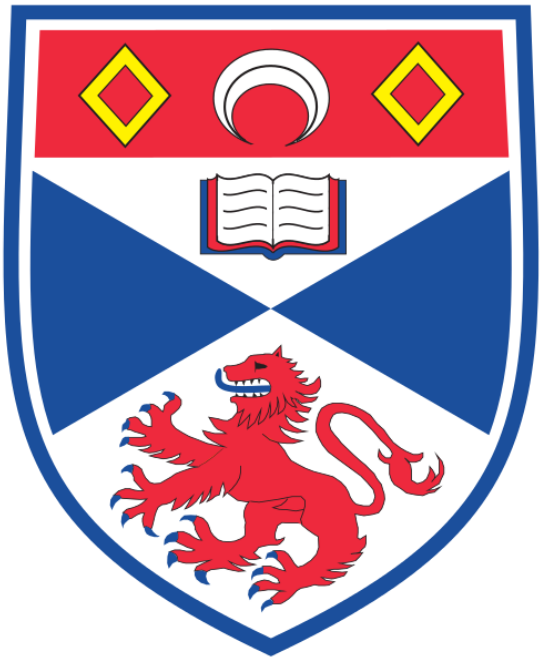
Modelling and simulating the complex relation between spread of disease, policy, and public obedience



Alexander Johnstone (170015593)

Supervised by Prof. Simon Dobson

12th of April 2021

Abstract

Over the course of the current COVID-19 pandemic there has been distinct variance between each countries response to the problem of suppressing the virus. Equally there has been disparity in the effect of the virus for each country, with great differences in proportions of cases and deaths. These differences have illuminated questions of the causation of these differences. Specifically, how government policies and public response (obedience) may affect the outcome of an epidemic. It is crucial to understand this relationship for future epidemics so that governments understand the best course of policy as well as how much public obedience they must generate to ensure control. In this paper this relationship is explored through simulation of a custom COVID-19 specific SIR compartmented model. The resulting (initial) experiment performed using this model suggests that individual disobedience is not as detrimental as proportional community-based disobedience. Furthermore, the results suggest understanding your populations social topology is crucial to creating good mitigation strategies.

Declaration

I declare that the material submitted for assessment is my own work except where credit is explicitly given to others by citation or acknowledgement. This work was performed during the current academic year except where otherwise stated. The main text of this project report is 17,278 words long, including project specification and plan. In submitting this project report to the University of St Andrews, I give permission for it to be made available for use in accordance with the regulations of the University Library. I also give permission for the title and abstract to be published and for copies of the report to be made and supplied at cost to any bona fide library or research worker, and to be made available on the World Wide Web. I retain the copyright in this work.

Acknowledgments

I would like to thank my supervisor, Professor Simon Dobson, for his continuous help, guidance, and reliability throughout the course of the project. Without his advice, knowledge of the subject, and jovial spirit I would not have been able to complete the project to the same level.

Contents

1 Introduction

Modelling and computer simulation is an extremely useful tool that allows researchers to explore how events would transpire in different situations and under different parameters in a simplified space. It is impossible to simulate with complete (or even near) precision most of the time due to a vast number of unknowns and occasionally lack of computing power. This has led researchers to consistently return to compartmented models to model specific scenarios where the assumptions are clear and stated. A disparity in different countries ability to suppress the spread of COVID-19 has asserted the question of why different countries have fared better than others. Modelling all these enumerable factors would be infeasible however a small, related subset could be vital to illuminating how effective policy is and potentially inform decision making. In a simplified sense, when attempting to suppress spread there are two modalities that can majorly affect the outcome. These are governments (and the rules/policies they impose) and the public (their obedience of these rules). A country with no mitigation plan will not supress a virus like COVID-19. Similarly, a country where the population does not respect or obey any mitigation plan will fare equally badly. The aim of this project, primarily, is to simulate the effect different levels of obedience to lockdown policy has on the outcome of a COVID-19 epidemic. The project aims to create software that allows exploration of obedience to this policy as well as how different strategies of the government affect the outcome. The objective of this project is to be able to use computer simulation techniques (compartmented models) to model a COVID-19 pandemic in which we can specify how obedient the public are to a lockdown as well as how the government decide to enter lockdowns (and other factors). This software will then be used to conduct experiment(s) that explore the relationship between government strategy, public obedience, and the outcome of a COVID-19 epidemic. Specifically, the objectives of the project are, in order of importance:

1. To create a compartmented SIR model for COVID-19 including obedience-based lockdowns and government lockdown instigation strategies.
2. To create software that allows diverse experiments to be run on this model.
3. To provide choice of base network to tailor the experiment further.
4. To incorporate other important/common policies in the model
5. To run an experiment that show the relationship between lockdown obedience, policy, and epidemic outcome.
6. To display the result of an epidemic in several formats

I have implemented objectives 1-4. I implemented an extensive model that includes common lockdown triggers, two lockdown obedience policies based on different perspectives of obedience. I also met objective 4 by implementing isolation and tracing policy to increase the depth though these are not implemented with regards to obedience. Furthermore, I have fulfilled objective 3 by implementing a series of diverse network generators that produce graphs with different combinations of properties. I have implemented a small front-end that allow the user to interface with the model and specify experiments, satisfying objective 2. Finally, I successfully ran an experiment using the software I created that illustrates the relationship between policy, obedience, and outcome as well as a supplementary experiment. This satisfies objective 5 and is included in the report. The current project does not meet objective 6. The software only presents the

results in one format which is a line chart of depicting many nodes are removed and how many are still susceptible at the end of each simulated epidemic.

2 Context Survey

2.1 COVID-19, Policy and Public Behaviour

Since the rise and spread of COVID-19 governments everywhere have sought various policies to restrict the spread of the virus. In this context a policy is a course of action proposed by a government to prevent, control, or reduce the spread of the virus. These policies are a significant factor in each countries effectiveness and ability to prevent and control the spread of COVID-19. There has been notable variation in each country's ability to contain the virus and ultimately to prevent mortality. Countries such as South Korea and China have controlled the spread of the disease impeccably, having approximately 100,000 confirmed cases each as of March 2021 [1] [2]. This accounts for 0.19% and 0.007% of their populations, respectively [3]. This is an astonishingly low rate when compared to the UK who have had approximately 4,500,000 [4] confirmed cases, accounting for approximately 7% of the population [3]. Countless factors contribute to a countries ability to control virus spread. Policy, accuracy of information, latency of information, public obedience, public wealth, population susceptibility, international connectivity, and much more. Modelling all these factors is a gargantuan task, however selecting a dominant subset would allow one to examine the complex interaction between the subset of factors to learn why certain countries fare better than others. Unsurprisingly countries that have handled the pandemic well are not new-comers, China and South Korea have both been exposed to viral deadly respiratory syndromes before (SARS and MERS respectively). This correlation is likely to be more than coincidence and suggests that countries with experience in attempting to control viral disease are better prepared for novel outbreaks. Logically this leads one to question why certain countries have been more effective than others. This is an interesting question however a far more interesting question would be to ask where the tipping points lie. For example, how much public disobedience of policy is too much?

A study conducted in March 2020 found that only strict adherence to lockdown measures as well as a small number of co-habitants per household was the only way to prevent exponential spread of disease in Italian communities [5]. The study has similar aims and approach to this one, however it is focussed solely on quarantine adherence rather than a wider combination of factors. The study focusses on two variables: household size and time spent in public per day. The also use the SEIR stochastic model (a more sophisticated SIR model) to model spread. Time spent in public per day is their measurement of obedience to quarantine measures. This has the advantage of being measurable in the real world, however, it does not account for variance in obedience (they assume everyone either obeys or disobeys to the same degree). Furthermore, it makes no variation upon household size and assumes every household is the same size in each test. Aside from this, the paper is compelling in conveying that small households and strict adherence are required for effective disease control. This study is a good example of papers in the literature and the results implore further study.

Given the aim of this study it is important to contextualise and describe the factors chosen for my model. Below is a subset of controllable or influenceable factors that affect the spread of disease.

2.1.1 R_0 Based Policy

The infamous R rate has perhaps been the most crucial and prevalent statistic related to COVID-19. The R rate is a colloquialism of R_0 which is known as the Basic Reproduction Rate. R_0 represents the number of infections expected to be produce as a direct result of a single initial infection [6]. This rate assumes a fully susceptible population with no immunisation. This reproduction rate is crucial as it determines how the disease spreads through the population. Any R_0 above 1 causes logarithmic growth in the number of cases, causing the virus to spread rapidly and uncontrollably. An R_0 beneath 1 however have the opposite characteristic and die out naturally. Unanimously and un-controversially it is maintained that to supress COVID-19, the reproductive rate must be kept beneath 1. Oversimplification in the media can lead to misconceptions on reproduction rates. Over the course of the pandemic the proposed reproduction rate for a country can change from week to week. This does not align with the definition of R_0 as it is a fixed rate based on a fully susceptible population. This is because the reported reproduction rate is closer to the effective reproduction rate R_t which is defined as the average number of new infections caused by a single case at time t [7]. This rate of reproduction is susceptible to more factors such as immunity and changes in social topology (e.g., isolation and lockdown). Measuring the actual reproduction rate of a population is complex and can only be done in hindsight, it is also unreliable. Using confirmed cases and COVID related mortality rates the reproductive rate can be inferred, however due to reporting lag any given reproductive number is a delayed figure [8]. Furthermore, there is considerable error in this measurement due to factors such as unconfirmed cases (asymptomatic cases). Another disadvantage of reproductive rates is that there are several ways to calculate them, leading to different outcomes based on the same data due to different assumptions. Despite its lag, uncertainty, and variability it is still an important value that many governments have used this value to dictate their plan of action. The United Kingdom cited a potential increase in reproductive rate as a reason for entering a lockdown on the 26th of December 2020 [9]. Similarly, Italy reimposed lockdown measures in March 2021 due to the reproductive rate rising above 1 according to their health minister [10]. These are a small sample of governments using reproductive rate to inform and justify spread prevention measures.

2.1.2 Case Based Policy

A complimentary, equally important, and far more intuitive figure reported often in the media is the number of cases. The cases figure, given daily or weekly, is the number of new confirmed cases, reported in the relevant time frame. This figure has less lag, because it is not concerned with mortality, however, it still faces the same problem of asymptomatic cases and those who do not confirm when they are infected. Along with rate of reproduction, many countries have cited daily cases as another reason to justify interventions such as lockdowns, quarantining and other social distancing. Paris entered a city lockdown in March 2021 with a large surge in daily cases being the driving factor [11]. Tokyo and surrounding areas were placed under emergency lockdowns after record case levels were reported in an unexpected spike [12]. This is a recurrent theme of the last year in which governments have exposed that daily cases, along with rate of reproduction, are a vital factor in determining when and how to act.

2.1.3 Lockdown Obedience

Unfortunately, a repetitive theme throughout the COVID-19 pandemic has been lack of obedience to policy and laws set out by governments. Lockdowns epitomise the strategy

of Governments to supress COVID-19 spread, which is to restrict the mixing of individuals as much as possible. In the UK lockdowns are a national mandate to stay at home as much as possible, specifying a limited number of essential reasons to be outside of your home and threatening fines for those who misbehave [13]. Other countries have been even stricter, for example in Spain, their national army were deployed onto the streets of Madrid to help enforce obedience of lockdown measures [14]. Regardless of the degree of lockdown severity and enforcement, disobedience stories have been prevalent. In the UK reported disobedience has ranged from an individual level (often notable figures) to illegal mass gatherings [15] [16]. Given that lockdowns are a pillar of most COVID-19 suppression strategies it is vital to understand the cost of different types of disobedience. Firstly, in the simpler case, how many disobedient individuals are required to undo lockdown efforts? In the more complex case, we may consider communities of disobedient individuals, are these communities more threatening and do they render lockdowns fruitless?

2.1.4 Isolation and Contact Tracing

Lockdowns can in some ways be thought of as a proactive approach. It is necessary however to introduce reactive approaches too. Isolation in epidemiology is the seclusion of a person from society who has been confirmed or suspected to have the virus. This is a reactive measure that can stop likely immediate spread. This spread limiting method is often compounded by tracing systems. These are systems and structures put in place that construct the set of recent contacts for an individual. These contacts can then be found and put under quarantine (similar measure to isolation) in case they too are infected. Isolation is an old technique however contact tracing is more modern and can only be achieved effectively using modern technology. Implementations of these systems determine their effectiveness. Many have attributed the success of South Korea to their state-of-the-art Epidemiological Investigation Support System (EISS) [17]. Oppositely, the UK efforts have been publicly criticised over mistakes in implementation leading to wasted efforts and wasted capital [18]. It has been shown in the literature, through simulation, that even when small-scale tracing is introduced it can have great effect in reducing spread and as such it is a potentially key policy that should be included in my model [19].

A recent study investigated the relationship between detection policy and lockdown intervention with emphasis on reducing ICU capacity. The study aimed to find an optimal policy for controlling the epidemic [20]. This study suggests that if mass resources are dedicated to detecting infected individuals, the need for social distancing is lesser, when compared to detecting immune individuals. This is another important recent study in the literature as it also models a complex interaction between a subset of factors related to COVID-19 control. Using multiple “levers” such as lockdown, isolation and quarantining the researchers were able to explore, through simulation, the interactions and conclude a series of optimal strategies that in tandem create control. This study concludes that quick and strong measures recover control of the epidemic, and that relaxation should be very slow to keep it under control. This study is optimised for cost (fiscal and social) as well as healthcare saturation. Though being very detailed and meticulous in approach, it is also perhaps slightly narrow. Quick and strong measures may be a control requirement; however, it does not help policy makers understand where the tipping point between control and epidemics lie in a complex system. This leaves room to explore these questions,

to find, in a lagged system, when intervention must be taken. If we intervene when the reproduction rate is above 1, it is likely already too late.

2.2 The S.I.R Model and Simulation

To simulate the proposed scenario (prior to additional constraints of policy) we must find a way to model the spread of an epidemic. One of the most popular methods of modelling in epidemiology is called Compartmented models. In a compartmented model, people may only belong to one compartment at a time, and over the course of a simulation they move from one compartment to another based on the structure of the model. The most common compartmented model is the S.I.R model which simplifies infectious diseases to a mathematical model. In the S.I.R model, there are three compartments [21]:

> (S) Susceptible individuals *x(t)*. This is the set of people who are yet to be infected with a disease at time *t*.
>
> (I) Infected individuals *y(t)*. This is the set of people who are infected with the disease and capable of spreading it at time *t*.
>
> (R) Removed individuals *z(t)*. This is the set of people who have been infected and are now recovered (also represents deceased). They are no longer infectious and cannot be re-infected.

Individuals may transition to another compartment if they belong to a compartment with a progression condition, transitions only happen at the instant of passing from one interval *t* to the next *t+1* [22]. A progression condition is a condition which allows an individual to transition from one compartment to another. In this context an individual may transition S->I or I->R with probabilities β and γ respectively. When specifying an instance of the model a starting population of infected individuals (one or more) must be specified, as well as a graph/network representing the population and their interconnections. Probabilities β and γ must also be specified according to the characteristics of the infectious disease with respect to the interval represented by each unit interval of *t*. More complex compartmented models exist which build upon the basic S.I.R model and often they are tailored to the disease or scenario, for example the S.E.I.S model which models diseases where individuals return to being susceptible instead of recovering. Criticism can be made over the use of a simplistic S.I.R model for COVID-19 as there is variation in how much immunity recovered individuals have as well as their ability to carry the virus still [23]. Using more complex models may not always be necessary and over complication could lead to uninterpretable results. Should a trend be found in a simple model, it is likely it will generalise to a more complex one.

2.3 Network Theory

To simulate an infectious disease using the SIR model it is necessary to have modelled an underlying population. This is almost exclusively fulfilled by using graphs. In fact, most complex systems can be represented using an undirected graph of nodes and edges. Undirected graphs can be formally defined as an ordered pair G = (V, E) where V is the set of vertices/nodes and E is a set of edges connecting pairs of nodes. Graphs may also be directed and/or weighted in which edges have directions and/or an associated real weight, respectively. Network theory is a specific field of graph theory in which the nodes are representative of real-objects or such that nodes have attributes. Network theory has various applications between social networks, the transfer of knowledge, shipping networks and much more. Social network theory focuses on the relations between nodes

such that the edges represent a social connection between two people such as: business relations, friendships, sexual contacts etc. There are two major sections within social network theory, these being network analysis and network modelling. Analysis of social networks is important so that we may understand fundamental characteristics of the network structure as well as more complex properties such as how diseases, information and innovations may spread. Social network modelling is crucial for analysis as it is very rare that we can observe real networks. To assess a social network requires large people participation which can raise several ethical problems and barriers. Furthermore, it can be hard to secure such participation from the desired network and it can be difficult to set boundaries on whom the network applies to. Finally, practical limitations like computing power, infrastructure and participant reward/payment limit the scope of observing real networks. This set of problems has given rise to the field of network modelling. There are a small number of observed social networks in the literature from which the properties found have been used to guide the modelling algorithms developed.

Definitions

Connected

A graph is connected if for any two nodes in the graph a path exists between them. In this context a path is defined as a route from one node to another along a series of edges in the graph

Path Length

Path length is logically the number of edges in a path. Normally the path length between two nodes refers to the minimum path. It is also quite useful to use average path length as a graph metric. The average path length is the average of every possible minimum path in the graph i.e., this means that the minimum path between every possible node pairing must be calculated and then the average of all those paths is taken.

Degree

The degree is a node attribute and is defined as the number of edges that are incident on that node.

Degree Distribution

The degree distribution of a network is the probability distribution of the node degrees of over the entire graph. The degree distribution of a graph is defined as:

$$P(k) = \frac{n_k}{n}$$

This means that it is the fraction of nodes n that have a certain degree n_k. In the realm of model generation different models will create different degree distributions and these tend to be useful for inferring related properties of a network.

Scale free networks

A scale free network is common property of models in the literature, and it is defined as a network with a power law degree distribution. A power law is defined as a relationship between two quantities in which a relative change in one quantity results in a proportional change in another [24]. In this case it is used to describe the case where a

small number of elements clustered at the top of the distribution dominate most resources [24]. In the context of network degree distributions this refers to a small number of nodes having a very large degree and most nodes having a much smaller degree. This degree distribution is denoted as:

$$P(k) = k^{-\gamma}$$

Where γ is some exponent usually between 2 and 3 [25]. Figure 1 shows an example of a scale free distribution. Degree distributions (and other distributions) are normally plotted against linear axis' however exponential axis' make the data points more legible. The plot clearly shows the relationship between the exponentially larger degree in nodes for an inversely exponentially small fraction of nodes.

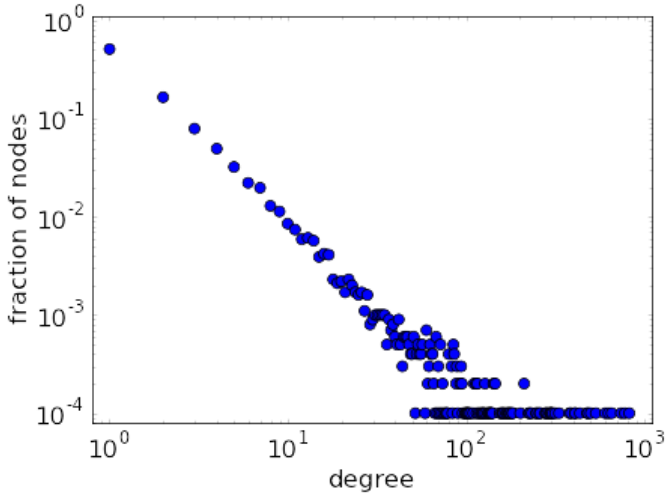


Figure 1 - Exemplar Scale Free distribution plot [26]

Clustering

Clustering is a term which describes the cliques of a network. In this instance we may consider a clique as a group of acquaintances in which each member knows every other member. It is useful to be able to measure the clustering in a network and a common method to do so is the clustering co-efficient proposed by Watts and Strogatz [27]. This is the average of the clustering co-efficient for each node. The clustering of single node is determined as follows:

Consider node n, it has k_n edges connecting to other, distinct, nodes. Suppose that a full clique is if there is a connection between every possible pairing of nodes in k_n. The total number of possible edges would be $k_n(k_n-1)/2$. Let E_n represent the number of edges that exist between the k_n nodes. The clustering co-efficient of that node is the ratio between these two values:

$$C_n = \frac{2E_n}{k_n(k_n - 1)}$$

The clustering co-efficient of the network is the average of the clustering co-efficients of the nodes. It is easy to misinterpret the meaning of the clustering co-efficient, it only determines the local cohesiveness of a neighbourhood of nodes. It has been argued that

this co-efficient fails to capture the presence of communities in a network [28]. This is an important drawback and while it is true that if you know two people, there is a high probability that they are acquainted too it does not capture the density of nodes in a larger community [29].

Small World Properties

Small world networks are a type of network defined by Watts and Strogatz [27]. They are defined as a network G = (V, E), that when compared with a random network $G_r = (V_r, E_r)$ such that $|V| = |V_r|$ & $|E| = |E_r|$, has higher clustering co-efficient- and the path length between two nodes scales as the logarithm of the number of nodes, $Log(|V|)$. Small worlds are often characterised using average path lengths and clustering co-efficients. This property is more commonly known as the 'six degrees of separation' which states that the average number of connections from one person to another is through 6 connections It was originally suggested to be present in human societies by Milgram et al who conducted several experiments in the US that backed up this claim [30].

Communities

A community within a network is a group of nodes within a network with a higher-than-average density of edges such that they mimic a social community. A community in a network is akin to town or city within a country. Formally communities within a graph are groups of nodes that have a high concentration of edges within themselves and have a low concentration of edges between the distinct groups [31]. Therefore, continuing with the previous example, if one were to model the population of the United Kingdom, people in Glasgow and London would have a high concentration of edges within their respective city but the relative concentration of edges between people in either city would be much lower.

