Imperial College London

Department of Computing

MEng Individual Project

A Spatio-Temporal Framework for the Modelling and Analysis of Biological and Socio-Technological Epidemics

Author:
Christos Karamanos

Supervisor:
Dr. William Knottenbelt
Second Marker:
Dr. Giuliano Casale

June 15, 2015

Abstract

This project proposes a framework, named STAGE, which facilitates the spatio-temporal visualisation, modelling, analysis and simulation of epidemics. By examining how a biological epidemic unfolds, it is possible to make short-term predictions and assess the effectiveness of various containment measures with obvious benefits for the authorities combating such epidemics. Recent studies have shown that the same mathematical models which have been used to describe how diseases spread, are often adequate to explain socio-technological epidemics too, such as information diffusion on an online social network. The analysis of such socio-technological epidemic phenomena has numerous commercial applications in areas like marketing, capacity planning and advertising.

The project concentrates on spatio-temporal modelling which, compared to classical epidemic modelling that only provides for the prediction of the number of infections at various points in time, allows us to answer complex questions such as "where is the disease likely to spread next?" Several frameworks capable of conducting such experiments have been developed recently to help epidemiologists gain more insight on how epidemics spread. They are however fundamentally tied to the case of biological epidemics as they are built on the assumption that transmission depends on physical contact and movement. They also tend to be strict on the types of models they can simulate, have over-complicated procedures for visualising data sets and constructing simulation scenarios and lack the mechanisms to provide quick feedback by visualising the model and the data side-by-side. This project addresses these weaknesses by presenting a framework capable of visualising, modelling and simulating general epidemics over both physical and logical space, making it suitable for researching socio-technological epidemic phenomena too.

The effectiveness of the proposed framework is tested by examining two case studies, one coming from the biological world and a second one from the socio-technological domain. The selected biological epidemic involves the ongoing Ebola outbreak in West Africa. By analysing data obtained from the World Health Organisation we verify that the spreading of the disease into new areas depends on geographical factors like distance and population density. Using the framework we form and simulate a spatio-temporal model consisting of a sophisticated disease model and a simple model of human mobility and show its ability to explain the spatio-temporal spread to a satisfactory level.

The second case study performed demonstrates the flexibility of the proposed framework in synthesising and testing spatio-temporal models for socio-technological epidemics. We investigate how information regarding events in London propagates through Twitter on three occasions and whether this is influenced by the physical location of the users. Indeed, the analysis of several aspects of the geotagged tweets, sent on the day of each event, shows that geography is important. In particular, we notice that the majority of the tweets were sent by users who were close to the location of the event, and that the majority of one's followers were located close to them. To account for geographical properties in the model, we present a novel way of using existing human mobility models to construct contact networks. By supporting both physical and logical space, the framework allows us to perform simulations in which information spreads through the logical contact network of the individuals involved but the space which is visualised concerns the physical locations of these individuals.

Acknowledgements

First of all I would like to thank my supervisor Dr. William J. Knottenbelt for his guidance and support throughout the project. Secondly, I would like to thank Dr. Dieter Fiems for his input during the initial stages. Last but not least, I would like to express my gratitude to my parents Neophytos and Athina for their continuous love and support throughout the years.

Contents

Chapter 1

Introduction

This project presents the Spatio-Temporal Analyser for General Epidemics (STAGE). It demonstrates its applicability to both biological and socio-technological epidemic phenomena by examining the 2014 West Africa Ebola outbreak and the activity between Twitter users regarding three 2015 London-based events.

1.1 Motivation

The study of biological epidemics is well-established and continues to attract a lot of attention. This is because despite the advancements in health care, outbreaks of diseases capable of devastating outcomes arise constantly. Only last month an outbreak of the Middle East Respiratory Syndrome (MERS) virus was observed in South Korea, a country considered to have a modern and sophisticated health care system [6]. This led to the closure of more than 700 schools and 1,369 people to be put under quarantine by the 3rd of June. Another ongoing epidemic concerns the Ebola virus which began in December 2013 and has caused more than 11,000 deaths in West Africa by the same date [30].

