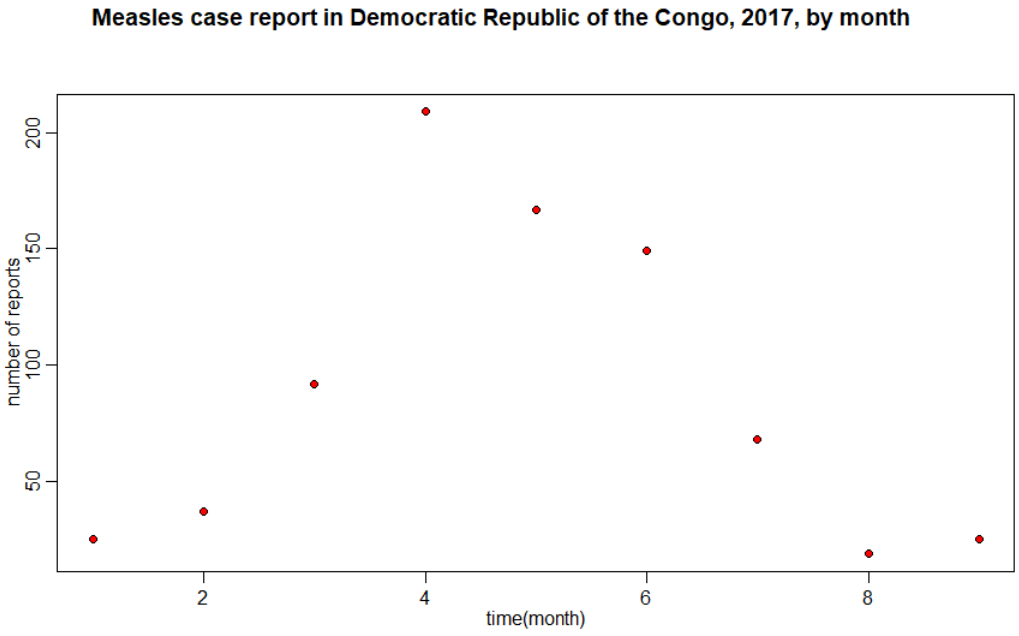
From the above observations and assessments, we find the SIR model is not suitable for optimization methods in the sense of finding optimal values for certain parameters, as β and γ has directly opposite roles. The most infectious disease would thus have a really high β and a really low γ, and vice versa. Instead, we propose a practical and useful use case for #optmethods. We would fit the data from a real disease to the SIR model, which means optimizing for the lowest least squared error function. [4]

2.4 Model Fitting and Discussion

1. **Model Fitting Problem:** We formulate the fitting problem and discuss the usability of common methods.

   As discussed above, we intend to fit the model to the measles incidence data set reported by the Democratic Republic of the Congo. The data is presented as a discrete time series composed with number of incidences reported by month. The data is presented as follow:

[4] #cs111-models: further analyze an epidemic model and assess the impact of each parameter in the model. Also, proactively connect a model to its use case and tools of analysis.



The purpose of the fitting procedure would be to find the basic reproduction number R_0 denoting the infectiousness of the disease, which could only be obtained by analyzing the spread of the disease in the infected population. Even though this approach of determining R_0 is retrospective - that is, we analyze the disease after it has run its course - the result obtained could be used as benchmark to compare the effectiveness of interventions in the next epidemic. This is especially true in the case of measles, which is recurring over years due to its common nature.

To assess the correctness of the fitting approach, we rely on the least square error function:

$$E = \sum_{i=1}^{n} (Y_i - y_i)^2$$

in which Y_i is the number of incidences recorded in the data set (real data), and y_i is the number of incidences predicted by the respective model. Intuitively, fitting the SIR model to real data in this case would mean determining the parameters β and γ so that the predicted y_i, which is a value dependent on β and γ, would be as close to the real observed value Y_i as possible.

While the gradient descent method and simplex method could be applied to find the optimal vector of parameters, they are not desirable for two reasons. One, they do not guarantee to converge at a global optimum, but rather just a local optimum. As the fitting problem for the measles epidemic is non-linear, the least square error function is also non-linear and thus does not guarantee to be convex. The result obtained from gradient descent and simplex method would be quite unreliable in this case, especially when we are not sure if our initial guess is reasonable enough so we could converge at the global optimum. Two, because of the multi-dimensional nature of the ODE, there is no graphical method to visually assess that the result obtained by gradient descent or the simplex method is indeed the global optimum.

For the above reasons, we propose a stochastic optimization method, called the Monte Carlo parameter sweeping method, to fit the SIR model derived above to real data. The method allows us to examine its correctness both visually and analytically

2. **Monte Carlo parameter sweeping method:** Like other Monte Carlo methods, the Monte Carlo parameter sweeping method's algorithm involves two rudimentary steps:

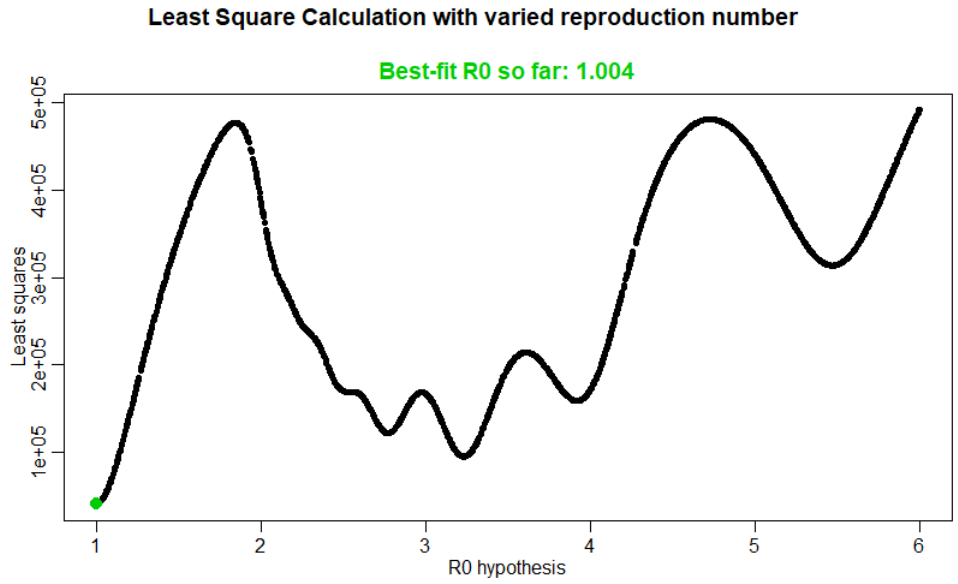
   - First, specify a range for each item of the parameter vector for subsequent sampling attempts.
   - Second, a loop in which each combination of parameter is sampled is iterated, assuming uniform distribution. For each combination of parameters, the least square error function is calculated. Values that successfully reduce the least square error is stored.