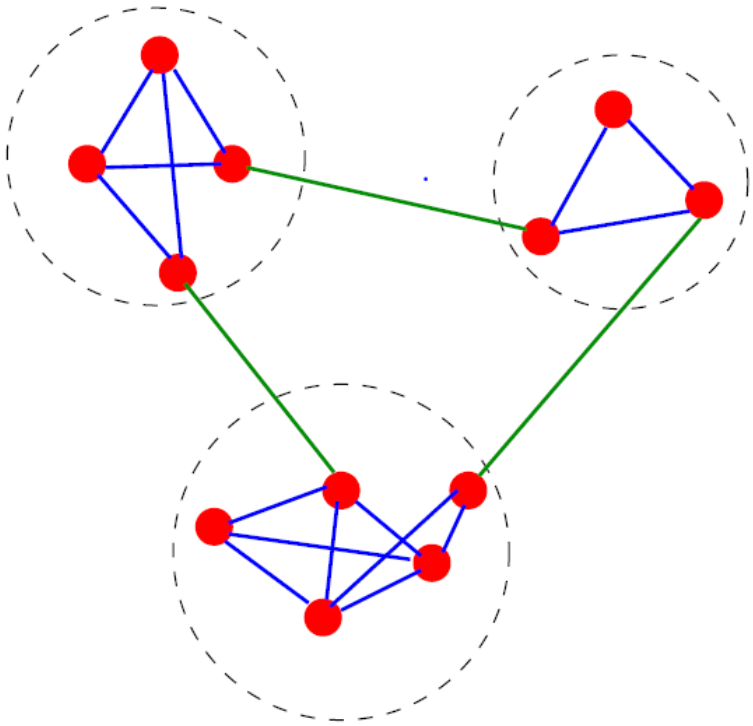


Figure 2 - Simple visualisation of a graph with three distinct communities, encircled by dashed circles [32]

There are various papers in the literature that detail the observed results of social networks however their relevance cannot be fully known as there will always be differences between said researched network and any specific context. These differences take many forms, from differences in size and space, to differences in culture and environment. There are however some properties of social networks that are prevalent in all networks which I will focus on incorporating. A specific study by Gábor Csányi and Balázs Szendrői was conducted on a social network where the number of individuals was size 10^4 [33]. This study is relevant as it uses a similar sized *real* data set, which lack of has often hindered the subject. Analysis of the data showed many features, including a scale-free network, small world properties, a large clustering co-efficient and showed evidence of a natural upper bound of the degree of a node. The natural upper bound provides reason to impose cut-off on the number of connections a node can have, reducing scale-free properties. What is important about these observed features is that they are not novel; each of these properties have been observed previously in other works. Although there are no definitively omnipresent social network characteristics, it is key to recognise observed characteristics and to be aware of them when modelling.

3 Requirement Specification

3.1 Primary Requirements

The primary requirements are at the core of the projects purpose. These are essential to being able to meet the project objective. To model public obedience, the model must firstly be able to detect and enter a lockdown as well as be able to enter a lockdown where obedience is not uniform. The model is useless unless there is a surrounding structure that allows experiments and simulations to be performed on the model. Finally, the purpose of building this model is to be able to explore what effect public obedience has on policy and as such a primary requirement is to conduct the experiment using the software.

- A COVID-19 specific SIR model that can use the lockdown triggers, R_0 and Daily Cases, to dictate when the model enters/leaves a lockdown.
- A COVID-19 specific SIR model that incorporates varied lockdown obedience.
- A program that allows a user to run simulations on the model.
- Run an experiment using the software that shows the relationship between public obedience and government policy with respect to epidemic outcome.

3.2 Secondary Requirements

The secondary requirements are important to significantly improve the scope of the project and increase the projects value. Using different networks is important as it allows experiments to be based on different network assumptions. The same experiment can be tried on a variety of networks to find trends. Using different lockdown variance approaches increases the experiment width as it allows for exploration of the effect of different forms of (dis)obedience. Isolation is also an important and prevalent characteristic of epidemic response and including this in the model increases its depth.

- Network generators (multiple algorithms) that generate graphs with different properties and characteristics (scale-free, small world, high average clustering).
- Lockdowns that are varied on an individual basis as well as a community basis.
- A COVID-19 specific SIR model that incorporates isolation of confirmed cases.

3.3 Tertiary Requirements

Finally, a list of tertiary requirements is presented that increase the scope of the project and improve usability. The tracing requirement increases the depth and scope of the model. The final two requirements increase the usefulness of the program and would allow users to retrieve results in a desired format as well as run experiments more efficiently.

- A COVID-19 specific SIR model that incorporates tracing of contacts after a confirmed case.
- An experiment setup that allows a user to run several simulations with different parameters in one experiment.
- Be able to present results from an experiment in multiple formats.

4 Software Engineering Process

4.1 Approach

There were several approaches available however in the end I settled on a prototyping approach. The exact direction of the project was slightly unclear at the beginning. My supervisor and I had a series of related questions of which we did not know the most interesting/important. As a result, this led to a prototypical approach in which a quick prototype was drawn up that contained useable implementations of each key part. Once a prototype had been created, we were able to explore and evaluate the question space and narrow the project's purpose. This was an iterative process. The agile development method would not have been suitable as there was no clear requirement specification at the beginning. Once the projects' purpose had been fully decided it was best to incorporate the best of each development process, having weekly meetings with my supervisor (akin to agile sprints) where we discussed the progress and changes to evaluate any refinements to the prototype.

4.2 Tools and Resources

A crucial aspect of developing software is tool picking. Some must be picked before beginning and others become necessary due to revelations later in development.

4.2.1 Python

Python is a high-level programming language that allows for quick development due to its large community and large package index https://pypi.org/. Python was chosen over similarly high-level languages such as Java as it has relevant packages that allow for quicker development.

https://www.python.org/

4.2.2 Epydemic

Epydemic is a small package that provides a framework for simulating epidemics on networks. This package has direct application to my project and is easy to extend to introduce extra criteria such as the policies required for the project.

https://pyepydemic.readthedocs.io/en/latest/

4.2.3 NetworkX

NetworkX is a crucial package for graph construction. The package provides all necessary functions to quickly create, represent and store graphs. The package also provides several algorithms for network creation, but these were not useful due to a need for customised algorithms. NetworkX is the compatible package for Epydemic making it the necessary choice.

https://networkx.org/

4.2.4 Matplotlib

To present the results a figure building tool is require. Matplotlib is an established and extensive graphing package that helps build visualisations dynamically and to present the results clearly and easily.

https://matplotlib.org/

4.2.5 Eel

Eel is an incredibly helpful small library for making simple HTML GUIs. The library allows you to host HTML on a local webserver, and through use of annotations one can make calls to python from the frontend or calls to the JavaScript from the backend. The need for a simple frontend became evident later in the development process as command line input became increasingly less viable due to the variety of inputs. As I have minimal experience with common Python GUI packages such as TKinter this made mocking up a frontend easier.

https://pypi.org/project/Eel/

4.2.6 HTML, CSS & JavaScript

The frontend scripting languages were dictated by eel. HTML and CSS were necessary to create an attractive and useable frontend. JavaScript was also necessary for input validation as well as for communicating with the backend.

https://html.spec.whatwg.org/

4.2.7 PyCharm

Finally, it is necessary to choose a suitable IDE for developing the software. I choose PyCharm due to an acquittance with other JetBrains products. PyCharm was also a suitable choice due to its flexibility which allowed for integrated development in the scripting languages.

https://www.jetbrains.com/pycharm/

5 Ethics

Regarding ethical implications of the project there were none to consider and none raised. This is because personal data was not used to guide the development of any aspect of the program. Some data was used from public sources about general community structures, famous COVID-19 response stories and COVID-19 characteristics however these do not have ethical implications. Initially the project included an element that would have attempted to mimic the undergraduate student network at the University of St. Andrews however this too was to be carried out by using publicly available sources. In the end this element was removed as it detracted from the projects purpose. The ethical approvement document can be found in Appendix 1.

6 Design

Before explaining the details of implementation, it is important to outline important features of the overall design. The overall design can be broken into two main aspects. The first is the design of the model, this is the SIR model and the logic required for the model to work. The second is the procedural communication between a frontend and backend which gives users an interface by which they can interact with the model and design their simulation.

6.1 Model Design

The model is integral to this project as the centrepiece and focus of the design. The rest of the design in many ways provides structure to interface with this model and as such the design of the model is vital. Figure 3 depicts how the model is implemented by the Epydemic package; this is used as the basis for my model. Figure 4 shows the logic of my model design which extends the original functionality. Due to the way Epydemic processes work at each interval the process will execute posted events as well as (potentially) a few

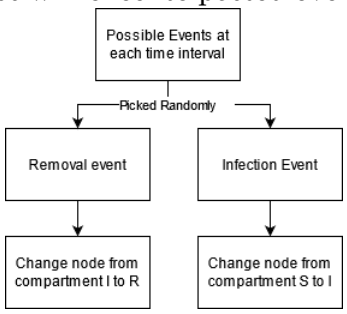


Figure 3 - The design of the SIR model as implemented by Epydemic.

random events. This provides a great multi-faceted approach which allows modelling of events as an event of chance (e.g., infection) or as a structured event (e.g., lockdown). The model I have designed is an extension of the provided SIR model from Epydemic [34]. As shown in figure 4 the infection events are crucial in my design as it is the gateway to all other events. This is because infections update the structures that determine other events or are directly related to events (such as isolation). The infection and removal events are picked at probability chance coherent with the characteristics of COVID-19. The other self-implemented events however are posted at future time intervals. This is partially to do with behaviour control and reproducibility but mainly because the events can be described sequentially as a chain of timed events after an infection. Not all combinations in the model are compatible, such as only one lockdown method may be specified (normal or individually varied obedience or community varied obedience).

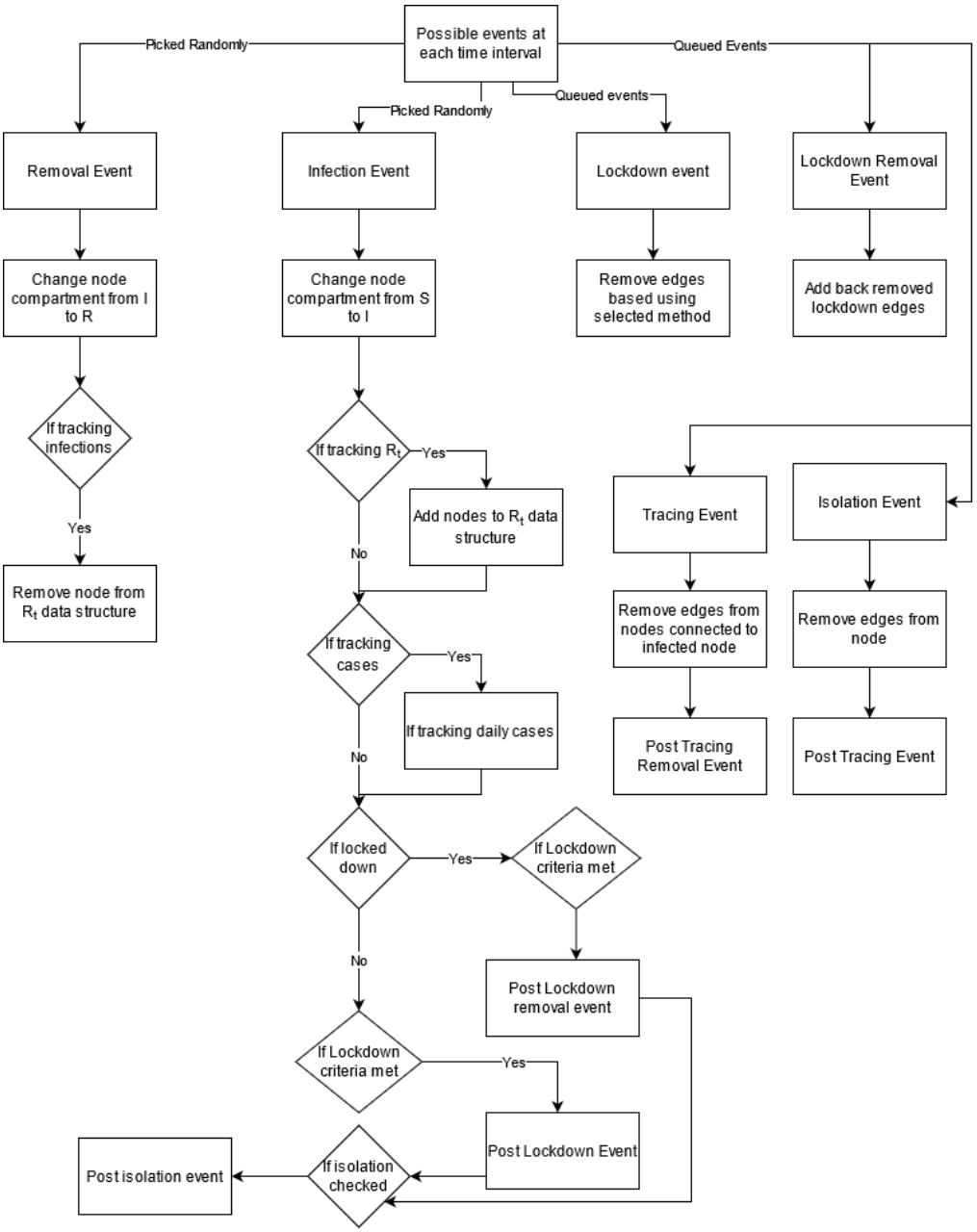


Figure 4 - A logic flowchart that shows the model structure for each possible event at a single time interval. This is an extension of the model shown in Figure 3 which I did not implement.

6.2 Overall Design and Structure

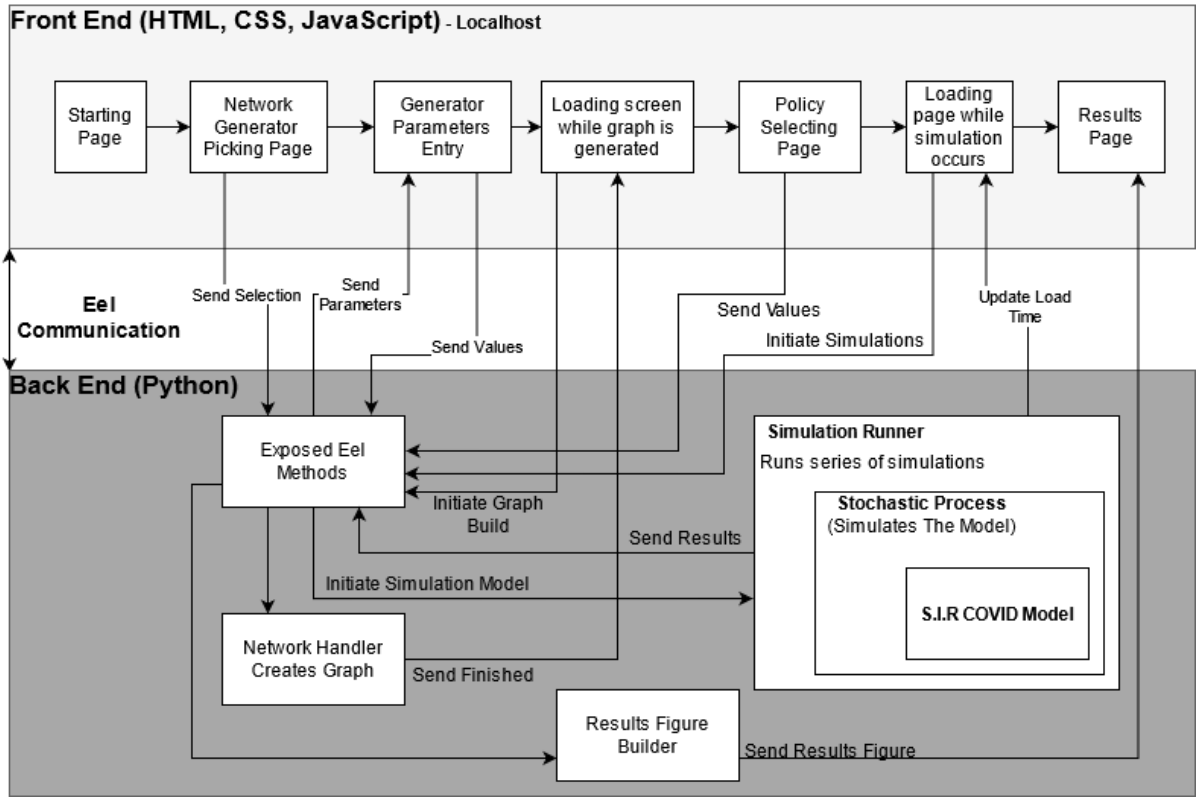


Figure 5 - A diagram showing the procedural flow of operations as a user creates and runs a simulation.

The flow of instructions that a user takes to specify a simulation and run it are very procedural and rigid as shown in figure 5. This is because a lot of input is required from the user, offering flexibility and variety. The overall design of the system is split into a frontend and backend that communicate via Eel. The frontend is not intelligent, however there are several scripts that sanitise input for the backend. In my design I have grouped all the exposed methods (callable by frontend) in one file on the backend which acts as a flow control and access point. These exposed methods orchestrate the other scripts in the backend to set up and run simulations. The backend also occasionally sends messages to the front end to signify the end of a process or to ask for values. The flow of the program from a user perspective is as follows:

1. Specify a network generator.
2. Specify network parameters.
3. Wait for network to be generated.
4. Specify policies and their parameters.
5. Specify the simulation values.
6. Wait for simulation to be run.
7. Observe results.

6.3 Simulation Design

With so many degrees of freedom and flexibility it is hard to design a simulation process that is equally flexible without providing an overwhelming number of options. This is unattractive from a user perspective and implementation perspective. As such it is

important to decide on a design for the simulation that allows user a suitable amount of freedom without overwhelming them. To do this I designed a simulation that is best explained via example, shown in figure 5. The simulation runner creates a series of models according to the input and then simulates each model separately.

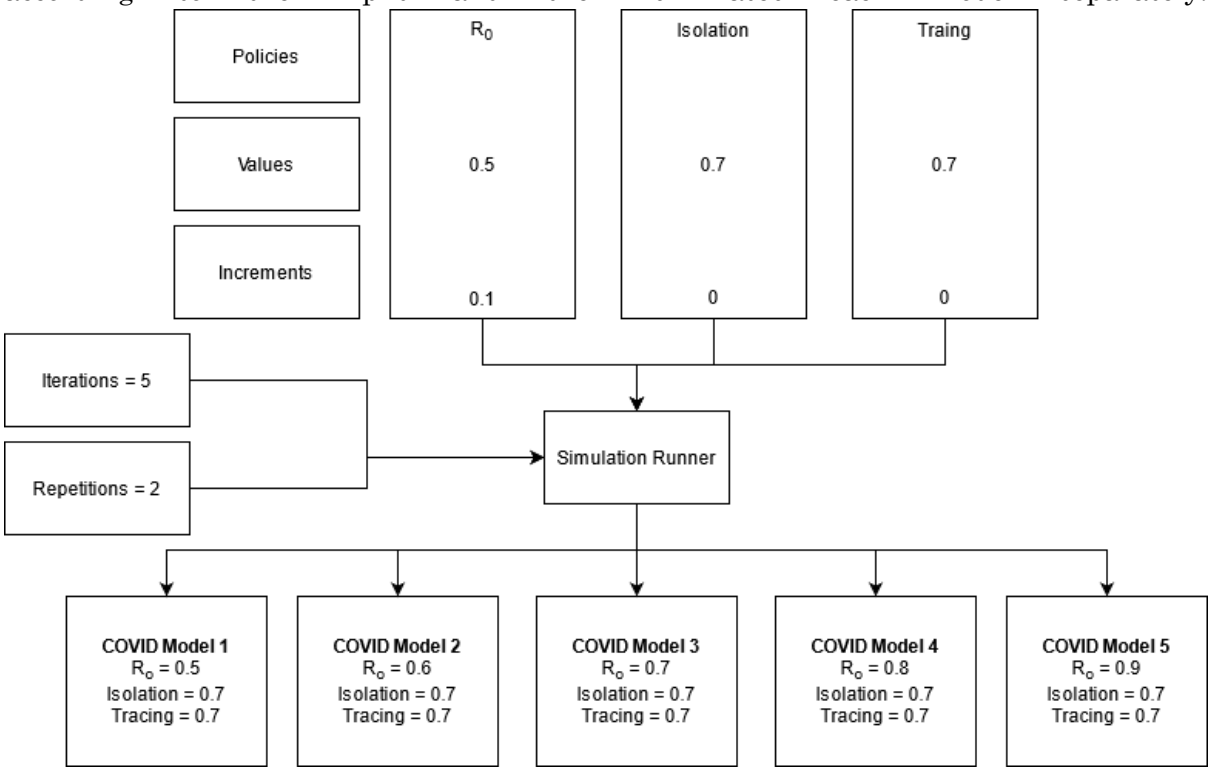


Figure 6 - A diagram showing an example of how models are built from user input

A graph of results can then be produced to show how as one (or more) variables change, the effect it has on the outcome. To specify a simulation a user will need to input the following:

- A network, this stays the same for each model produced.
- The set of policies to be modelled.
    - For each policy:
        - A starting value. The first model produced will have this value.
        - An incremental value. Each subsequent model produced after the first will have value $V_i = V_{i-1} +$ increment
- The number of increments to be made. This determines how many different models will be produced.
- The number of repetitions to be made. This value determines how many times each produced model is simulated. If greater than one the average result is taken.

The input-to-effect correlation can be viewed in figure 6 where the consequence of each input can be seen. The incremental value is set to 5, hence the 5 created models. The repetition value is set to 2 so each individual model would be simulated twice, and the results averaged. An incremental value is set to 0.1 for R_0 which results in the incremental value of R_0 in each subsequent model. The isolation and tracing policies however have an incremental value of 0 so they stay the same in each model. The user may specify more than one incremental value for a simulation. This means that the program would run a

series of models to show the effect in change of parameters, if a user wanted to see results for a specific value, then they could specify an increment of 0 for each policy. This seems to be a good trade-off between usability and freedom to design the simulation. I decided to design the project with the perspective of simulations as this will speed up use for running a series of simulations and allow easy exploration of tipping points and change.