Human history is full of examples of such epidemics. To realise the scale of disaster which can occur if we do not take effective measures in time one has to consider the case of the 1918 influenza pandemic ("Spanish flu") which infected 500 million people and killed more people than World War I [37]. The World Health Organisation (WHO) estimates that 25% of human deaths are caused by infectious diseases [25].

Epidemiologists analyse the evolution of infectious diseases over time and attempt to create mathematical models which characterise their dynamics. These models are assessed based on their ability to explain what already happened as well as predict the number of infectious individuals in the near future. By varying a model's parameters and conducting simulations the results of possible containment measures are evaluated. This kind of analysis provides valuable input to the authorities and health organisations. When the spatial dimension is also examined and modelled explicitly, the capabilities of the model are increased significantly, as it can now make forecasts by area. The increase in the sophistication of the model accounts for the fact that disease parameters are not

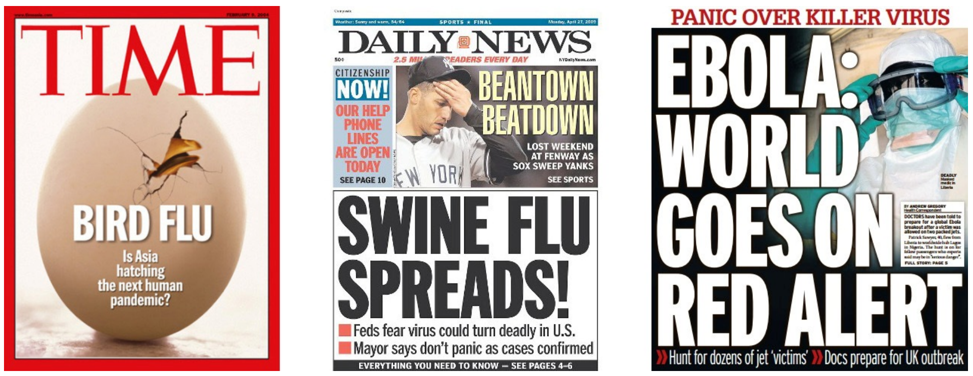

TIME
BIRD FLU
Is Asia hatching the next human pandemic?
DAILY NEWS
CITIZENSHIP NOW! OUR HELP PHONE LINES ARE OPEN TODAY
SEE PAGE 10
BEANTOWN BEATDOWN
LOST WEEKEND AT FENWAY AS SOX SWEEP YANKS
SEE SPORTS
SWINE FLU SPREADS!
Feds fear virus could turn deadly in U.S.
Mayor says don't panic as cases confirmed
EVERYTHING YOU NEED TO KNOW – SEE PAGES 4–6
PANIC OVER KILLER VIRUS
EBOLA: WORLD GOES ON RED ALERT
Hunt for dozens of jet 'victims'
Docs prepare for UK outbreak

Figure 1.1: Headlines about recent epidemics.

homogeneous across regions, but it inherently affects the computational complexity of the simulation. This is due to the existence of additional components (and parameters) which describe how people move between regions.

The process of building such models and using them to predict where a disease will strike next involves a high degree of uncertainly. This is mainly due to the fact that there is only one trace of cases which is observed in an outbreak to which the model must fit. If the observed trace does not represent accurately the actual events during the epidemic the model's predictions will lie on a false trajectory. Unfortunately, the accuracy of data collected by health organisations is often questioned as there exist cases which are reported late or not at all. Furthermore, a good spatio-temporal model requires accurate data on human mobility patterns, which is difficult to obtain due to practical and legal obstacles. The task of epidemiologists is therefore very challenging.

Disease epidemics are the most serious as they are a matter of life and death. They are not however the only kind of epidemic phenomena which is worth examining in the present era. The technological evolution has made the world susceptible to a new family of epidemics called *socio-technological* [29]. The Internet, in particular, has increased the speed and breadth of information dissemination to the extent that anyone can begin a pandemic from any place using just a connected device. Famous examples include the 2012 video clip "Gangnam style" which now has more than 2 billion views on YouTube, the 2013 "Harlem shake" meme and the 2014 "ALS ice bucket challenge" which immediately went viral on online social media. Computer viruses constitute another example of Internet-based epidemic phenomena. For instance, the 2001 Code Red worm at its peak was infecting more than 2,000 new hosts every minute [23]. The analysis and modelling of such phenomena has many applications including marketing, capacity planning, Online Social Network (OSN) studies and advertising.