   More formally put, the Monte Carlo method is an approximation method based on Bayesian inference, which, in turns, aims to update a posterior likelihood $P(\theta|y)$ of a parameter θ given a model y and our prior belief $P(\theta)$. This approach overcomes the analytical and tractability concern of other standard Bayesian analysis methods via repetitive simulation. Hence, we accept that the result might not be strictly rigorous, and this approach is rather a heuristic one. Yet, due to its higher likelihood of converging to a global optima, its arguably less computational requirements (as we do not have to constantly update the Hessian or Jacobian of the model) and the capability for visual inspection as the algorithm is running, it is considered a superior method to what we studied in class, namely Gradient Descent and Newton's method. [5]

To model our target disease, measles in Congo, we gather the demographic and background information as follow:

- Congo population: $N = 78,000,000$ (people) (in 2011, which is the timeframe of our dataset)
- Measles length of infection: 15 days. Thus, $\frac{1}{\gamma} = 15(days)$.
- The initial population of infected $I_0 = 1$ (person)

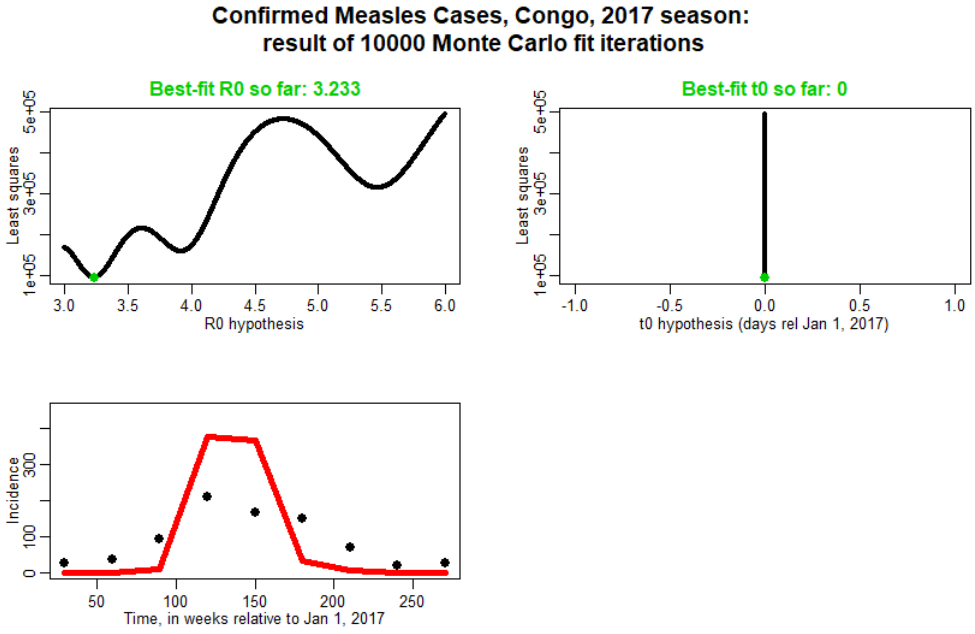
The fitting procedure is conducted with the R code (given in appendix) inspired by Sherry Stowers of Polymatheia [5], with various tweaks to better represent our dataset. The result is displayed as follow:



By sampling for R_0 and calculate the least squared error for each R_0 value, we successfully confirm that

[5] #cs111-stochastic: applied a stochastic method for optimization, assessed its applicability and superiority to other optimization methods.

the least squared error function in this case is non-convex (see plot above), which renders Gradient Descent impractical.



After $10,000$ iterations of the algorithm, we have fitted the model to the given data. Fitting results shows the best fitted value for the reproduction number is $R_0 = 3.233$, which is much lower than the standard reproduction number of measles $R_0 = 15$. And even though there is an outbreak, the disease has not spread well enough to have any country wide impact (at the peak of the disease, there are only around 200 reported cases a month). The model also did not correctly predict the propagation pattern (as denoted by the red line in the graph, the disease abruptly reached a peak and died down in just about 150 days, even though in reality, the epidemic occurred for approximately 280 days. This could be explained by one of the following reasons:

1. The disease occurred in the beginning of 2017, when medical effort has already been prominent in Africa. There should have been prevention methods, such as vaccination, at the time the disease occurred. Thus, the disease was successfully contained at that time. Our model fitting attempt only return the reproduction number **under prevention**. The natural reproduction number would be much higher.

2. The disease only struck a particular area or social group before being contained. There might have been heterogeneity, either socially or geographically, that did not introduce the disease to other geographical areas or social groups (even though geographical heterogeneity is more likely as Congo is confirmed to be socially homogeneous).

3. The number of data is too low. Should the data is generated by weeks or days, the algorithm would have a more precise fitting result. Additionally, it might be the case that people in far-flung area did not report if they contracted the disease. Thus, the data might not be correct.[6]

2.5 Strengths and Weaknesses

The above deterministic model is successful in:

[6] #cs111-optmethods: Discussed, applied, and assessed the result of an optimization method on least squared error function.

1. Describe disease propagation in a community without demographics or prevention methods. The model works well in determining a reproduction number for a disease. A more complicated model with more factors concerned might not work any better for determining R_0, which fundamentally is just a value to compare the severity of diseases.

2. Simple to analyze yet effective for real use cases. Easy for numerical and qualitative analysis.

On the other hand, the model has certain shortcomings:

1. Is not useful outside its domain. That is, the model could only be used for retrospectively analysis of a disease. For predictive analysis, the method would not return useful information as 1/ the fitting procedure is not suitable for predictive purpose, and 2/ the main target of this model, the reproduction number R_0, is mostly useful for retrospective comparison.

2. Is deterministic, which means there is no space for white noise. Arguably, a stochastic model is more suitable for realistic modeling, albeit harder to analyze, for it allows uncertainty to happen. As we could see, the above model did not predict well the cycle of the disease in Congo, where unexpected factors such as vaccination and heterogeneity happened. [7]

3 Stochastic Model

Stochastic model allows for uncertainty in the model. In this section, we attempt to formulate a Markov Chain model from the SIR model and evaluate its usefulness, particularly in the context of disease modeling.

3.1 Introduction

Markov Chain solves two problem that was eminent in the previous model. One, the parameter estimation is retrospective. Thus, while it serves as a useful benchmark, it is not as helpful in real time analysis of the disease. Two, in a context of biological modeling, it is unlikely that the dynamics of consideration is solely driven by deterministic mechanisms, but rather exposed to factors not already understood [6]. The spread of the disease, as we could see in the real data in the last section, is not steady nor easily predictable. As a result, we hope that the stochastic model could provide a more realistic representation of disease propagation in this case. We would experiment with predictive disease modeling (i.e. calculating probability of disease outbreak) and stochastic modeling of the measles disease.

To achieve that goal, we model the system as a set of states $S = \{s_1, s_2, \ldots, s_i\}$. The system would start from one of these states, say state s_x and move to another state s_y with a probability p_{xy}. The probability p_{xy} is independent of the states the system has been in before state s_x. This is the memoryless property of the Markov Chain. We would then base on the SIR model formulated above to build a transition matrix P, denoting the probability of transitioning from one compartment to another.

3.2 Observations and Assumptions

1. As discussed above, the Markovian property, or memoryless property, is the most popular and essential assumption for Markov processes, thus including Markov chain. The Markovian property assumes that the probability of transitioning from one state to another depends only on the present state, and not on any other past states, for the information contained in the current state is accumulated from all the past states the system has been to.

2. By analysis of the transition matrix, we could classify the Markov Chain to certain properties, such as ergodic, absorbing, or regular, which determines the behavior of the Markov Chain given a certain starting state.

[7] #cs111-models: Overall, discussed and analyzed an ODE model with details and insights.

