Some Topics in Modelling South African COVID-19 Epidemiological Data

by

Sinoxolo Moyomuhle Nene (214522049)

A dissertation presented in the partial fulfilment of the requirements for the degree of Master of Science in Physics

School of Chemistry and Physics
College of Agriculture, Engineering and Science
University of KwaZulu-Natal
Westville Campus
South Africa

Supervised by Prof. Martin Bucher and Prof. Francesco Petruccione

30 November 2022
Corrected version: 3 April 2023

As the candidate's Supervisor I agree to the resubmission of this thesis with corrections.

Supervisor's Signature:

Martin Bucher

Declaration - Plagiarism

I, Sinoxolo Moyomuhle Nene, declare that

1. The research reported in this thesis is my own work, except where stated otherwise.
2. This thesis has not been submitted for any degree or examination at any other university.
3. This thesis does not contain any data, pictures, graphs, or other information from other people unless they are specifically acknowledged as coming from other people.
4. This thesis contains no writing from other people unless it is specifically acknowledged as being sourced from other researchers. Where other written sources have been cited, then:
   (a) Their words have been re-written, but the general information attributed to them has been referenced
   (b) When their exact words were used, their writing was placed in italics, enclosed in quotation marks, and referenced.
5. This thesis does not include any text, graphics, or tables copied and pasted from the Internet unless the source is specifically acknowledged and detailed in the thesis and References sections.

Signed :

Sinoxolo Moyomuhle Nene

Acknowledgments

First and foremost, I want to thank both of my supervisors. Thank you for believing in me and providing me with the opportunity to grow. Prof. Martin Bucher, I appreciate your constant support, patience, and guidance. Also, thank you for genuinely caring about my well-being and looking out for me.

Prof Francesco Petruccione and Miss Neli Mncube, thank you for ensuring that I had a working space on campus and helping me with all of the admin. I want to thank NITHeCS for their financial support.

Lastly, I also want to express my gratitude to my family, connect group and friends for their emotional and mental support.

Abstract

The ongoing COVID-19 epidemic produces a wide variety of quality data, which can be analyzed using simple mathematical models inspired by statistical physics. We explore some underlying methods and apply them to the simulated data using parameters extracted from the previously analyzed data by Pulliam et al. to determine whether the Omicron variant reinfects at a higher rate compared to other previous Variants of Concern. First, using simple prototype models, we investigate whether simple dynamical systems of low dimensions are inherently predictable. The concept of a strange attractor is defined and numerically explored using the Duffing oscillator as an example. The statistical theory of linear parametric models is then investigated mathematically and applied to some standard datasets with the R statistical programming language. We also study the Markov Chain Monte Carlo technique, exploring complicated models and presenting mathematical theory, numerical implementation, optimization, and convergence diagnostics. Finally, we apply the MCMC techniques to estimate the parameters of our model using simulated data for reinfections. Based on the analysis of the mock data, we found that the Omicron variant does not have a higher reinfection rate, as expected since our simulated data for reinfections had no difference in the reinfection rate. Although Pulliam et al. claimed to make all their data available, the required data for analyzing relative reinfection rates was not made publicly available, supposedly on account of privacy concerns. This is why we were forced to use mock data, which in any case would need to be generated to test and validate the code.

Contents

List of Tables

List of plots

1 Introduction

Since the COVID-19 pandemic began in December 2019, the severe acute respiratory syndrome coronavirus 2 (SARS-CoV-2) has continuously mutated, resulting in the emergence of numerous variants around the world [1]. These variants were classified by the World Health Organization (WHO) into three categories: variant of interest (VOI), variant of concern (VOC), and variant under monitoring (VUM) [2], [32]. South Africa had experienced four distinct pandemic waves by February 2022, all caused by the ancestral SARS-CoV-2 and three VOCs (Beta, Delta and Omicron BA.1) [3], [10]. The introduction of the Alpha, Beta, and Delta SARS-CoV-2 VOCs was linked to new waves of infections, sometimes spreading over the whole globe. The first case discovered in South Africa was in early March 2020, followed by a wave that peaked in July and ended in September. The Beta (B.1.351 / 501Y.V2 / 20H) variant, first discovered in South Africa in October 2020, drove the second wave, which peaked in January 2021 and waned in February. The Delta (B.1.617.2 / 478K.V1 / 21A) variant was dominant in the third wave, which peaked in July and ended in September 2021 [8].

As 2021 was coming to an end, with natural and vaccine-induced immunity increasing, people were optimistic, thinking that the worst was over and the pandemic was coming to an end. However, after a sudden rapid increase in daily confirmed cases linked to the Omicron variant of SARS-CoV-2, panic and anxiety set in, causing global concern in late November 2021. The Omicron variant was discovered in South Africa, Gauteng Province, and caused a dramatic spike in COVID-19 cases [8]. The World Health Organization (WHO) recognized Omicron as a variant under monitoring (VUM) on November 24, 2021, but it was reclassified as a variant of concern (VOC) two days later [5]. The Omicron variant has many mutations, including 15 mutations in the spike receptor-binding domain (RBD). It has several mutations in common with the preceding VOC Alpha, Beta, and Gamma variants, which sparked widespread concern regarding viral transmissibility, pathogenicity, and immune evasion. The Omicron variant is the most highly mutated VOC thus far, paving the path for increased transmissibility and partial resistance to immunity acquired by COVID-19 vaccinations [5].

Many mutations in variants like Delta and Alpha are linked to immune escape and the possibility of increased transmissibility. However, there are significant uncertainties regarding the Omicron's transmissibility, severity, and pathogenicity, as well as the effect of vaccine protection against Omicron infection [6]. The previous VOCs (Alpha, Beta, and Delta) emerged in a world where natural immunity to COVID-19 infections was prevalent. However, the Omicron variant emerged as global vaccination was increasing; thus, the introduction of this variant raised the critical question of whether the Omicron variant has a higher rate of reinfections than the previous VOCs.

The paper by Pulliam et al. [9] investigates whether SARS-CoV-2 reinfection risk changed over time in South Africa in the context of the emergence of the Beta (B.1.351), Delta (B.1.617.2), and Omicron (B.1.1.529) variants. They examined a large routine national dataset containing all laboratory-confirmed SARS-CoV-2 cases in the country from South Africa's National Notifiable Medical Conditions Surveillance System, allowing for a thorough examination of suspected reinfections. The data had positive tests of 2,796,982 people with SARS-CoV-2 who had a positive test result at least 90 days before November 27, 2021 [8]. The tests had specimen receipt dates ranging from March 4, 2020, to January 31, 2022. They devised two methods for monitoring routine epidemiological surveillance data. Individuals with consecutive positive tests at least 90 days apart were assumed to have suspected reinfections. Their routine reinfection risk monitoring included comparing reinfection rates to the expectation under a null model (approach 1) and estimating the time-varying hazards of infection and reinfection throughout the epidemic (approach 2) based on a model-based reconstruction of the susceptible populations eligible for primary and secondary infections. They discovered no evidence of increased reinfection risk associated with the circulation of the Beta (B.1.351) or Delta (B.1.617.2) variants. However, they did discover clear population-level evidence of immune evasion by the Omicron (B.1.1.529) variant in previously infected South Africans.

Since COVID-19 was discovered in South Africa in March 2020, there have been approximately 4 million confirmed cases as of June 2022. The new daily confirmed cases increased exponentially in the early days of the outbreak, but after that, the curve started to exhibit the up and down trends of new daily confirmed cases as shown in Figure 1. The curve has multiple peaks that are caused by different SARS-CoV-2 variants and we look at simple prototype models to assess whether the dynamics of the curve displaying the trends of new daily confirmed cases are inherently predictable. Various mathematical modeling methods have been used to predict the course of COVID-19 [15], including the classical susceptible-infected-recovered (SIR) model and its modifications. For a long time, models based on ordinary differential equations (ODEs) have been used to model the classical dynamics of epidemics. Kermack and McKendrick proposed the SIR model in 1927 to simulate the spread of infectious diseases like measles and rubella [13]. Since then, it has been used to model epidemics ranging from AIDS to SARS (severe acute respiratory syndrome). SIR modeling uses a set of differential equations to estimate how the number of infected and recovered people varies over time, given a particular rate of infection and recovery. SIR models give insights and predictions regarding the pandemic and the probability of reinfections.

We categorize SIR models into three host compartments: susceptible, infected, and removed, with a susceptible individual moving to the infected compartment upon infection and an infected individual moving to the removed compartment following removal/recovery [12]. Individual progress is schematically illustrated by the sequence

$$S \to I \to R.$$

In any model, the assumptions about the infection transmission and incubation period are critical; equations and the parameters represent these assumptions. In classical SIR models, we assume that:

1. The growth in the infected class, βSI, is proportional to the number of infected and susceptibles where $\beta > 0$ is a constant parameter.

2. The rate of infected persons moving to the removed class is proportional to the number of infected, that is, γI, where $\gamma > 0$ is a constant and $1/\gamma$ is a measure of the average time spent in the infectious state.

3. The incubation time is brief enough that it is negligible; that is, a susceptible who gets the disease becomes infected immediately.

The susceptible population declines monotonically towards zero in the standard SIR model. The model mechanism based on the assumptions stated above is

$$\begin{aligned} \frac{dS}{dt} &= -\beta SI, \\ \frac{dI}{dt} &= \beta SI - \gamma I, \\ \frac{dR}{dt} &= \gamma I \end{aligned} \tag{1}$$

where $\beta > 0$ represents the infection rate, and $\gamma > 0$ represents the removal rate from the infected state. The constant population size N is built into the system (1) since, on adding the equations

$$\frac{dS}{dt} + \frac{dI}{dt} + \frac{dR}{dt} = 0 \Rightarrow S(t) + I(t) + R(t) = N, \tag{2}$$

where N is the total size of the population. One of the fundamental assumptions of the classical SIR model is that the infected and susceptible populations mix uniformly and that the total population remains constant throughout time. While this assumption helps simplify dynamics and computing outcomes, it does not apply to the transmission of the COVID-19 virus since finding these parameters is not straightforward. This population-level random matching assumption ignores critical aspects of reality and localization. People are more likely to engage with members of their social network, characterized generally by familial relationships, employment and social contexts, and geography. Moreover, individual behavioral responses, as well as health and economic policies aimed at disease mitigation, such as social isolation, self-quarantine, and wearing a face mask, can affect the rate of viral transmission through a person's network of contacts differently than the population as a whole [14]. Not everyone is the same: some have many contacts, while others have few. To simulate this variability, several compartments like SEIR are required. According to the Grant

et al. ([16]) study, SEIR models provide accurate estimates of the position of equilibrium points when the rate at which individuals enter each stage is equal to the rate at which they exit it. They do not, however, accurately capture the distribution of time that an individual spends in each compartment and thus do not accurately capture the transient dynamics of epidemics. Despite the simplicity and effectiveness of SIR-based models in forecasting other infectious diseases, their ability to forecast the COVID-19 pandemic is debatable [15].

The following is an outline of this thesis. Chapters 2 and 3 deal with the theory of dynamical systems and questions of predictability. The SIR model above and its compartmentalized generalizations is an example of a dynamical system. In principle, if the ordinary differential equation defining the dynamical system is known, we could predict the future course of the epidemic. As shown in the picture below, the number of new daily confirmed cases of COVID-19 in South Africa increased exponentially in the early days of the outbreak and began to exhibit up and down trends, we look at dynamical systems to see if the dynamics of COVID-19 are inherently predictable. We also look at nonlinear dynamical systems to see how the dynamics of COVID-19 behave over time because the system described in Figure 1 is not linear and defined the concept of strange attractors, numerically looking at the Duffing oscillator as an example since nonlinear dynamical systems tend to depict chaotic behavior. Through these model systems we investigated the kinds of behavior that could result. However, not knowing the right ODE we cannot determine the predictability of the epidemic but only cast some light on the possible behaviors. In Chapter 4 we turn to the statistical theory of linear parametric models. Some publicly available datasets are analyzed using routines from the R statistical programming language to illustrate and validate the techniques presented. In Chapter 5 we explore the Markov Chain Monte Carlo technique, investigating complex models and presenting both mathematical theory, numerical implementation, optimization, and convergence diagnostics. Finally, in Chapter 6 we apply the MCMC technique to the simulated data using parameters extracted from the previously analyzed data by Pulliam et al.

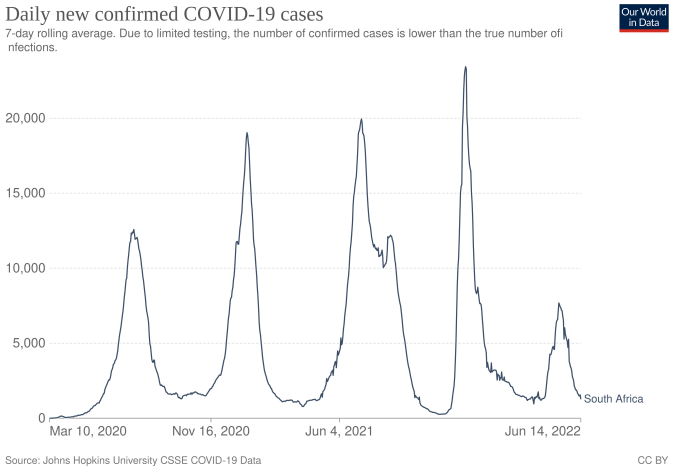


Figure 1: Daily new confirmed COVID-19 cases in South Africa from March 2020 to June 14, 2022. The first wave peak was driven by the original strain (aka the wild type), the second by the Alpha variant, the third by the Beta variant, the fourth by the Delta variant, and thd last by the Omicron variant.

2 Dynamical Systems

A dynamical system has a state that changes over time (t). Suppose we have an initial point $X \in \mathbb{R}^n$, a discrete dynamical system on $\mathbb{R}^n$ informs us where X is one unit of time later, two units later and so on. Moreover $X_1, X_2, ..., X_n$ denotes the new X positions. At time zero, X is at location X_0. X_t gives the "trajectory" of X ([17]). There are two types of dynamical systems in applications: those with discrete-time variables ($t \in \mathbb{Z}$, or $\mathbb{N}$) and those with continuous-time variables ($t \in \mathbb{R}$). Discrete dynamical systems are represented as a function iteration, such as

$$x_{t+1} = f(x_t), t \in \mathbb{Z}, or \mathbb{N}. \tag{3}$$

When t is continuous, the dynamics are commonly given by the differential equation

$$\frac{dx}{dt} = \dot{x} = X(x). \tag{4}$$

In (3 and 4), x represents the system's state and takes values from the state space or phase space. The phase space is sometimes Euclidean space or a subset of it, but it can also be a non-Euclidean structure like a circle, sphere, torus, or some other differentiable manifold. The function that converts t to X_t returns either a series of points or a curve in $\mathbb{R}^n$ that depicts X's life history as time passes from negative infinity to positive infinity. Various examples of dynamical systems impose various rules about how the function X_t depends on t. Ergodic theory, for example, deals with such functions under the premise that they maintain a measure on $\mathbb{R}^n$ [19]. Topological dynamics is concerned with such functions under the premise that X_t only changes continuously. We commonly assume that the function X_t is continuously differentiable in the case of differential equations. A function is continuously differentiable if all its partial derivatives exist and are continuous across its domain. Let us consider a dynamical system that is defined by the ordinary differential equation

$$\dot{x} = f(x, t; \mu), \tag{5}$$

where x is a dynamical variable that can be an element of the n-dimensional Euclidian space $\mathbb{R}^n$. The dynamical variable x is also known as the system's state point. The dot denotes differentiation with respect to time t. The vector field f is the right-hand side of equation that smoothly maps $\mathbb{R}^n \times \mathbb{R}$ to $\mathbb{R}^n$. When the vector field f explicitly includes time t as an argument ($\dot{x} = f(x, t; \mu)$), the dynamical system is said to be non-autonomous, but it is said to be autonomous when it does not ($\dot{x} = f(x; \mu)$). Equation 5 is a linear dynamical system where f is a linear function that may be represented by a $n \times n$ matrix. It is a nonlinear dynamical system where f is a nonlinear function.