7 Implementation

7.1 Network Generators

There are various approaches to building a network graph however before that can be done, I must try to (loosely) define an ideal network structure. Most graph generation models incorporate a random element, as such it is hard to guarantee all networks will have similar properties. It is also not useful to evaluate networks that only exhibit certain properties and structures, given that most real networks cannot be observed it is important that any eventual results reflect the various possibilities that could arise in the real network. From the context of the literature, I should seek to generate graphs that include scale-free networks, small world properties and a large clustering co-efficient. I should also seek to find a suitable upper bound on the number of contacts a node can have. It should be noted that generated graphs can subscribe to all, a few, or even none of these properties. It is important to have a mixture of different networks being generated within reasonable bounds. As such I decided to implement 4 different algorithms each with different properties to add an essential degree of flexibility to the software.

7.1.1 Implementing an Erdős–Rényi Network

The simplest method of generating a network is simply to generate an empty graph and connect nodes at random. The Erdős–Rényi model is an ambiguous term for a model that generates such random graphs. This term is ambiguous as it is often used to denote the model proposed by Paul Erdős and Alfréd Rényi as well as a model proposed by Edgar Gilbert [35, 36]. For disambiguation I will use the term 'Erdős–Rényi' to denote the $G(n, p)$ model, which refers to a graph constructed by connecting nodes n randomly at probability p independently from other edges. In this model we have a fixed set of nodes and each edge has an identical, independent probability of existing in the graph. I chose this model as most other models in the literature include probabilities of edges existing and therefore it makes more sense to proceed with this version rather than specifying the number of edges to be randomly picked amongst the nodes. The other benefit of this model is that Erdős and Rényi subsequently went on to describe the behaviour of this model for specific P values [37]. Most notably they provided a point by which probabilities above would create connected graphs. This point is given by:

$$p > \frac{(1+\varepsilon)\ln n}{n}$$

Where ε is a small positive real number. As I have no use for isolated nodes this is incredibly useful lower limit on p. Given a desired value n (e.g., 10,000) and taking $\varepsilon = 0.1$ we can derive a lower limit for a connected networks edge probability.

$$p > \frac{(1.01)\ln 10000}{10000}$$

$$\Rightarrow p > 0.0009$$

This value is extremely useful as it gives a minimal value by which to create connected graphs which are essential for the simulation as unconnected graphs cannot model an epidemic well.

Given that the Erdős–Rényi is probability based, theoretically any graph can be produced from a positive probability. However, the graphs produced using this method rarely deviate from a normal degree distribution and often lacks any of the discussed properties (scale-free, small world properties etc).

NetworkX offers an implementation of this algorithm however I decided to implement the algorithm myself primarily from an understanding viewpoint as well as consistency (I implemented all the other algorithms too).

7.1.2 Implementing a Barabási-Albert Network

The Barabási -Albert model is an algorithm developed by Albert-László Barabási and Réka Albert, which has the special property of generating scale-free networks [25]. These networks are generated using a technique called 'preferential attachment' which is a commonly observed property of network growth. Preferential attachment is the process by which high degree nodes (very sociable people) are more likely to be attached to new nodes. The phenomenon of preferential attachment was shown to appear in many networks such as the Internet, Science Collaboration graphs and the web of human sexual contacts [25]. When generating a Barabási-Albert network one must specify:

N, the number of nodes for the initial network

M, the number of connections a new node receives.

L, the limit on network growth.

The algorithm is a growth model, by which we start with an initial network and grow the network to size *L*. The initial network must be connected, so that resulting network is also connected. The simplest way to do this is to create a linear network of connections in the initial network of size N. After this, new nodes are introduced one at a time and given *M* unique connections. The algorithm consists of creating an initial, connected, network and then applying the preferential attachment function to each node until it satisfies the specified number of nodes. This model can be very slow to generate graphs due to the method by which new nodes are attached. For each new node we must assign *M* unique connections. The probability by which these connections are made is often miniscule which means it can take a significant time to find enough connections for each new node. The probability is as follows:

$$p_k = \frac{k_{degree}}{\sum_{j=0}^{n-1} j_{degree}}$$

Above shows that the probability p_k of creating an edge between a new node and node k is equal to the degree of node k divided by the sum of the degrees of all other nodes. Essentially this is the ratio between edges in a node and the edges of the whole network. This reveals that if we have many nodes, edges or both we will have a very small probability of creating an edge for each node unless the node has a very significant degree. For example, if we have 300 nodes (a relatively small network) with an average degree of 7 and node k has degree 15, then p_k would be:

$$p_k = \frac{k_{degree}}{\sum_{j=0}^{n-1} j_{degree}} = \frac{15}{2100} = 0.007 \ldots$$

The above example shows that even an above average node in a small network has a probability of just 0.7%. The algorithm picks nodes at random and then creates an edge according to the probability, this continues until there are M unique connections. This means that it can take a long time to make connections for each node as the probabilities are small and it also means that as the network grows it will become even slower as the probabilities decrease. To get around this we can use a similar approach to the NetworkX implementation in which they create a list of repeated nodes and chose randomly from that list [38]. Every time a node is picked for an edge it is (re)added to the list. This way the more times a node is picked, the more it will appear in the list and the more likely it will be to be picked again, yielding a scale free network.

Given the nature of the algorithm there is no way to dictate the degree of each node, which often leads to a single node having an absurdly large number of connections. This is problematic as it suggests that it is possible for one node to be in contact with most people in the network. If we have the perspective that the network represents the set of contacts in a single day, then this it is highly unlikely one node interacts with all others. Therefore, it is necessary to impose a cut-off on the degree of nodes. This preserves the scale free property however increases the number of high degree nodes. For this algorithm (and other scale free networks) the cut-off has been set to 30. The reason why I implemented the algorithm myself instead of using the NetworkX implementation was so that I could impose a cut-off which is necessary for my project.

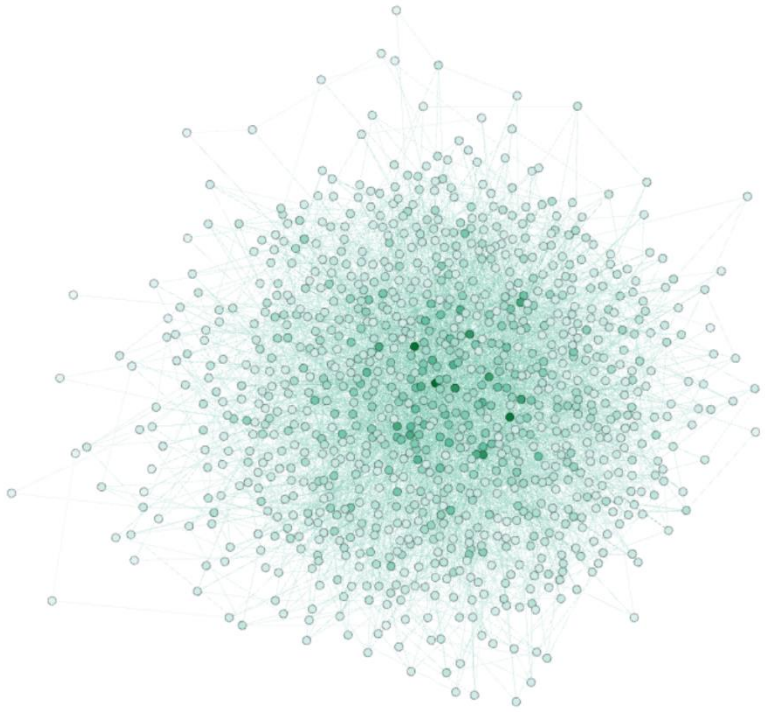


Figure 7 - Network Generated using Barabási-Albert algorithm with N = 200, M = 4, L = 1000 coloured according to a green degree gradient

In figure 7 a Barabási-Albert network is shown for 1000 nodes. The nodes are coloured according to a gradient such that deep green nodes represent a high degree and lighter nodes have a small degree. Using the gradient information, the scale-free aspect can be observed as there are many light-coloured nodes with few connections and a very small number of dark nodes that have many more connections.

7.1.3 Implementing a Watts-Strogatz Network

The Watts-Strogatz graph generation model is a great model for producing small-world properties, clustering, and short average path lengths [27]. After reviewing the properties of common Erdős-Rényi graphs they concluded that they do not contain the aforementioned properties, so they developed a model that does. The model, proposed by Duncan Watts and Steven Strogatz, is a two-step algorithm that takes inputs:

N, the number of nodes

K, the mean degree

B, the probability of rewiring an edge.

The algorithm begins by creating a ring lattice structure. This structure is formed by organising the nodes in a ring and connecting each node to *K*/2 closest neighbours in the ring on either side. This way each node receives *K* (or *K*-1 nodes for odd *K*). For example, given $K = 4$ and $N = 100$ then node '0' would gain edges: (0,1), (0,2), (0,98), (0,99). Figure 8 shows the resulting network from step 1 of the algorithm.

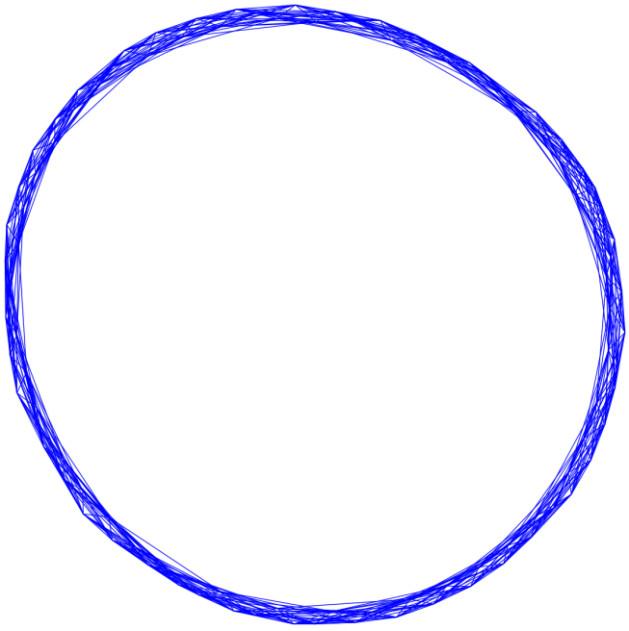


Figure 8 - Initial Ring Lattice formed in step 1 of the Watts-Strogatz algorithm using inputs N = 300, K =15

From this figure the lattice can be partially observed. The ring has a large width due to the relatively large *K*, and the ring structure is obvious. After step one a process called rewiring takes place. Rewiring is a process by which for each node we rewire its rightmost neighbour at probability *B*. If an edge is to be rewired, then the edge between the node and its rightmost neighbour is removed and the node gains a new connection with a new (randomly chosen) node in the graph. The aim of this second step is to reduce the distance

in the graph between nodes that are far away. In Figure 9 the rewired nodes and connections have been highlighted in green. The original ring lattice is still observable although it is now tangled. The rewired connections pull far away parts of the graph closer together and resultingly generates small world properties.

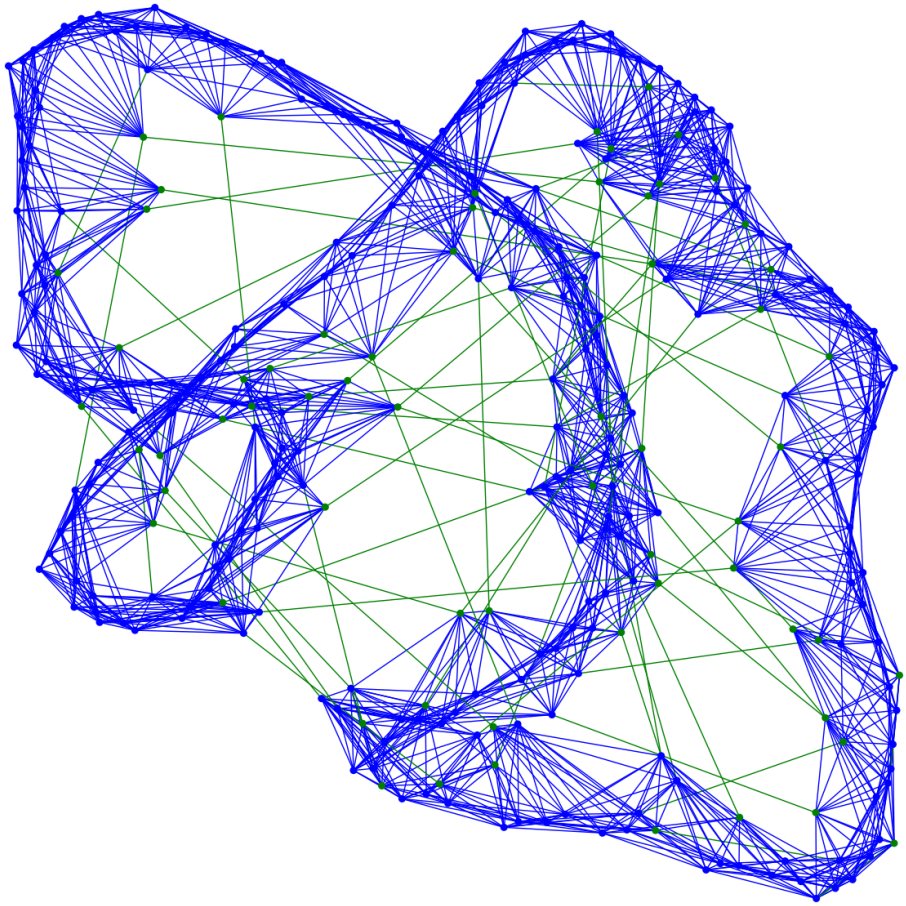


Figure 9 - A graph produced by the Watts-Strogatz algorithm with N = 300, K = 15 and B = 0.25. Green nodes and edges represent those that have been re-wired.

It was not necessary to introduce cut-off for this generator as the user can specify the mean degree and it does not exhibit scale free properties. I implemented this algorithm myself to be able to show the steps taken to create the graph as well as to implement the lattice creation in a slightly different way. My implementation always favours the clockwise direction in a graph with an odd mean degree, as opposed to the left side or a mixture.

7.1.4 Implementing a Random Psuedofractal Network

The Random Psuedofractal Network (RPN) uses a simple recursive growth rule to produce networks with both scale free and small world properties. The model, developed by Wang et al is a variation of pseudofractal network growth [39]. Pseduofractal networks and consequently RPN are edge focussed rather than node focussed like most models. A pseudofractal network is generated by adding a new node for each edge at each step of growth. For each edge in the existing graph the nodes at either end of the edge are connected to a new node. Figure 10 usefully shows the first three stages of such growth as it is easy to perceive new nodes and the edges they correspond to.

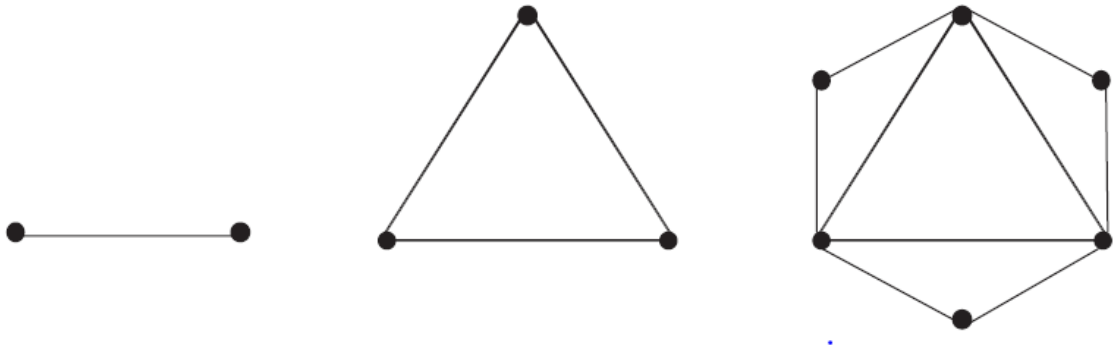


Figure 10 - The first three phases of a pseudofractal graph produced by Wang et al [39].

RPN's are similar however at each step in an RPN we randomly pick an edge in the graph and attach a new node to the existing nodes at either side of the edge. The graph starts as a dyad of two nodes. The scale free properties arise as the early nodes are more likely to have more edges, as they have more edges, they have a higher probability of being picked due to being present in several edges. This leads to a rich get richer consequence. Figure 11 clearly shows that a small number of nodes have become hubs with many connections whereas most other nodes only have a few connections.

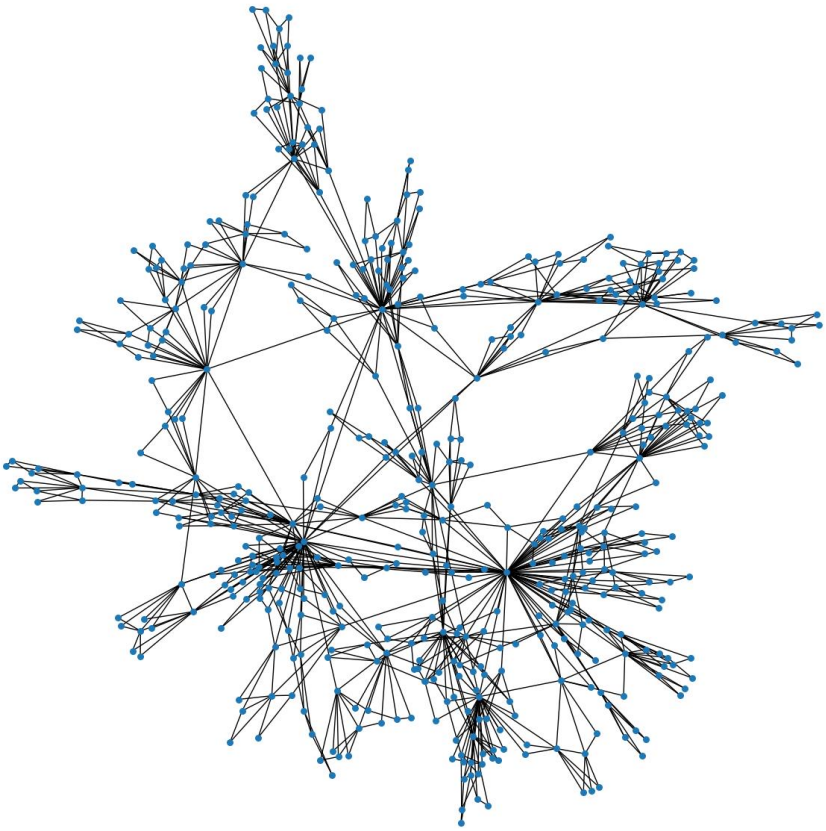


Figure 11 - RPN generated graph with 500 nodes

These graphs also have very high average clustering due to the nature at which nodes are added to create triads. The model is intended to only take one parameter, being the desired number of nodes for the graph. This however is not very useful for research purposes and as such I have adapted it so that one may specify the number of edges (*e*), a new node will form a triad with. This means that the exact number of edges a node inherits is variable with a maximum value of *2e* and a minimum value of *e + 1*. This is because the set of randomly sampled edges may contain edges that include the same node (which is probable in a scale free network). A notable aspect of this alteration is that as the graph begins with a single edge between two nodes, new nodes must inherit less edges until the number of edges exceeds the specified value. A further modification of this algorithm was to introduce cut-off as it exhibits scale free properties. This cut-off is set to 30 also. I had to self-implement this algorithm as it is not provided by NetworkX and I had to amend it to include extra parameters and cut-off.

7.2 SIR Model

The model for the simulation itself is perhaps the most crucial aspect of the whole project. It is hard to balance each of the policies such that they can be specified dynamically and interact without conflict whilst also achieving expected behaviours. When modelling the policy, it can be useful to implement them so that they exhibit correct behaviours for extremes which should help intermediates generalise.

7.2.1 Epydemic Framework

The eventual model that includes policy will be extended from the generic SIR model provided by Epydemic. It is therefore useful to explain how Epydemic works before detailing the extensions I have implemented for the policies. As pointed out in the tutorial provided for the library the process can be split into three sequential steps [40]:

- Providing a network
- Running the Simulation
- Retrieving the results

The point of focus is running the simulation. To run the simulation a *process dynamic* is used which is the simulation approach. The process allows for events to be posted according to time or probability, where events are functions that generally changes the compartment of a node. In this case, they may also change the topology and data structures surrounding the network. This means at any interval an event may be instigated due to an event scheduled for that time or if the random event criteria are met. The process works on a compartmented model which in this case is the SIR model. The process continues until the number of infected individuals reaches 0 and the simulated epidemic has ended. Due to the way Epydemic policies must be modelled using events that 'fire' when the correct criteria are met.

7.2.2 Implementing Lockdown Triggers

It is necessary to have triggers or 'levers' that may instigate a lockdown. Daily cases and the rate of reproduction are inordinately discussed statistics worldwide and are often used to back up policy decisions. To model these triggers there must be implemented structures to track and measure each statistic as well as implemented logic that dictate when lockdowns are instigated. For each trigger, the user will be able to specify a threshold above which the model will lockdown. After the lockdown if the specified statistic falls back beneath the threshold the model will 'open up" again, and the original network will

be restored. Both triggers can be specified in a single model however the user is provided no logic control. The logic of a dual lever system is that either lever may triggered a lockdown. If a lockdown is triggered due to one statistic rising above the specified threshold, it can only be removed if that same statistic falls back beneath the statistic. For example: a user specifies thresholds $R_t = 0.8$ and *Daily Cases = 2%,* at a particular interval t, $R_t > 0.8$ and $Daily\ Cases_t < 2\%$, then the subsequent lockdown can only be removed if $R_t < 0.8$ at a later interval t regardless of daily cases.

Rate of Reproduction

As explored in the context survey, within contemporary infrastructure it is impossible to gauge a perfect measure of R_t. In our model however this is possible. To track the rate of reproduction we must use the right data-structure to be able to store, access and edit the infections quickly and logically. As Epydemic does not track who infects whom but rather who is infected at any one point it is necessary to implement a new data structure for this. To do this I chose to use a python dictionary as it is a hashmap which means it has good time complexity.