3. For our model, another assumption is discrete space-time. This means that there is a finite set of state spaces and time steps, as opposed to the continuous space-time case. This means the model is simpler and easier to analyze, as well as being more suitable to our modeling task. As we could see from our data set, disease propagation is not monitored in continuous time, but rather by time intervals. Thus, continuous Markov Chain would be impractical for our case. [8]

3.3 Derivation and Interpretation

Because we focus on transforming an SIR model to a Discrete Process Markov Chain, the states are described by the proportions of the population N that belong to each of the compartment, namely S, I, and R respectively. The transition matrix P would take the general form:

$$\mathbf{P} = \begin{array}{c} \\ S \\ I \\ R \end{array} \begin{array}{c} \begin{array}{ccc} S & I & R \end{array} \\ \left\| \begin{array}{ccc} (1-a) & a & 0 \\ 0 & (1-b) & b \\ 0 & 0 & 1 \end{array} \right\| \end{array}$$

in which a and b respectively are the probability of moving from the susceptible compartment S to the infected compartment I, and the probability of moving from the infected compartment I to the removed compartment R.

From the general form of P, we see that R is an absorbing state, as $P_{RR} = 1$. That is, no state other than R itself is accessible from it. By that definition, S and I are non-absorbing state. And as it is possible to move from one non-absorbing state to the absorbing state R, the Markov Chain we are formulating is an absorbing Markov Chain. Also, due to the logic of the SIR model, once a unit moves from one compartment to another, it would not move back to its previous compartment. Thus, the Markov Chain is not ergodic nor transient.

Because we have seen that β and γ in the last section denote the transition rates from the S compartment to the I compartment, and from the I compartment to the R compartment, respectively. It is expected that the probability a would be equivalent to β, and the probability b would be equivalent to γ. Indeed, we could represent the Markov Chain as a set of equations equivalent to that of an ODE model.

Let the vector

$$p(0) = \begin{bmatrix} p_s(0) & p_i(0) & p_r(0) \end{bmatrix}$$

denotes the initial proportions of the susceptible, infected, and removed population. The $(t+1)$ steps forward proportion vector $p(t+1)$ would be written as follow:

$$p(t+1) = p(t) \cdot P$$

in which $t \in \mathbb{Z}_+$, thus satisfying the discrete space-time property of the Markov Chain in our case. By explicitly multiplying the proportion matrix $p(t)$ by the transition matrix P, we have the following set of equations:

$$p_s(t+1) = p_s(t) \cdot (1-a) = p_s(t) - a \cdot p_s(t)$$

$$p_i(t+1) = p_s(t) \cdot a + (1-b) \cdot p_i(t) = a \cdot p_s(t) + p_i(t) - b \cdot p_i(t)$$

$$p_r(t) = b \cdot p_i(t) + p_r(t)$$

The above set of equations, on the other hand, could be rewritten in a similar manner to a set of differential equations:

$$p_s(t+1) - p_s(t) = -a \cdot p_s(t)$$

$$p_i(t+1) - p(i) = a \cdot p_s(t) - b \cdot p_i(t)$$

[8] #cs111-models: proposed a new model, discussed assumptions, and verified its validity for one particular modeling task.

$$p_r(t+1) - p_r(t) = b \cdot p_i(t)$$

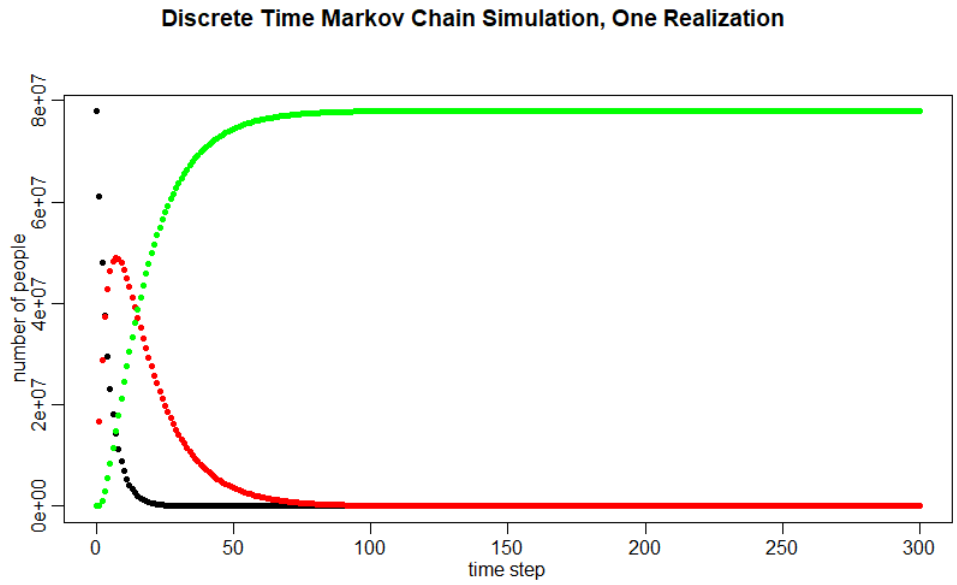
which, as we could see, bear resemblance to the ODEs we formulate in the last section. They are fundamentally different, however, in that the set of equations above are discrete, while the ODEs are continuous, but this proves that the Markov chain is a viable and capable model to describe disease propagation.

3.4 Modeling and Simulation

Because of the relationship between a and β, and b and γ, we set $a = \beta$ and $b = \gamma$, for $\beta = R_0 \cdot \gamma = 3.233 \cdot 1.15 = 0.21$ and $\gamma = 1/15 = 0.07$, as found in the last section for modeling purpose. Thus, we get a transition matrix as follow:

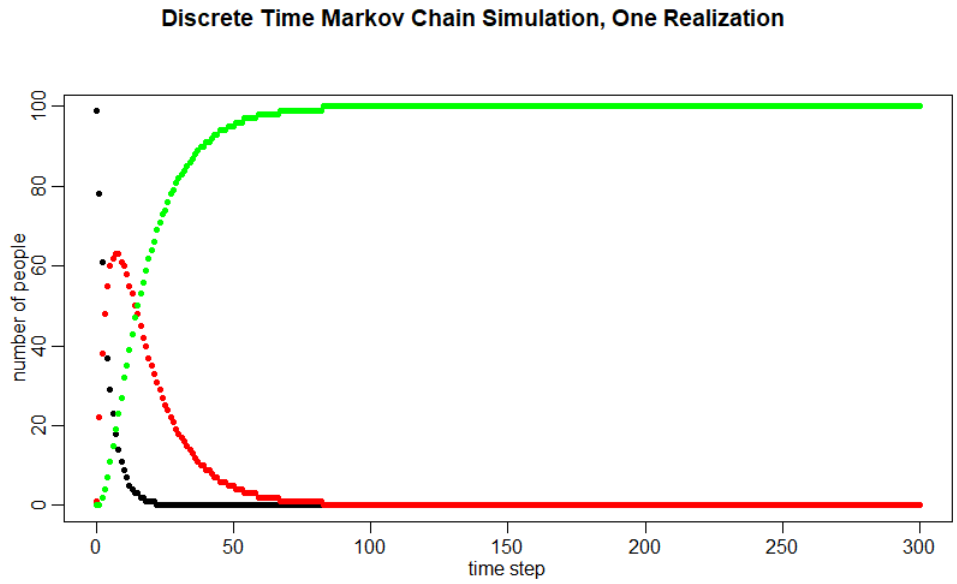
$$\mathbf{P} = \begin{array}{c} \\ S \\ I \\ R \end{array} \begin{array}{c} \begin{array}{ccc} S & I & R \end{array} \\ \left\| \begin{array}{ccc} 0.79 & 0.21 & 0 \\ 0 & 0.93 & 0.07 \\ 0 & 0 & 1 \end{array} \right\| \end{array}$$