The study of geometrical properties of trajectories and long-term behavior is the focus of dynamical systems. Dynamical systems may model a wide range of behaviors, including the movements of planets in solar systems, the spread

of diseases in a population, the form and development of plants, the interaction of optical pulses, and the mechanisms that govern electrical circuits and heartbeats. Consider Figure 1, which depicts the long-term dynamics of the new daily COVID-19 confirmed cases in South Africa. Do they behave consistently and predictably? We look at the nonlinear dynamical systems to see how they behave in the long term since the system described in Figure 1 is not linear.

2.1 Nonlinear Dynamical Systems

Nonlinear dynamical systems, as opposed to linear systems, do not obey the superposition principle, which means that their outputs are not directly proportional to their inputs. A nonlinear system of equations is typically used to define the behavior of a nonlinear system. They have radically changed how scientists perceive the world. For a long time, it was considered that determinism implies predictability or that if the behavior of a system was specified, for example, via a differential equation, then the future state of the system's solutions could be predicted indefinitely. Far into the future, nonlinear dynamics demonstrated that this assumption was wrong. Even if the evolution of a system is entirely specified by an ordinary differential equation (ODE) and an initial condition, its trajectory is impossible to predict because non-linearity causes instability, leading to a lack of predictability. Suppose $x^* \in \mathbb{R}^n$ is an equilibrium point for the differential equation

$$\dot{x} = f(x). \tag{6}$$

Then x^* is a stable equilibrium if, for every neighborhood $\mathcal{O}$ of x^* in $\mathbb{R}^n$, there exists a neighborhood $\mathcal{O}_1$ of x^* in $\mathcal{O}$ such that any solution $x(t)$ with $x(0) = x_0$ in $\mathcal{O}_1$ is defined and remains in $\mathcal{O}$ for all $t > 0$. Another type of stability is asymptotic stability. x^* is asymptotically stable when $\mathcal{O}_1$ is selected so that we obtain $\lim_{t\to\infty} x(t) = x^*$ in addition to the stability properties. When the equilibrium x^* is not stable, it is referred to as unstable. This means that for any neighborhood $\mathcal{O}_1$ of x^* in $\mathcal{O}$, there is at least one solution $x(t)$ starting at $x(0) \in \mathcal{O}_1$ that does not lie entirely in $\mathcal{O}$ for every $t > 0$.

Fixed points represent equilibrium solutions [20]. Stable fixed points are often called attractors and unstable fixed points are also known as repellers or sources. Stable fixed points cause nearby trajectories converge to them, whereas unstable fixed points cause nearby trajectories diverge from them. To numerically determine the stability of fixed points, the Jacobian matrix, an analogue of the derivative in multidimensional space, is required. Let $\mathbf{f}$ be a map on $\mathbb{R}^2$ and $x, y \in \mathbb{R}^2$, the Jacobian matrix $\mathbf{J_f}(x, y)$ of $\mathbf{f}$ at point (x, y) is

$$\mathbf{J_f}(x, y) = \begin{bmatrix} \frac{\partial f_1}{\partial x} & \frac{\partial f_1}{\partial y} \\ \frac{\partial f_2}{\partial x} & \frac{\partial f_2}{\partial y} \end{bmatrix}. \tag{7}$$

The Jacobian determinant is

$$\Delta = \det(\mathbf{J_f}(x,y)) = \frac{\partial f_1}{\partial x} \times \frac{\partial f_2}{\partial y} - \frac{\partial f_2}{\partial x} \times \frac{\partial f_1}{\partial y}, \tag{8}$$

and the trace of the Jacobian is

$$\tau = \mathrm{Tr}(\mathbf{J_f}(x,y)) = \frac{\partial f_1}{\partial x} + \frac{\partial f_2}{\partial y}. \tag{9}$$

The eigenvalues of a Jacobian matrix are determined by the characteristic equation $\det(\mathbf{J_f}(x,y) - \lambda I) = 0$, where I is the identity matrix. For a 2×2 Jacobian matrix, the two eigenvalues are

$$\lambda_1 = \frac{\tau + \sqrt{\tau^2 - 4\Delta}}{2}, \tag{10}$$

and

$$\lambda_2 = \frac{\tau - \sqrt{\tau^2 - 4\Delta}}{2}. \tag{11}$$

The eigenvalues of the Jacobian matrix only depend on the determinant and trace of the Jacobian matrix. If $\mathbf{f}$ be a map on $\mathbb{R}^n$ and let $\mathbf{p}$ be a fixed point $\mathbf{f}$. Then,

1. If the real component of each $\mathbf{J_f}(\mathbf{p})$ eigenvalue is negative, $\mathbf{p}$ is a sink.
2. If the real component of each $\mathbf{J_f}(\mathbf{p})$ eigenvalue is positive, $\mathbf{p}$ is a source.
3. If at least one eigenvalue of $\mathbf{J_f}(\mathbf{p})$ has a negative real component and another has a positive real component, then $\mathbf{p}$ is a saddle.

Sources and saddles are examples of unstable equilibria. To show how the fixed points of a dynamical system can be associated very different behavior in nearby solution curves, we consider the system below.

$$\dot{x} = x(3 - x - 2y) \tag{12}$$

$$\dot{y} = y(2 - x - y) \tag{13}$$

First, we determine the fixed points. They occur at $\dot{x} = 0$ and $\dot{y} = 0$ at the same time, hence this system has four fixed points: (0,0), (3,0), (0,2), and (1,1). The Jacobian matrix A used to categorize the fixed points as per Figure 2 is

$$A(x^*, y^*) = \begin{bmatrix} 3 - 2x - 2y & -2x \\ -y & 2 - x - 2y \end{bmatrix} \tag{14}$$

At the fixed points, the Jacobian matrix is then evaluated. The Jacobian at the origin (0,0) is

$$A(0,0) = \begin{bmatrix} 3 & 0 \\ 0 & 2 \end{bmatrix} \tag{15}$$

Δ and τ are 6 and 5, respectively, we then calculated the eigenvalues ($\lambda_{1,2} = 3, 2$) using equations 10 and 11. Since $\lambda_{1,2}, \Delta, \tau > 0$, from Figure 2 that the origin is an unstable node. The Jacobian matrix for the fixed point (3,0) is

$$A(3,0) = \begin{bmatrix} -3 & -6 \\ 0 & -1 \end{bmatrix} \tag{16}$$

The matrix at (3,0) has the following eigenvalues$\lambda_{1,2} = -1, -3$. Because both the eigevalues are < 0, $\Delta = 3 > 0$ and $\tau = -4 < 0$, (3,0) is a stable node. Then

$$A(0,2) = \begin{bmatrix} -1 & 0 \\ -2 & -2 \end{bmatrix} \tag{17}$$

The eigenvalues for the matrix at (0,2) are $\lambda_{1,2} = -1, -2$, this is also a stable point since $\Delta = 3$ is positive and $\tau = -4$ negative. Lastly we compute the Jacobian matrix at (1,1)

$$A(1,1) = \begin{bmatrix} -1 & -2 \\ -1 & -1 \end{bmatrix} \tag{18}$$

This matrix has eigenvalues, $\lambda_{1,2} = -1 + \sqrt{2}, -1 - \sqrt{2}$, $\Delta = -1$, and $\tau = -2$. The eigenvalues have opposite signs and both the determinant and trace are $<$ 0, so this point is a saddle point. As two of the fixed points are stable nodes (attractors or sinks), some of the trajectories that start near the origin diverge from the origin since it is a source and go to the stable node on the x-axis, while others go to the stable node on the y-axis.

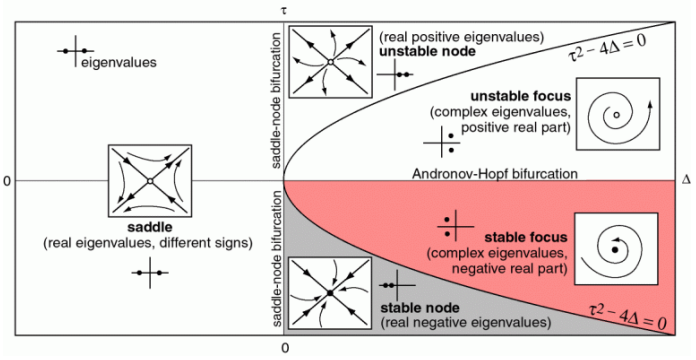


Figure 2: Fixed point classification based on the trace τ and determinant Δ of the Jacobian matrix.

In 2D phase flow, not all attractors are fixed points. There is just one other kind of attractor: the limit cycle. A limit cycle is a closed, isolated trajectory. Isolated trajectories are those in which neighboring trajectories are not closed and spiral either toward or away from the limit cycle. If all trajectories near a limit cycle converge to it as $t \to \infty$, it is said to be asymptotic stable. Otherwise, it is unstable, or in exceptional cases, partially stable [20, 24]. Stable limit cycles

are particularly significant in science because they simulate systems that exhibit self-sustaining oscillations. In other words, even in the absence of external periodic stimulation, these systems oscillate (i.e., the beating of a heart) [20]. Limit cycles are nonlinear phenomena that cannot occur in linear systems. Of course, the linear system $\dot{\boldsymbol{x}} = A\boldsymbol{x}$ can have closed orbits, but they are not isolated; if $\boldsymbol{x}(t)$ is a periodic solution, then $c\boldsymbol{x}(t)$ is as well, for any constant $c \neq 0$. As a result, $\boldsymbol{x}(t)$ is surrounded by a one-parameter family of closed orbits. The amplitude of a linear oscillation is entirely defined by its initial conditions; any minor perturbation in the amplitude will remain indefinitely. On the other hand, limit cycle oscillations are governed by the system's structure.

3 Strange Attractors

Suppose that $\mathcal{A} \subset \mathbb{R}^n$ is an attractor. If $\mathcal{A}$ is chaotic, it is known as a strange attractor, where chaos is the aperiodic long-term behavior in a deterministic system sensitive to initial conditions. "Aperiodic long-term behavior" refers to trajectories that do not settle down to fixed points, periodic orbits, or quasiperiodic orbits as $t \to \infty$. The word "deterministic" refers to the system's absence of random or noisy input parameters. The nonlinearity of the system, rather than noisy driving forces, causes irregular behavior. Sensitive dependency on initial conditions refers to neighboring trajectories diverging exponentially fast. Strange attractors have complex structures arising from simple dynamical systems. A dynamical system is chaotic if it has just one positive Liapunov exponent.The concept of the Liapunov exponent is that given an initial condition x_0, consider a neighboring point $x_0 + \delta_0$ where the initial separation δ_0 is extremely small. Let δ_n be the separation after n iterations. If $|\delta_n| \approx |\delta_0|^{n\lambda}$, λ is called a Liapunov exponent.

The equation of a particular dynamical system specifies the behavior of a specific period. To explain why an equation of a particular dynamical system cannot specify the behavior for all times, let us consider the weather as an example. The atmosphere governs the weather, and the atmosphere follows deterministic physical laws. However, long-term weather predictions have made little progress despite thorough, precise observations and the unleashing of vast computing resources. This unpredictability is because weather is extremely sensitive to initial conditions. A very small change in the weather today (the initial conditions) generates a large change in the weather tomorrow and an even larger change in the weather the next day. This sensitivity to initial conditions has been called the butterfly effect because a butterfly flapping its wings in Brazil might hypothetically bring off tornadoes in Texas. Long-term prediction is difficult because we can never know the initial conditions with arbitrary precision, even if the physical laws are deterministic and precisely known. The predictability horizon in weather forecasting has been demonstrated to be no more than two or three weeks [23].

Strange attractors differ from other phase-space attractors because the system's position on the attractor is unpredictable. They combine aspects of order and disorder, predictability and unpredictability, from the same systems and equations in an exciting way. The motion of the attractor is locally unstable because nearby orbits are pushed apart, but globally stable because the orbits will always trace out a strange attractor in the same way.

In this study, we explore chaotic behavior using the motions of the Duffing Oscillator as an example, which is particularly powerful due to its resemblance to the classic linear damped driven oscillator. The Duffing Oscillator is more than just an equation that depicts chaos; it is a powerful model that may be applied in various fields. As a result, it is critical to comprehend its behaviors and

properties to get any relevant predictions from the model. Fundamentally, the Duffing oscillator is described by Duffing's equation, a second-order nonlinear differential equation of the form

$$\ddot{x} + \delta\dot{x} + \alpha x + \beta x^3 = \gamma \cos \omega t \tag{19}$$

where δ determines the amount of damping, α controls the linear stiffness, and β controls the amount of nonlinearity in the restoring force. If $\beta = 0$, the Duffing equation describes a damped and driven simple harmonic oscillator; γ determines the strength of the driving force, which oscillates at an angular frequency ω. If $\gamma = 0$, the system is without a driving force and when the strength of the driving force (γ) is increased, the system transitions to chaos.

Solutions to the Duffing Oscillator can exhibit very interesting and nontrivial behavior depending on the values of the parameters. Let us first consider a case where all the parameters of the Duffing equation are set to 1. As shown in Figure 3, we get a nice orbit when all parameters are set to 1. The system's solution tends to one limit cycle at longer times for initial values within or outside the limit cycle attractor. When the damping and nonlinear restoring forces are both zero, the system has imaginary eigenvalues and behaves like an underdamped oscillator. As a result, the orbit is pushed farther and further out from the center, as seen in Figure 4. Since γ is the damping force, it can be adjusted to vary the solution paths. So we are interested in observing what happens to the Duffing oscillator when the other parameters are fixed while γ is varied. In Figure 5, all other parameters were fixed to 1 while γ was set to 0.1, causing the system to shoot forth and be driven outward. Over time, the damping and driving force reaches an equilibrium and approaches a limit cycle. We observed a somewhat different pattern when we increased the driving force to $\gamma = 12.34$, as seen in Figure 6. When γ is 12.34, the system oscillates about a few points and has a larger amplitude than when γ is 0.1. Even though it starts at the same position in Figures 5 and 6, it propelled far further, resulting in two very different dynamics. There are areas in the system where the small changes in γ change the orbit, and as γ get larger and larger over time, the spring orbit get driven further and further outwards.

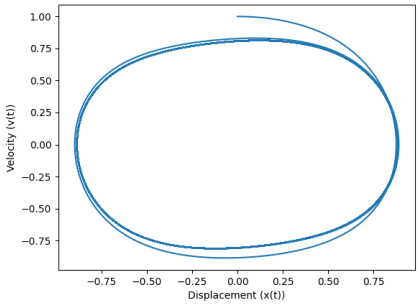


Figure 3: Duffing oscillator ($\alpha = \beta = \delta = \gamma = \omega = 1$). After the decay of an initial transient, the solution becomes periodic.