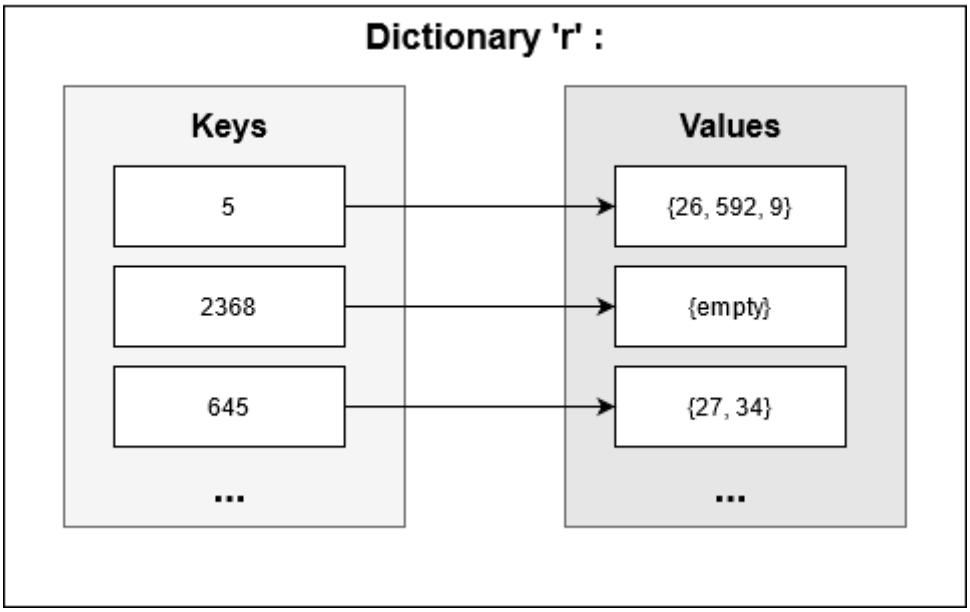


Figure 12 - A visualisation of the rate of reproduction dictionary

The dictionary is implemented so that each key is an integer representing a unique node in the graph. Each node in networkX is already represented this way which makes this approach intuitive. The value for each key is a set of nodes that have been directly infected by the key node. This means that the rate of reproduction at any one instant is the average length of the values set. A visualisation of the dictionary can be seen in figure 12.

To create this data structure requires constant updating and maintenance. Upon initialisation it is empty and is unable to track the original infections chosen by Epydemic. When an infection event takes place two actions must be performed on the dictionary. The first is to add a new key to the dictionary with the newly infected node value as the key and an empty list as its value. Secondly the node that makes the infections key is retrieved and the node it has infected is appended to its value set. When a node is moved to the *removed* compartment its key-value pairing is removed so

that the dictionary represents the rate of reproduction at each time frame. Therefore, over a simulation run it begins empty, grows as the infection spreads, and then wanes until empty as the epidemic ends. Although there is considerable error in the measurement produced in the reality, I decided not to model this was an unattractive extra degree of freedom. The model however is designed to access old rates of reproduction such that it mimics the delays caused by reporting. Finally, the rate of reproduction can only be accessed if the number of cases rises above a small portion of the network. This prevents a poor behaviour that was caught during testing where the initial infection node(s) made several infections leading to nonsensical values of R_t.

Daily Cases

Daily cases are a simpler problem. Finding the daily cases in this system can be done using a much simpler data structure than for Rate of Reproduction. To track and measure this statistic a simple list is required where the index is an integer time interval, and the values are the number of infections that occurred at that time frame. Thus, when an infection takes place, we use the time of infection to find the appropriate list entry and increase the number of infections for that day by 1. This statistic requires less maintenance as old values do not need to be removed. When using this statistic, no minimum number of infections is required for behaviour as this is essentially the statistic. This statistic has more volatile behaviour however and as such it is necessary to impose greater lockdown easing criteria. As infections are random it is possible (and common) to have a day where the infection rate is lower than average which would trigger lockdown easing. To fix this volatility the model checks that all reported cases in the past two weeks (14 intervals) are beneath the threshold to assert that the daily cases has fallen truly.

7.2.3 Implementing Lockdowns with Varied Obedience

One of the requirements was to introduce an aspect to the model that represented lockdown obedience. I have implemented three separate forms of lockdown in the network. The first is a uniform lockdown, in which every node loses the same percentage of edges. The second is a varied lockdown in which each node is given a random percentage of edges to lose, where the percentage is draw from a normal distribution. The final implementation is a varied lockdown in which each node is assigned a community/cluster via a community finding algorithm and then each community will be assigned a percentage of nodes to lose, where the percentage is also drawn from a normal distribution. Logically for the modelling a lockdown trigger must be used in conjunction with lockdown policies. Furthermore, each lockdown policy is independent of the others and only one can be used in a simulation as it is illogical to impose two lockdowns at once. When a lockdown is imposed the model tracks all the removed edges. A lockdown is removed when the trigger criteria is no longer removed (e.g., R_t falls beneath 1) at which point the original topology is restored using the tracked edges.

Uniform (standard) Lockdown

In the rudimentary case where only a lockdown trigger is supplied but no obedience measure is specified, a strong standardised lockdown should be imposed. There are a few ways this could be achieved however I picked the implementation based on speed and effectiveness. For a lockdown to be effective (strong) the topology needs to be significantly reduced in terms of number of edges so that if it were in place for most of a simulation the spread would be minimal. Given the desired behaviour I decided to remove 90% of edges in the graph. This is a significant number and should have a large effect on graphs with

a reasonable number of edges. This number is also suitable as lockdown are not the same as isolation and therefore essential contact is still preserved such as essential workers, shopping, exercise etc. The most efficient way to remove edges is to retrieve the list of edges in the graph and select a random sample of the specified size. Each edge in the sampled list is then removed from the network. These edges are stored in a list so that when the lockdown criteria are reversed, they can be reintroduced to the network.

Individually Varied Lockdown Obedience

Another requirement of the project as to model lockdown policy such that public obedience is not uniform. As reported by media, many individuals have been caught contravening lockdown rules, sometimes intentionally and other times through misinterpretation. To model this phenomenon each node in the network must be assigned a value that represents how strictly they obey lockdown law. The value should be picked such that there is wide variance and so that the variance can be measured or characterised. This value should then determine how many edges they lose. The easiest option would be to pick the number of edges completely randomly, however this would not be very repeatable or measurable. There would be no way to impose a skew on the variance (toward more or less obedience) and the randomness may lead to erratic results. The value is thus chosen from a normal distribution where the user may specify a mean between 0 and 1, and the distribution variance is always set to 0.1. The normal distribution creates a better chance of having consistent results as well as allowing the user to specify a skew on the variance, to model more obedient or less obedient societies. Logically if a distribution is specified such that values outside (0,1) are picked from the distribution then they are rounded to the nearest limit. As shown by figure 13 the deviation is suitable as when the middle value is picked the distribution spans the whole set of values however the majority will be picked from the centre of the distribution. I found this to be a good trade-off in repeatability, control, and variance.

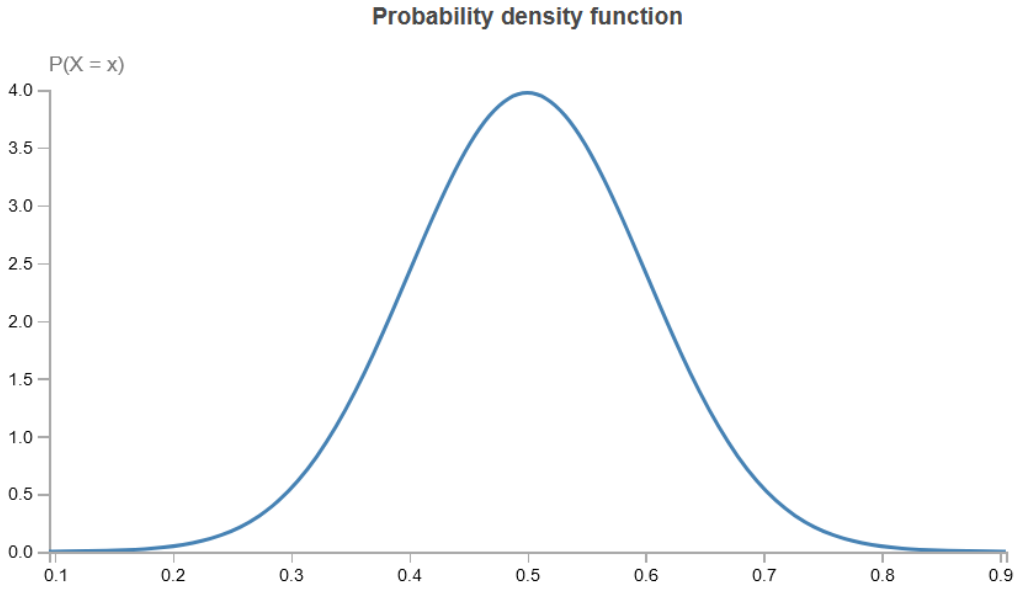


Figure 13 - Normal distribution N(0.5, 0.1) [41]

The effect of an individually varied lockdown is shown in figure 14. The figure is coloured according to the community finding algorithm used for clustered obedience. This is to help

contrast the effect of the lockdown against figure 15. Figure 14 shows that when variance is picked based on individuals then there are a few notable effects. Some nodes are completely isolated, some are a part of a small group of isolated nodes and a large portion are still connected in a group with severely reduced connectivity. The proportions of these groups change depending on the distribution shape.

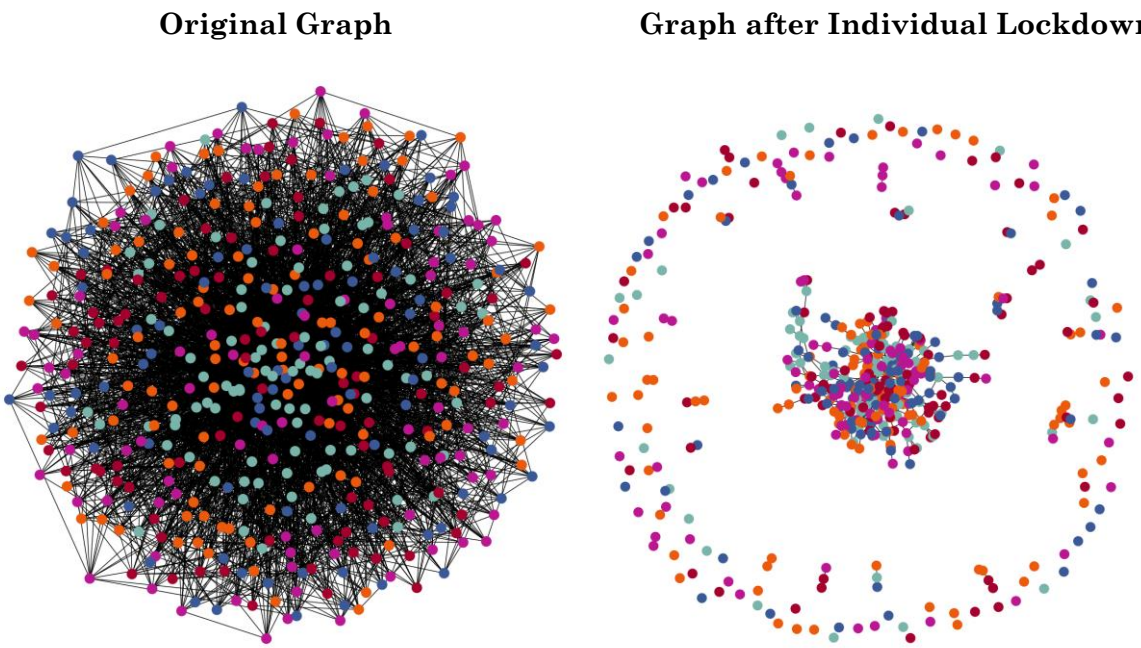


Figure 14 - Network of 500 nodes, coloured by a split into 5 communities using the fluid community algorithm. The right graph shows the original network after edges are removed using the individual-based lockdown with normal distribution (0.7, 0.1). The right graph shows the original network after edges are removed using the individual-based lockdown with normal distribution (0.7, 0.1). The colouring shows that communities are not preserved, and nodes are likely to be very disconnected or connected to a sparse group after an individual-based lockdown in contrast to figure 15.

Community Varied Lockdown Obedience

As discussed in the context survey mass gatherings of people who do not social distance have arisen. This suggests that the lockdown obedience may also be well modelled in a community sense, where some communities are stricter on rules than others. Modelling this phenomenon is more complex as it has the added complexity of finding communities. Finding a community is tough as it has a loose definition and nodes may belong to multiple communities. Furthermore, many community search algorithms are exhaustive and have poor asymptotic time complexity. Given the context of the problem, an algorithm is required that will split the graph into a desired number of communities quickly. It is more important that each node is assigned a unique community quickly than that the communities identified be optimal.

Fluid Community Algorithms

The algorithm chosen to perform the community split is the fluid community's algorithm [42]. The algorithm is named after the way fluids interact in a shared environment. The algorithm starts with k single node communities. The algorithm then iterates over each vertex updating the community it belongs to using an update rule. The algorithm converges (finishes) when two full iterations have taken place and no nodes have changed community. This algorithm is extremely advantageous as it is extremely quick, and it allows specification of the number of communities to be found.

Original Graph **Graph after Community Lockdown**

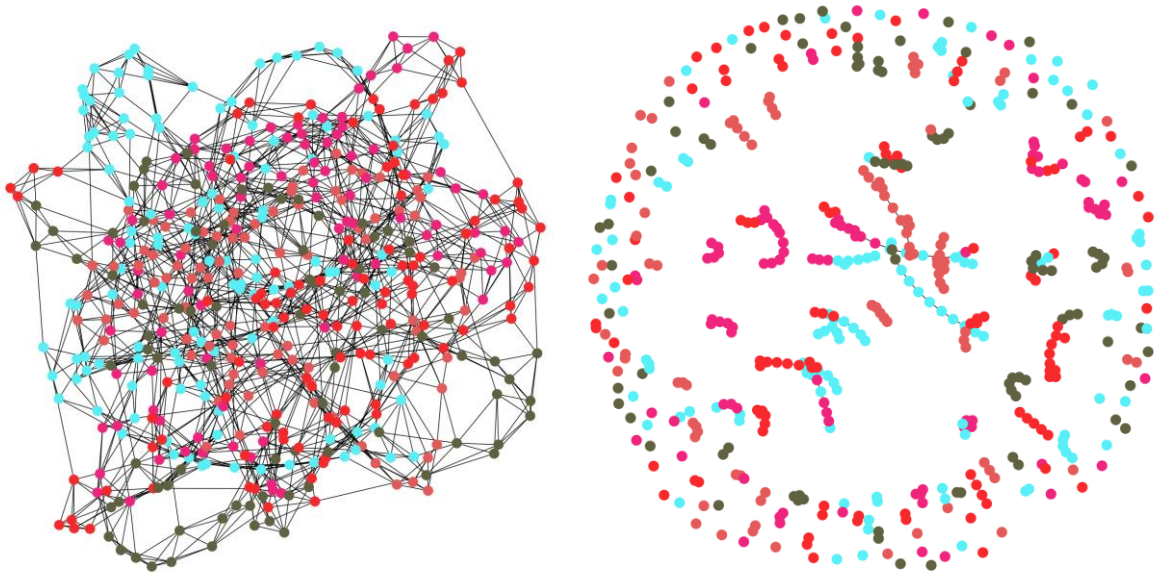


Figure 15 - Networks of 500 nodes coloured by a community split using the fluid community algorithm. On the left is the normal network supplied to the model. The right graph shows the original network after edges are removed using the community-based lockdown with normal distribution (0.7, 0.1). The colouring shows that communities are preserved, and nodes are more likely to be connected to other nodes in their community after a community-based lockdown in contrast to figure 14.

The implementation of the community-based obedience can be split into a few components. Firstly, the graph is split into communities. The number of communities is dependent on the graph size. A specific paper on human communities found that communities of size 50, 150 and 500 are disproportionately more common and have greater longevity [43]. Based on these findings, I decided to split small graphs (less than 5000 nodes) so that the communities would have roughly 50 nodes and large graphs so that the communities would have 500 nodes. The difference in size is necessary to produce sufficiently many communities while also preserving speed as the greater the number of communities required the longer the algorithm may take. The graph is split into communities once at the beginning of the simulation. When a lockdown is triggered, a similar approach is taken to the individual obedience model. Each community is assigned a value (percentage) that is drawn from a normal distribution. This percentage represents the number of edges to be removed from that community. The normal distributions mean is specifiable, and the variance is set to 0.1 as before. The effect of a lockdown using the community perspective is shown in figure 15. In this figure the communities are tangled in the original graph, however after a lockdown is triggered the communities present themselves more clearly. Some of the nodes end up completely disconnected, some end up in homogenous community groups and a few larger inhomogeneous groups have formed where separate communities are still identifiable. Generally, the communities tend to fracture away from other communities more than their own. The distribution used to create figure 14 and 15 is the same and as such they are comparable. When comparing of the two figures, the two methods prove to be quite different. The community method leads to separation into many disconnected groups rather than one large sparse group as in the individual method.

7.2.4 Implementing Isolation and Tracing

Isolation and contact tracing can be an effective way to stunt the spread of a virus. In the implementation isolation and tracing will be treated as separate events. Isolation is an independently specifiable event however tracing is dependent upon isolation as it does not

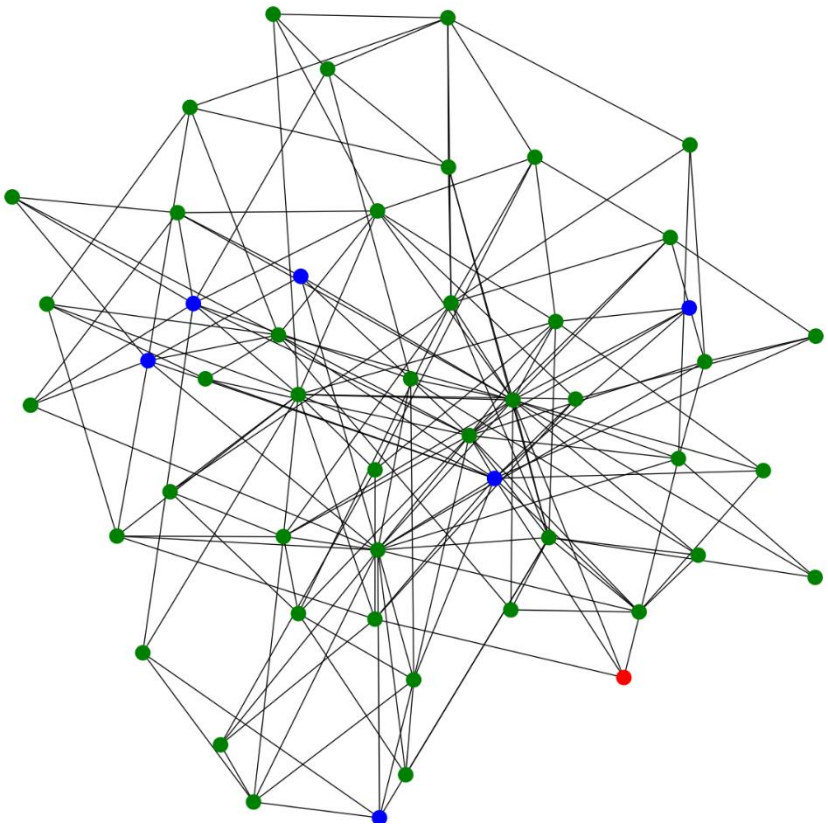


Figure 16 - An example of contact tracing. The red node represents the new infection, and the blue nodes are the traced contacts.

make sense to have a tracing system if the initial contact is not found to be a confirmed case (and thus isolated).

Isolation

Isolation is the state of being removed from a network due to being a confirmed case. The isolation model is not behaviour focussed as we assume that people are more obedient if they are a confirmed case. Therefore, the specifiable value and the explorable phenomenon is that of how many cases are caught (via symptoms and tracing). When specified in a model an isolation event is posted after an infection event. At the beginning of the event using random generators and the specified probability each node has a chance of becoming a confirmed case. In my implementation if a node is a confirmed case, then they lose all edges. After a node is confirmed a tracing event is posted in the future. If the node is not a confirmed case, then nothing happens.

Tracing

The specified tracing value does not infer how many edges each traced node loses but rather the percentage of nodes that are traced from an infection. This means that each traced node loses all its edges however not all nodes in contact with the infected are traced.

7.2.5 Implementing Timings and Parameters

With all the policies implemented the final aspect of implementation for the model is timing which shall tie them all together. Each modelled policy is a posted event. Since each posted event can be posted at a future time interval this can mimic the aspect of delay. For example, when a node becomes infected there is delay before symptoms present, and infectivity begins between infectivity and symptom onset. Thus, to model an infection being caught, we must post an isolation event suitably far ahead that it represents this period. To implement timings for the COVID-19 model they need to be based on evidence from research as well as consistent with one another. The first step is to decide what a single unit interval t represents. It must be sufficiently small so events may be posted at integer intervals while not being too small to overload the model and slow it down. As such I decided that each interval would represent a day, all the posted events will be at least one day later and as such this is suitable.

SIR Parameters

Firstly, the parameters of the SIR model must be specified. This is an incredibly difficult decision due to several factors. Primarily there is large variance among results in the literature. Secondly there is disparity in the complexity of these results and the simplicity of an SIR model. Below is a list of values that after combing through the literature are commonly reported and near the mean of reported values.

Table 1 - A table of COVID-19 characteristics taken from the recent research

Covid-19 Characteristic	Value
Incubation Period	Average of 5.1 days [44]
Infectivity Start	Mean of 5.8 days [45]
Infectivity Peak	2 days before symptom onset [45]
Infectious Period (Symptomatic)	Average of 13.4 [46]
Infectious Period (Asymptomatic)	Average of 9.5 [46]
Percentage of Cases That Are Asymptomatic	33% [47]
R_0	2.87 [48]

Given that when a node becomes infectious in SIR, they are immediately infectious, this creates a discrepancy between reality and the model. Therefore, we must account for this in our model. However, there are also discrepancies between findings shown in table 1 in incubation period and infectivity start. This illuminates the lack of agreement between research and the variance in effect on different populations. As such I will not account for the incubation period due to the lack of agreement in research and the difficulty this presents for a simplified model.