By define the starting state to be $p(0) = \begin{bmatrix} 77,999,999 & 1 & 0 \end{bmatrix}$ We would then model this system using R and compare its predictive power to that of the ODE model. [9]

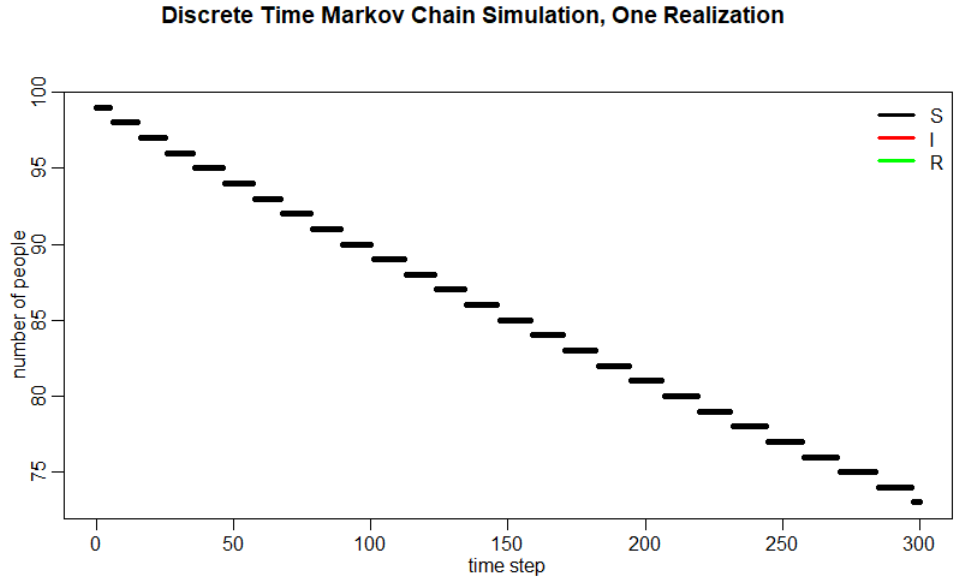


In this realization of the Markov Chain model, we could find that because the transition matrix P is regular, the system would eventually reach a stable state, at which the proportion of the population belonging to each of the compartment is unchanged. This is expected, judging from the phase plane analysis for our SIR model. The code successfully found that the steady state could be reached by the 280th step, at which $S = I = 0$, and $R = 78,000,000$. This corresponds perfectly the Congo measles data. The peak, however, rises too quickly and ends up to more than 2 months earlier than the real data. Also, a shortcoming in this model is that the stochasticity is not well seen as white noise is limited and the population is too large. In the plot below, where the population is 100 people, stochasticity could be detected much easier. This proves that this Markov Chain model might not be a good fit to describe a phenomenon in a large population, but rather a better analysis tool for small ones, such as a neighborhood or a household. Indeed, the literature confirms this conclusion [7].

[9] #cs111-stochasticity and #cs111-odes: formulates a stochastic model from a deterministic model and explore the relationship between the two



Because the performance of the model for the Congo measles data somehow closely resembles that of the SIR model (arguably with more precise cycle prediction result), for experimental purpose, we also run the model for $R_0 < 1$ for small population. The result is depicted below:



We could clearly see the difference between the deterministic SIR model and the Markov Chain model in this particular situation. According to the deterministic model, a disease with $R_0 < 1$ would quickly dies down and have no effect whatsoever on the community. However, as could be seen on the above plot, the population in S slowly decreases over the course of 300 days. This is because even though the chance is low, there is always a probability that the disease still exist in a few members of the society (probably in incubation period) and sometimes spread from one member to another. This pattern is fairly common

for less infectious diseases which have long incubation periods and are not acute enough for one patient to develop immunity after having recovered. Modeling such patterns might be a better use case for the Markov Chain model.

3.5 Strengths and Weaknesses

In the last section, we found that the stochastic Markov Chain model formulated from the classic SIR model is effective in:

1. Analyze disease propagation in small population such as neighborhoods or households. Suitable for predictive purposes, that is, finding out the proportion of infected people in the next state and time step, given the memoryless property.

2. Describe the pattern of propagation for certain types of disease, particularly those that have long incubation periods and less acute symptoms, which would easily flow around a community without causing epidemic outbreak. The model might also be effect to model STDs such as HIV/AIDS.

The disadvantages of the model are as follow:

1. Not effective for large population as stochasticity is not noticeable in this case. Thus, for large population, the model is not much better than the SIR model.

2. More difficult to analyze due to the introduction of uncertainty. [10]

4 Concluding Remarks

In this paper, we have explored two approaches for disease modeling, namely the differential equation approach through the classic Kermark-McKendrick SIR model, and the Markov Chain approach (which, in turns, is also based on the SIR model) for descriptive and predictive purposes.

We find that the ODE model is highly suitable to describe disease propagation in large homogeneous population despite its simplicity, even though its deterministic nature means that it might not be strictly realistic. The Markov Chain approach, on the other hand, includes stochastic factors in the time series, and could be a better fit for predictive modeling than ODEs. However, its stochasticity wears down with large population (source), and it would be more suited for small-scale tasks such as household disease propagation modeling. Both approaches have their own merits, but, given more time and space allotted, we believe that the following additions might provide more useful insights, as well as interesting topics for further research:

1. Expansion of the SIR model for disease prevention, in particular, the introduction of vaccination and demographics: Our SIR model is classic and fundamental, but we have seen that it is more useful as a retrospective modeling techniques. If we introduce vaccination and demographics, two benefits are eminent. One, we end up with a more realistic model that corresponds to the change in population. For large population, such as the Congo, this might be a large change to the population variable N. Also, because measles are popular on newborn babies, this addition would increase the accuracy of our model. Two, vaccination allows us to make a better use out of the model. It is difficult to apply optimization methods to the classic SIR model, but as long as an intervention such as vaccination is introduced, we could optimize for the minimal population which needs intervention to prevent endemic outbreak.

2. Modeling the system with stochastic differential equations instead of Markov Chain: We found that our Markov Chain approach is ill-fitted, analytically and computationally, for large population. From [10] SDEs look to be the perfect complementary model for Markov Chain as it both introduces stochasticity and works well for large populations.

[10] #cs111-models: Overall, presented a stochastic model, analyzed its effectiveness in a practical modeling task, as well as an experimental one, and suggested use cases for the model.

3. Add white noises to Markov Chains to improve stochasticity in large populations: We could see the diminishing effect to stochasticity as population gets larger for our Markov Chain approach. Thus, adding white noises is a fix to get the degree of stochasticity high in large population, which is promising if we look to find insights from a Markov process-based stochastic model for disease propagation modeling.