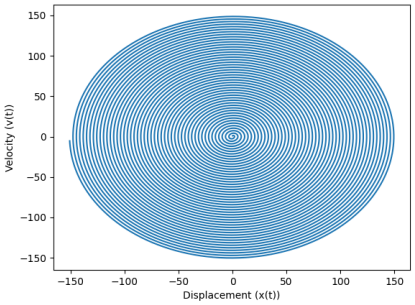


Figure 4: Duffing oscillator similar to Figure 3, but with $\beta = \delta = 0$ instead of $\beta = \delta = 1$. With no damping nor nonlinearity, one observes an outward spiral.

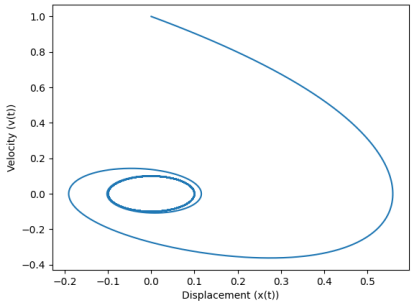


Figure 5: Duffing oscillator similar to Figure 3, but with $\gamma = 0.1$ instead of $\gamma = 1$.

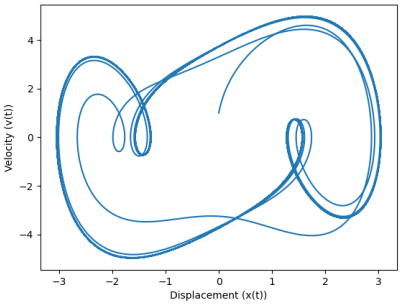


Figure 6: Duffing oscillator same as Figure 5, but with a stronger driving force.

So far, we have been adjusting the damping force γ while maintaining all other parameters at 1, and now we want to see what happens to the system when we change the other parameters. In Figure 7, the butterfly effect is quite visible, we have two attractors that begin at the same position as Figures 3, 4, 5, and 6. It orbits one of the attractors, skips over to the other, and bounces back and forth; This is where we get the concept of chaotic systems from, as this system is nonlinear and gives rise to mathematical chaos. If we have two simulations with remarkably similar initial conditions, their behavior will diverge quite a bit over time. To analyze the chaotic motion shown in Figure 7, we look at the Poincaré section. Instead of considering the phase space trajectory for all times, which results in a continuous curve, the Poincaré section is just the discrete collection of phase space locations of the particle at each period of the driving force, i.e., at $t = \pi/\omega, 2\pi/\omega, 3\pi/\omega, ..., n\pi/\omega$. The Poincaré section is a single point for a periodic orbit, but when the period doubles, it becomes two points, and so on.

Figure 8 shows a Poincaré section, a chaotic attractor of the Duffing oscillator for $\alpha = 1, \beta = 5, \delta = 0.02, \gamma = 12.34$ and $\omega = 0.5$.

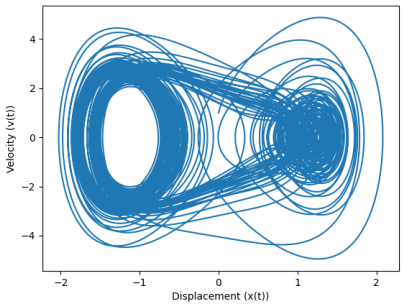


Figure 7: The plot of the Duffing oscillator exhibiting chaotic behaviour ($\alpha = 1$, $\beta = 5$, $\delta = 0.02$, $\gamma = 12.34$, and $\omega = 0.5$).

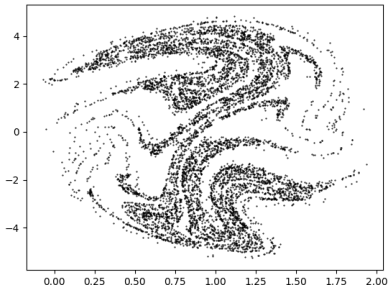


Figure 8: A Poincaré section of the forced Duffing equation, exhibiting chaotic behaviour ($\alpha = 1, \beta = 5, \delta = 0.02$, $\gamma = 12.34$ and $\omega = 0.5$).

A Duffing oscillator is an attractor in the sense that trajectories starting from different initial conditions converge to it, and it is strange that it is neither a continuous line in phase space, like a limit cycle, nor does it fill the x-v plane. Because the system is deterministic, the two simulations are pretty comparable for the first several steps. However, the 0.00001 difference soon becomes so large that it separates the long-term outcomes of the two simulation runs. Because chaotic systems are so sensitive to initial conditions as shown in Figure 9, it is nearly impossible for humans to predict their long-term behavior.

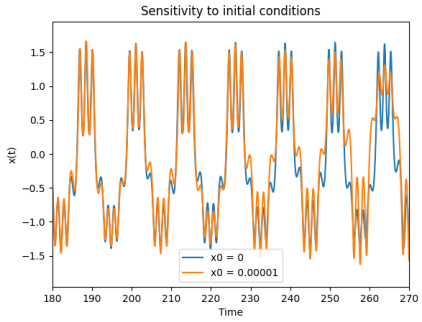


Figure 9: The plot shows the sensitivity of the non-linear dynamical system to the initial condition with the parameters of the Duffing equation set to $\alpha = 1$, $\beta = 5$, $\delta = 0,02$, $\gamma = 8$, and $\omega = 0.5$.

4 Linear Models

Given a set of independent variables and the associated dependent variables, a linear model estimates the parameters that describe how they are related to each other. Let us take a look at this linear equation

$$y = \beta_0 + \beta_1 x + \varepsilon \tag{20}$$

where y represents the dependent or response variable, and x represents the independent or predictor variable. The error term in the model is the random variable ε. In this context, "error" does not refer to a mistake but rather to a stochastic variable representing random fluctuations, measurement inaccuracies, or the influence of variables outside our control [25]. By (20), a linear model comparing the response y to multiple predictors has the general form

$$y_i = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_k x_k + \varepsilon, \tag{21}$$

where the parameters $\beta_0, \beta_1, \beta_2, \ldots, \beta_k$ are the regression coefficients. As in (20), ε accounts for random variation in y that is not explained by the x variables. Using matrix notation, a linear model such as (21) for each of n observations in a dataset may be written as

$$\mathbf{y} = \beta \mathbf{X} + \varepsilon \tag{22}$$

where

$$\mathbf{y} = \begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ \vdots \\ y_n \end{bmatrix}, \tag{23}$$

$$\beta = \begin{bmatrix} \beta_0 \\ \beta_1 \\ \vdots \\ \beta_k \end{bmatrix}, \tag{24}$$

$$\mathbf{X} = \begin{bmatrix} 1 & x_{11} & \ldots & x_{1k} \\ 1 & x_{21} & \ldots & x_{2k} \\ 1 & x_{31} & \ldots & x_{3k} \\ \vdots & \vdots & \vdots & \vdots \\ 1 & x_{n1} & \ldots & x_{nk} \end{bmatrix}, \tag{25}$$

and

$$\varepsilon = \begin{bmatrix} \varepsilon_1 \\ \varepsilon_2 \\ \varepsilon_3 \\ \vdots \\ \varepsilon_n \end{bmatrix}. \tag{26}$$

We assume the following additional assumptions to complete the model in (21):

1. $E(\varepsilon_i) = 0$ for all $i = 1, 2, ..., n$, or, equivalently, $E(y_i) = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_k x_k +$.
2. $\text{var}(\varepsilon_i) = \sigma^2$ for all $i = 1, 2, ..., n$, or, equivalently, $\text{var}(y_i) = \sigma^2$.
3. $\text{cov}(\varepsilon_i, \varepsilon_j) = 0$ for all $i \neq j$, or, equivalently, $\text{cov}(y_i, y_j) = 0$.
4. ε is $N_n(\mathbf{0}, \sigma^2\mathbf{I})$ or $\mathbf{y}$ is $N_n(\mathbf{X}\beta, \sigma^2\mathbf{I})$.

Assumption 1 asserts that the model (21) is valid, suggesting that y_i is only dependent on x_i and that all other variation in y_i is random. Assumption 2 states that the variance of ε or y is independent of x_i values. Assumption 2 is also known as the homoscedasticity assumption, homogeneous variance assumption, or constant variance assumption. The ε variables and the y variables are uncorrelated under assumption 3. $\sigma_{ij} = 0$ implies that the y (or ε) variables are both independent and uncorrelated under normality (assumption 4). The above three assumptions on ε_i or y_i can be represented in terms of the model in equation 22

1. $\mathbf{E}(\varepsilon) = \mathbf{0}$
2. $\text{cov}(\varepsilon) = \sigma^2\mathbf{I}$

Both the prior assumptions, $\text{var}(\varepsilon_i) = \sigma^2$ and $\text{cov}(\varepsilon_i, \varepsilon_j) = 0$, are included in assumption $\text{cov}(\varepsilon) = \sigma^2\mathbf{I}$.

The challenge we must address is how we should "fit" the line equation to a set of data. We want to determine the β_0 and β_1 values that form a line as "close" to the points as feasible. Suppose that for each data point (x_i, y_i) on line (20), a point $(x_i, \beta_0 + \beta_1 x_i)$ is the point on the line with the same x -coordinate. We call y_i the observed value of y, $\beta_0 + \beta_1 x_i$ the predicted y-value, and the difference between an observed y-value and a predicted y-value is a residual. To determine how "close" the line is to the data, we sum the squares of the residuals. It is worth noting that the residuals might be positive or negative; by squaring them, we ensure that our error estimates do not cancel out. Equation 20 is the least-squares line that minimizes the residual sum of squares. Regression coefficients are the coefficients of the line.

Of course, if the data points are not exactly on a line, there are no parameters β for which the predicted y-values in $\beta\mathbf{X}$ equal the observed y-values in $\mathbf{y}$, thus $\mathbf{y} = \beta\mathbf{X}$ has no solution. Now, because the data does not lie perfectly on a line, we have opted to seek the β that minimizes the sum of squared residuals of deviations of the n observed y's from their predicted values $\hat{y}$. we are looking for $\hat{\beta}_0$, $\hat{\beta}_1$, $\hat{\beta}_2$, ..., $\hat{\beta}_k$ that minimize

$$\sum_i^n \hat{\varepsilon_i}^2 = \sum_i^n (y_i - \hat{y}_i)^2 = \sum_i^n (y_i - \hat{\beta}_0 - \hat{\beta}_1 x_{i1} - \hat{\beta}_2 x_{i2} - \cdots - \hat{\beta}_k x_{ik})^2. \quad (27)$$

$E(y_i)$ is estimated from the predicted value $\hat{y}_i = \hat{\beta}_0 + \hat{\beta}_1 x_{i1} + \hat{\beta}_2 x_{i2} + \cdots + \hat{\beta}_k x_{ik}$. The four conditions must be met for a linear model to be considered a good model, and when all the assumptions are met, the least-squares estimators of β's show some good properties. If one or more assumptions fail, the estimators may be poor. With real data, any of the four assumptions might fail.

Let us consider the Whiteside dataset from the R package MASS as an example of real data. In the 1960s, Mr. Derek Whiteside of the UK Building Research Station documented the weekly gas use and average external temperature at his own house in South-East England for two heating seasons. The recordings are before and after the cavity-wall insulation was built. The exercise aimed to determine the impact of insulation on gas usage. The variables in the data frame are "Insul", a factor with levels "Before" and "After"; "Temp", for the weekly average external temperature in degrees Celsius; and "Gas", for the weekly gas usage in 1000 cubic feet. Figure 10 shows an inversely proportional relationship between average temperature and gas consumption before and after insulation. The insulation reduces the gas consumption for equal external temperatures and the rate at which gas consumption increases as external temperature falls. The linearity assumption holds for this data. We quantitatively investigated the other assumptions for linear models by using R's primary function *lm* to fit a linear model. We fitted both the before and after regression models in the same "lm" model and obtained the residual summary of the fitted model. The residual summary statistics provide information regarding the symmetry of the residual distribution. The residual summary of our fitted model is

```
> gasBA <- lm(Gas ~ Insul/Temp - 1, whiteside)
> summary(gasBA)

Call:
lm(formula = Gas ~ Insul/Temp - 1, data = whiteside)

Residuals:
     Min       1Q   Median       3Q      Max
-0.97802 -0.18011  0.03757  0.20930  0.63803

Coefficients:
                 Estimate Std. Error t value Pr(>|t|)
InsulBefore       6.85383    0.13596   50.41   <2e-16 ***
InsulAfter        4.72385    0.11810   40.00   <2e-16 ***
InsulBefore:Temp -0.39324    0.02249  -17.49   <2e-16 ***
InsulAfter:Temp  -0.27793    0.02292  -12.12   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.323 on 52 degrees of freedom
Multiple R-squared:  0.9946,Adjusted R-squared:  0.9942
```

```
F-statistic:  2391 on 4 and 52 DF,  p-value: < 2.2e-16
```

The median of our fitted model is close to zero, indicating a symmetric residual distribution (median=mean). Furthermore, the 3Q and 1Q have similar magnitudes. A symmetric zero-mean distribution requires them to be equal in the limit of infinite sample size. The maximum and minimum values are almost comparable in magnitude, indicating no outliers. The estimates are statistically significant since the p-value < 0.0125. Temperature measurements can forecast gas consumption values (Multiple R-squared is close to 1).

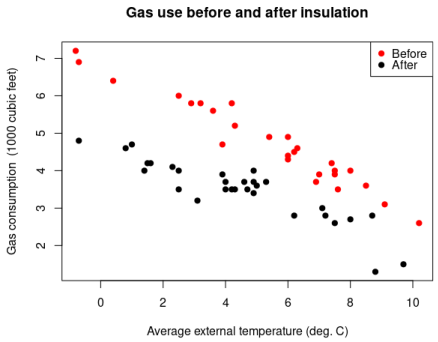


Figure 10: Scatter plot of Whiteside's data demonstrating the influence of insulation on household gas consumption.

Figures 11 and 12 help us identify whether our fitted model fulfills the homoscedasticity assumption. On this assumption, each variable in a set of random variables has the same finite variance. The residuals seem to bounce randomly and somewhat form a horizontal band around the 0 line, suggesting that the assumption that the linear relationship is reasonable and that the variances of the error terms are equal. In addition, the residuals do not appear to follow a clear pattern in Figure 12, indicating that our model does not visibly violate the homoscedasticity assumption. We also check Cook's distance which is used to identify data points that may have a negative impact on our regression model. It calculates how much the model's fitted values change when one data point is removed. Even though observations 36, 46, and 55 have the highest standardized residuals, it is seen in Figure 13 that they are not outside the Cook's distance (red dashed line) and therefore are not influential. In Figure 14, residuals are approximately lying on the straight dashed line, suggesting that they are normally distributed. The model we fitted on the Whiteside dataset met all the assumptions of linear models; therefore, the fitted model is a good approximation of the Whiteside data.

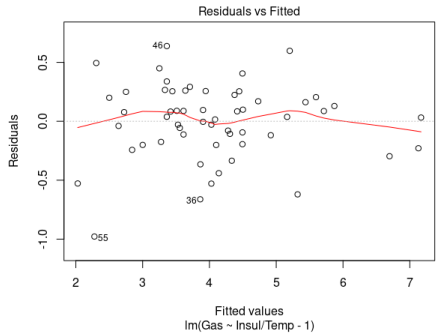


Figure 11: Residuals versus Fitted values plot.