Probability of Removal

The infectious period of a person differs based on whether they present symptoms. The SIR model however can only use one value to generalise both scenarios. Initially we must create a unified infectious period. To do this I took the weighted average of the symptomatic and asymptomatic period:

$$Infectious\ Period = 13.4 \times \frac{2}{3} + 9.5 \times \frac{1}{3} = 12.1\ldots$$

This value can then be used to create a probability of removal (probability that a node is no longer infectious and removed from model). The probability of removal is:

$$P\ Remove = \frac{1}{12.1} = 0.0826$$

Probability of Infection

The probability infection is given by the rate of reproduction which has probability 0.0287.

Probability of Initial Infection

This probability is dictated by me and determines the proportion of initial infections. I choose for 1% of the population to begin infected. This is so that without any policy the virus will not die out and infect most nodes. This is so that simulations are not skewed by a proportion where the virus is eradicated immediately.

Timings

Given that a time perspective has been set it is essential to implement the timings for each of the event posting. As the model can track its own state in at every interval, events are posted in the future to mimic a delay. Table 2 shows the delays and the implementation reasoning.

Event	Time Delay	Reason
Lockdown	7 days after lockdown threshold is met	7 days was chosen as it reflects a delay in reporting as well as time for a government to make and implement the policy.
Un-Lockdown	7 days after lockdown threshold is no longer met	
Isolation	5 days after infection	This was chosen due to characteristics in table 1
Tracing	1 day after isolation of a node	This assumes quick tracing mimicking an individual informing their contacts and a government tracing team
Undo Tracing (Release a node from a tracing induced quarantine)	14 days after tracing quarantine	This is chosen to be a suitably long time to mimic the maximum time it could take to develop symptoms

7.3 Figure Builder

As several simulations are being run in a single experiment, a suitable method of reporting the results must be found. Given that the experiment lends for one or more values to be scaled the results can be presented as a line chart that shows how the outcome

of a simulation changes as the parameter(s) of interest change. The results of a single simulation are the three compartments' values after the epidemic. As the simulations end when the 'Infected' compartment is empty this means the results are the 'Susceptible' and 'Removed' compartments, representing how many did and did not get infected, respectively. The figure builder takes a series of data for these model outcomes and plots a line for the susceptible and removed compartments showing how they change as the parameters change. Given that a maximum of 5 different variables could be scaled in a single experiment this presents a problem for plotting. Matplotlib allows for two x-axes to be shown (which is reasonable), so in the worst case we must pick 2 from 5. The figure builder works by only considering parameters for an incremental value greater than 0 (the parameter is being changed). In the case where every parameter is not incremented the results are plotted against the iteration number. Therefore, if there are 2 or less incremental values for specified policies we simply plot against those policies. In the case where there are more than 2, there is a scale of precedence:

1. Rate of Reproduction
2. Daily Cases
3. Lockdown Obedience (Individual)
4. Lockdown Obedience (Community)
5. Isolation
6. Tracing

As policies 3 and 4 are incompatible we can have a maximum of 5 potential x-axes in a singles experiment. Figure 17 shows an example of plot that has two x-axes. An interesting aspect of the figure builder is that passing the image via eel was too convoluted and, in the end, I had to work around this by saving the plot as an image to a specific location which the frontend then reads it in from.

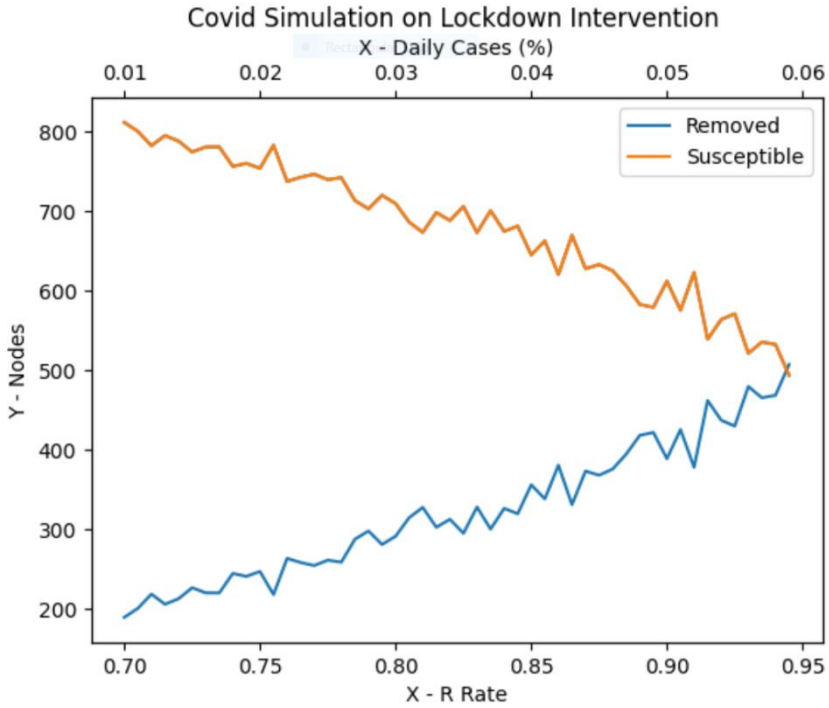


Figure 17 - An example of results produced after an experiment with two x-axes. As the rate at which governments instigate lockdowns increases, the number of removed nodes (those who've been infected) increases.

7.4 Testing

A comprehensive summary of the tests carried out is attached in appendix 4. Testing of a model that involves so many degrees of randomity can be hard as we cannot expect many values. As such I had to take an unusual approach to certain parts of the test suite to ensure the model worked as expected. The tests are split into unit and integration tests.

7.4.1 Unit Tests

Unit tests are the backbone of a test suite that ensure individual functions and modules work correctly on their own. These tests are implemented using pythons unittest library and can be found in the test directory. I have implemented unit tests for each of the network algorithms, the figure builder, and the SIR model. These tests assert the validity of basic algorithms and functions. These unit functions are unfortunately slightly higher level in parts than I would like due to the nature of the project. Correct behaviour of modules and functions that create the networks do not have expected exact outcomes but rather expected behaviours. As such due to randomity some unit tests assert that the outcome obeys an expected behaviour rather than an expected value. Extensive testing is applied to the SIR model however the setup required for these tests is unusual and extensive. Due to the way Epydemic works, the model functions can only be called after extensive set up. As such it is quicker and easier to run a normal simulation to set up the model and then call the functions after setup to test, they work as units.

7.4.2 Integration Tests

Integration tests assert the combination of units to satisfy that the individual parts work together. The integration tests I have implemented test simulations of the model, exposed methods, graph generation and simulation behaviour. These tests satisfy combinations of units as well as mimic the way in which users interact with the software. Testing the behaviour of simulations is an important aspect of the test suite as it ensures that the model is calibrated well and performs as expected in polar extremes. These behaviours are such as that under complete obedience and comprehensive policy the virus should have minimal spread. If the model behaves as well in extremes this is positive as it asserts that we will have tipping points in between which is important for the objective of the project.

8 Exploring the effect of obedience on lockdown policy.

8.1 Aim

The aim of the experiment is to find/show the relationship between lockdown policy and public obedience. The purpose of this is to be able to assess what type of disobedience is more reductive (community based or individual) as well as what level of obedience it takes to contain a virus and at which level government efforts become redundant.

8.2 Method

To explore this relationship, it is important to utilise sensible values for the lockdown levers. The lever(s) should be such that if very good obedience were in place the epidemic would be contained. It should also not be so effective that the model is in a constant state of lockdown. The lever value is based upon findings of a separate experiment attached in appendix 2. The results of that experiment show varied outcomes for daily cases, whereas R_0 produced much more stable and consistent results across all the topologies. Based on

this information I decided to use solely R_0 as a lever for the experiment. The value I chose for R_0 in this experiment is 0.9. I choose this value because it is near the cross-over point for each of the topologies as well as it a realistic value that governments may use for intervention as it is near 1. This value is worse for random pseudofractal networks as it is just after the cross-over point and therefore this will need to be accounted for in the conclusion.

It is important to analyse a large range of values for obedience. I will therefore use an obedience range of 0.3 to 1.0, with increments of 0.02. Each interval will be repeated 10 times, inferring 350 simulations per test run (to produce each graph). Each test will be repeated on individual based obedience and community-based obedience.

The experiment will be repeated on each of the four network generators. I fix the mean node degree to 10±1, this means for each graph the mean degree may be between 9 and 11. Secondly the graph size is fixed to exactly 10,000 nodes.

8.3 Results

Erdős–Rényi Network

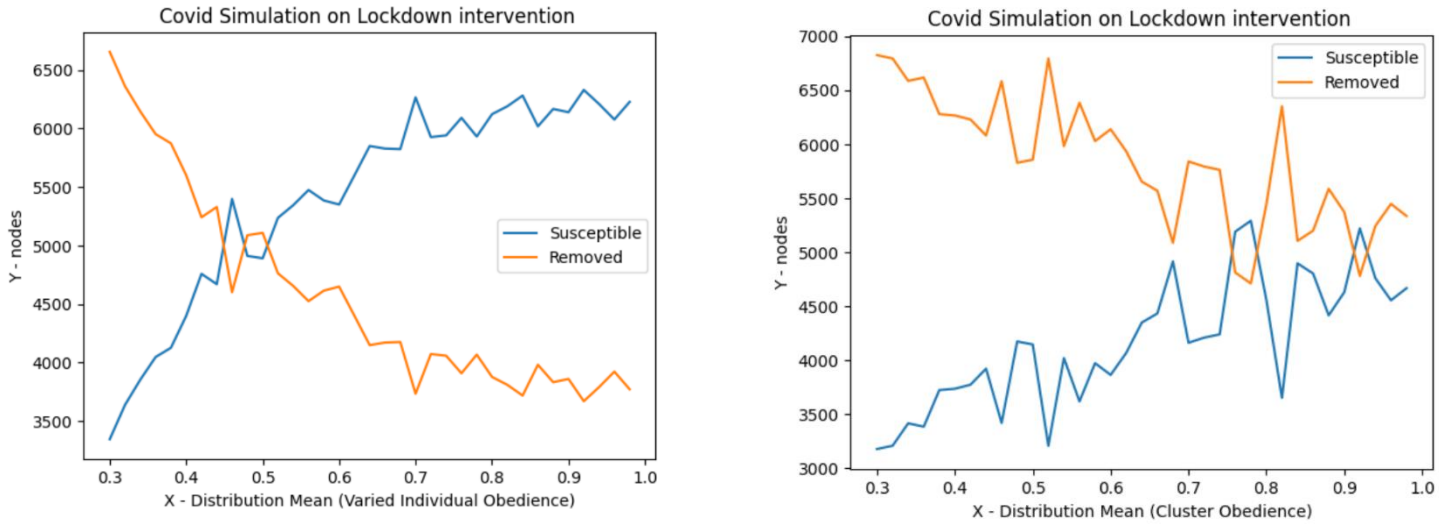


Figure 18 – Two figures depicting relationships between epidemic outcome and public obedience to lockdown. The left graph shows how the epidemic outcome changes as obedience to lockdowns increases on an individual basis. The right graph shows how the epidemic outcome changes as obedience to lockdowns increases on community basis. The network was created using the Erdős–Rényi with parameters: 10000 nodes, 10 edges (0.001%)

Watts-Strogatz Network

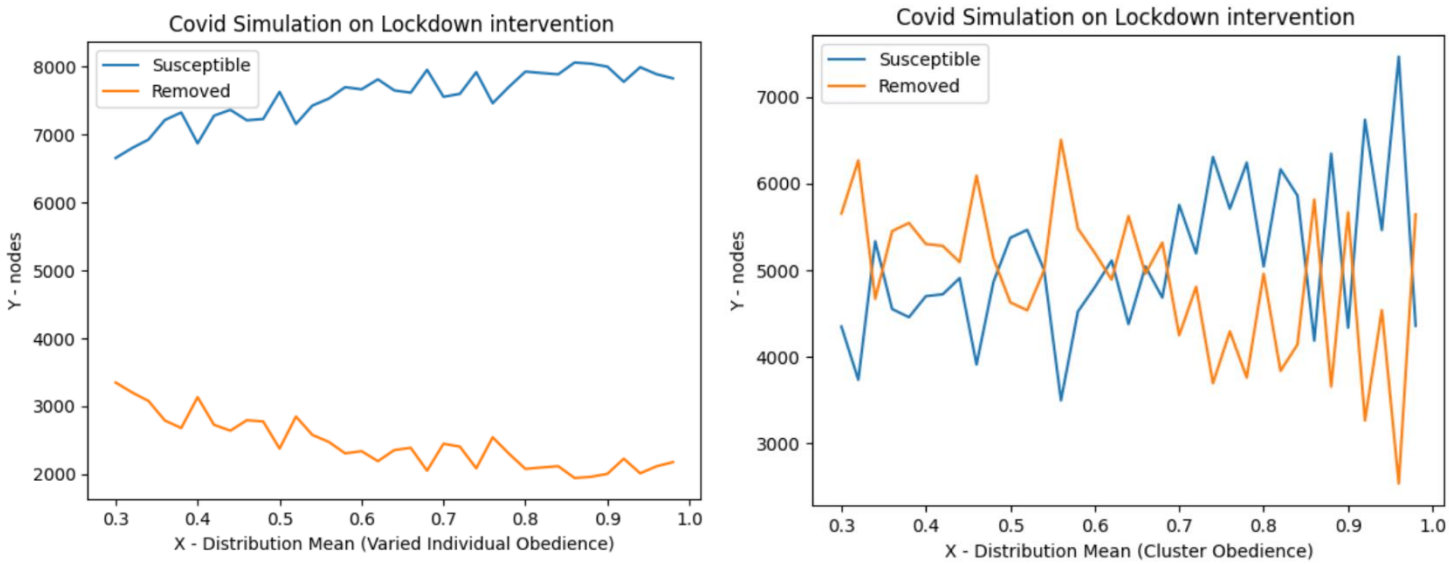


Figure 19 - Two figures depicting relationships between epidemic outcome and public obedience to lockdown. The left graph shows how the epidemic outcome changes as obedience to lockdowns increases on an individual basis. The right graph shows how the epidemic outcome changes as obedience to lockdowns increases on community basis. The network was created using the Watts-Strogatz algorithm with parameters: 10000 nodes, mean degree of 10, rewiring of 0.35.

Barabási -Albert Network

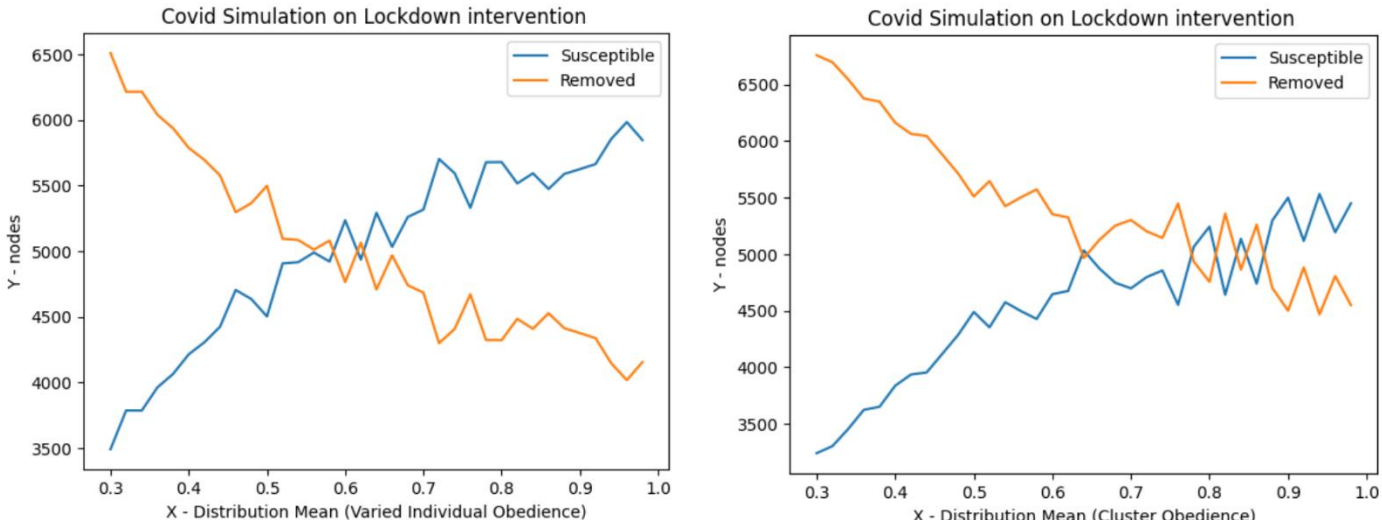


Figure 20 - Two figures depicting relationships between epidemic outcome and public obedience to lockdown. The left graph shows how the epidemic outcome changes as obedience to lockdowns increases on an individual basis. The right graph shows how the epidemic outcome changes as obedience to lockdowns increases on community basis. The network was created using the Barabási -Albert algorithm with parameters: 100 starting nodes, 5 edges for new nodes, 10000 node limit.

Random Pseudofractal Network

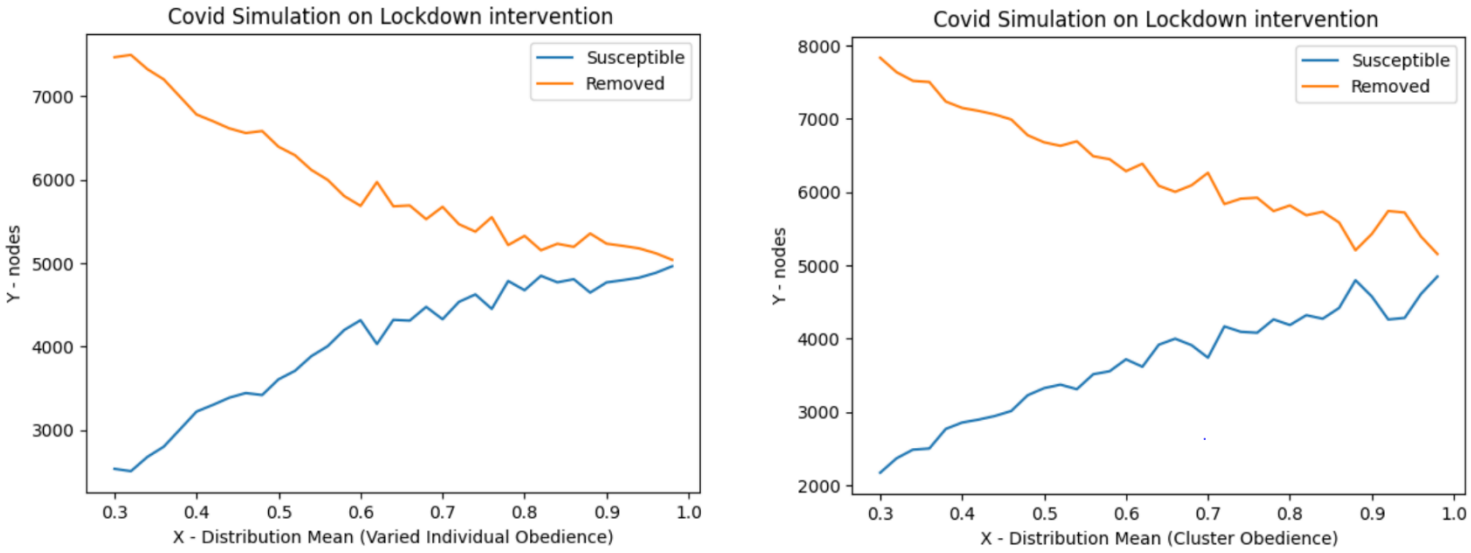


Figure 21 - Two figures depicting relationships between epidemic outcome and public obedience to lockdown. The left graph shows how the epidemic outcome changes as obedience to lockdowns increases on an individual basis. The right graph shows how the epidemic outcome changes as obedience to lockdowns increases on community basis. The network was created using the Random Pseudofractal algorithm with parameters: 10000 starting nodes, 5 edges for new nodes.

8.4 Interpretation of Results

Figures 18-21 show the difference in outcome between individually varied obedience and community varied obedience for each of the different network generators. Below is a list of interpretations for each figure (and thus topology).

Erdős–Rényi (Figure 18)

The two graphs show a clear difference between the different types of obedience. The individually varied obedience has a long cross-over point beneath 0.5. This suggests that in a network with no power-law and no small world properties (high clustering or average path lengths) that individual disobedience is not destructive to the outcome of the epidemic so long as it is not severe disobedience. The community-based disobedience is less accommodating. The graph clearly shows that for any value beneath 0.6 the epidemic is out of control. Furthermore, above this level the chart is extremely unstable which means the results were highly volatile, suggesting only extremely high obedience from a community perspective can reduce spread in this topology.

Watts-Strogatz (Figure 19)

The Watts-Strogatz results are the most polarised. The individually based obedience result is easy to interpret. The individually based disobedience appears to result in controlled epidemics for all explored values. The tails at the beginning of the graph suggest that only extremely high disobedience would lead to outbreak. This suggests that graphs that exhibit small world properties, in the absence of power-law characteristics, can be effectively controlled so long as obedience is on an individual basis. The results for community-based disobedience are much more difficult to interpret. Firstly, the graph is extremely erratic, this suggests that for all levels of the result was volatile. If a line of best fit were to be drawn it would suggest that moderate to high disobedience may lead to

loss of control, though not a complete outbreak. This data suggests that high obedience in a community sense would also lead to control.

Barabási -Albert (Figure 20)

These graphs have very similar structure. The individually varied results for this topology show that crossover section between 0.5 and 0.7, suggesting that for power-laws, small individual disobedience is okay, but moderate to high disobedience on an individual basis can undo the efforts of lockdown policy. The community disobedience graph shows more severe results. As shown in the right-hand graph the community-based obedience reaches a plateau even at high obedience levels. The difference between the two suggests that community-based disobedience is more destructive than individual disobedience for power-law networks. As discussed in the methods section.