5 Reference

1 http://math.unm.edu/~sulsky/mathcamp/SIR.pdf

2 Kurzemsky, A. L. Thermodynamic Limit in Statistical Physics.

3 https://web.stanford.edu/~jhj1/teachingdocs/Jones-Epidemics050308.pdf

4 https://en.wikipedia.org/wiki/List_of_countries_ranked_by_ethnic_and_cultural_diversity_level

5 http://sherrytowers.com/2013/01/29/neiu-lecture-vi-fitting-the-parameters-of-an-sir-model-to-influenza-data/

6 Ditlevsen S., Samson A., Introduction to Stochastic Modeling in Biology

7 Eisenberg, A. The Application of Markov Chain Monte Carlo to Infectious Diseases (2011)

8 http://thepasqualian.com/?p=1862

9 http://sherrytowers.com/2012/12/11/simple-epidemic-modelling-with-an-sir-model/#numeric

10 http://sherrytowers.com/2015/10/01/difference-between-markov-chain-monte-carlo-stochastic-differential-equations-a

11 Kousar N., Mahmood R., Ghalib M. A Numerical Study of SIR Model.

12 https://www.dartmouth.edu/~chance/teaching_aids/books_articles/probability_book/Chapter11.pdf

13 http://math.unm.edu/~sulsky/mathcamp/SIR.pdf

14 https://www.marcoaltini.com/blog/parameter-estimates-for-regression-least-squares-gradient-descent-and-monte-carl

6 Appendix

This appendix includes codes for various aspects of the paper.

6.1 R Code to numerically solve SIR differential equations:

By Sherry Towers, Polymatheia, 2012. The model Towers looked at is different from the one we analyzed in the paper, so I have changed the parameters in the code below to better suit our purpose.

```
#################################################################################
# An R script to solve ODE's of an SIR model
# http://www.sherrytowers.com/sir_func.R
#
# Author: Sherry Towers
#       smtowers@asu.edu
#
# Created: Dec 1st, 2012
# Copyright Sherry Towers, 2012
#
# This script is not guaranteed to be free of bugs and/or errors.
#
# This script can be freely used and shared as long as the author and
# copyright information in this header remain intact.
```

```
#
# In order to use this script in R, you must first have the deSolve library
# installed. To do this, type in the R command line:
#   install.packages("deSolve")
# then pick a mirror site located close to you.
################################################################################

require("deSolve")

################################################################################
################################################################################
# This is a function which, given a value of S, I, and R at time t
# calculates the time derivatives of S I and R
#
# The function gets called over and over again by functions in the R deSolve
# library to do the numerical integration over many time steps to solve
# the system of ODE's
#
# t is the current time at a particular step
# The vector x contains the current values in each compartment
# The list object vparameters contains the parameters of the model, like the
# recovery period, gamma, and the transmission rate, beta (in this case)
# In the main program, the vparameters list object is filled with the parameter
# values and their names.
#
# This function gets passed to the functions in the deSolve package
#
################################################################################
derivative_calc_func=function(t, x, vparameters){
  ##############################################################################
  # The vector x is the same length as the number of compartments. In your main
  # program you identify which compartment corresponds to which element of x.
  # You need to make sure that these are in the same order.
  ##############################################################################
  S = x[1]
  I = x[2]
  R = x[3]

  ##############################################################################
  # vparameters is a list object, filled in the main program, and passed
  # Keep this next line the same when you are writing your own function to
  # solve a system of ODE's
  ##############################################################################
  with(as.list(vparameters),{

    ############################################################################
    # calculate the population size, which for a simple SIR model is
    # npop = S+I+R
    # we will need this to calculate our SIR model derivatives below
    ############################################################################
    npop = S+I+R

    ############################################################################
    # Now give the expressions for the derivatives of S, I, and R wrt time
    # these come from the model equations. The following equations are for
    # an SIR model. When you write your own function, replace these with
    # your model equations
    ############################################################################
    dS = -beta*S*I/npop
```

```
    dI = +beta*S*I/npop - gamma*I
    dR = +gamma*I

    ############################################################################
    # vout is an output vector that contains the values of the derivates
    # the compartments in the vector on the RHS need to be in the same order as the
    # compartments used to fill the x vector!
    ############################################################################
    vout = c(dS,dI,dR)
    list(vout)
  })
}

##############################################################################################
##############################################################################################
##############################################################################################
# this is the same as the above function, except now it includes births and
# deaths (both with rate mu) in the model equations
##############################################################################################
derivative_calc_func_with_demographics=function(t, x, vparameters){
  S = x[1]
  I = x[2]
  R = x[3]

  with(as.list(vparameters),{
    npop = S+I+R
    dS = -beta*S*I/npop - mu*S + npop*mu
    dI = +beta*S*I/npop - gamma*I - mu*I
    dR = +gamma*I - mu*R
    out = c(dS,dI,dR)
    list(out)
  })
}

##############################################################################################
##############################################################################################
# this is the same as the derivative_calc_function, except now this involves
# calculating the derivatives of an SIR model with vaccination
# Rvac is the vaccinated (and now immune) compartment
# The vaccination begins at time_vaccination_begins, and ends at
# time_vaccination_ends
##############################################################################################
derivative_calc_func_with_vaccination=function(t, x, vparameters){
  S    = x[1]
  I    = x[2]
  R    = x[3]
  Rvac = x[4]

  with(as.list(vparameters),{
    npop = S+I+R+Rvac
    dS   = -beta*S*I/npop
    dI   = +beta*S*I/npop - gamma*I
    dR   = +gamma*I
    dRvac = 0
    if (t>=time_vaccination_begins&t<=time_vaccination_ends){
      dS   = dS - rho*S
      dRvac = +rho*S
    }
    out = c(dS,dI,dR,dRvac)
```

```
    list(out)
  })
}
```

6.2 R Code for Visualization of SIR Models with varied R_0 and N:

```
require("sfsmisc")
source("sir_func.R")
npop = 10000000
I_0 = 100000
R_0 = 0
S_0 = npop-I_0-R_0
tbegin = 0
tend  = 150
vt = seq(tbegin,tend,1)

gamma = 1/3
#R0   = 1.50
R0 = 10
beta = R0*gamma
vparameters = c(gamma=gamma,beta=beta)
inits = c(S=S_0,I=I_0,R=R_0)

solved_model = as.data.frame(lsoda(inits, vt, derivative_calc_func, vparameters))

cat("The item names in the solved_model object are:",names(solved_model),"\n")

vS = solved_model$S
vI = solved_model$I
vR = solved_model$R
vtime = solved_model$time
vnpop = vS+vI+vR

mult.fig(4,main="SIR model of pandemic influenza with R0=1.5")

ymax = 1.4*max(vI/npop)
plot(vtime,vI/npop,type="l",xlab="time",ylab="fraction
    infected",ylim=c(0,ymax),lwd=3,col=4,main="Infected")

n=length(vtime)
lines(vtime[2:n],-diff(vS)/(diff(vtime)*vnpop[1:(n-1)]),type="l",lwd=3,col=2)

legend("topright",legend=c("total infected (prevalence)","newly infected/day
    (incidence)"),bty="n",lwd=3,col=c(4,2))

ymin = 0.9*min(vS/vnpop)
plot(vtime,vS/vnpop,type="l",xlab="time",ylab="fraction
    susceptible",ylim=c(ymin,1),lwd=3,main="Susceptible")

iind = which.min(abs(vS/vnpop-1/R0)) # find the index at which S/N is equal to 1/R0
lines(c(vtime[iind],vtime[iind]),c(-1000,1000),col=3,lwd=3)
legend("bottomleft",legend=c("time at which S=1/R0"),bty="n",lwd=3,col=c(3),cex=0.7)

plot(vtime,log(vI/vnpop),type="l",xlab="time",ylab="log(fraction
    infected)",lwd=3,col=4,main="log(Infected)")

text(40,-14,"Initial\n exponential\n rise",cex=0.7)
```

```
lines(c(vtime[iind],vtime[iind]),c(-1000,1000),col=3,lwd=3)
legend("topleft",legend=c("time at which S=1/R0","log(Infected)"),bty="n",lwd=3,col=c(3,4),cex=0.7)

epsilon = 0.00001
vsinf = seq(0,1-epsilon,epsilon)

LHS = -log(vsinf)+log(S_0/npop)
RHS = R0*(1-vsinf)
iind = which.min(abs(RHS-LHS))
sinf_predicted = vsinf[iind]

cat("The final fraction of susceptibles at the end of the epidemic from the model simulation is
    ",min(vS/vnpop),"\n")
cat("The final fraction of susceptibles at the end of the epidemic predicted by the final size
    relation is ",sinf_predicted,"\n")
```