Several studies have shown that the same family of mathematical models which is used to describe biological epidemics is applicable to the case of socio-technological ones too. These include an analysis of how information spreads online [9], a framework for fitting such models to Internet-based phenomena on-the-fly [28] and the modelling of the Code Red worm [47]. By simulating these models we obtain estimates to questions like "when will the epidemic reach its peak?" and "when will it die out?" In the case of diseases, spatio-temporal modelling and analysis allows us to answer questions regarding the role of geography in the spreading too such as "where is the disease likely to hit next?" Very few studies exist however that attempt to assess whether, and to what degree, the physical space is an important factor in socio-technological epidemics. If we are able to show that the physical location of Internet users plays some role indeed then we can visualise an appropriate model on a map and make predictions about for example which area will have more activity. Such insight may improve our marketing, advertising and capacity planning strategies.

1.2 Problem statement

There exist several spatio-temporal simulation frameworks which have been developed to assist researchers in modelling and analysing biological epidemics such as Surveillance[1], STEM[2] and GLEAMviz[3]. They are however fundamentally tied to the kind of epidemic phenomena for which they were created, lacking the flexibility which would allow one to study the spatio-temporal spreading of other types. The reason they are not suitable for modelling socio-technological epidemics is because they are built on the assumption that transmission depends on physical contact, and contact depends on physical movement. Transmission over technological channels is of course not constrained by these factors. Furthermore, even for the case of biological epidemics, existing frameworks tend to be strict on the types of models they can simulate and they also lack the mechanisms to provide quick feedback by visualising the model and the data set side-by-side. Moreover, they often have over-complicated procedures for basic operations such as visualising data sets and constructing simulation scenarios.

The primary objective of this project is to implement a framework which is suitable for modelling and analysing both biological and socio-technological epidemics. We also aim to deliver a product which tackles some important limitations and omissions of existing applications. The main challenge lies in the architecture of the framework; it should be flexible enough to allow several modes of simulation. Moreover, it should enable the users to test the significance of geographical factors like distance and population density in the evolution of both kinds of epidemics by providing a common model capable of characterising both physical movement and information diffusion.

The main idea is to make the notion of space a key factor in the analysis. By having a flexible, amendable and extensible spatial model we can perform simulations using the perspectives of physical areas, logical networks of individuals or a mixture of both. The first perspective allows us to simulate biological epidemics in a similar manner as the existing frameworks. The second one allows us to simulate epidemics over an abstract graph in which the connections between individuals are explicitly modelled. The third perspective is the most powerful as it simulates the spreading over both notions of space simultaneously, allowing us to define a network of individuals and allocate them on a map based on some geographical factors. This allows us to model and simulate socio-technological epidemics where information propagates through the logical network of the individuals involved while the space which is visualised concerns their physical locations.

[1]http://surveillance.r-forge.r-project.org
[2]https://www.eclipse.org/stem/
[3]http://www.gleamviz.org

We also present a novel way of using existing models which describe the spatial spreading of biological diseases to account for the role of geography in socio-technological epidemics. In essence, we extend these so-called *mobility models* to capture their characteristics in additional output formats. For example, consider a human mobility model which favours movements towards close-by or urban areas. We can use it in the case of disease modelling to compute estimations of the average number of people moving between two regions in a unit of time. We can also use it to construct the contact network of some individuals in such a way that the majority of one's connections reflect people in places which are close to them or have high population density.

1.3 Objectives

The objectives on this project are:

- Develop a framework which is suitable for the spatio-temporal visualisation, modelling, simulation and analysis of general epidemics from a single trace. It should provide the means to easily visualise data sets and test various kinds of disease and mobility models in both physical and logical space.
- Demonstrate the framework's applicability to biological disease modelling by conducting a case study of a contemporary epidemic.
- Demonstrate the framework's applicability to socio-technological epidemic phenomena by conducting a case study of information diffusion in an online social network.