Random Pseudofractal (Figure 21)

Finally, the random pseudofractal graphs suggest that both forms of obedience are destructive and even suggest that full obedience will not contain an epidemic. This is false, however. As discussed in the methods section the intervention value for lockdowns was unfavourable for this topology and must be accounted for in the results. As such what we can infer is the difference between the two. Both graphs have similar structure however the community-based disobedience has a greater gap at high obedience levels. This suggests that community-based disobedience is more destructive than individual based disobedience in networks with power-law properties and small world-properties.

8.4.1 Conclusion:

Obedience in general

The set of graphs illuminate the need for obedience. Most of the graphs are very erratic, which suggests that the results are very volatile and could lead to varied outcomes dependent on chance. From 8 tests only 1 suggested that disobedience will not lead to loss of control. The exact amount of obedience required is hard to infer from this test, and even if it were inferable, it would be harder to translate into the real world. The results suggest that a minimum of 50% of contacts must be cut for lockdowns to have any effect, however by looking at the cross-over points, over 70% of close-contacts is more likely to lead to effective lockdowns. These values however are not significant.

Individual vs. Community

This experiment does provide interesting insight into different forms of obedience. From the results, in every case, community disobedience appears to have more adverse impact than individual disobedience for the same level of edge removal. In the model individual-obedience is measured as everyone losing a percentage of contacts taken from a normal distribution. The community-obedience is measure however, as each identified community losing a percentage of edges taken from the same distribution. In this sense for the same obedience level each network loses a similar number of edges (not exactly equal due to chance) however the way in which edges are removed are different. Translating this into real world terms, it is better to have everyone break lockdown rules individually than to have a mixture of more-obedient and less-obedient communities. Therefore, to curb the spread of a virus government could allow for reduced individual mixing in the hope of preventing community-based disobedience.

Topology

The last conclusion from this experiment relates to the topology and characteristics of the underlying networks and their relation to disobedience. The results show that networks that exhibit small-world properties (short average path lengths and high clustering) are more negatively impacted by community disobedience. Figure 19 suggests that networks with small world-properties and no power-law are almost immune to individual disobedience. Networks that exhibit power-laws appear to be very susceptible to either type of disobedience which is shown in figures 20 & 21.

8.4.2 Repeatability:

All the parameters used to create these graphs are provided I the method section and network specific parameters can be found under each graph.

9 Evaluation and Critical Appraisal

9.1 Original Objectives

The original objectives were to:

1. To create a compartmented SIR model for COVID-19 including obedience-based lockdowns and government lockdown instigation strategies.
2. To create software that allows diverse experiments to be run on this model.
3. To provide choice of base network to tailor the experiment further.
4. To incorporate other important/common policies in the model
5. To run an experiment that show the relationship between lockdown obedience, policy, and epidemic outcome.
6. To display the result of an epidemic in several formats

As displayed in the report, most of these objectives have been met. Unfortunately, due to timing issues I was not able to complete objective 6. Objective 6 is important as it intends to increase the number of options for a user so that more data produced by the simulation is exposed and retrievable. This is a weakness in the project, and it would have been improved by additional plot formats and options. The other 5 objectives however have all been met. The model I implemented contains two forms by which obedience can be measured as well as the two most common and reported statistics to guide lockdown instigation (R_0 and daily cases). The front-end I created allows the user to tailor their experiments in an open and unintrusive way. I have provided a lot of control to the user without an overwhelming set of options on display. The project currently offers 4 network generators, each producing graph with varied properties drawn from the literature. This is an advantage of the project that allows the user to mimic their population if they are aware of the population's characteristics. The report includes an experiment which details the effects of public disobedience, the difference in effect between the two types of modelled obedience as well as the effect topology plays in outcome. These three findings make the project a success. Most significantly I was able to find and show a clear difference in outcome between the two types of obedience. The project has a few other advantages such as including isolation and tracing policies and being able to run several experiments with different values all in one go. Overall, comparatively to the objectives, the project has been successful, yielding a detailed and diverse model, accompanied by a useful experiment set-up structure and an experiment with a promising precursor to further research on obedience and policy.

9.2 Comparison against similar works

9.2.1 Related Papers

The most relevant paper (previously discussed in the context survey) I could find was a study conducted by the Department of Public Health and Clinical Medicine in Sweden [5]. This study is similar in that it models lockdown adherence. This study and my project differ in their perspective on adherence/obedience, my project assumes obedience based on how many contacts are kept, their model however measures adherence as number of hours spent outside. This paper also chose to vary the base network; however, their network was modelled to represent households and as such the size of households was varied across simulations. Both perspectives have advantages and disadvantages. Modelling adherence as time spent outside does not make sense for colder climates and as such is specific to Italy which the study is based on. It also does not account for direct rule breaking and intentional close-contact socialising. My model, conversely, does not account for random infections that could be made in public spaces and only models close contacts that do not change throughout the model. The set of contacts a node has changes each day which is not represented. A criticism I would make of their model in comparison to mine is the lack of variance. My model not only incorporates different levels of obedience but also accounts for the fact that there will be variation in how much people adhere to rules. This paper assumes in each simulation a constant house size and a constant level of obedience. This does not reflect reality however it is simpler which increases reproducibility which is important. Finally, when considering result reporting this paper and my project are quite similar. Both provide a single method of result retrieval and they are both line graphs as shown in figures 17 and 22. The results of the paper however are concerned with secondary infections over a short period whereas mine reflects the result of a whole epidemic. Again, arguments can be made for both methods however each equally show if an epidemic is out of control or under control.

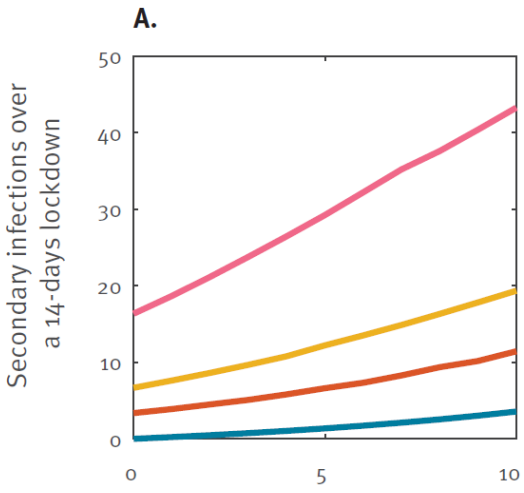


Figure 22 - A line graph produced by adherence-based study [5]

9.2.2 Related Software

It is also useful to compare against similar software. Specifically, software that allows you to specify SIR simulations. A similar project simulates the spread of various influenza virus' in Poland [49]. The comparison should be limited to the software itself rather than

any underlying logic and purpose-driven differences. This software has several advantages and disadvantages in relation to mine. In terms of epidemic input there are similarities and dissimilarities. As shown in figure 23, the input and results are all contained on the same page. This is not intuitive or attractive. My project presents a sequential set of steps for input and then a results page which in my opinion is a less cluttered and more readily understood. The actual input boxes themselves are near identical to mine and both projects autofill useful values which is good. There is also a lack of explanation on the page. My frontend provides descriptors when the mouse is hovering above an input which helps users understand the effect of each feature. There are also cases of poor design, such as when the simulation starts there are no infected nodes. To introduce infected nodes, you must click a button to add 300 infected nodes, meaning you cannot start a simulation with immediately infected nodes. Equally there are some advantages also to this software over mine. Firstly, the software provides real-time results with each simulation and a map visualisation to show where the epidemic is spreading. This is a useful and attractive feature of this software. This software also returns results in several forms. This is objectively better for the user however the implementation and presentation are imperfect, as the interface has become cluttered. This software also has a feature that allows the user to export the test history which is its greatest advantage over mine. My software has no mechanism to store results other than the last figure that was created. This figure is overwritten with each simulation.

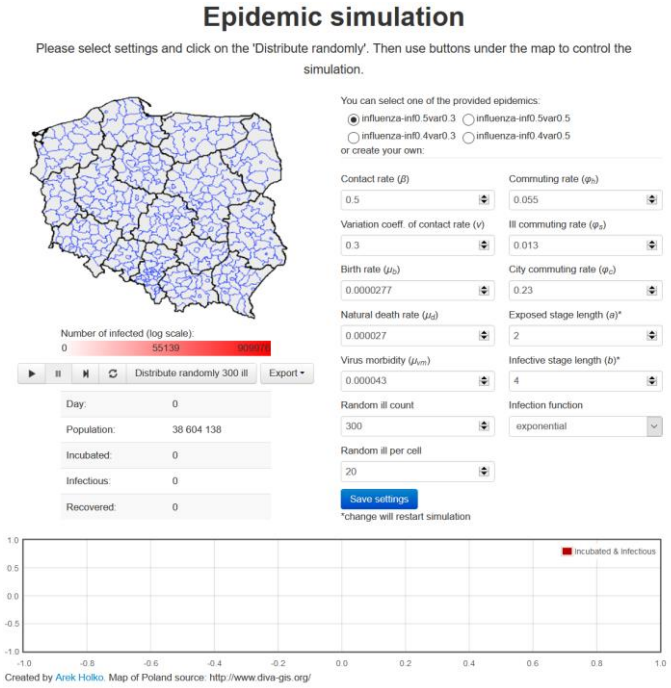


Figure 23 - User Interface for a similar epidemic simulation [49]

9.2.3 Related Tool

A more directly related tool is the impressive COVID-19 simulator, a collaboration between Harvard Medical School and Massachusetts General Hospital [50]. This online resource provides many tools. The most relevant of these is the COVID-19 Policy Simulator. This simulator is designed specifically to operate for the United States of America and therefore all values pertain to the USA. The following sections discuss and evaluate the differences between each model.

Parameters

The parameters of this model are more flexible than mine. As this simulator is specific the simulator can obtain the estimated effective reproduction rate for each population. This allows for contemporary results however does not allow room to explore a whole epidemic, but rather the course of a part of an epidemic which may not be informative for future epidemics.

Network Specificity

My model allows the user to build their network to mimic any assumptions they may have about a population. This simulator however uses live data to mimic specific networks. The benefit of my methodology is flexibility, which allows the user to (less accurately) model many populations. This model however models a small subset of populations more accurately. Ultimately this difference is by nature of purpose. My approach is the result of wanting to explore policy abstractly, whereas this simulator seeks to predict the effect of different strategies for the USA specifically.

Model

The model employed by this simulator is the SEIR model (Susceptible, Exposed, Infected, Recovered) this adds the extra dimension of an incubation period. This model is then extended to include hospitalisations, critical care, and death. By adding more compartments that are common to COVID-19 this increases the depth of their model. The SEIR model is a better model for accurately portraying the virus however it has the same pitfall that recovered individuals do not return to the susceptible state. Many individuals do not gain immunity after having had the virus and this is a criticism of both models.

Policies

This simulator has four forms of policy. Minimal Restrictions, Stay at Home Orders, Lockdown and Current Interventions. Using an epidemic tool this simulator obtains the current effect reproduction rate for each state to infer the effectiveness of restrictions (spread rate). The simulator also offers the user the choice of two policies. There are a few differences in approach regarding policy. Primarily this simulator offers more forms of intervention. Although my model offers the user three forms of lockdown with specifiable obedience this may be similar in a sense to different severity of restriction. The documentation does not mention isolation or tracing, so it is hard to know if this is being modelled. This is an important aspect of policy and should be included which is a downside to the simulator.

Duration vs. Threshold

A major difference between the two policy simulators is instigation of intervention methods. My simulator begins a lockdown when a threshold is exceeded and exits when it is no longer exceeded. This simulator however allows the user to specify how long the intervention lasts and it begins on the following Monday of the date you run the simulation. The drawback with the duration approach is that if the virus is still rampant it should be unlikely that intervention methods be removed. A criticism of the threshold model could be that absurdly short lockdowns may arise however by using event posting and information lag my model avoids this pitfall.

Data Presentation

Finally, the two differ on data presentation. This simulator produces 9 informative figures describing the results of the simulation compared to my single figure. These figures include daily cases, cumulative cases, daily deaths and much more. These figures are better in generation and presentation. As they use result of one simulation the curves are smoother. The only criticism is that they could run several simulations using the same parameters to increase the reliability of results.

10 Conclusions

10.1 Achievements

Throughout the project I have accomplished a set of significant achievements. I successfully synthesised and combined network theory and information surrounding public policy response into two models that represent how the public disobedience affects the effect of a lockdown. This model is also complimented by a system that can mimic how governments operate in a simplified sense, simulating the same levers that governments use to dictate lockdown policy. Another key achievement relates to the surrounding structure that allows users to specify and run experiments in a flexible and intuitive way. An overshadowed achievement is that of the network generator algorithms. These are key algorithms that capture the observed properties of social network and are modified so that they are more useable with the purpose of this project (e.g., cut off for suitable contact numbers in scale free networks and added parameters). Being able to have use different base networks is essential for users to be able to include any assumptions they may have on a population they are modelling.

10.2 Drawbacks

There are several drawbacks in this works, some related to the software as a product and others are of design decisions of the model. The graph builder is not as diverse as required by the tertiary requirements which is disappointing and could increase the usefulness of the software. If I had had more time, I would have liked to introduce more graphing options, in particular a way to see the number of lockdowns instigated and the percentage of time spent in lockdowns. Another addition would be to introduce a scatter plot of all values produced in a simulation and to draw lines of best fit over the top. This would remove the jagged lines exhibited in most results. Another drawback in the model is the timing between infection and onset of symptoms being rigid. This does not model reality well where onset of symptoms is quite varied. At the detriment of complexity, the timing for symptom onset of a node could be drawn from a distribution that matches the real data. Similar drawbacks exist in the model such as the use of a simple SIR model, it could be easily argued that an SEIR, SIS or SEIS model are more appropriate. Another significant drawback of this model (and all contemporary COVID-19 models) is that there has not been enough time to learn the characteristics of the disease and as such there is a lack of unification in results in the literature. A final drawback is the number of degrees of freedom and the repeatability. Although the model shows consistent behaviour that is aligned with the literature it is far from perfect. Minimising the effect of unnecessary degrees of freedom could be useful. This could be such as being able to use the same network in multiple different experiments to eliminate variance caused by the base graphs.

10.3 Future Work

This project is a great start in the subject however there are many directions for future work. Currently the project is specific to COVID-19, this is not particularly useful for new epidemics. A good direction for future work could be to introduce a feature that allows users to tailor the model to the characteristics of any virus. An obvious direction for future work would be introduce and model more policy. Lockdown is a severe and heavy policy that governments seek to avoid, modelling less severe policies with respect to obedience may help find better strategies for containing epidemics. Another interesting direction for future work would be to use this project as an educational tool. After improvement of the user interface this could be useful to show and inform people about epidemic containment and spread of disease. The software could be extended to show each simulation and the spread of the virus through the nodes. Finally, it would be good to explore more experimentation using the software. The experiment I have conducted is a good start however it only scratches the surface of the model's variability. Future experiments could incorporate many more combinations of policy and their values to produce further findings.

References

[1] World Health Organisation, “South Korea COVID-19 WHO,” 2021. [Online]. Available: https://covid19.who.int/region/wpro/country/kr. [Accessed 21 March 2021].

[2] World Health Organisation, “China COVID-19 WHO,” 2021. [Online]. Available: https://covid19.who.int/region/wpro/country/cn. [Accessed 21 March 2021].

[3] Worldometer, “World Populations By Country,” 2021. [Online]. Available: https://www.worldometers.info/world-population/population-by-country/. [Accessed 21 March 2021].

[4] World Health Organisation, “UK COVID-19 WHO,” 2021. [Online]. Available: https://covid19.who.int/region/euro/country/gb. [Accessed 21 March 2021].

[5] H. Sjödin, . A. Wilder-Smith, S. Osman, Z. Farooq and J. Rocklöv, “Only strict quarantine measures can curb the coronavirus disease (COVID-19) outbreak in Italy, 2020,” *Eurosurveillance,* vol. 25, no. 13, 2020.

[6] C. Fraser, C. A. Donnelly, S. Chauchemez, W. P. Hanage and M. D. Van Kerkhove, “Pandemic Potential of a Strain of Influenza A (H1N1): Early Findings,” *Science,* vol. 324, no. 5934, 2009.

[7] Health Knowledge, “epidemic-theory,” Health Knowledge, 2018. [Online]. Available: https://www.healthknowledge.org.uk/public-health-textbook/research-methods/1a-epidemiology/epidemic-theory. [Accessed March 2021].

[8] Nature, “A guide to R,” 03 July 2020. [Online]. Available: https://www.nature.com/articles/d41586-020-02009-w. [Accessed March 2021].

[9] BBC, “Covid-19: Christmas rules tightened for England, Scotland and Wales,” BBC, 2020 December 2020. [Online]. Available: https://www.bbc.co.uk/news/uk-55379220. [Accessed 22 March 2021].

[10] M. Day, “Covid-19: Italy reimposes widespread lockdown as transmission rate rises again,” *BMJ,* vol. 372, no. 726, 2021.

[11] France 24, “Paris enters new month-long partial lockdown amid soaring Covid-19 cases,” 20 March 2021. [Online]. Available: https://www.france24.com/en/france/20210320-paris-enters-new-month-long-partial-lockdown-amid-soaring-covid-19-cases. [Accessed 23 March 2021].

[12] Y. Kageyama, “Japan declares emergency for Tokyo area as cases spike,” AP News, 7 January 2021. [Online]. Available: https://apnews.com/article/tokyo-emergency-coronavirus-spike-japan-01e0916762eaa06d3d65510bcd271967. [Accessed 23 March 2021].

[13] Cabinet Office, “National lockdown: Stay at Home,” 4 January 2021. [Online]. Available: https://www.gov.uk/guidance/national-lockdown-stay-at-home. [Accessed 23 March 2021].

[14] K. Ng, “Spanish army to enforce lockdown in Madrid,” Independent, 22 September 2020. [Online]. Available: https://www.independent.co.uk/news/world/europe/madrid-spain-coronavirus-lockdown-restrictions-army-b526693.html. [Accessed 23 March 2021].

[15] J. Craig, “Coronavirus: How Dominic Cummings' trip to Durham damaged trust in the government,” BBC, 7 August 2020. [Online]. Available: https://news.sky.com/story/coronavirus-how-dominic-cummings-trip-to-durham-damaged-trust-in-the-government-12044015. [Accessed 23 March 2021].

[16] BBC, “Bristol illegal rave attended by 700 people,” BBC, 01 November 2020. [Online]. Available: https://www.bbc.co.uk/news/av/uk-54773047. [Accessed 23 March 2020].

[17] T. Cheshire, “Coronavirus: How South Korea's track and trace system has kept death count below 500,” Sky News, 12 October 2020. [Online]. Available: https://news.sky.com/story/coronavirus-how-south-koreas-track-and-trace-system-has-kept-death-count-below-500-12103124. [Accessed 05 April 2021].

[18] L. Kelion, “Covid: Test error 'should never have happened' - Hancock,” BBC, 5 October 2020. [Online]. Available: https://www.bbc.co.uk/news/uk-54422505. [Accessed 05 April 2021].

[19] S. Kojaku, L. Hébert-Dufresne and E. Mones, “The effectiveness of backward contact tracing in networks,” *Nature Physics,* 2021.

[20] “COVID-19 pandemic control: Balancing detection policy and lockdown intervention under ICU sustainability,” *Mathematical Modelling of Natural Phenomena,* vol. 15, no. 57, 2020.

[21] “Exact analytical solutions of the Susceptible-Infected-Recovered (SIR) epidemic model and of the SIR model with equal death and birth rates,” *Applied Mathematics and Computation,* vol. 236, pp. 184-194, 2014.

[22] W. O. Kermack and A. McKendrick, “A contribution to the mathematical theory of epidemics,” *Proceedings of the Royal Society A,* vol. 115, no. 772, pp. 700-721, 1927.

[23] Public Health England, “Past COVID-19 infection provides some immunity but people may still carry and transmit virus,” 14 January 2021. [Online]. Available: https://www.gov.uk/government/news/past-covid-19-infection-provides-some-immunity-but-people-may-still-carry-and-transmit-virus. [Accessed 23 March 2021].

[24] S. Glen, “Power Law,” Statistic How To, 15 July 2016. [Online]. Available: https://www.statisticshowto.com/power-law/. [Accessed 2021].

[25] A.-L. Barabási and R. Albert, “"Statistical mechanics of complex networks",” *Reviews of Modern Physics,* vol. 74, pp. 47-97, 2002.

[26] D. Q. Nykamp, “Scale Free Network,” Math Insight, [Online]. Available: https://mathinsight.org/scale_free_network. [Accessed January 2021].

[27] D. J. Watts and S. Strogatz, “Collective dynamics of "small-world" networks,” *Nature,* vol. 393, pp. 440-442, 1998.

[28] U. Brandes and T. Erlebach, Network Analysis : Methodological Foundations (Lecture Notes in Computer Science), 2005.

[29] A. Sallaberry, F. Zeidi and G. Melançon, "Model for Generating Artificial Social Networks," *Social Network Analysis and Mining, SPringer,* vol. 3, 2013.

[30] S. Milgram, "The Small World Problem," *Pyschology Today,* May 1967.

[31] M. Girvan and M. Newman, "Community structure in social and biological networks," *Proceedings of the National Academy of Sciences of the United States of America,* vol. 99, no. 12, pp. 7821-7826, 2002.

[32] S. Fortunato, "Community detection in graphs," *Physics Reports,* vol. 486, no. 3-5, pp. 75-174, 2010.

[33] G. Csányi and B. Szendrői, "Structure of a large social network," *Physical Review E,* vol. 69, no. 3, 2004.

[34] S. Dobson, "SIR Model," 9 December 2020. [Online]. Available: https://github.com/simoninireland/epydemic/blob/master/epydemic/sir_model.py. [Accessed 01 April 2021].

[35] E. N. Gilbert, "Random Graphs," 1959. [Online]. Available: https://projecteuclid.org/euclid.aoms/1177706098. [Accessed 2020].

[36] P. Erdős and A. Rényi, "On Random Graphs," in *Publicationes Mathematicae*, Institute of Mathematics, University of Debrecen, 1959, pp. 290-297.