6.3 R Code for Plotting SIR models with varied β and γ:

```
source("sir_func.R")

npop = 499
I_0 = 1
R_0 = 0
S_0 = npop-I_0-R_0
tbegin = 0
tend  = 150
vt = seq(tbegin,tend,1)

gamma = c(0.1, 0.2, 0.3)
beta = c(0.5, 0.5, 0.5)
vparameters1 = c(gamma=gamma[1],beta=beta[1])
vparameters2 = c(gamma=gamma[2],beta=beta[2])
vparameters3 = c(gamma=gamma[3],beta=beta[3])
inits = c(S=S_0,I=I_0,R=R_0)

solved_model1 = as.data.frame(lsoda(inits, vt, derivative_calc_func, vparameters1))
solved_model2 = as.data.frame(lsoda(inits, vt, derivative_calc_func, vparameters2))
solved_model3 = as.data.frame(lsoda(inits, vt, derivative_calc_func, vparameters3))

colnames(solved_model1) <- c("time", "S1", "I1", "R1")
colnames(solved_model2) <- c("time", "S2", "I2", "R2")
colnames(solved_model3) <- c("time", "S3", "I3", "R3")

plt1 <- cbind(solved_model1$time, solved_model1$S1, solved_model2$S2, solved_model3$S3)
plt2 <- cbind(solved_model1$time, solved_model1$I1, solved_model2$I2, solved_model3$I3)
plt3 <- cbind(solved_model1$time, solved_model1$R1, solved_model2$R2, solved_model3$R3)

vS = solved_model1$S1
vI = solved_model1$I1
vR = solved_model1$R1
vtime = solved_model1$time
vnpop = vS+vI+vR

################################
#gamma plot                    #
################################
```

```
mult.fig(4,main="SIR model with varied gamma, by S, I, and R separately")

ymax = max(plt1[,2])*1.2
ymin = min(plt1[,2])*0.8
plot(plt1[,1],plt1[,2],type="l",xlab="time",ylab="S(t)",ylim=c(ymin,ymax),lwd=3,col=4,main="Susceptible
    Population")
lines(plt1[,1], plt1[,3], type='l', lwd=3, col=2)
lines(plt1[,1], plt1[,4], type='l', lwd=3, col="green")

legend("topright",legend=c("gamma=0.1", "gamma=0.2", "gamma=0.3"),bty="n",lwd=3,col=c(4,2,
    "green"))

ymax = max(plt2[,2])*1.2
ymin = min(plt2[,2])*0.8
plot(plt2[,1],plt2[,2],type="l",xlab="time",ylab="I(t)",ylim=c(ymin,ymax),lwd=3,col=4,main="Infected
    Population")
lines(plt2[,1], plt2[,3], type='l', lwd=3, col=2)
lines(plt2[,1], plt2[,4], type='l', lwd=3, col="green")

legend("topright",legend=c("gamma=0.1", "gamma=0.2", "gamma=0.3"),bty="n",lwd=3,col=c(4,2,
    "green"))

ymax = max(plt3[,2])*1.2
ymin = min(plt3[,2])*0.8
plot(plt3[,1],plt3[,2],type="l",xlab="time",ylab="R(t)",ylim=c(ymin,ymax),lwd=3,col=4,main="Removed
    Population")
lines(plt3[,1], plt3[,3], type='l', lwd=3, col=2)
lines(plt3[,1], plt3[,4], type='l', lwd=3, col="green")

legend("bottomright",legend=c("gamma=0.1", "gamma=0.2", "gamma=0.3"),bty="n",lwd=3,col=c(4,2,
    "green"))

#################################
#beta plot                      #
#################################
npop = 499
I_0 = 1
R_0 = 0
S_0 = npop-I_0-R_0
tbegin = 0
tend  = 150
vt = seq(tbegin,tend,1)

gamma = c(0.1, 0.1, 0.1)
beta = c(0.15, 0.2, 0.3)
vparameters1 = c(gamma=gamma[1],beta=beta[1])
vparameters2 = c(gamma=gamma[2],beta=beta[2])
vparameters3 = c(gamma=gamma[3],beta=beta[3])
inits = c(S=S_0,I=I_0,R=R_0)

solved_model1 = as.data.frame(lsoda(inits, vt, derivative_calc_func, vparameters1))
solved_model2 = as.data.frame(lsoda(inits, vt, derivative_calc_func, vparameters2))
solved_model3 = as.data.frame(lsoda(inits, vt, derivative_calc_func, vparameters3))

colnames(solved_model1) <- c("time", "S1", "I1", "R1")
colnames(solved_model2) <- c("time", "S2", "I2", "R2")
colnames(solved_model3) <- c("time", "S3", "I3", "R3")
```

```
plt1 <- cbind(solved_model1$time, solved_model1$S1, solved_model2$S2, solved_model3$S3)
plt2 <- cbind(solved_model1$time, solved_model1$I1, solved_model2$I2, solved_model3$I3)
plt3 <- cbind(solved_model1$time, solved_model1$R1, solved_model2$R2, solved_model3$R3)

vS = solved_model1$S1
vI = solved_model1$I1
vR = solved_model1$R1
vtime = solved_model1$time
vnpop = vS+vI+vR

mult.fig(4,main="SIR model with varied beta, by S, I, and R separately")

ymax = max(plt1[,2])*1.2
ymin = min(plt3[,2])*0.8
plot(plt1[,1],plt1[,2],type="l",xlab="time",ylab="S(t)",ylim=c(ymin,ymax),lwd=3,col=4,main="Susceptible
    Population")
lines(plt1[,1], plt1[,3], type='l', lwd=3, col=2)
lines(plt1[,1], plt1[,4], type='l', lwd=3, col="green")

legend("topright",legend=c("beta=0.15", "beta=0.2", "beta=0.3"),bty="n",lwd=3,col=c(4,2, "green"))

ymax = max(plt1[,3])*1.2
ymin = min(plt3[,3])*0.8
plot(plt2[,1],plt2[,2],type="l",xlab="time",ylab="I(t)",ylim=c(ymin,ymax),lwd=3,col=4,main="Infected
    Population")
lines(plt2[,1], plt2[,3], type='l', lwd=3, col=2)
lines(plt2[,1], plt2[,4], type='l', lwd=3, col="green")

legend("topright",legend=c("beta=0.15", "beta=0.2", "beta=0.3"),bty="n",lwd=3,col=c(4,2, "green"))

ymax = max(plt1[,4])*1.2
ymin = min(plt3[,4])*0.8
plot(plt3[,1],plt3[,2],type="l",xlab="time",ylab="R(t)",ylim=c(ymin,ymax),lwd=3,col=4,main="Removed
    Population")
lines(plt3[,1], plt3[,3], type='l', lwd=3, col=2)
lines(plt3[,1], plt3[,4], type='l', lwd=3, col="green")

legend("topleft",legend=c("beta=0.15", "beta=0.2", "beta=0.3"),bty="n",lwd=3,col=c(4,2, "green"))
```