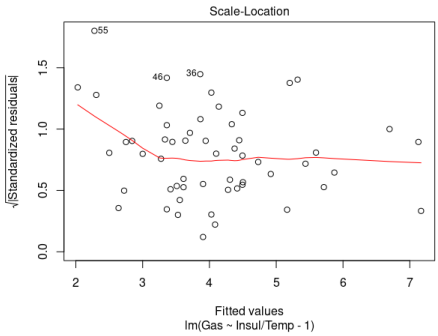


Figure 12: The magnitude of the standardized residuals' variability as a function of the fitted values.

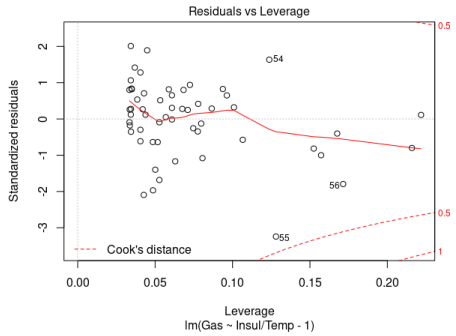


Figure 13: Cook's distance plot. There are no points outside of the red dashed line indicating zero influential points.

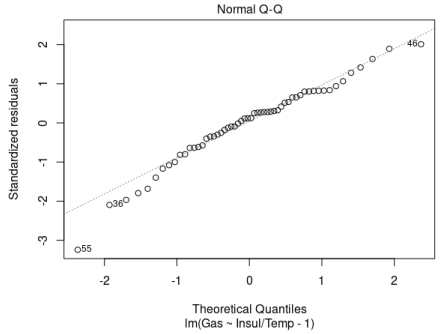


Figure 14: The Normal Q-Q plot for gas consumption of both before and after insulation. It visually checks whether the assumption of normally distributed residuals is plausible. The points form a roughly straight line, implying that the residuals are normally distributed.

Let us now look at a slightly different example where we fit a multiple regression model to the Scottish hills data from the MASS R package. The hills data frame contains Scottish hill race record timings where the columns show the overall race distance, total height climbed, and the record time as shown in Table 1.

Table 1: Record time for Scottish Hill races.

Observation Number	Name of race	x_1:distance	x_2:climb	y: record time
1	Greenmantle	2.5	650	16.083
2	Carnethy	6.0	2500	48.350
3	Craig Dunain	6.0	900	33.650
4	Ben Rha	7.5	800	45.600
5	Ben Lomond	8.0	3070	62.267
6	Goatfell	8.0	2866	73.217
7	Bens of Jura	16.0	7500	204.617
8	Cairnpapple	6.0	800	36.367
9	Scolty	5.0	800	29.750
10	Traprain	6.0	650	39.750
11	Lairig Ghru	28.0	2100	192.667
12	Dollar	5.0	2000	43.050
13	Lomonds	9.5	2200	65.000
14	Cairn Table	6.0	500	44.133
15	Eildon Two	4.5	1500	26.933
16	Cairngorm	10.0	3000	72.250
17	Seven Hills	14.0	2200	98.417
18	Knock Hill	3.0	350	78.650
19	Black Hill	4.5	1000	17.417
20	Creag Beag	5.5	600	32.567
21	Kildcon Hill	3.0	300	15.950
22	Meall Ant-Suidhe	3.5	1500	27.900
23	Half Ben Nevis	6.0	2200	47.633
24	Cow Hill	2.0	900	17.933
25	N Berwick Law	3.0	600	18.683
26	Creag Dubh	4.0	2000	26.217
27	Burnswark	6.0	800	34.433
28	Largo Law	5.0	950	28.567
29	Criffel	6.5	1750	50.500
30	Acmony	5.0	500	20.950
31	Ben Nevis	10.0	4400	85.583
32	Knockfarrel	6.0	600	32.383
33	Two Breweries	18.0	5200	170.250
34	Cockleroi	4.5	850	28.100
35	Moffat Chase	20.0	5000	159.833

We fitted a multiple regression model (hills.lm) of time on dist and climb and looked at the summary of our model below.

```
> summary(hills.lm)
Call:
lm(formula = time ~ dist + climb, data = hills)

Residuals:
    Min      1Q  Median      3Q     Max
-16.215  -7.129  -1.186   2.371  65.121

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) -8.992039   4.302734  -2.090   0.0447 *
dist         6.217956   0.601148  10.343 9.86e-12 ***
climb        0.011048   0.002051   5.387 6.45e-06 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 14.68 on 32 degrees of freedom
Multiple R-squared:  0.9191,Adjusted R-squared:  0.914
F-statistic: 181.7 on 2 and 32 DF,  p-value: < 2.2e-16
```

The relationships between dist and climb are statistically significant since their p-values < 0.05, and the adjusted R^2 value indicates that the model provides a good fit for the data. However, high R^2 does not mean that our fitted model meets the model assumptions. The magnitudes of 1Q and 3Q and Min and Max are not close to each other, indicating outliers in our dataset. The median is also not close to zero, suggesting that the residual distribution is not symmetric. Figures 15, 16, 17, 18, and 19 help us verify if the model is adequate and fits the analysis's assumptions. Instead of residuals, we used studentized residuals in Figure 15 because they identify outliers. From Figure 15 we can see that about two observations have a studentized residual with an absolute value larger than three, indicating two apparent outliers in our model. Figure 19 shows that the data points are not randomly distributed around zero and that the two outliers are Knock hill and Bens of Jura. Moreover, one of the two, Bens of Jura, proves to be an influential observation, as seen in Figure 16. The normal Q-Q plot in Figure 18 shows that most points typically follow a straight line, indicating that most residuals are normally distributed despite outliers.

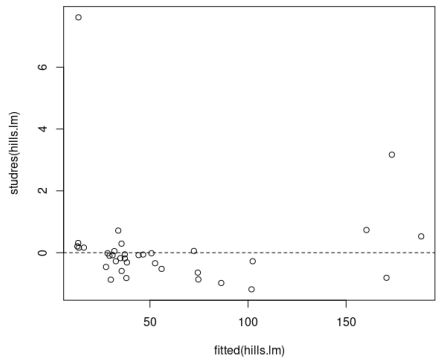
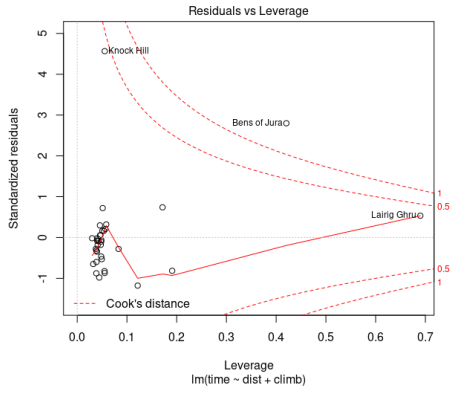


Figure 15: Plot identifying outliers. The two 2 points with studentized residuals greater than 3 are outliers.



gt

Figure 16: Diagnostic plot identifying influential observations or points. Bens of Jura is outside of the Cook's distance (red dashed line) which suggest that is it an influential observation.

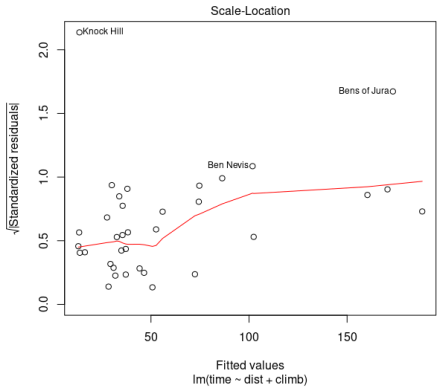


Figure 17: Diagnostic plot for Scottish data, unweighted model.

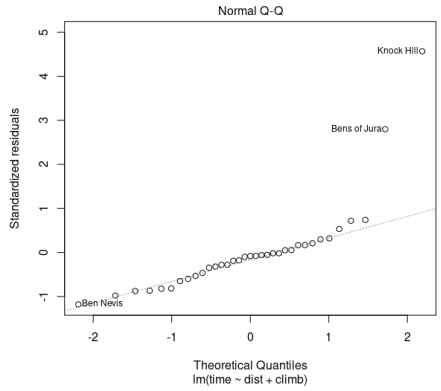


Figure 18: The Normal Q-Q plot for Scottish data.

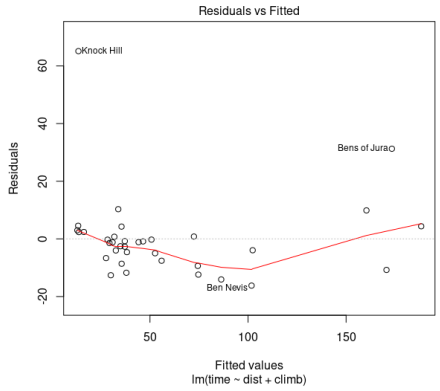


Figure 19: The plot Residuals versus Fitted values for the Scottish data.

Since the model's assumptions are not met, we updated our model by first deleting Knock hill and then deleting both Knock hill and Bens of Jura from our dataset, respectively, to see how it affected our data. As seen from the residual summary below, removing Knock hill observation from our data did not significantly affect our fitted model.

```
Call:
lm(formula = time ~ dist + climb, data = hills, subset = -18)

Coefficients:
(Intercept)         dist        climb
  -13.53035      6.36456      0.01185
```

Knock hill did not have high leverage because our fitted model did not change that much. However, when we removed Knock hill and Bens of Jura from our data, the change in the coefficients was quite significant (see residual summary below). Bens of Jura has both high leverage and a large residual. Hence it affected our model.

```
Call:
lm(formula = time ~ dist + climb, data = hills, subset = -c(7,18))

Coefficients:
(Intercept)         dist        climb
 -10.361646     6.692114     0.008047
```

There are issues with our fitted model, so we fitted a weighted model to help us deal with the heteroscedasticity. Each data point is given a weight depending on the variance of its fitted value in the weighted regression. We weight the fit by using distance as a proxy for time, with the weights being inversely proportional

to the variance. We omitted observation number 18 corresponding to Knock Hill because we suspect there was a mistake; the time in Table 1 is significantly too long for the provided distance and climb, and it may have been misreported by an hour. The summary of the weigthed model is

```
Call:
lm(formula = time ~ dist + climb, data = hills, subset = -18,
    weights = 1/dist^2)

Weighted Residuals:
     Min       1Q   Median       3Q      Max
-2.66603 -0.58817  0.00677  0.53522  3.09838

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) -5.808538   2.034248  -2.855   0.0076 **
dist         5.820760   0.536084  10.858 4.33e-12 ***
climb        0.009029   0.001537   5.873 1.76e-06 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 1.156 on 31 degrees of freedom
Multiple R-squared:  0.9249,Adjusted R-squared:  0.9201
F-statistic: 190.9 on 2 and 31 DF,  p-value: < 2.2e-16
```

We wanted to fit a model with a zero intercept on the physical ground. However, the intercept is still non-zero. The prediction equation was then divided by distance, and inverse speed (time/distance) was regressed on gradient (climb/distance):

```
Call:
lm(formula = hills$ispeed ~ hills$grad, data = hills[-18, ])

Coefficients:
(Intercept)   hills$grad
   6.563132     0.004057
```

Figures 20 and 21 are the diagnostic plots for the weighted model. The observations in the Normal Q-Q plot roughly follow a straight line, which leads us to assume that the residuals are normally distributed. Also, in Figure 20, most residuals are not far from the zero line.

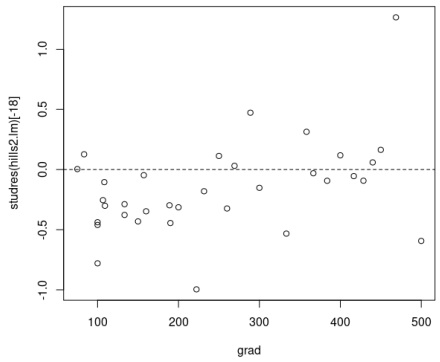


Figure 20: Diagnostic plot of the weighted model.

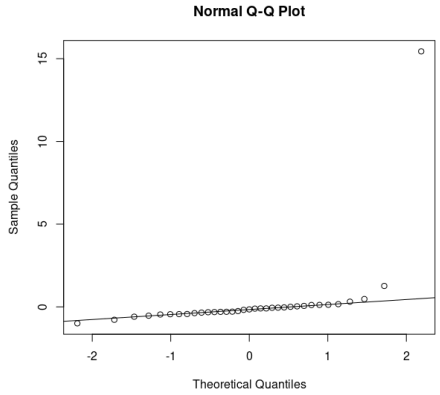


Figure 21: The Normal Q-Q plot for the weighted model.

5 Markov Chain Monte Carlo (MCMC)

Suppose we have a sequence of dependent random variables, $\{X_0, X_1, X_2, ...\}$, such that when $t \geq 0$, the next state X_{t+1} is sampled from a probability distribution $P(X_{t+1}|X_t)$ that depends solely on the chain's current state, X_t. Given X_t, the following state X_{t+1} is independent of the chain's history $\{X_0, X_1, ..., X_{t-1}\}$. This sequence is a Markov chain, and $X_{t+1}|X_0, X_1, X_2, ..., X_t \sim K(X_t, K_{t+1})$ is the chain's transition kernel K. We shall assume that the chain is time-homogeneous, which means that $K(X_t, K_{t+1})$ is independent of t. A simple random walk Markov chain, for example, fulfills

$$X_{t+1} = X_t + \epsilon_t, \tag{28}$$

where $\epsilon_t \sim N(0, 1)$ independent of X_t; hence, the Markov kernel $P(\cdot|\cdot)$ corresponds to a $N(X_t, 1)$ density. There exists a stationary probability distribution f such that if $X_t \sim f$, then $X_{t+1} \sim f$. As a result, the transition kernel and stationary distribution fulfill the equation

$$\int_{\chi} K(x, y)f(x)dx = f(y). \tag{29}$$

The presence of a stationary distribution imposes a primary constraint on K known as irreducibility in Markov chain theory; which states that the kernel K allows for free moves all over the state space, namely that the sequence X_t has a positive probability of eventually reaching any region of the state space regardless of the starting value X_0. A stationary distribution has significant implications for chain X_t behavior. One implication is that most of the chains used in MCMC algorithms are recurrent, meaning they will return to any given non-trivial set an infinite number. The stationary distribution is likewise a limiting distribution in the case of recurrent chains since the limiting distribution of X_t is f for almost any initial value X_0. This phenomenon is also known as ergodicity, and it has clear simulation implications in that if a given kernel K generates an ergodic Markov chain with stationary distribution f. Constructing a chain from this kernel K would gradually produce simulations from f. The standard average, in particular, for integrable functions h

$$\frac{1}{T}\sum_{t=1}^{T} h(X_t) \longrightarrow E_f[h(X)], \tag{30}$$

which means that the Law of Large Numbers (which states that as a sample size increases, the sample mean approximates the expected value) is applicable in MCMC settings. Equation (30) is thus known as the Ergodic Theorem. The working idea of Markov chain Monte Carlo algorithms is simple to explain. Given a target density f, we construct a Markov kernel K with a stationary distribution f. We then use this kernel to produce a Markov chain X_t with the limiting distribution f and integrals that can be estimated using the Ergodic Theorem (30). The challenge should thus be in constructing a kernel K that is

connected with any density f. However, there are universal methods for deriving such kernels because they are theoretically true for any density f! One of these algorithms is the Metropolis-Hastings algorithm.