[37] P. Erdős and . A. Rényi, " On the evolution of random graphs," in *A MTA MATEMATIKAI KUTATÓ INTÉZETÉNEK KÖZLEMÉNYEI 5. ÉVFOLYAM*, 1960, pp. 17-61.

[38] NetworkX, "Generators," 22 August 2020. [Online]. Available: https://networkx.org/documentation/stable//_modules/networkx/generators/random_graphs.html#barabasi_albert_graph. [Accessed January 2021].

[39] L. Wang, F. Du, H. Dai and Y. Sun, "Random pseudofractal scale-free networks with small-world effect," *The European Physics Journal B,* vol. 53, pp. 361-366, 2006.

[40] S. Dobson, "Running a process," 2020. [Online]. Available: https://pyepydemic.readthedocs.io/en/latest/tutorial/use-standard-model.html. [Accessed 29 03 2021].

[41] scherry, "Plot Distributions," EssyCode, [Online]. Available: https://www.essycode.com/distribution-viewer/. [Accessed 30 03 2021].

[42] F. Parés, D. Garcia-Gasulla and A. Vilalta, "Fluid Communities: A Competitive, Scalable," in *International Workshop on Complex Networks and their Applications*, 2018.

[43] I. Dunbar and R. Sosis, "Optimising human community sizes," *Evolution and Human Behavior,* vol. 39, no. 1, pp. 106-111, 2018.

[44] S. Laur, K. Grantz and Q. Bi, "The Incubation Period of Coronavirus Disease 2019 (COVID-19) From Publicly Reported Confirmed Cases: Estimation and Application," *Annals of Internal Medicine,* vol. 172, no. 9, 2020.

[45] X. He, E. Lau and P. Wu, "Temporal dynamics in viral shedding and transmissibility of COVID-19," *nature medicine,* vol. 26, pp. 672-675, 2020.

[46] A. Byrne, D. McEvoy and A. Collines, "Inferred duration of infectious period of SARS-CoV-2: rapid scoping review and analysis of available evidence for asymptomatic and symptomatic COVID-19 cases," *BMJ Open,* vol. 10, no. 8, 2020.

[47] D. Oran, AM and E. Topol, "The Proportion of SARS-CoV-2 Infections That Are Asymptomatic," *Annals of Internal Medicine,* 2021.

[48] A. Billah, M. Miah and N. Khan, "Reproductive number of coronavirus: A systematic review and meta-analysis based on global level evidence," *PLOS ONE,* vol. 15, no. 11, 2020.

[49] A. Holko, "Epidemic Simulation," [Online]. Available: https://holko.pl/epidemic-simulation/. [Accessed 02 April 2021].

[50] Harvard Medical School, "COVID-19 Simulator," [Online]. Available: https://analytics-tools.shinyapps.io/covid19simulator05/. [Accessed 12th April April 2021].

Appendix 1

UNIVERSITY OF ST ANDREWS
TEACHING AND RESEARCH ETHICS COMMITTEE (UTREC)
SCHOOL OF COMPUTER SCIENCE
PRELIMINARY ETHICS SELF-ASSESSMENT FORM

This Preliminary Ethics Self-Assessment Form is to be conducted by the researcher, and completed in conjunction with the Guidelines for Ethical Research Practice. All staff and students of the School of Computer Science must complete it prior to commencing research.

This Form will act as a formal record of your ethical considerations.
Tick one box

- ☐ **Staff Project**
- ☐ **Postgraduate Project**
- ☒ **Undergraduate Project**

Title of project

Exploring the observable factors of an unobservable social network

Name of researcher(s)

Alexander (Sandy) Johnstone

Name of supervisor (for student research)

Simon Dobson

OVERALL ASSESSMENT (to be signed after questions, overleaf, have been completed)

Self audit has been conducted **YES** ☑ **NO** ☐

There are no ethical issues raised by this project

Signature Student or Researcher

Print Name

Alexander Johnstone

Date

26/09/2020

Signature Lead Researcher or Supervisor

Print Name

Date

This form must be date stamped and held in the files of the Lead Researcher or Supervisor. If fieldwork is required, a copy must also be lodged with appropriate Risk Assessment forms. The School Ethics Committee will be responsible for monitoring assessments.

Computer Science Preliminary Ethics Self-Assessment Form

Research with human subjects

Does your research involve human subjects or have potential adverse consequences for human welfare and wellbeing?

YES ☐ **NO** ☒

If YES, full ethics review required
For example:
Will you be surveying, observing or interviewing human subjects?
Will you be analysing secondary data that could significantly affect human subjects?
Does your research have the potential to have a significant negative effect on people in the study area?

Potential physical or psychological harm, discomfort or stress

Are there any foreseeable risks to the researcher, or to any participants in this research?

YES ☐ **NO** ☒

If YES, full ethics review required
For example:
Is there any potential that there could be physical harm for anyone involved in the research?
Is there any potential for psychological harm, discomfort or stress for anyone involved in the research?

Conflicts of interest

Do any conflicts of interest arise?

YES ☐ **NO** ☒

If YES, full ethics review required
For example:
Might research objectivity be compromised by sponsorship?
Might any issues of intellectual property or roles in research be raised?

Funding

Is your research funded externally?

YES ☐ **NO** ☒

If YES, does the funder appear on the 'currently automatically approved' list on the UTREC website?

YES ☐ **NO** ☐

If NO, you will need to submit a Funding Approval Application as per instructions on the UTREC website.

Research with animals

Does your research involve the use of living animals?

YES ☐ **NO** ☒

If YES, your proposal must be referred to the University's Animal Welfare and Ethics Committee (AWEC)

University Teaching and Research Ethics Committee (UTREC) pages
http://www.st-andrews.ac.uk/utrec/

Appendix 2

The relationship between government policy lockdown strategy and epidemic outcome

Aim:
The aim of the experiment is to use the model created to explore how different government strategies affect the outcome of an epidemic.

Purpose:
In this model government strategy can be viewed as a value for a key statistic (R_0 or Daily Cases), if the value is exceeded the model enters a lockdown. As the model incorporates lag it is useful to know if the government takes actions at different levels how this affects the outcome of an epidemic. The outcome of an epidemic is related to how many nodes are removed (infected during the epidemic) and how many nodes are still susceptible. The purpose therefore is to find levels at which, with good obedience, the epidemic can be controlled to find suitable values for further experimentation on obedience.

Assumptions of the model/experiment:
This model assumes that government strategy does not change over the course of an epidemic. This means one value is used as a threshold per simulation.

The experiment does not incorporate obedience-based lockdowns. This means that the instigated lockdowns are instigated on a network basis, in which a random sample of 90% of the nodes are removed from the network when a lockdown is present.

The model assumes a government takes 7 days after a threshold breach to instigate a lockdown. This accounts for delays in reporting and for time to decide/implement a lockdown.

The model does not incorporate error in reported statistics.

The model (SIR) assumes nodes cannot be re-infected.

Method:
To carry out this experiment requires 3 sets of simulations in the software:

Simulation 1 – R_0 Only

Increments: From 0.7 to 1.1 with intervals of 0.01

Simulation 2 – Daily Cases Only

Increments: From 0.0001 to 0.004 with intervals of 0.0001

Simulation 3 - R_0 and Daily Cases

Increments: As before.

These are the three possible combinations of policy. For each simulation I attempt to fix as many things as possible. Firstly, I fix the mean node degree to 10±1, this means for each graph the mean degree may be between 9 and 11. Secondly the graph size is fixed to exactly 10,000 nodes. For each simulation I use 40 value increments (increments shown above) with 10 repetitions per simulation. The increment start and end points were found

empirically before the experiment, Finally, each simulation is carried out on each of the 4 base graph generators.

Results:

Simulation 1 – R_0 Only

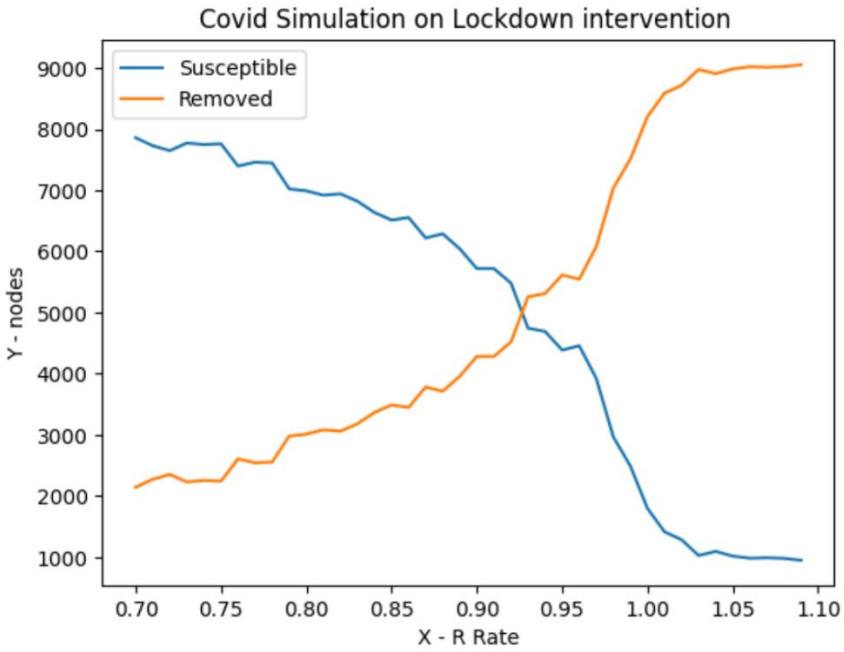


Figure 24 – Resulting graph of a simulation with parameters: R_0 = 0.7, Increment = 0.01, Iterations = 40, Repetitions = 10. Network created using Erdős–Rényi Algorithm with parameters: 10,000 nodes and mean degree 10. The graph shows the change in epidemic outcome as the intervention value increases.

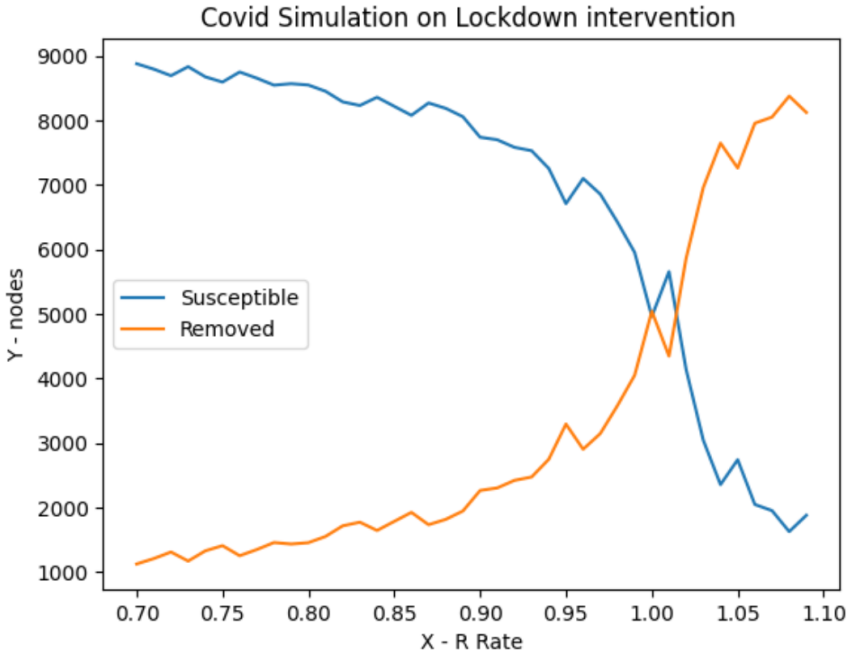


Figure 25 – Results of a simulation with parameters: R_0 = 0.7, Increment = 0.01, Iterations = 40, Repetitions = 10. Network created using Watts-Strogatz Algorithm with parameters: 10,000 nodes, mean degree 10 and rewiring parameter 0.35. The graph shows the change in epidemic outcome as the intervention value increases.

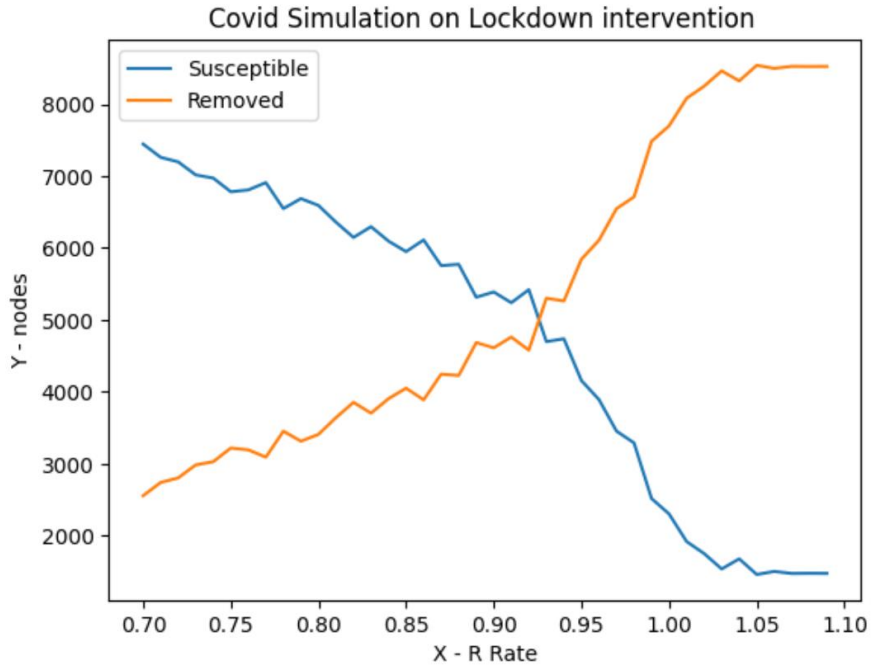


Figure 26 – Results of a simulation with parameters: $R_0 = 0.7$, Increment = 0.01, Iterations = 40, Repetitions = 10. Network created using Barabási -Albert Network Algorithm with parameters: 100 starting nodes, 5 new edges and growth limit of 10,000. The graph shows the change in epidemic outcome as the intervention value increases.

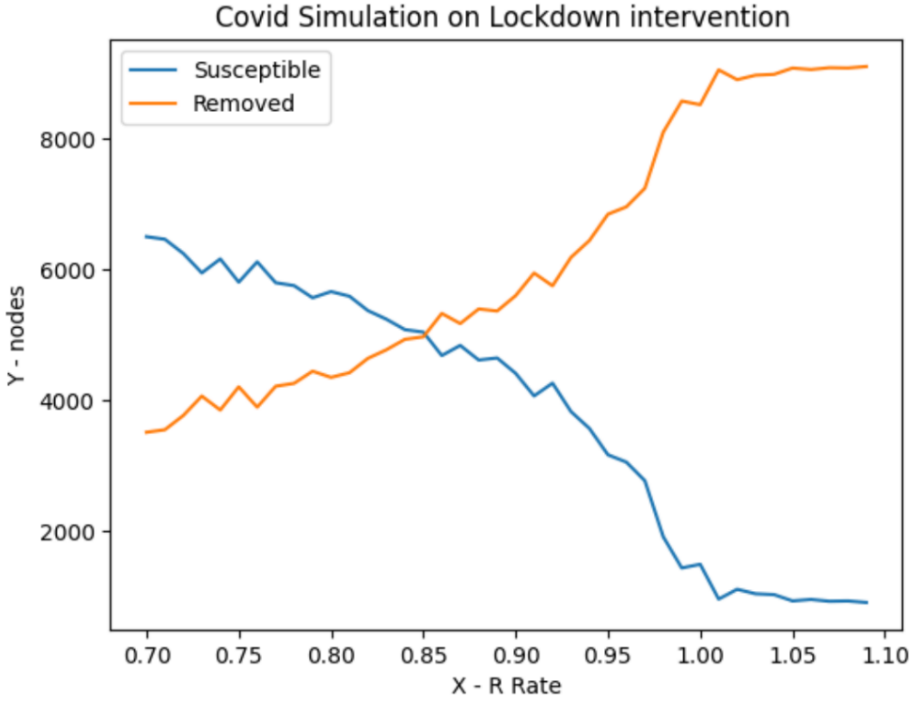


Figure 27 - Results of a simulation with parameters: $R_0 = 0.7$, Increment = 0.01, Iterations = 40, Repetitions = 10. Network created using Random Pseudofractal Algorithm with parameters: 5 new edges for nodes, growth limit of 10,000. The graph shows the change in epidemic outcome as the intervention value increases.

Simulation 2 – Daily Cases Only

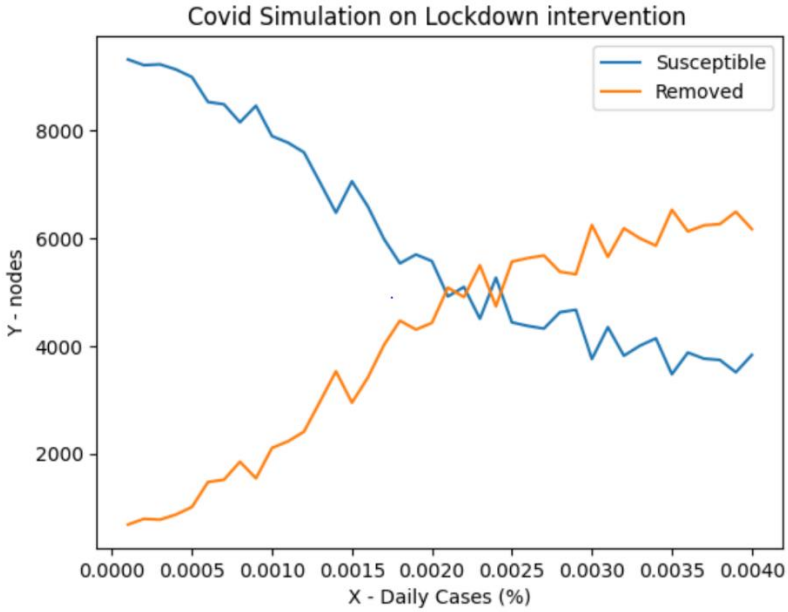


Figure 28 - Resulting graph of a simulation with parameters: Daily Cases = 0.0001, Increment = 0.0001, Iterations = 40, Repetitions = 10. Network created using Erdős–Rényi Algorithm with parameters: 10,000 nodes and mean degree 10. The graph shows the change in epidemic outcome as the intervention value increases.

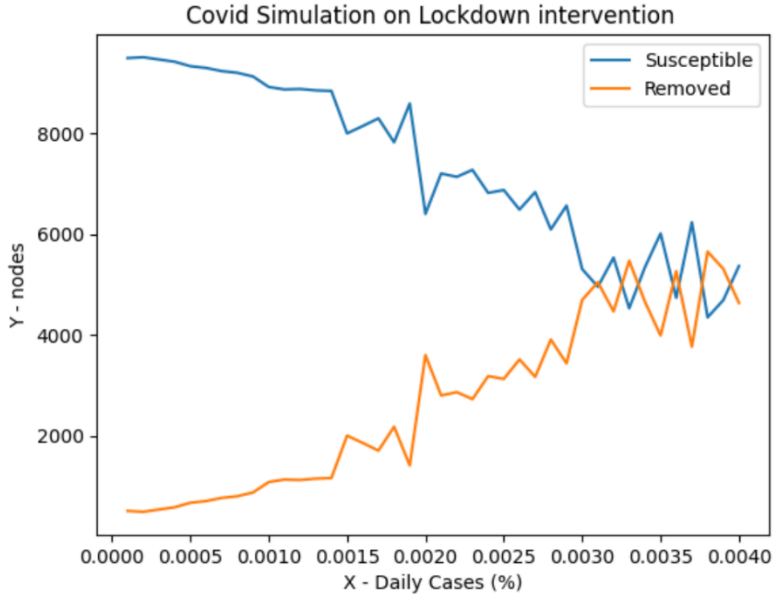


Figure 29- Results of a simulation with parameters: Daily Cases = 0.0001, Increment = 0.0001, Iterations = 40, Repetitions = 10. Network created using Watts-Strogatz Algorithm with parameters: 10,000 nodes, mean degree 10 and rewiring parameter 0.35. The graph shows the change in epidemic outcome as the intervention value increases.

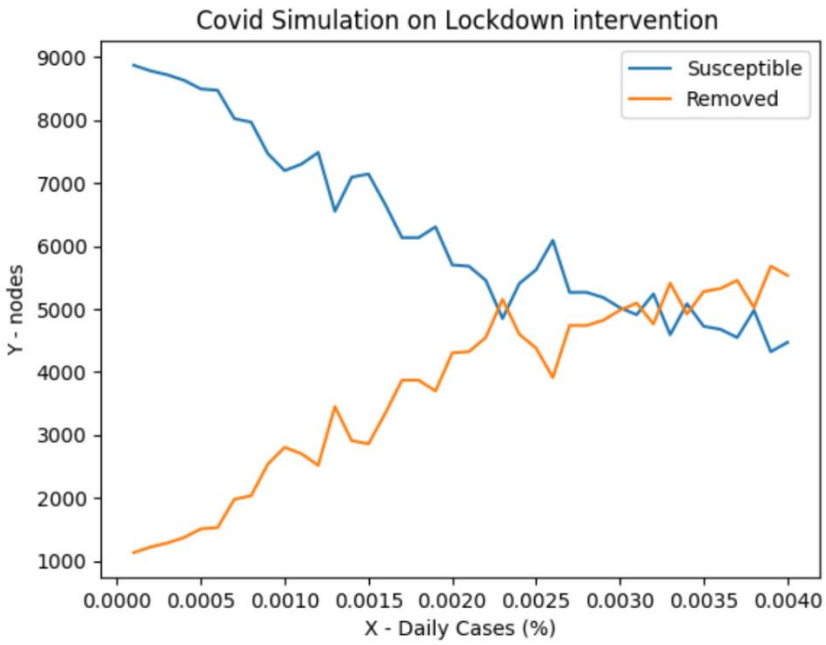


Figure 30 – Results of a simulation with parameters: Daily Cases = 0.0001, Increment = 0.0001, Iterations = 40, Repetitions = 10. Network created using Barabási -Albert NetworkAlgorithm with parameters: 100 starting nodes, 5 new edges and growth limit of 10,000. The graph shows the change in epidemic outcome as the intervention value increases.