6.4 R Code for Phase Plane Analysis:

```
require("phaseR")

sir <- function(t, y, parameters) {
  x <- y[1]
  y <- y[2]
  #n <- y[3]
  beta <- parameters[1]
  gamma <- parameters[2]
  dy <- numeric(2)
  dy[1] <- - beta*x*y
  dy[2] <- beta*x*y - gamma*y
  list(dy)
}

flowField <- flowField(sir, x.lim = c(-1,100), y.lim = c(-1,100),
                    parameters = c(0.3,0.1), points = 19, add = FALSE,
```

```
                    xlab = "S", ylab = "I", main = "Phase Plane with NullClines, beta=0.3,
                        gamma=0.1, R>1")
nullclines <-
  nullclines(sir, x.lim = c(-1, 100), y.lim = c(-1, 100),
             parameters = c(1/5,1/15), points = 500)
y0 <- matrix(c(99, 1, 40, 80, 30, 70, 20,20), ncol = 2, nrow = 4, byrow = TRUE)
trajectory <-
  trajectory(sir, y0 = y0, t.end = 300,
             parameters = c(0.03,0.1), colour = rep("black", 3))

flowField <- flowField(sir, x.lim = c(-1,100), y.lim = c(-1,100),
                    parameters = c(0.001,0.1), points = 19, add = FALSE,
                    xlab = "S", ylab = "I", main = "Phase Plane with NullClines, beta=0.001,
                        gamma=0.1, R<1")
nullclines <-
  nullclines(sir, x.lim = c(-1, 100), y.lim = c(-1, 100),
           parameters = c(0.001,0.1), points = 500)
y0 <- matrix(c(99, 1, 40, 80, 30, 70, 20,20), ncol = 2, nrow = 4, byrow = TRUE)
trajectory <-
  trajectory(sir, y0 = y0, t.end = 300,
           parameters = c(0.001,0.1), colour = rep("black", 3))
```