The Metropolis-Hastings algorithm chooses the next state X_{t+1} at each time t by first sampling a candidate point Y from a proposal distribution $q(\cdot|X_t)$. The proposal distribution may be affected by the current point X_t. The candidate point Y is then accepted with probability $\alpha(X_t, Y)$ where

$$\alpha(X, Y) = \min(1, \frac{\pi(Y)q(X|Y)}{\pi(X)q(Y|X)}) \tag{31}$$

The next state is $X_{t+1} = Y$ if the candidate point is accepted. If the candidate is rejected, the chain does not move, i.e., $X_{t+1} = X_t$. To run the Metropolis-Hastings algorithm on a computer, we need to run the proposal chain $q(\cdot|X)$, followed by the accept/reject step. As a result, running the algorithm is quite feasible. However, this algorithm raises new questions. Most notably, how do we choose the proposal distributions $q(\cdot|X)$? Concerning the first question, there are several kinds of methods for choosing proposal distributions, such as:

1. Symmetric Metropolis Algorithm : Here, $q(X|Y) = q(Y|X)$, and the acceptance probability is simplified to $\alpha(X, Y) = \min(1, \frac{\pi(Y)}{\pi(X)})$.

2. Random walk Metropolis-Hastings: Here, $q(X|Y) = q(|Y - X)|$. For example, $q(X|\cdot) = N(X, \sigma^2)$, or $q(X|\cdot) = \text{Uniform}(X - 1, X + 1)$.

3. Independence sampler: Here, $q(X|Y) = q(Y)$, implying that $q(X|\cdot)$ is independent of X.

Let us generate samples of standard normal distribution, for example. We do not know how to sample from the target distribution $N(0, 1)$ but do know how to sample from the proposed distribution $N(\mu, 1)$. We run a random walk Metropolis-Hastings algorithm on a computer. The R code we wrote to run the random walk Metropolis-Hastings algorithm in the preceding example is as follows:

```
target <- function(x){dnorm(x,mean = 0, sd = 1)}
proposed <- function(x){rnorm(1,mean = x, sd=1)}
stored_chain <- rep(NA,10000)
x_current<- -20
for(i in 1:length(stored_chain)){
  x_proposed <- proposed(x_current)
  ratio <- min(1,target(x_proposed)/target(x_current))
  accept <- runif(1) < ratio
  stored_chain[i] <- ifelse(accept,x_proposed,x_current)
  x_current <- stored_chain[i]
}
plot(stored_chain,type = "l",col = "blue")
```

Figure 22 shows that the starting value is much beyond the range of typical x_0 values. The algorithm has a burn-in period. It takes roughly 40-time steps for the chain to reach a typical state. We must discard the burn-in samples. The following evolution plot (Figure 23) illustrates the random walk Metropolis-Hastings algorithm with $x_0 = 0$ and $\sigma = 1$ and no burn-in samples. The algorithm performs well with this starting condition and proposal distribution. The samples are correlated, yet the Markov chain mixes well.

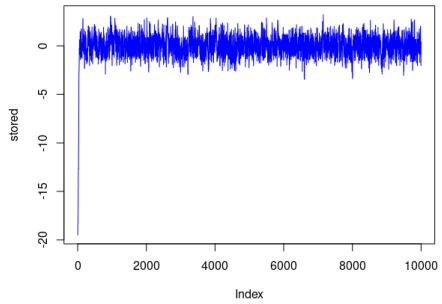


Figure 22: The random walk Metropolis-Hastings for the normal distribution with $\sigma = 1$ and the burn in period.

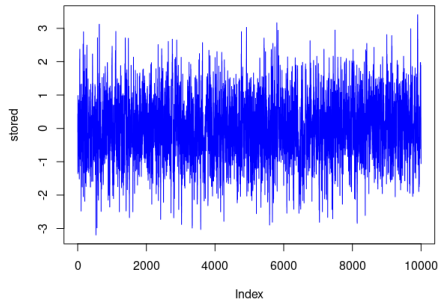


Figure 23: The random walk Metropolis-Hastings for the normal distribution with $\sigma = 1$ and starting point $x_0 = 0$.

When implementing MCMC, several issues arise, the most pressing of which is the choice of proposal distribution. The question is, how do we choose a proposal distribution or what happens when the step size is either too small or too large? The algorithm does not mix well with the given proposal distribution

when the step size is too small. The state moves slowly through state space, and there is a strong correlation between samples over lengthy periods. It also results in a very high acceptance rate since very few proposals are noticeably poorer than the ones they compete with. We would need a considerable number of iterations for the chain to converge and approximate the posterior distribution given the setup shown in the evolution plot (Figure 24) for the random walk Metropolis-Hastings algorithm with $\sigma = 0.1$. On the other hand, The large proposal distribution generates many proposals with very low or zero posterior probability, resulting in a very high rejection rate. As a result, the chain remains in its current state for many steps. This is apparent as flat lines in the evolution plot (Figure 25) for the random walk Metropolis-Hastings algorithm with $\sigma = 100$. In practice, more draws will be required to provide an acceptable approximation of the posterior distribution.

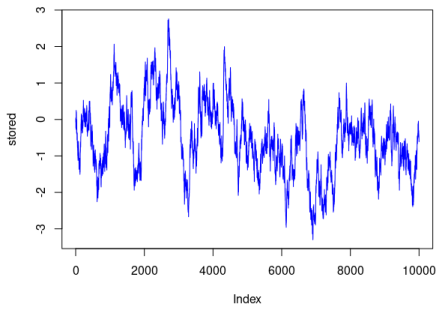


Figure 24: The random walk Metropolis-Hastings for the normal distribution with $\sigma = 0.1$ and starting point $x_0 = 0$.

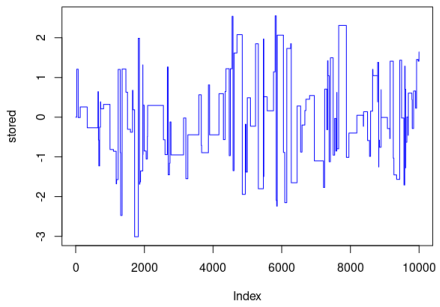


Figure 25: The random walk Metropolis-Hastings for the normal distribution with $\sigma = 100$ and starting point $x_0 = 0$

A proposal distribution that is too narrow or too wide (i.e., has a smaller or larger standard deviation (σ)) results in higher autocovariance and slower convergence. This phenomena is illustrated in Figure 26. Ideally, the proposal distribution must be scaled such that both extremes are avoided. The above example shows that tuning the random walk's scale σ is crucial for attaining a good approximation to the target distribution in a reasonable number of iterations.

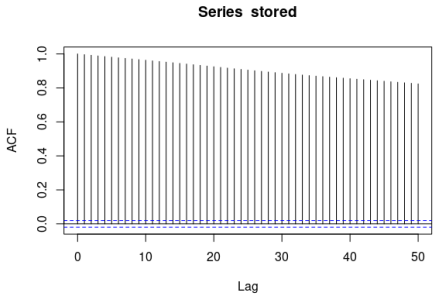


Figure 26: The autocorrelation plot from random walk Metropolis-Hastings with a stationary distribution $N(0, 1)$ and proposal distribution $N(X_t, 0.1)$.

A proposal distribution that can be easily sampled and evaluated should be chosen for computing efficiency. Tuning the proposal distribution is not always easy and in some cases the random walk Metropolis-Hastings algorithm is not efficient. To evaluate the chains' efficiency, we compared chains drawn at random and independently from the target distribution. When the chains are sampled independently, the proposal distribution is independent of the present

state of the chain. In Figure 27, when the proposal distribution q is $N(X_t, 10)$ and chains are randomly drawn, many flat lines on the simulation plot indicate a higher rejection rate than in Figure 28. The independently drawn chains (with the proposal distribution $N(0, 10)$) mix well, and the sample converges to the target distribution considerably faster than when the chains are drawn randomly as seen in Figures 29 and 30.

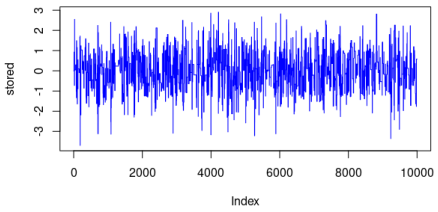


Figure 27: The random walk Metropolis-Hastings for the normal distribution with $\sigma = 10$ and starting point $x_0 = 0$

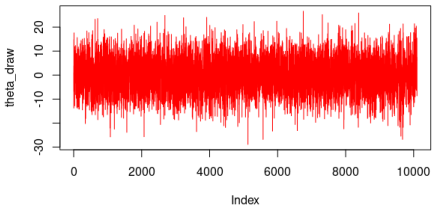


Figure 28: The Independent Sampler for the normal distribution with zero mean and the standard deviation of $\sigma = 10$.

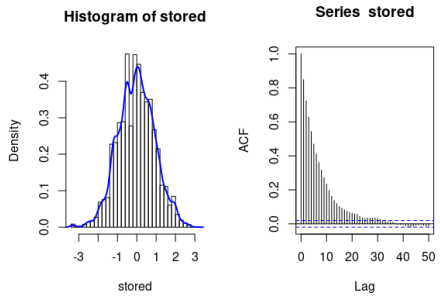


Figure 29: Histogram and autocovariance function from a random walk Metropolis-Hastings with the proposal distribution $N(X_t, 10)$.

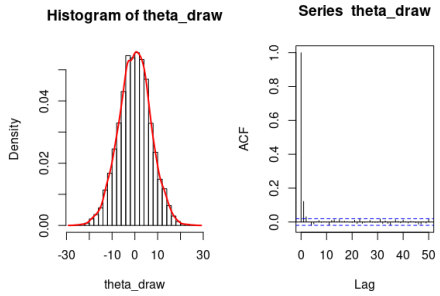


Figure 30: Histogram and autocovariance function from an Independent Sampler with the proposal distribution $N(0, 10)$.

The independence sampler works well when the proposal distribution is an excellent approximation to the target distribution. Still, it is most safe when the proposal distribution is heavier-tailed than the target distribution.

Both the Metropolis-Hastings algorithms and the Monte Carlo Accept–Reject algorithm, which are general simulation methods, possess commonalities that allow them to be compared. As a result, the Monte Carlo Accept–Reject method is implemented for sampling one-dimensional probabilistic models. The MCMC algorithms are most valuable in higher dimensions because, for high-dimensional probabilistic models, Monte Carlo sampling is inefficient and may be difficult. In contrast, Markov Chain Monte Carlo gives an alternative approach to random sampling, a high-dimensional probability distribution where the following sample is dependent on the current sample.

We looked at the MCMC sampling for one-dimensional probabilistic models in the above examples, but MCMC methods are most useful when sampling

from high-dimensional probability distributions. Let us now sample from our two-dimensional posterior distribution using the Independent sampler. Figures 31 and 32 show the trace plot of x and y points (first and second chain), respectively, when the starting point is (0,0). We can see from Figure 32 that zero is not the best starting point for the chain. It takes time for the chain to mix. So we changed the starting point of the Y chain to 0.8, as shown in Figure 33. Both X and Y chains seem to mix very slowly, the parameters are strongly correlated, and the autocorrelation between states of Markov chains is high. Ideally, we want it to drop quickly to zero to show that the chain has converged to a posterior distribution. However, as shown in Figure 34, this is not the case. We ran the chains for longer iterations, as shown in Figure 35, and the chains seem to "mix well," but that is not enough to conclude if the chains have converged to the posterior distribution.

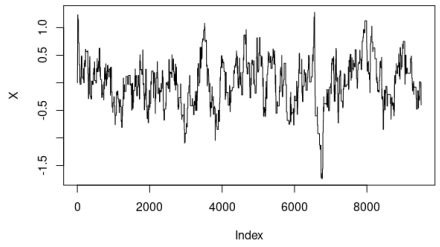


Figure 31: Trace plot of X (the first MCMC chain) with $\sigma = 0.2$ and the starting point of $X_0 = 0$.

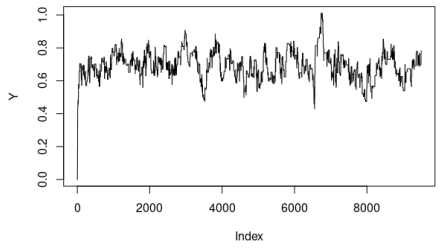


Figure 32: Trace plot of Y (the second MCMC chain) with $\sigma = 0.2$ and starting point of $Y_0 = 0$.

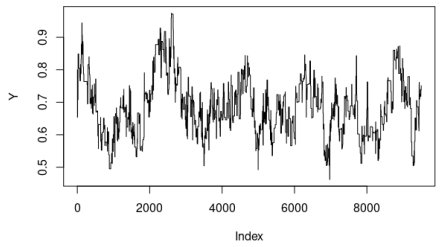


Figure 33: Trace plot of Y (the second MCMC chain) with $\sigma = 0.2$ and starting point of $Y_0 = 0.8$.

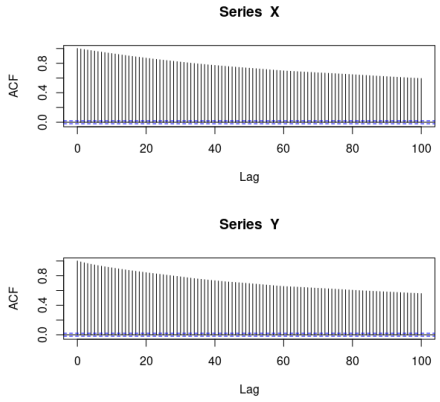


Figure 34: Autocorrelation plot of X and Y MCMC chains

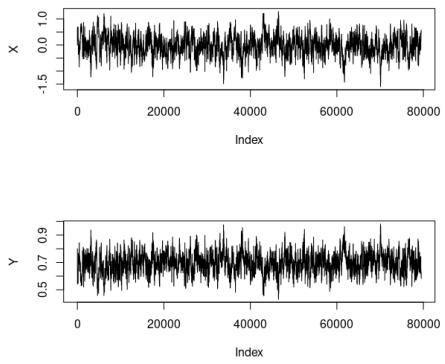


Figure 35: Trace plot of X and Y MCMC chains with $\sigma = 0.2$ and starting point of $(0, 0.8)$ for longer iterations.