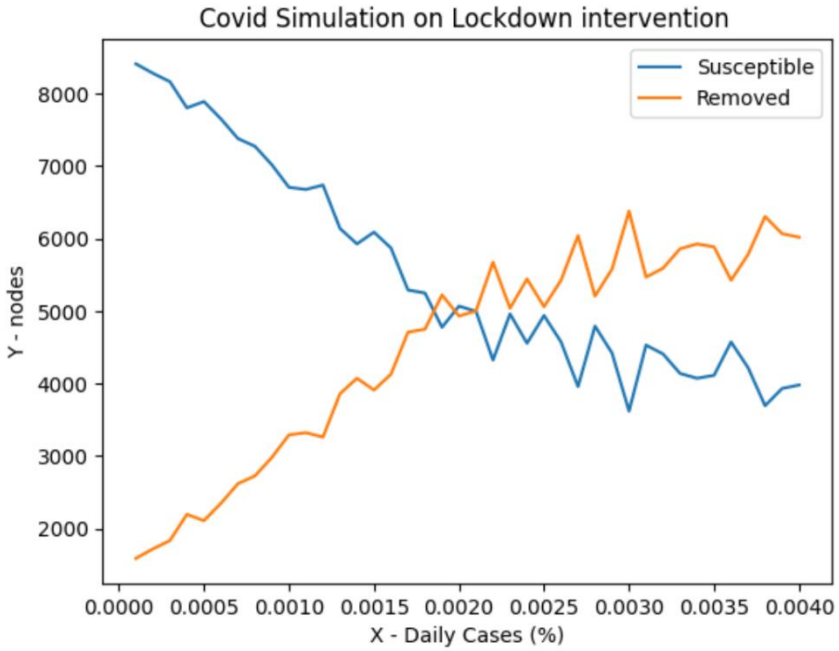


Figure 31 Results of a simulation with parameters: Daily Cases = 0.0001, Increment = 0.0001, Iterations = 40, Repetitions = 10. Network created using Random Pseudofractal Algorithm with parameters: 5 new edges for nodes, growth limit of 10,000. The graph shows the change in epidemic outcome as the intervention value increases.

Simulation 3 – R_0 and Daily Cases

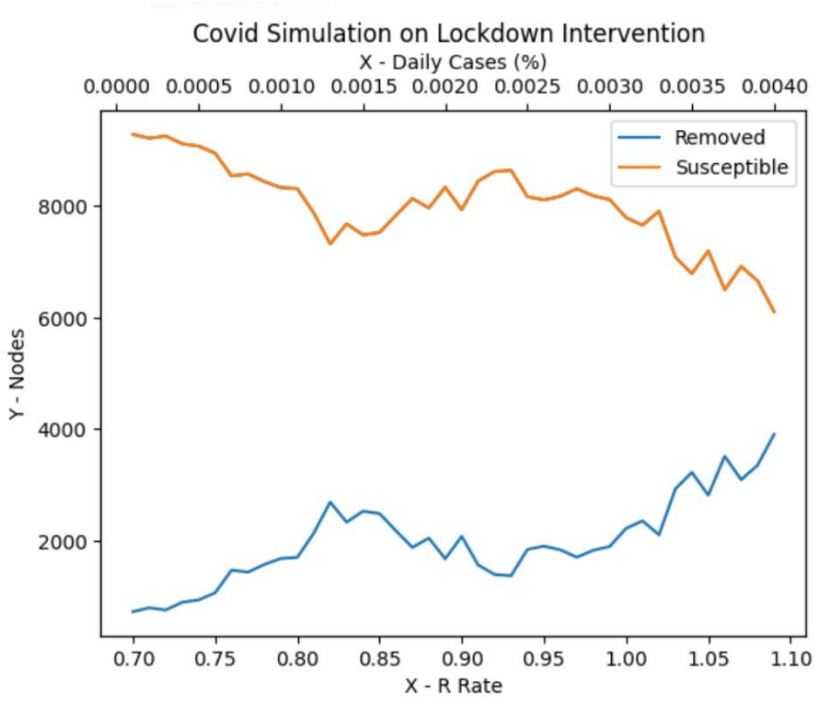


Figure 32 - Resulting graph of a simulation with parameters: (R_0 = 0.7, Increment = 0.01), (Daily Cases = 0.0001, Increment = 0.0001), Iterations = 40, Repetitions = 10. Network created using Erdős–Rényi Algorithm with parameters: 10,000 nodes and mean degree 10. The graph shows the change in epidemic outcome as the intervention value increases.

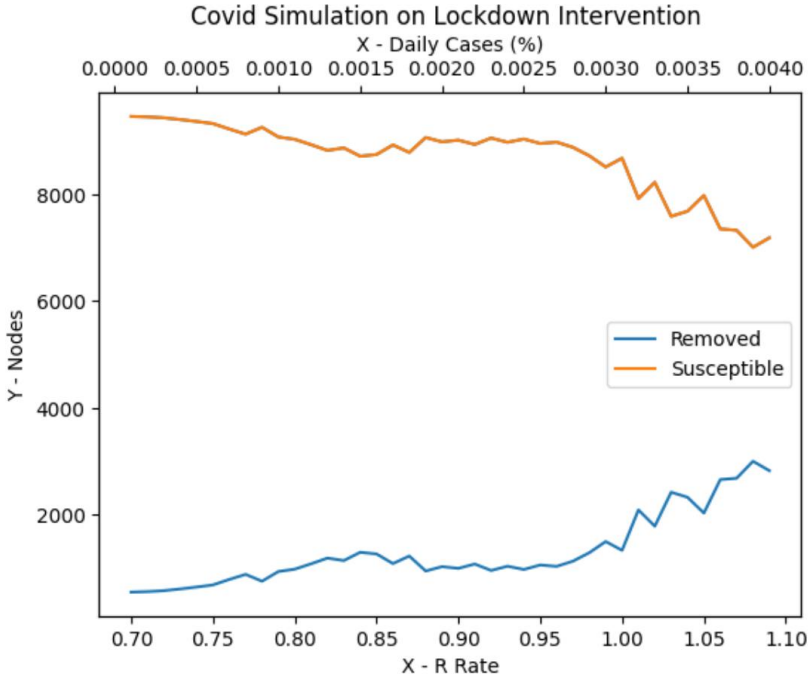


Figure 33 - Resulting graph of a simulation with parameters: (R_0 = 0.7, Increment = 0.01), (Daily Cases = 0.0001, Increment = 0.0001), Iterations = 40, Repetitions = 10. Network created using Watts-Strogatz Algorithm with parameters: 10,000 nodes, mean degree 10 and rewiring parameter 0.35. The graph shows the change in epidemic outcome as the intervention value increases.

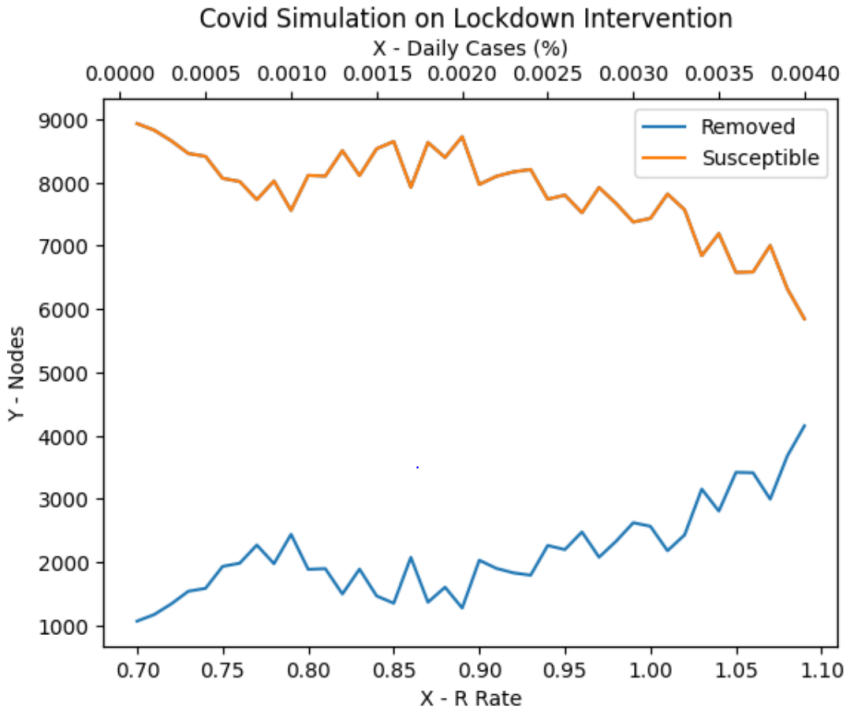


Figure 34 - Resulting graph of a simulation with parameters: (R_0 = 0.7, Increment = 0.01), (Daily Cases = 0.0001, Increment = 0.0001), Iterations = 40, Repetitions = 10. Network created using Barabási -Albert Network Algorithm with parameters: 100 starting nodes, 5 new edges and growth limit of 10,000. The graph shows the change in epidemic outcome as the intervention value increases.

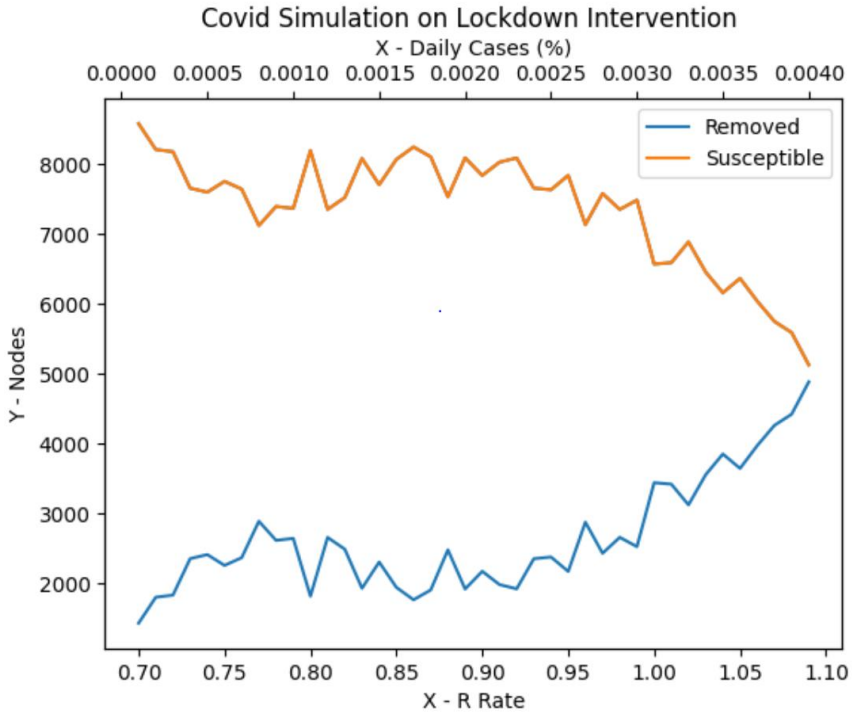


Figure 35 - - Resulting graph of a simulation with parameters: (R_0 = 0.7, Increment = 0.01), (Daily Cases = 0.0001, Increment = 0.0001), Iterations = 40, Repetitions = 10. Network created using Random Pseudofractal Algorithm with parameters: 5 new edges for nodes, growth limit of 10,000. The graph shows the change in epidemic outcome as the intervention value increases.

Interpretation of Results

Simulation 1 – R_0

The results of R_0 as the sole intervention value are consistent and stable. Figures 24-27 show cleaner curves and clear tipping (cross-over points) for simulations. The cross over point ranges from 0.85 to 1.0, suggesting that underlying topology influences outcome. The results also suggest that power-law networks need a lower (stricter) value of R_0 to contain spread and that graphs with clustering and no power-law fare the best (Figure 25).

Simulation 2 – Daily Cases

The results of simulation 2 are more varied than simulation 1. In the investigated space there is large variation, and the lines are more jagged suggesting more varied results. For each of the figures (28-31) a cross-over point can be seen and they share a similar shape. For figures 28, 30 & 31 the cross-over points are similar, all starting in the interval 0.0015 to 0.0025. The topology that fared best using daily cases alone was the Watts-Strogatz graph. This suggests a virus spreading in a population with no power-law aspect can be well described by daily cases. Figure 30 & 31 both have power-law properties and as such require lower thresholds than those that have no power-law properties. Overall, the results of daily cases are much more jagged than in simulation 1 suggesting the outcome of any given simulation is more likely to vary between extremes.

Simulation 3 – R_0 & Daily Cases

Finally, simulation 3 shows some interesting outcomes. The model is designed to enter a lockdown if either lever is triggered, I would expect this to lead to lever domination however the combination of the two produces a near identical shape regardless of topology as shown by figures 32 to 35. This structure is however a little unusual. In each of the relevant figures the outcome decreases, increases again and then decreases again. This is unusual as it does not produce a regular curve like the other simulations. The reason behind this initial decline is unexplained. One explanation *could* be that at these points the model gets stuck between policies which leads to longer epidemics and more infections. Networks with power-law properties produce the most erratic results, as shown in figure 34 & 35 the plotted lines are much more jagged suggesting more variation in the results relative to figures 32 and 33. Overall the shape of these results is very promising and suggests a combination of policies is much more powerful than each individually which I would expect.

Conclusion:

Several conclusions can be drawn from these simulations. Primarily this experiment highlights the idea that understanding a population topology is key to suppressing an epidemic. The most important factor of topologies is the power-law aspect. If a population exhibits a power-law, then R_0 is a less effective metric to inform lockdown decisions than daily cases. Oppositely if a power-law is not present in the network then daily cases is a less effective metric, and R_0 is a better metric to gauge spread and inform lockdown decisions. Using a mixture of both levers however appears to be the optimal choice to suppress the virus, this is to be expected as lockdowns may be triggered for either lever and therefore there is higher chance of instigating a lockdown at the right time. This only works however if they are used in an 'either or' approach. If both levers had to be met to instigate a lockdown this would most likely lead to worse outcomes than single metrics.

Repeatability:
All the parameters used to create these graphs are provided I the method section and network specific parameters can be found under each graph.

Appendix 3

Operations Manual

Prerequisites:
To run the source code, the following are essential:

Python 3.9.0 - *https://www.python.org/*

PyPi or similar package manager - *https://packaging.python.org/tutorials/installing-packages/#installing-from-pypi*

Installing:
Step 1:

Installation of the code primarily requires the source code itself. The source code is available on MMS in the correct structure. It is important that this structure is unchanged for relative paths.

Step 2:

The second step is to ensure that you have all the relevant packages require by the source code. I have included a file named "requirement.txt" in the source folder that lists all requirements for the project. To install these packages with PyPi use the command:

```
$ pip install -r requirements.txt
```

Execution:
To execute the code from command line, use the command:

```
$ python _main_.py
```

Guidance:
The system is mostly intuitive. If an input aspect is unclear there is likely an informative tag that displays when you hover over the relevant box. The following instructions should aid use of the software:

One important aspect of the software is that the frontend should **not be used in full screen mode** this is purely for the results page as it can reduce the visibility of the graph.

Network Setup

The network set up firstly provides a series of radio buttons relating to different generators. Selecting different generators changes the topology of the underlying network. Once a generator has been selected the next screen asks for generator parameters. These parameters may seem obscure however an explanation is provided for each in this report.

Policy Setup

The policy setup page is less intuitive. For each policy, the first value corresponds to the initial value of the policy and the second value corresponds to how much it will increase by between simulations. If no increase is desired this value should be set to 0. The user must select the policies they want to include using radio buttons and alert boxes will prevent errors.

Simulation Setup

Finally, the simulation set provides to input boxes. The first relates to how many different models will be made, each new model will be created using the initial value and the increments for the parameters. The second box relates to how many times each simulation will be repeated.

Appendix 4

Execution of Tests:
To execute the test suite, navigate to the "Integration" folder or the "Unit" folder within the "test" folder and execute the command:

```
$ python -m unittest discover
```

Testing Summary:
Unit

TestUnitER - Unit tests for Erdős–Rényi Generator	
Test Scenario	**Outcome**
Test that when probability is 0 the graph is fully unconnected	Pass
Test that when probability is 1 the graph is fully connected	Pass
Test that when probability is beneath 0 is behaves like it is 0	Pass
Test that when probability is over 1 it behaves like it is 1	Pass
Test that when probability is between 1 and 0 the edges percentage is similar to probability	Pass

TestUnitBA - Unit tests for Barabasi-Albert Generator	
Test Scenario	**Outcome**
Test that when new connections are 0 the graph is fully unconnected and empty	Pass
Test that when new input is normal the graph has expected properties (node and edge count)	Pass
Test that when parameter M is negative the algorithm behaves as if M is 0	Pass
Test that the preferential attachment method picks expected number from a list	Pass
Test that in a power law many have few connections, and few have many	Pass
Test that cut-off works	Pass
TestUnitWS - Unit tests for Watts-Strogatz Generator	
Test Scenario	**Outcome**
Test that when mean degree is 0 the graph is fully unconnected	Pass
Test that when mean degree is highest possible the graph is fully connected	Pass
Test that when rewiring that the edges change	Pass
Test that checks rewiring works as expected	Pass
Test that checks rewiring works as expected when only one possible rewiring is left	Pass
Test that the clustering of the network is high	Pass
Test that small world properties are produced	Pass

TestUnitRPN - Unit tests for Random Psuedofractal Generator	
Test Scenario	**Outcome**
Test that when new connections is 0 the graph is unconnected	Pass
Test that when new connections is higher than nodes that the graph is fully connected	Pass
Test that power law effect occurs	Pass
Test that small world properties are produced	Pass
Test that cut-off works	Pass

TestUnitSIR - Unit tests for Covid SIR model	
Test Scenario	**Outcome**
Test when an infection happens case tracking is updated	Pass
Test when an infection happens r tracking is updated	Pass
Test when a node is removed r tracking is updated	Pass
Test asserts when all lockdown criteria are met for rate of reproduction then a lockdown is posted	Pass
Test asserts when rate of reproduction is too low the lockdown is not posted	Pass
Test asserts when there are too few cases R is not used	Pass
Test asserts when there are already removed edges (a lockdown in place) it is not posted	Pass
Test asserts when there is a lockdown posted for cases a new one is not posted	Pass
Test asserts when there is a lockdown posted for R already a new one is not posted	Pass
Test asserts when all lockdown criteria are met for cases then a lockdown is posted	Pass
Test asserts when there are insufficient cases then a lockdown is not posted	Pass
Test asserts when there are sufficient cases but removed edges then a lockdown is not posted	Pass
Test asserts when there are sufficient cases but a lockdown for R then a lockdown isn't posted	Pass
Test when a normal lockdown is posted the expected number of nodes are removed	Pass
Test when a normal lockdown is posted then removed that the number of edges is restored	Pass
Test an individually varied lockdown	Pass
Test a clustered varied lockdown	Pass
Test that the community algorithm creates communities of an expected size	Pass
Test isolation event is posted	Pass
Test isolation event works as expected	Pass
Test tracing event is posted	Pass
Test tracing event works as expected	Pass
Test that the tracing event posts an undo trace event	Pass
Test an undo trace restores network edges	Pass

TestUnitFigureBuilder - Unit tests for Figure Builder	
Test Scenario	**Outcome**
Test that builder can plot correctly with no axis	Pass
Test that builder can plot cases axis correctly	Pass
Test that builder can plot individual variance axis correctly	Pass
Test that builder can plot clustered variance axis correctly	Pass
Test that builder can plot isolation axis correctly	Pass
Test that builder can plot track and trace axis correctly	Pass
Test that builder can identify when one axis is to small when two are specified	Pass
Test that builder can plot r and case axes together	Pass
Test that builder can plot r and variance together	Pass
Test that builder can pick two axis and plot when more than two policies are specified	Pass

Integration

TestIntegrationSIR- Integration test for SIR model	
Test Scenario	**Outcome**
Test that setting distributed variance works as expected	Pass
Test that setting clustered variance works creates communities	Pass
Test that setting very low r leads to lockdown (minimal spread)	Pass
Test that setting very high r leads to lockdown (widespread)	Pass
Test that setting very low cases leads to lockdown (minimal spread)	Pass
Test that setting very high cases leads to no lockdown (widespread)	Pass
Test that setting very high obedience leads to lockdowns with minimal spread	Pass
Test that setting very low obedience leads to ineffective lockdowns	Pass
Test that setting very high obedience leads to lockdowns with minimal spread	Pass
Test that setting very low obedience leads to ineffective lockdowns	Pass
Test that setting very high case finding leads to isolation with minimal spread	Pass
Test that setting very low case finding leads to ineffective isolation	Pass
Test that setting very high track and trace leads to minimal spread	Pass
Test that setting very low track and trace leads to ineffective isolations	Pass

TestIntegrationSimRunner – Integration tests for sim runner	
Test Scenario	**Outcome**
Test to check sim runner does correct number of increments and the correct increments	Pass
Test to check sim runner does correct number of increments for no increment	Pass
Test to check sim runner performs correctly for large input	Pass

TestIntegrationSimRunner – Integration tests for sim runner	
Test Scenario	**Outcome**
Test to check sim runner does correct number of increments and the correct increments	Pass
Test to check sim runner does correct number of increments for no increment	Pass
Test to check sim runner performs correctly for large input	Pass

TestIntegrationGraphGen – Integration tests for graph handler	
Test Scenario	**Outcome**
Test that generation Erdos-Renyi network using graph generator works	Pass
Test that generation Watts-Strogatz network using graph generator works	Pass
Test that generation Barabasi-Albert network using graph generator works	Pass
Test that generation Random Pseudo-Fractal network using graph generator works	

TestIntegrationEelMethods– Integration tests for eel methods	
Test Scenario	**Outcome**
Test ER generation	Pass
Test WS generation	Pass
Test BA generation	Pass
Test RPN generation	Pass
Test frontend setters work as expected	Pass
Test simulation handling works as expected	Pass
Test figure can be built from simulation	Pass