6.5 R Code for Model Fitting:

```
##############################################################################
##############################################################################
#Credit: Sherry Towers, ASU, Polymatheia
#A script emplying MC parameter sweeping algorithm
#Modified for visual proof of non-convexity of least square function
#and to fit for Congo Measles Data
#By Long Le
##############################################################################

##############################################################################
# par(pch=20) sets a solid round dot style for the plots
# The chron package contains utilities to calculate dates
##############################################################################
rm(list = ls(all = TRUE)) # resets R to fresh
require("chron")
require("sfsmisc")
par(pch=20)
source("sir_func.R")
set.seed(732552)

x <- c(25,37,92,209,167,149,68,19)
t <- 30*c(1,2,3,4,5,6,7,8,9)
adat <- as.data.frame(cbind(t,x))
#plot(df, pch=21, bg="red", xlab="time(month)", ylab="number of reports")
colnames(adat) <- c("times_of_observed", "incidence_observed")

##############################################################################
# read in 2007 influenza surveillance data (confirmed cases) for
# the Midwest (CDC region 5) from
# http://www.cdc.gov/flu/weekly/regions2007-2008/datafinalHHS/whoregX.htm
# where X=5 is midwest
# X=1 is northeast
# X=2 is NY and NJ
```

```
# X=3 are eastern seabord states like PA, DE, etc
#
# the weeks are number of weeks relative to Jan 1, 2007
# week 1 in 2007 ended Jan 6, 2007
##############################################################################
#adat = read.table("midwest_influenza_2007_to_2008.dat",header=T,sep=",")
#cat("\n")
#cat("The data file contains: ",names(adat),"\n")
#cat("\n")

##############################################################################
# The CDC data is weekly, with the date of each point corresponding to the
# date of the end of the week over which the data were collected.
# Let's convert these dates to time in days, relative to Jan 1, 2007
# We will be using this vector of dates, vtime_data, to obtain the model estimates
# of the incidence at that time.
# adat$week is relative to Jan 1, 2007, with week #1 occuring the first week in
# January, 2007.
##############################################################################
#adat$time_in_days_rel_jan_1_2007 = julian(1,6,2007)+(adat$week-1)*7-julian(1,1,2007)

##############################################################################
# Specifically for this data:
# make sure we are far enough into the season that there is at least one case per week
##############################################################################
#adat=subset(adat,week>=47)
incidence_observed = adat$incidence_observed #adat$B
times_of_observed = adat$times_of_observed #adat$time_in_days_rel_jan_1_2007
time_binning = min(diff(times_of_observed))

##############################################################################
##############################################################################
# now, set up the iterations of the Monte Carlo method
# At each iteration, we will randomly sample a hypothesis for the
# reproduction number, R0, and the time-of-introduction of the virus to the
# population, t0.
#
# With these hypotheses, we will solve for the model predicted incidence
# and calculate the least squares statistic comparing this to the observed
# incidence. We store the hypotheses for R0 and t0 in the vectors vR0 and vt0,
# and the resulting least squares statistic in the vector vleastsq
#
# At each iteration, we'll check if the predicted incidence is the best fit
# so far, and if so, we'll store that in vbest_leastsq_fit_incidence_prediction
#
# best_leastsq_so_far keeps track of the best-fit least squares so far obtained
# in the iterations.
##############################################################################
vR0 = numeric(0)
vt0 = numeric(0)
vleastsq = numeric(0)

best_leastsq_so_far = 1e10
vbest_leastsq_fit_incidence_prediction = rep(0,length(incidence_observed))

niter = 10000
for (iter in 1:niter){

  ############################################################################
```

```
# This process is computationally intensive, so once in a while during the
# iterations it is nice to inform user the script is doing something,
# and not hung
###############################################################################
if (iter%%100==0){
  cat("Doing iteration ",iter," out of ",niter,"\n")
}

###############################################################################
###############################################################################
# set up the model parameters
# npop is approximately the population of IL IN MI MN OH WI (CDC region 5)
# I_0 is the initial number infected
# R_0 is the inital number recovered and immune
# S_0 is the susceptibles
#
# 1/gamma is the average recovery period of the disease
# R0     is the reproduction number
# t0     is the time-of-introduction of the disease to the population, measured
#        in days from Jan 1, 2007
#
# For the SIR model, R0=beta/gamma, thus given our hypotheses for gamma and R0,
# we calculate beta as beta=R0*gamma (note that beta and gamma appear in the model
# equations, not gamma and R0, which is why we need to calculate beta given R0
# and gamma).
###############################################################################
npop = 78000000
I_0 = 1
R_0 = 0
S_0 = npop-I_0-R_0

gamma = 1/15

###############################################################################
# randomly sample R0 and t0 uniformly
###############################################################################
R0 = runif(1,3,6)
#t0 = as.integer(runif(1,160,(min(times_of_observed)-time_binning)))

###############################################################################
# or, you can use the Normal distribution to preferentially sample close to
# a particular value
###############################################################################
#R0 = rnorm(1,1.20,0.10)
#t0 = as.integer(rnorm(1,200,20))

###############################################################################
# calculate beta for the SIR model, and fill the vparameters and inits
# vectors that get passed to the lsoda method that solves our system of
# equations
###############################################################################
beta = R0*gamma

vparameters = c(gamma=gamma,beta=beta)
inits = c(S=S_0,I=I_0,R=R_0)

###############################################################################
# We get the model solution for all days from t0 to the last week of the
# data time series. If t0 is greater than the minimum date in the data time
```

```
# series, we need to print out a warning, because the time of introduction had
# to be before we actually started observing cases in the data!
##############################################################################
#t0 = adat$time[1]
t0 = 0
tmin = t0
tmax = max(times_of_observed)

if (tmin>(min(times_of_observed)-time_binning)){
  cat("\n")
  cat("*****************************************************************************\n")
  cat("*****************************************************************************\n")
  cat("The time-of-introduction is _after_ the first cases
      appeared!",t0,min(times_of_observed)-time_binning,"\n")
  cat("*****************************************************************************\n")
  cat("*****************************************************************************\n")
  cat("\n")
}
tmin = min(t0,min(times_of_observed)-time_binning)

vt = seq(tmin,tmax)

##############################################################################
# Now solve the system of differential equations numerically with lsoda in the
# deSolve package. Put the results in solved_model
# The derivative_calc_func for the SIR model is in the sir_func.R script
##############################################################################
solved_model = as.data.frame(lsoda(inits, vt, derivative_calc_func, vparameters))

##############################################################################
# the B influenza data is incidence, not prevalence (actually, the B influenza
# data is the true incidence times the fraction that the CDC actually confirms)
#
# The incidence over some time step is the difference in S over that time
# step (in a model with no births or deaths, immigration or emigration, or
# recurring susceptibility).
#
# The time step are derived from the dates at the end of the week of each
# data point (vtime_data)
#
# solved_model$time%in%vtime_data returns the indices of the elements of simodel$time that
# are also found in vtime_data
##############################################################################
tmin_data = min(times_of_observed)-time_binning
tmax_data = max(times_of_observed)
vtime_data = seq(tmin_data,tmax_data,time_binning)

susceptible_predicted = solved_model$S[solved_model$time%in%vtime_data]
incidence_predicted = -diff(susceptible_predicted)

##############################################################################
# from the model estimate of the incidence and the data, we can
# estimate the fraction of cases that were confirmed
##############################################################################
frac_confirmed = sum(incidence_observed)/sum(incidence_predicted)

##############################################################################
# normalize the model prediction so area under curve
# equals the sum of the data incidence
```

```
#############################################################################
incidence_predicted = incidence_predicted*frac_confirmed

#############################################################################
# now calculate the least-squares
# statistic that compares the data to this model calculated
# under a particular hypothesis of R0 and t0
#############################################################################

if (length(incidence_predicted)==length(incidence_observed)
   &!is.na(sum(incidence_predicted))){

  ###########################################################################
  # calculate the least squares statistic
  ###########################################################################
  leastsquares = sum((incidence_predicted-incidence_observed)^2)
  vR0 = append(vR0,R0)
  vt0 = append(vt0,t0)
  vleastsq = append(vleastsq,leastsquares)

  if (leastsquares<best_leastsq_so_far){
    best_leastsq_so_far = leastsquares
    vbest_leastsq_fit_incidence_prediction = incidence_predicted
    R0_best = R0
    t0_best = t0
    cat("The best value of R0 so far is:",R0_best,"\n")
    cat("The best value of t0 so far is:",t0_best,"\n")
  }

  ###########################################################################
  # plot the best-fit results every once in a while
  # cex is the point size
  ###########################################################################
  if (iter%%100==0){
    text_main = paste("Confirmed Measles Cases, Congo, 2017 season:\n result of",iter,"Monte
        Carlo fit iterations")
    #text_main = paste("Least Square Calculation with varied reproduction number")
    mult.fig(4,main=text_main,oma=c(1,2,4,1))

    num_points_to_show = 250
    ymax = max(vleastsq)
    #if (length(vleastsq)>num_points_to_show) ymax = sort(vleastsq)[num_points_to_show]
    l = which(vleastsq<=ymax)

    lmin = which.min(vleastsq)

    plot(vR0[l]
         ,vleastsq[l]
         ,ylab="Least squares"
         ,xlab="R0 hypothesis"
         ,main=paste("Best-fit R0 so far:",round(R0_best,3))
         ,col.main=3)
    points(vR0[lmin],vleastsq[lmin],col=3,cex=2)

    plot(vt0[l]
         ,vleastsq[l]
         ,ylab="Least squares"
         ,xlab="t0 hypothesis (days rel Jan 1, 2017)"
         ,main=paste("Best-fit t0 so far:",t0_best)
```

```
        ,col.main=3)
    points(vt0[lmin],vleastsq[lmin],col=3,cex=2)

    ymax = max(c(incidence_observed,vbest_leastsq_fit_incidence_prediction))
    plot(times_of_observed
        ,incidence_observed
        ,ylim=c(0,1.2*ymax)
        ,xlab="Time, in weeks relative to Jan 1, 2017"
        ,ylab="Incidence"
        ,cex=2)
    lines(times_of_observed
         ,vbest_leastsq_fit_incidence_prediction
         ,col=2
         ,lwd=5)
  }

 } # end check that the predicted incidence vector is the same length as the observed and doesn't
     contain NA's
} # end loop over the Monte Carlo iterations
```

6.6 R Code for Markov Chain Analysis:

```
########################################
#Author: Long Le                 #
#A script to simulate Markov Chain Model#
#and find the number of steps    #
#to reach an absorbing state (if any) #
########################################
install.packages("markovchain")
require("markovchain")

mcSIR <- new("markovchain", states=c("S","I","R"),
           transitionMatrix=matrix(data=c(1-3.233*1/15,3.233*1/15,0,0,1-1/15,1/15,0,0,1),
           byrow=TRUE, nrow=3), name="SIR")
initialState <- c(77999999,1,0)

timesteps <- 300
sir.df <- data.frame( "timestep" = numeric(),
                  "S" = numeric(), "I" = numeric(),
                  "R" = numeric(), stringsAsFactors=FALSE)
for (i in 0:timesteps) {
  newrow <- as.list(c(i,round(as.numeric(initialState * mcSIR ^ i),0)))
  sir.df[nrow(sir.df) + 1, ] <- newrow
}

text_main = paste("Discrete Time Markov Chain Simulation, One Realization")
mult.fig(1,main=text_main,oma=c(1,2,4,1))
plot(sir.df$timestep,sir.df$S)
points(sir.df$timestep,sir.df$I, col="red")
points(sir.df$timestep,sir.df$R, col="green")


absorbingStates(mcSIR)
transientStates(mcSIR)
ab.state <- absorbingStates(mcSIR)
occurs.at <- min(which(sir.df[,ab.state]==max(sir.df[,ab.state])))
```

occurs.at