1.4 Contributions

The project has made the following contributions:

- Development of a framework in Java which is capable of performing spatio-temporal visualisations and simulations of both biological and socio-technological epidemics (Chapter 3). It offers flexibility in the kind of spatial, disease and mobility models which can be used. It generates interactive visualisations on various types of real-world maps as well as on graphs of different properties. It provides the means to compare the model against the real data or compare two models side-by-side for direct and immediate feedback. It allows the user to simulate the results of certain containment measures. It features a flexible model fitting procedure and a built-in synthetic data generator. Moreover, its modular design makes it easily extensible to cater for additional models.
- Analysis, modelling and simulation of the 2014 West Africa Ebola outbreak (Chapter 4). We visualise the spread of the epidemic in Guinea, Sierra Leone and Liberia. By analysing the data obtained from WHO we demonstrate that there is correlation between the number of cases in a region and (i) its distance from the epicentre or the

closest capital, (ii) its population and (iii) its population density. We then proceed to show that a sophisticated disease model combined with a simple model of human mobility such as the *Gravity model* are capable of explaining the spatio-temporal spread to a satisfactory level.

- Analysis, modelling and simulation of Twitter activity regarding the following events in London in 2015: the marathon, the birth of Princess Charlotte and the general elections (Chapter 5). We collect and visualise the geotagged tweets related to these events. By analysing the activity on the day of each event we investigate whether a spatio-temporal model may be applicable to explain it. We then propose an appropriate form of a spatio-temporal model regarding tweeting and retweeting which is compatible with our findings, simulate it by estimating its parameters from the actual data and evaluate the goodness of fit.

- Assessment of the role of geography in location-based events involving Twitter (Chapter 5). We examine the role of eye-witnesses in the beginning of each epidemic in our data sets. We verify that, even at the scale of a single city, the contact network among Twitter users is influenced by the physical distance between them. Specifically, we observe that the majority of the senders' followers in our data sets are located close to them. We then demonstrate how to account for geography during the construction of a network of users using the same mobility models developed for biological epidemics.

Chapter 2

Background

2.1 Epidemiology

Epidemiology is the study of the causes, patterns and effects of contagious diseases. It has emerged as a science to assist health organisations in their efforts to prevent or limit the impact of epidemics e.g. via containment or mass vaccination. In general, infections occur by direct contact with an infected person or animal, via contaminated food or water and by airborne diseases.

Epidemic outbreaks have an *index case*, which refers to the initial patient. This patient infects the people around them and, depending on the movement habits of the population, the disease spreads to other regions. If the epidemic spreads across a significantly large area (such as multiple continents) then it becomes a pandemic, meaning that it can potentially affect all people.

Epidemiologists quantify how contagious a disease is by calculating the *basic reproductive number*, R_0. This is the average number of infections an infected individual causes. This metric is used to determine whether the disease can spread and how hard it is to contain it. When it drops below 1 the epidemic ceases to exist.

2.2 Mathematical modelling

Models for biological epidemics are typically divided into deterministic and stochastic. Deterministic models describe the average behaviour of the population under study and thus lead to reproducible results. On the other hand, stochastic models produce different results each time as they contain some element of randomness. Because the evolution of an epidemic depends on the random interactions of people within the population, stochastic models tend to be more accurate. However, as a consequence of the "law of large numbers" [8], when the population is sufficiently large these random interactions cancel each other out, making the simpler deterministic models good enough to use.

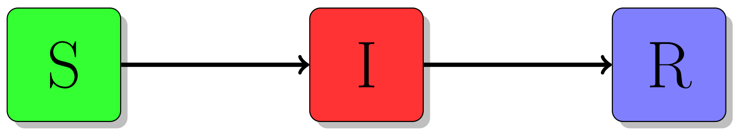


Figure 2.1: Transitions between compartments in the SIR model.

Stochastic simulation algorithms such as the Gillespie algorithm [10] are useful for generating random traces of epidemics using deterministic models. These traces can be used to evaluate the performance of the deterministic model by comparing the true parameter values against the estimated ones obtained in the process of fitting the model to the data.