There are formal but not definitive statistical tests for convergence to assess convergence. The Gelman-Rubin statistic assesses parallel chains with dispersed initial values to test whether they converge to the same target distribution. To define the method, we shall consider a single summary at a time and label it ψ. We shall assume m parallel simulations, each of length n. For each scalar summary of interest, we would like a numerical equivalent of the comparison in Figure 35 that states: The two sequences are much farther apart than we could expect, based on their internal variability. For each scalar summary ψ, we label the m parallel sequences of length n as $(\psi_{ij}), j = 1, ..., n; i = 1, ..., m$, and we compute two quantities, the between-sequence variance B and the within-sequence variances W:

$$B = \frac{n}{m-1} \sum_{i=1}^{m} (\bar{\psi_i} - \bar{\psi_{..}})^2 \tag{32}$$

where

$$\bar{\psi_i} = \frac{1}{n} \sum_{j=1}^{n} \psi_{ij}, \tag{33}$$

$$\bar{\psi_{..}} = \frac{1}{m} \sum_{i=1}^{m} \bar{\psi_i}. \tag{34}$$

$$W = \frac{1}{m} \sum_{i=1}^{m} s_i^2 \tag{35}$$

where,

$$s_i^2 = \frac{1}{(n-1)} \sum_{j=1}^{n} (\psi_{ij} - \bar{\psi_i})^2. \tag{36}$$

The between-sequence variance B contains a factor of n because it is based on the variance of the within-sequence means, $\bar{\psi}$, each of which is an average of n values ψ_{ij}. We then construct two estimates of the variance of ψ in the target distribution from the two variance components. First,

$$\hat{Var}(\psi) = \left(1 - \frac{1}{n}\right) W + \frac{B}{n} \tag{37}$$

would be an unbiased estimate of the variance under stationarity if the starting points of the simulations were drawn from the target distribution. However, under the more realistic assumption, it is an overestimate that the starting points are overdispersed. $\hat{Var}(\psi)$ is a "conservative" estimate of the variation of ψ under overdispersion. We can now track the Markov chain's convergence by estimating the factor by which the conservative estimate of the ψ distribution might be reduced: the ratio between the estimated upper and lower bounds for the ψ standard deviation, which we call the "estimated potential scale reduction factor (PSRF),"

$$\sqrt{\hat{R}} = \sqrt{\frac{\hat{Var}(\psi)}{W}}. \tag{38}$$

If $\hat{R}$ is large, it implies that further simulations may further reduce either the variance estimate or that more simulation would increase W because the simulated sequences have not yet completed a full tour of the target distribution. If, on the other hand, the PSRF is close to one, we may conclude that each of the m sets of n simulated observations is close to the target distribution. As previously stated, the chains in Figure 35 appeared to mix well. We used the Gelman-Rubin statistic to assess if our X MCMC chain had converged to the posterior distribution after 80000 iterations. We divided our X sample into $m = 4$ sequences of length $2n = 20000$ each and calculated the Gelman-Rubin $\hat{R}$. We obtained $\hat{R} = 1.02569 < 1.1$, indicating that our X MCMC chain has converged to the posterior distribution.

6 Estimating Reinfection Risk with the Emergence of Beta, Delta, and Omicron Variants

The Omicron (B.1.1.529 / 21K) variant was discovered in Gauteng Province in late November 2021 and connected with rapidly increasing case numbers [8]. Although prior VOCs emerged in a world where natural immunity to COVID-19 infections was prevalent, this VOC (Omicron variant) originated when global vaccination immunity was rising [33]. In this study we will be attempting to determine whether the Omicron variant reinfects at a higher rate than other VOCs using the paper by Pulliam et al. as the basis for this study. SARS-CoV-2 infection-induced immunity has been shown in studies to last at least 5-6 months after infection, with some small case studies indicating that repeat infections could occur as soon as 1-3 months after the initial infection[34]. Following the methodology of Pulliam et al., we shall consider individuals with consecutive positive tests at least 90 days apart to have been reinfected. When the interval between positive tests is shorter, a continuation of the initial infection rather than a reinfection cannot be excluded.

Let us assume that for a case that tests positive on day t (by specimen receipt date), the reinfection hazard is 0 for days $t_i + 1$ to $t_i + 90$, and the reinfection hazard is $\lambda \hat{C}_t$ for days with $t > t_i + 90$, where $\hat{C}_t$ is the 7-day moving average of the detected case incidence (initial infections and reinfections) that day [8]. Let

$$\bar{k_t} = \lambda \hat{C_t} e^{-\sum_{t'=90}^{t_i-1} \lambda \hat{C_{t'}}}. \tag{39}$$

It follows that the probability of a case testing positive on day t_i having a diagnosed reinfection by day t is

$$p(k_t, \bar{k_t}) = \frac{e^{-\bar{k_t}} \bar{k_t}^{k_t}}{k_t!}. \tag{40}$$

The probability mass function in equation 40 for the Poisson distribution gives us the likelihood function

$$p(k_{t=91}, \ldots, k_{t_{end}} | \lambda, \hat{C_t}) = \prod_{t=91}^{t_{end}} p(k_t; \bar{k_t}) = \prod_{t=91}^{t_{end}} \frac{e^{-\bar{k_t}} \bar{k_t}^{k_t}}{k_t!}. \tag{41}$$

We further expand equation 41 to

$$p(k_{t=91}, \ldots, k_{t_{end}} | \lambda, \hat{C_t}) = \frac{\prod_{t=91}^{t_{end}} e^{-\bar{k_t}} \prod_{t=91}^{t_{end}} \bar{k_t}^{k_t}}{\prod_{t=91}^{t_{end}} k_t!}. \tag{42}$$

When there are many data points, it is usually easier to rewrite this likelihood as the log of the likelihood with no loss of generality because maximizing a log-likelihood is the same as maximizing a likelihood. The products are not numerically stable; they quickly result in underflow given the limitations of the float or double type. The logarithm transforms a product of densities into a

sum and exponents to products. Then the log-likelihood function of equation 42 is

$$\mathcal{L}(p) = \log p(k_{t=91}, \ldots, k_{t_{end}} | \lambda, \hat{C}_t) = \log \left(\prod_{t=91}^{t_{end}} \frac{e^{-\bar{k_t}} \bar{k_t}^{k_t}}{k_t!} \right) \tag{43}$$

$$= \sum_{t=91}^{t_{end}} k_t \log \bar{k_t} - \sum_{t=91}^{t_{end}} \bar{k_t} - \log \left(\prod_{t=91}^{t_{end}} k_t! \right). \tag{44}$$

We began this investigation by attempting to determine whether the Omicron variant reinfects at a higher rate than other VOCs by developing a model and applying it to the data of individuals with SARS-CoV-2 who received a positive test result at least 90 days before November 27, 2021, previously analyzed by Pulliam et al. However, later in our research, we discovered that Pulliam et al. ([35]) publicized only some of the data they used. They provided some of the data required to reproduce their analysis, but the crucial data regarding the infection and reinfection dates are missing, supposedly due to privacy concerns. As a result, we used the available data to generate mock data for reinfections. The data we used is a count data with three columns; the first column (date) contains the specimen receipt dates from March 4, 2020, to November 27, 2022. The second column (inc_1) contains the national daily time series of newly detected putative primary infections. The third column (inc_2) contains suspected reinfections. The code used simulate the mock data is in Appendix B.

Suppose that for a person who tests positive for COVID-19 on day t, the risk of reinfection is zero each subsequent day from that t to day 90 after testing positive. We assume that the risk of reinfection is zero because a person may have multiple positive tests from the same infection. Based on our data, we can't tell the difference between that and reinfection after a short period. Then the probability of a person testing positive on day t and getting reinfected on day x is

$$p_x = \left(e^{-\int_{t+90}^{x-1} dt \lambda I(t)} - e^{-\int_{t+90}^{x} dt \lambda I(t)} \right) \tag{45}$$

where $I(t)$ is the smoothed daily number of cases and π accounts for the possibility of a person becoming infected again without being tested or testing and not reporting it. Suppose N people tested positive on day t. The probability that exactly k_x of those people test positive on day x is given by the binomial distribution

$$P(k_x) = \frac{N!}{k_x!(N-k_x)!} p_x^{k_x} \ (1-p_x)^{N-k_x}. \tag{46}$$

The expectation value for k_x is Np_x. It follows that given the observed counts $k_{t+90}, \ldots, k_{t_{end}}$, the likelihood for the theory with probabilities $p_{t+90}, \ldots, p_{t_{end}}$

is given by

$$L(k_{t+90}, k_{t+91}, \ldots, k_{t_{end}} | p_{t+90}, p_{t+91}, \ldots, p_{t_{end}})$$
$$= \frac{N!}{(N - \sum_{t'=t+90}^{t_{end}} k_{t'})!} \left(\prod_{t'=t+90}^{t_{end}} \frac{p_{t'}^{k_{t'}}}{k_{t'}!} \right) \left(1 - \sum_{t'=t+90}^{t_{end}} p_{t'} \right)^{N - \sum_{t'=t+90}^{t_{end}} k_{t'}} \quad (47)$$

where t_{end} is the last available data point. So because multinomial distribution belongs to the exponential family, we know that evaluating the log-likelihood will give us a more easy expression. The log-likelihood is given by

$$\log L(k_{t+90}, k_{t+91}, \ldots, k_{t_{end}} | p_{t+90}, p_{t+91}, \ldots, p_{t_{end}})$$
$$= \log \left(\frac{N!}{(N - \sum_{t'=t+90}^{t_{end}} k_{t'})!} \left(\prod_{t'=t+90}^{t_{end}} \frac{p_{t'}^{k_{t'}}}{k_{t'}!} \right) \left(1 - \sum_{t'=t+90}^{t_{end}} p_{t'} \right)^{N - \sum_{t'=t+90}^{t_{end}} k_{t'}} \right). \quad (48)$$

We further simplify equation 48 and get the log-likelihood

$$\mathcal{L}(k_{t+90}, k_{t+91}, \ldots, k_{t_{end}} | p_{t+90}, p_{t+91}, \ldots, p_{t_{end}}) = \log N! - \log \left(N - \sum_{t'=t+90}^{t_{end}} k_{t'} \right)!$$
$$+ \sum_{t'=t+90}^{t_{end}} k_{t'} \log p_{t'} - \sum_{t'=t+90}^{t_{end}} \log k_{t'}! + \left(N - \sum_{t'=t+90}^{t_{end}} k_{t'} \right) \log \left(1 - \sum_{t'=t+90}^{t_{end}} p_{t'} \right). \quad (49)$$

Figure 36 shows much fluctuation between day-to-day cases (black dots) and a few cases reported on weekends, which may have been caused by laboratories not being open on certain days for people to report their cases. And then, the next day, they may have received a mix of reports from the previous day and the current day. So, to produce something smoother, we assume $\bar{X}_i$ is the 7-day moving average of the number of primary infections reported each day. Let $S_r[i, j]$ be the number of people who tested positive on day i who are susceptible to reinfection on day j. The initial condition for this variable is

$$S_r[i, i] = \bar{X}_i. \tag{50}$$

For $j > i$, the update rule is

$$S_r[i, j] = S_r[i, j-1] - R_{nd}[i, j], \tag{51}$$

where

$$R_{nd}[i, j] = \alpha S_r[i, j]\bar{X}_j \tag{52}$$

is the number of new daily reinfections for those having previously tested positve on day i and α the rate of reinfection. We for the moment assume that α is constant across all strains and does not change over time. The new daily detected reinfections is then given by

$$R_{dd}[i, j] = \beta R_{nd}[i, j] \tag{53}$$

where β is the rate of new daily detected reinfections. Both α and β are the model parameters and are unknown. We use the reinfection data simulated by the code in Appendix B to program the log-likelihood in Appendix C in an MCMC, Metropolis Hastings algorithm in order to estimate the model parameters α and β.

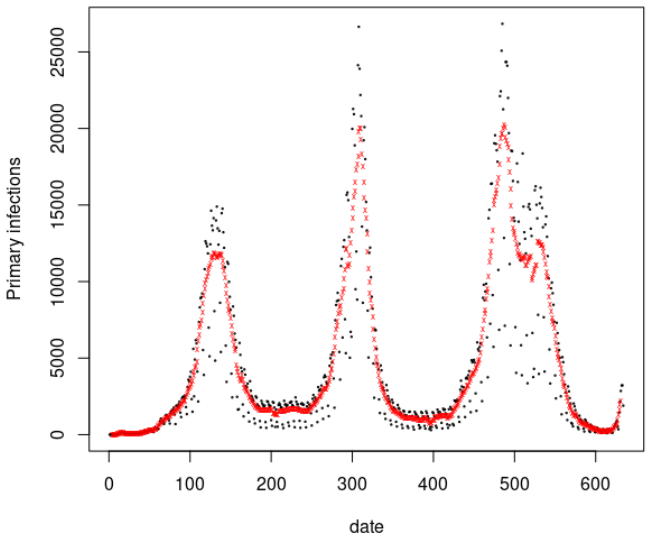


Figure 36: A time series of primary infections reported from March 4th, 2020 to November 27th, 2021. The red points represent the 7-day moving average, while the black points represent the daily data. The peaks represent the wave periods driven by Beta, Delta and Omicron variants.

The posterior distributions of α and β were estimated using Markov Chain Monte Carlo (MCMC) methods. For each run of the MCMC algorithm, we ran 10000 iterations, discarding the first 1500 iterations (burn-in). Figure 37 show the traces of MCMC chains with slow mixing. There appears to be a high degree of serial correlation between the subsequent draws, which is gradually decreasing to zero as shown in the plot. The chains traverse the sample space slowly. The trace plots should ideally show that our MCMC chains are "mixing well" and that the correlation is decreasing to zero, given the number of iterations we ran. Both MCMC chains, however, do not appear to have converged to the posterior distribution. When we plot α against β, we see that there is an almost degenerate direction, causing our MCMC to fail to converge. We need to take long trial steps in the long direction and short steps in the transverse direction to fix this. The draws bounce off the walls and diffuse along the long direction, which is why such long chains are required. Therefore, to fix the convergence issue on our MCMC chains, we calculated the covariance matrix of our chains which we took from an initial run and used to calculate a better choice of trial step in our MCMC algorithm. We then ran the MCMC for the same 10000 iterations discarding the burn-in period of 1500 iterations as we did previously. We can see that taking the covariance matrix helped because Figure 38 show

that our MCMC chains mix well, indicating that the chains converged to the posterior distributions.

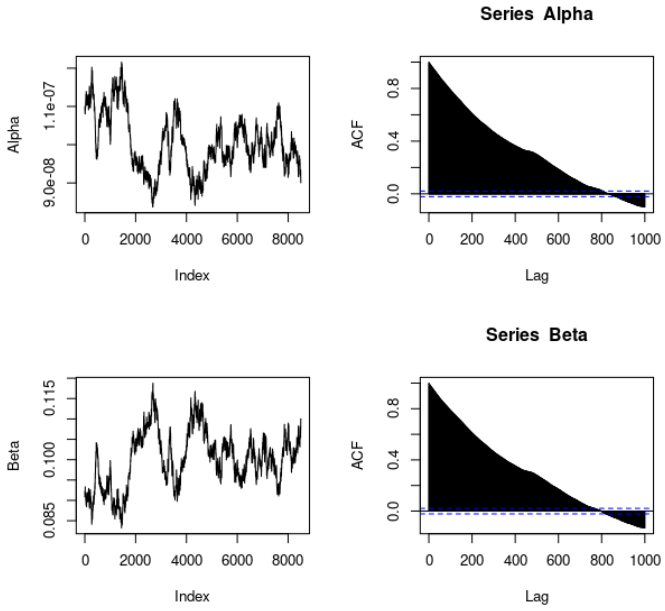


Figure 37: MCMC chains for the posterior distributions of α and β. The high serial correlation between the draws and poor mixing of the chains indicate failure to converge.

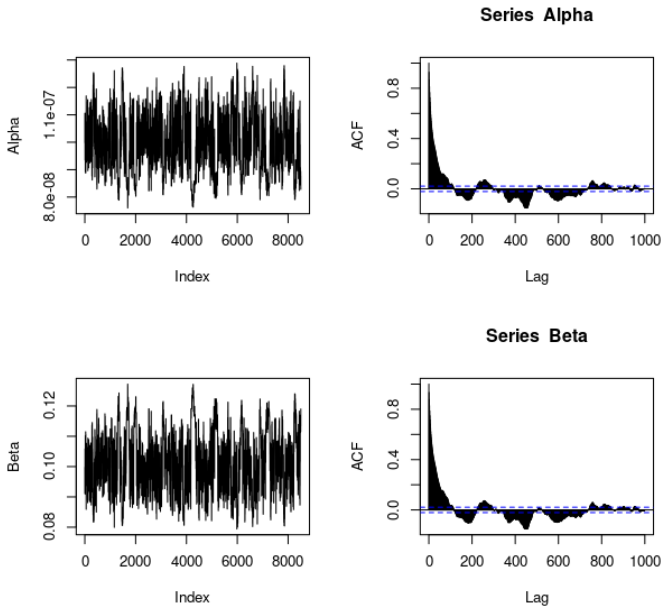


Figure 38: MCMC trace plots that recovers α and β when the calculated covariance matrix is used to generate a better choice of trial step and their corresponding correlation plots at lag up to 1000. The sufficient mixing of chains and quick drop to zero in correlation suggest convergence.

We further extended our model by assuming that α is not constant and changes with time. This is to reflect that various strains can have varying reinfection rates, which we are assuming in this case to determine whether the Omicron variant has an increased reinfection rate. Let i be the dominant COVID-19 variants, ranging from 1 to 4.

$$\begin{aligned} j < j_{12}, i[j] = 1 \\ j_{12} < j < j_{23}, i[j] = 2. \end{aligned} \tag{54}$$

Then $\alpha[j] = \bar{\alpha}[i[j]]$. $\bar{\alpha}[i = 1], \bar{\alpha}[i = 2]$, and $\bar{\alpha}[i = 3]$ are older strains while $\bar{\alpha}[i = 4]$ is the Omicron variant.

The Omicron variant emerged when most people were vaccinated, and we were hoping that the vaccine-induced immunity would cause the pandemic to die out. However, there was a rapid spread of daily reported cases after discovering the Omicron variant, which has led us to suspect that the Omicron variant has an increased prevalence of reinfection than the prior variants of concern (VOCs). So to investigate whether the Omicron variant has an increased reinfection rate, we use the Metropolis-Hastings Markov Chain Monte Carlo (MCMC) to estimate the unknown parameters for the extended model where the Omicron has

a different α from the previous COVID-19 waves. We look at the posterior distribution of the α ratio for earlier waves and the wave caused by the Omicron variant assuming that it is not affected by the fraction of people who got reinfected but did not get a test. Figure 39 shows quite a few flat lines indicating low successive accepted draws. The chain is taking time to explore the sample space, and there is poor mixing. However, the successive draws are bouncing around 1, and Figure 40 shows a short serial correlation between the draws, there is a high correlation at short lags, but it drops to zero rather quickly. Hence we run the chain for longer iterations to see if it has converged to the posterior distribution. Running the chain for longer iterations helped improve the mixing. The chain explores the sample space many times; also, there is a low serial correlation between the successive draws. The MCMC chain is converging to 1, which is consistent with the previous plot. The MCMC chain is converging to one, which is consistent with the previous plot. Contrary to our prediction, the Omicron variant does not have an increased reinfection rate since the posterior of the ratio converges to one, indicating that the reinfection rate does not change across all waves. We used the mock data where there is no difference in the reinfection rate therefore, the MCMC recovered the parameters of the mock data to the expected accuracy since the posterior of the ratio converges to one.

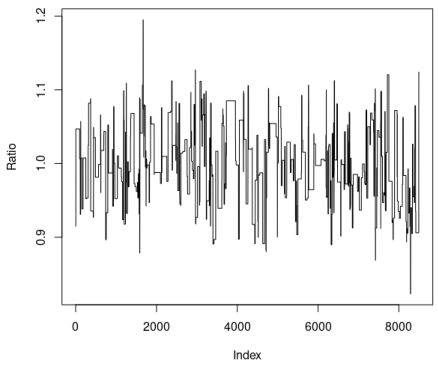


Figure 39: MCMC trace plot for the posterior of the reinfection rate ratio for the Omicron variant and the previous variants.

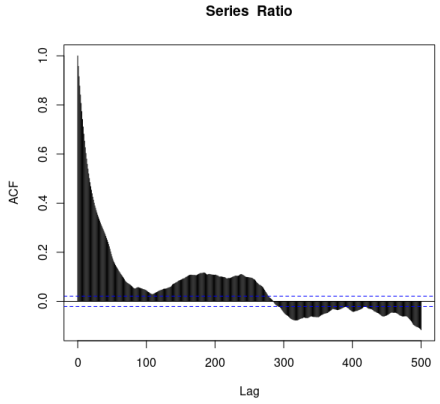


Figure 40: Autocorrelation plot. There is high correlation at short lags but it quickly drops to zero.

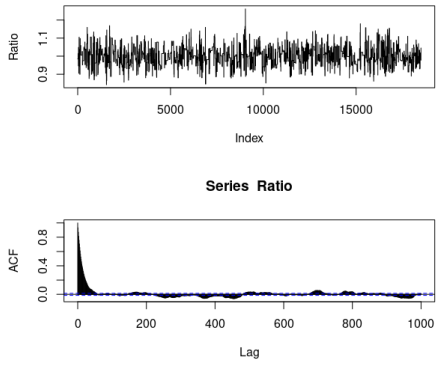


Figure 41: MCMC trace plot for 20000 iterations and a burn-in period of 1500 iterations. There is sufficient mixing, low serial correlation and the chain is converging to one.

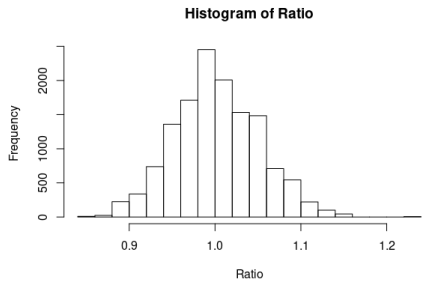


Figure 42: Histogram showing that the ratio can be measured to a 10% accuracy at about 2σ.

7 Conclusion

With the rapid increase in COVID-19 daily reported cases in South Africa in late November 2021 following the emergence of the Omicron variant, the question arose of whether Omicron had an increased reinfection risk compared to the previous Beta and Delta variants. We had wanted to investigate this question using real data as in the Pulliam et al. paper cited above. However, because of restrictions arising from privacy concerns the data made available was incomplete, so we generated simulated data with no difference in the reinfection rate. We were able to limit the relative reinfection rate at 1.0 ± 0.1, or $\pm 10\%$. If the ratio of reinfection rates differed by more than that, the nonequality of the rate could be established with statistical significance. Our analysis successfully recovered the inputs to the simulated data. The MCMC algorithm recovered the parameters that were consistent with the expected accuracy since the posterior of the ratio converges to one.

References

[1] Özüdoğru O., Bahçe Y. G., Acer Ö. SARS CoV-2 reinfection rate is higher in the Omicron variant than in the Alpha and Delta variants, Ir J Med Sci, 2022 Jun 17, doi: 10.1007/s11845-022-03060-4.

[2] Choi J. Y., Smith D. M. SARS-CoV-2 Variants of Concern, Yonsei Med J., 2021 November, 62(11): 961–968, doi: 10.3349/ymj.2021.62.11.961.

[3] Yang W., Shaman J. L. COVID-19 pandemic dynamics in South Africa and epidemiological characteristics of three variants of concern (Beta, Delta, and Omicron), eLife. 2022 August 09, 11: e78933, doi: 10.7554/eLife.78933.

[4] Our World in Data, South Africa: Coronavirus pandemic country profile. https://ourworldindata.org/coronavirus/country/south-africa.

[5] SARS-CoV-2 Omicron variant: Characteristics and prevention He X., Hong W., Pan X., Lu G., Wei X. SARS-CoV-2 Omicron variant: Characteristics and prevention, MedComm (2020), 2021 December 16, 2(4): 838–845, doi: 10.1002/mco2.110.

[6] Ren S. Y., Wang W. B., Gao R, D., Zhou A. M. Omicron variant (B.1.1.529) of SARS-CoV-2: Mutation, infectivity, transmission, and vaccine resistance, World J Clin Cases, 2022 January 7, 10(1): 1–11, doi: 10.12998/wjcc.v10.i1.1.

[7] Araf Y., Akter F., Tang Y. D., Fatemi R., Parvez M. S. A., Zheng C., Hossain M. G. Omicron variant of SARS-CoV-2: Genomics, transmissibility, and responses to current COVID-19 vaccines, J Med Virol, 2022 May, 94(5):1825-1832, doi: 10.1002/jmv.27588.

[8] Pulliam J. R. C., van Schalkwyk C., Govender N.,von Gottberg A., Cohen C., Groome M. J., Dushoff J., Mlisana K., Moultrie H. Increased risk of SARS-CoV-2 reinfection associated with emergence of the Omicron variant in South Africa, 2021 December 02, doi: https://doi.org/10.1101/2021.11.11.21266068.

[9] Pulliam J. R. C., van Schalkwyk C., Govender N.,von Gottberg A., Cohen C., Groome M. J., Dushoff J., Mlisana K., Moultrie H. Increased risk of SARS-CoV-2 reinfection associated with emergence of the Omicron variant in South Africa, Science, 2022 March 15, 376(6593), do: 10.1126/science.abn4947.

[10] Sun K., Tempia S., Kleynhans J., Gottberg A. V., McMorrow M. L., Wolter N., Bhiman J. N., Moyes J., Plessis M., Carrim M., Buys A., Martinson N. A., Kahn K., Tollman S., Lebina L., Wafawanaka F., du Toit J. D., Gómez-Olivé F. X., Mkhencele T., Viboud C., Cohen C. SARS-CoV-2 transmission, persistence of immunity, and estimates of Omicron's impact in South African population cohorts, Science Translational Medicine, 2022 May 31, Vol 14, Issue 659, DOI: 10.1126/scitranslmed.abo7081.

[11] Suzuki R., Yamasoba D., Kimura I., Wang L., Kishimoto M., Ito J., Morioka Y., Nao N., Nasser H., Uriu K., Kosugi Y., Tsuda M., Orba Y., Sasaki M., Shimizu R., Kawabata R., Yoshimatsu K., Asakura H., Nagashima M., Sadamasu K., Yoshimura K., The Genotype to Phenotype Japan (G2P-Japan) Consortium, Sawa H., Ikeda T., Irie T., Matsuno K., Tanaka S., Fukuhara T., Sato K. Attenuated fusogenicity and pathogenicity of SARS-CoV-2 Omicron variant, Nature, 2022, 603(7902): 700–705, doi: 10.1038/s41586-022-04462-1.

[12] Murray, J. D. Mathematical biology, 3rd ed. 2 v. New York : Springer, 2002–2003.

[13] Brauer F., Castillo-Chavez C. Mathematical Models in Population Biology and Epidemiology, 2nd ed. New York: Springer, 2012.

[14] Karaivanov A. A social network model of COVID-19, PLoS One, 2020, 15(10): e0240878, doi: 10.1371/journal.pone.0240878.

[15] Moein S., Nickaeen N., Roointan A., Borhani N., Heidary Z., Javanmard S. H., Ghaisari J.,Gheisari Y. Inefficiency of SIR models in forecasting COVID-19 epidemic: a case study of Isfahan, Sci Rep, 2021, 11: 4725, doi: 10.1038/s41598-021-84055-6.

[16] Grant A. Dynamics of COVID-19 epidemics: SEIR models underestimate peak infection rates and overestimate epidemic duration, medRxiv, 2020 April 02, doi: https://doi.org/10.1101/2020.04.02.20050674.

[17] Morris W. H., Smale S., Devaney R. L. Differential Equations, Dynamical Systems, and an introduction to Chaos, 3rd ed. Elsevier Science: Academic Press, 2012-2013.

[18] Morris W. H., Smale S., Devaney R. L. Differential Equations, Dynamical Systems, and an introduction to Chaos, 2nd ed. Elsevier Science: Academic Press, 2004.

[19] https://en.wikipedia.org/wiki/Ergodic _theory

[20] Strogatz S. H. Nonlinear Dynamics and Chaos, With Applications to Physics, Biology, Chemistry, and Engineering, 1st ed. Westview Press, 1994.

[21] Jack Terwilliger, (A Bit of) Biological Neural Networks – Part I, Spiking Neurons. http://jackterwilliger.com/biological-neural-networks-part-i-spiking-neurons.

[22] Sprott J. C. Strange Attractors: creating patterns in chaos cover, M T Books, 1993.

[23] Wiggins S. Introduction to Applied Nonlinear Dynamical Systems and Chaos, 2nd ed. New York: Springer, 2003.

[24] Fathabadi H. S. Behavior of Limit Cycles in Nonlinear Systems, International Journal of Information and Electronics Engineering, 2012, 2(4): 523-526, ISSN: 2010-3719, doi: 10.7763/IJIEE.2012.V2.152.

[25] Rencher A. C., Schaalje G. B. Linear Models In Statistics, 2nd ed. Wiley-Interscience, 2008.

[26] Venables W. N., Ripley B. D. Morden Applied Statistics with S, 4th ed. New York: Springer, 2002.

[27] Gilks W. R., Richardson S. Markov Chain Monte Carlo In Practice, London: Chapman Hall, 1995 - 1996.

[28] Hogg R. V., McKean J. W., Craig A. T. Introduction to Mathematical Statistics, 7th ed. Pearson, 2012.

[29] Robert C. P., Casella G., Introducing Monte Carlo Methods with R, New York: Springer, 2010.

[30] Gamerman D., Lopes H. F. Markov Chain Monte Carlo: Stochastic Simulation for Bayesian Inference, 2nd ed. Chapman Hall/CRC, 2006.

[31] Brooks S. P., Gelman A. General Methods for Monitoring Convergence of Iterative Simulations, Journal of Computational and Graphical Statistics, 1998, 7(4): 434-455, doi:10.1080/10618600.1998.10474787.

[32] Aleem A., Akbar Samad A. B., Slenker A. K. Emerging Variants of SARS-CoV-2 And Novel Therapeutics Against Coronavirus (COVID-19). https://pubmed.ncbi.nlm.nih.gov/34033342.

[33] Abdool Karima S. S ., Abdool Karima Q. Omicron SARS-CoV-2 Variant: a new chapter in the COVID-19 pandemic, Lancet, 2021, 398(10317): 2126–2128, doi:10.1016/S0140-6736(21)02758-6.

[34] Deng L., Li P., Zhang X., Jiang Q., Turner D., Zhou C., Gao Y, Qian F., Zhang C., Lu H., Zou H., Vermund S. H., Qian H. Risk of SARS-CoV-2 reinfection: a systematic review and meta-analysis, Scientific Reports 12, 20763 (2022), https://doi.org/10.1038/s41598-022-24220-7.

[35] https://github.com/jrcpulliam/reinfections/blob/master/data/ts_data.csv

A Poincaré Code

This is my appendix
In appendix A

```
#!/usr/bin/python 3

import numpy as np
from scipy.integrate import odeint
from scipy.integrate import solve_ivp
import matplotlib.pyplot as plt
from scipy.interpolate import griddata

def duffing(y,t,alpha,beta,delta,gamma,omega):
    x,v = y
    dxdt = v
    dvdt = gamma*np.cos(omega*t)-delta*v -alpha*x -beta*x**3
    return dxdt,dvdt
alpha = 1
beta = 5
delta = 0.02
gamma = 12.34
omega = 0.5

t = np.arange(0,500,0.01)
y0 = [0,1]
y1 = [0.00001,1]
sol = odeint(duffing,y0,t,args=(alpha,beta,delta,gamma,omega))
sol = np.array(sol)

fig = plt.figure()
ax = fig.add_subplot()
ax.plot(sol[:,0],sol[:,1])
ax.set_xlabel('Displacement (x(t))')
ax.set_ylabel('Velocity (v(t))')
plt.show()

def poincare2(alpha,beta,delta,gamma):
    def func(t,w):
        x,v = w
        return [v,gamma*np.cos(t)-delta*v -alpha*x -beta*x**3]
    w0 = np.array([1.0,0.0])
    t = (0,2*np.pi)
    n_iter=10000
    w_out=[]
    for i in range(n_iter):
```

```
      global sol
      sol = solve_ivp(func,t,w0,method='RK45')
      yvec=sol.y
      w0=(yvec[0,-1],yvec[1,-1])
      w_out.append(w0)

    return(w_out)

w_out = poincare2(alpha,beta,delta,gamma)
x,y = zip(*w_out)
drop = 2000

plt.figure()
plt.plot(x[drop:],y[drop:],'.',color="black",markersize=1)
#plt.title('Poincare Section')
plt.show()
```

B Simulated Data Code

```
library(zoo)

data=read.csv("ts_data.csv")
data$date <-as.Date.character(data$date,format = "%Y-%m-%d")

dailyCasesRaw=(data[2])[[1]]
dailyCases7daySmooth=rollmean(dailyCasesRaw,7,fill=NA)
reinfections=(data[3])[[1]]
reinfections7daySmooth=rollmean(reinfections,7,fill=NA)

undercountFactor=10
ncases=undercountFactor*dailyCases7daySmooth

ncases[is.na(ncases)]=0.
ndim=length(ncases)

reinfectionArray=array(0,c(ndim,ndim))
fracArray=array(0,c(ndim,ndim))

lambda=0.00000001
for (i in (1:ndim)){
  n=ncases[i]
  fracSusceptible=1.
  for (j in (min(ndim,i+90):ndim)){
     # need to include varying lambda according to number of wave
     dailyReinfectProb=lambda*ncases[j]
```

```
    reInfect=dailyReinfectProb*fracSusceptible*n
    reinfectionArray[i,j]=rpois(1,reInfect)
    fracArray[i,j]=fracSusceptible
    fracSusceptible=fracSusceptible*exp(-dailyReinfectProb)
  }
}
```

C Log-Likelihood Code

```
library(zoo)
library(testit)
library(graphics)

data=read.csv("ts_data.csv")

dailyCasesRaw=(data[2])[[1]]
dailyCases7daySmooth=rollmean(dailyCasesRaw,7,align="center")
ndim=length(dailyCases7daySmooth)

reinfectionArray=array(0,c(ndim,ndim))   # This array is the mock data.
fracArray=array(0,c(ndim,ndim))

simulateData=function(Alpha,Beta){
  reinfectionArray=array(0,c(ndim,ndim))
  for (i in (1:ndim)){
      fracSusceptible=1.
      for (j in (min(ndim,i+90):ndim)){
        dailyReinfectProb=Alpha*dailyCases7daySmooth[j]
        reInfect=dailyReinfectProb*fracSusceptible*dailyCases7daySmooth[i]
        reinfectionArray[i,j]=Beta*reInfect
        fracArray[i,j] <-fracSusceptible
        fracSusceptible=fracSusceptible*exp(-dailyReinfectProb)
      }
    }
  return(reinfectionArray)
}

Alpha=1.e-7
Beta=0.1

reinfectData=simulateData(Alpha,Beta)

logLikelihood1 <- function(k,k_bar){ # k is the observed outcome of the Poisson process, k_b
    if ((k == 0) & (k_bar == 0 )){
```

```
       print(c(k,k_bar,0))
       return(0.)
    }
    result=(k*log1p(k_bar)- k_bar)
    return(result)
}

ModelPredict2 <- function(Alpha,Beta){
  number_of_cases=length(dailyCases7daySmooth)
  new_daily_reinfections          =array(0.,dim=c(number_of_cases,number_of_cases))
  new_daily_detected_reinfections =array(0.,dim=c(number_of_cases,number_of_cases))
  for (i in 1:number_of_cases){
    susceptible_to_reinfections = dailyCases7daySmooth[i]
    for (j in  (min(number_of_cases,i+90):number_of_cases)){
      new_daily_reinfections[i,j] = Alpha * susceptible_to_reinfections  * dailyCases7daySmo
      susceptible_to_reinfections = susceptible_to_reinfections - new_daily_reinfections[i,j
      new_daily_detected_reinfections[i,j] = Beta * new_daily_reinfections[i,j]
    }
  }
  return(new_daily_detected_reinfections)
}

logLikelihood2 <- function(Alpha,Beta){     # assume that reinfectData exists as a global va
  reinfect_data_expectation=ModelPredict2(Alpha,Beta)
  ni <- dim(reinfectData)[1]
  SUM = 0.
  for (i in 1:ni){
    jstart = min(i+90,ni)
    for (j in (jstart:ni)){
       SUM=SUM +logLikelihood1(reinfectData[i,j],reinfect_data_expectation[i,j])
    }
  }
  return(SUM)
}

fun=function(x,y){
 n=length(x)
 result=array(0.,n)
 for( i in (1:n)){
   result[i]=logLikelihood2(x[i]*Alpha,y[i]*Beta)
 }
 return(result)
}
```

D MCMC Code

```
library(MASS)
source(file="LogLikelihood.R")

step_generator1=function(){
  kdim=2
  step <-rnorm(kdim,mean =0, sd=0.01)
  step[1]=1.e-7*step[1]
  step[2]=1.e-1*step[2]
  return(step)
}

step_generator2=function(){
  step=mvrnorm(1,c(0.,0.),myCov)
  return(step)
}

acceptance_rate=function(W){
   same=head(W,-1)==tail(W,-1)
   accepted=length(same[same==FALSE])
   total=length(same)
   return(accepted/total)
}

make_chain=function(n_chain, step_generator, x_start, logLikelihood){
  x_current=x_start
  Stored_Chains =list(x_current)
  for(i in 1:(n_chain)){
    print(i)
    step=step_generator()
    x_trial <- x_current + step
    loglike_current <- logLikelihood(x_current[1],x_current[2])
    loglike_trial <- logLikelihood(x_trial[1],x_trial[2])
    if ((is.nan(loglike_trial)==FALSE) && loglike_trial > loglike_current){
      x_current <- x_trial
    }
    else {
      p <- exp(loglike_trial-loglike_current)
      if ( runif(1) < p ){
        x_current = x_trial
      }
    }
    Stored_Chains=c(Stored_Chains, list(x_current))
  }
  x_pts <- lapply(Stored_Chains, "[", 1)
```

```
  y_pts <- lapply(Stored_Chains, "[", 2)
  X <- unlist(x_pts)
  Y <- unlist(y_pts)
  return(list("X"=X,"Y"=Y))
}

set.seed(123)
n_chain<-10000
burn_in <- 1500
x_start<-c(1.e-7,0.1)

Chains1=make_chain(n_chain, step_generator1, x_start, logLikelihood2)
X1=Chains1$X
Y1=Chains1$Y

plot(X1,Y1)
plot(X1,type="l")
plot(Y1,type="l")

print(acceptance_rate(X1))

df=data.frame(X1,Y1)
myCov=cov(df)
eigen(myCov)

Chains2=make_chain(n_chain, step_generator2, x_start, logLikelihood2)
X2=Chains2$X
Y2=Chains2$Y
Alpha = X2[1500:10000]
Beta = Y2[1500:10000]

par(mfcol=c(2,1))
plot(Alpha,type="l")
acf(Alpha, lag.max = 500, type = "correlation")

par(mfcol=c(2,1))
plot(Beta,type="l")
acf(Beta, lag.max = 100, type = "correlation")

mean(X2)
mean(Y2)
df2=data.frame(X2,Y2)
myCov2=cov(df2)
print(myCov2)

print(acceptance_rate(X2))
```

E Log-Likelihood Code for Extended Model

```
library(zoo)
library(testit)
library(graphics)

data=read.csv("ts_data.csv")

dailyCasesRaw=(data[2])[[1]]
dailyCases7daySmooth=rollmean(dailyCasesRaw,7,align="center")
ndim=length(dailyCases7daySmooth)

reinfectionArray=array(0,c(ndim,ndim))   # This array is the mock data.
fracArray=array(0,c(ndim,ndim))

simulateData=function(Alpha,Beta){
  reinfectionArray=array(0,c(ndim,ndim))
  for (i in (1:ndim)){
      fracSusceptible=1.
      for (j in (min(ndim,i+90):ndim)){
        dailyReinfectProb=Alpha*dailyCases7daySmooth[j]
        reInfect=dailyReinfectProb*fracSusceptible*dailyCases7daySmooth[i]
        reinfectionArray[i,j]=Beta*reInfect
        fracArray[i,j]<<-fracSusceptible # this type of assignment is required in R to chang
        #print(c(i,j,fracSusceptible))
        fracSusceptible=fracSusceptible*exp(-dailyReinfectProb)
      }
    }
  return(reinfectionArray)
}

Alpha=1.e-7
Alpha1=1.e-7
Alpha2=2.e-7
Beta=0.1

reinfectData=simulateData(Alpha,Beta)

logLikelihood1 <- function(k,k_bar){ # k is the observed outcome of the Poisson process, k_b
    if ((k == 0) & (k_bar == 0 )){
       print(c(k,k_bar,0))
       return(0.)
    }
```

```
    result=(k*log(k_bar)- k_bar)
    return(result)
}

ModelPredict2 <- function(Alpha,Beta){
  nc=length(dailyCases7daySmooth)
  new_daily_reinfections          =array(0.,dim=c(nc,nc))
  new_daily_detected_reinfections =array(0.,dim=c(nc,nc))
  for (i in 1:nc){
    susceptible_to_reinfections = dailyCases7daySmooth[i]
    for (j in  (min(nc,i+90):nc)){
      new_daily_reinfections[i,j] = Alpha * susceptible_to_reinfections  * dailyCases7daySmo
      susceptible_to_reinfections = susceptible_to_reinfections - new_daily_reinfections[i,j
      new_daily_detected_reinfections[i,j] = Beta * new_daily_reinfections[i,j]
    }
  }
  return(new_daily_detected_reinfections)
}

logLikelihood2 <- function(Alpha,Beta){     # assume that reinfectData exists as a global va
  if ( Alpha <=0 ) return(-Inf)
  if ( Beta  <=0 ) return(-Inf)
  reinfect_data_expectation=ModelPredict2(Alpha,Beta)
  ni <- dim(reinfectData)[1]
  SUM = 0.
  for (i in 1:ni){
    jstart = min(i+90,ni)
    for (j in (jstart:ni)){
       SUM=SUM +logLikelihood1(reinfectData[i,j],reinfect_data_expectation[i,j])
    }
  }
  return(SUM)
}

ModelPredict3 <- function(Alpha1, Alpha2, Beta){
  nc=length(dailyCases7daySmooth)
  AlphaVec=c(Alpha1*rep(1.,600),Alpha2*rep(1.,93))
  new_daily_reinfections          =array(0.,dim=c(nc,nc))
  new_daily_detected_reinfections =array(0.,dim=c(nc,nc))
  for (i in 1:nc){
    susceptible_to_reinfections = dailyCases7daySmooth[i]
    for (j in  (min(nc,i+90):nc)){
      new_daily_reinfections[i,j] = AlphaVec[j] * susceptible_to_reinfections  * dailyCases7
      susceptible_to_reinfections = susceptible_to_reinfections - new_daily_reinfections[i,j
      new_daily_detected_reinfections[i,j] = Beta * new_daily_reinfections[i,j]
    }
```

```
  }
  return(new_daily_detected_reinfections)
}

logLikelihood3 <- function(Alpha1,Alpha2,Beta){     # assume that reinfectData exists as a g
  if ( Alpha1 <=0 ) return(-Inf)
  if ( Alpha2 <=0 ) return(-Inf)
  if ( Beta  <=0 ) return(-Inf)
  reinfect_data_expectation=ModelPredict3(Alpha1,Alpha2,Beta)
  ni <- dim(reinfectData)[1]
  SUM = 0.
  for (i in 1:ni){
    jstart = min(i+90,ni)
    for (j in (jstart:ni)){
       SUM=SUM +logLikelihood1(reinfectData[i,j],reinfect_data_expectation[i,j])
    }
  }
  return(SUM)
}
```

F MCMC Code for Extended Model

```
    library(MASS)
source(file="LogLikelihood6.R")

step_generator1=function(){
  kdim=3
  step <-rnorm(kdim,mean =0, sd=0.02)
  step[1]=1.e-7*step[1]
  step[2]=1.e-7*step[2]
  step[3]=1.e-1*step[3]
  return(step)
}

step_generator2=function(){
  step=2.5*mvrnorm(1,c(0.,0.,0.),myCov)
  return(step)
}

acceptance_rate=function(W){
   same=head(W,-1)==tail(W,-1)
   accepted=length(same[same==FALSE])
   total=length(same)
   return(accepted/total)
}
```

```
make_chain=function(n_chain, step_generator, x_start, logLikelihood){
  x_current=x_start
  Stored_Chains =list(x_current)
  for(i in 1:(n_chain)){
    print(i)
    step=step_generator()
    x_trial <- x_current + step
    loglike_current <- logLikelihood(x_current[1],x_current[2],x_current[3])
    loglike_trial <- logLikelihood(x_trial[1],x_trial[2],x_trial[3])
    if ((is.nan(loglike_trial)==FALSE) && loglike_trial > loglike_current){
      x_current <- x_trial
    }
    else {
      p <- exp(loglike_trial-loglike_current)
      if ( runif(1) < p ){
        x_current = x_trial
      }
    }
    Stored_Chains=c(Stored_Chains, list(x_current))
  }
  x1_pts <- lapply(Stored_Chains, "[", 1)
  x2_pts <- lapply(Stored_Chains, "[", 2)
  y_pts <- lapply(Stored_Chains, "[", 3)
  X1 <- unlist(x1_pts)
  X2 <- unlist(x2_pts)
  Y <- unlist(y_pts)
  return(list( "X1"=X1, "X2"=X2, "Y"=Y))
}
set.seed(123)
n_chain<-20000
burn_in <- 1500
x_start<-c( 1.e-7, 1.e-7, 0.1)

ChainsA=make_chain(n_chain, step_generator1, x_start, logLikelihood3)
X1_A=ChainsA$X1
X2_A=ChainsA$X2
Y_A=ChainsA$Y

df=data.frame(X1_A, X2_A, Y_A)
myCov=cov(df)
eigen(myCov)

ChainsB=make_chain(n_chain, step_generator2, x_start, logLikelihood3)
X1_B=ChainsB$X1
X2_B=ChainsB$X2
```

```
Y_B=ChainsB$Y

mean(X1_B)
mean(X2_B)
mean(Y_B)
dfB=data.frame(X1_B, X2_B, Y_B)
myCovB=cov(dfB)
print(myCovB)

ratio=X1_B/X2_B
Ratio = ratio[1501:20000]
par(mfcol=c(2,1))
plot(Ratio,type="l")
acf(Ratio,lag.max = 1000,type = "correlation")
```