2.3 Deterministic compartmental models

The most popular deterministic models are the *compartmental* ones. These divide the population into disjoint sub populations such as the Susceptible, Infected and Recovered individuals. People transition from one compartment to another with rates given as parameters. The rates at which the populations in the compartments change over time are modelled using Ordinary Differential Equations (ODEs), the solution of which at regular time intervals gives the size of each compartment. By plotting the number of individuals in each compartment over time we can estimate the duration of the epidemic, as well as when it will peak.

2.3.1 SIR model

The Susceptible-Infected-Recovered (SIR) model is a classical epidemic model devised by Kermack and McKendrick in 1927 [15]. It divides a population into three compartments:

S(t) the number of individuals who are susceptible to the disease at time t.

I(t) the number of infected individuals at time t.

R(t) the number of individuals who have recovered (or died) at time t.

The total number of individuals in the population (N) is constant and equal to the sum of S(t) + I(t) + R(t) for every t. Figure 2.1 shows the transitions between the compartments of this model.

The following ODEs define the sizes of the compartments over time:

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

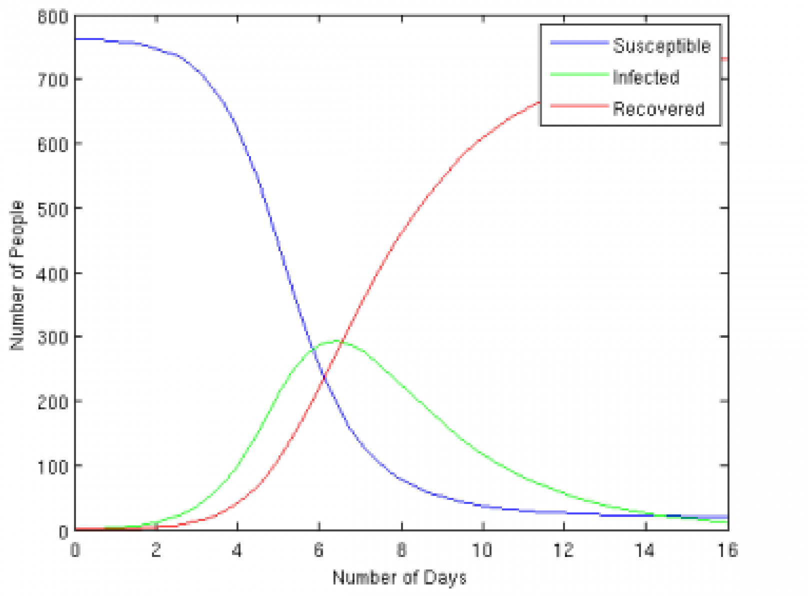


Figure 2.2: The number of individuals in each compartment over time for an epidemic simulated using the SIR model.

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

where β is the rate at which an infected individual comes into contact with a susceptible one and γ is the rate at which an infected individual recovers. Recovered individuals have immunity. Figure 2.2 shows a possible solution of the ODEs.

The basic reproductive number for this model is:

$$R_0 = \frac{\beta}{\gamma}$$

2.3.2 SEIR model

There exist several variations of the SIR model. One of them is the Susceptible-Exposed-Infected-Recovered (SEIR) model which includes an Exposed compartment, as shown in Figure 2.3. This consists of the individuals who are infected but not yet infectious. The ODEs are:

$$\frac{dS}{dt} = -\frac{\beta SI}{N}$$
$$\frac{dE}{dt} = \frac{\beta SI}{N} - \alpha E$$
$$\frac{dI}{dt} = \alpha E - \gamma I$$

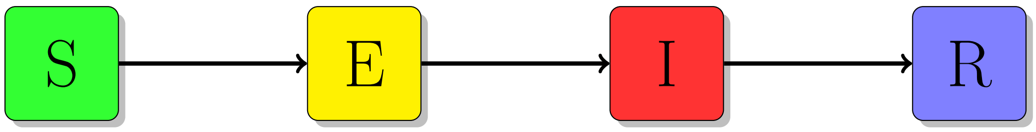


Figure 2.3: Transitions between compartments in the SEIR model.

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

where α is the rate at which an exposed individual becomes infectious.

The choice between different compartmental models depends on the application. For example, the SEIR model can potentially capture the dynamics of spreading news (or rumours) better than the SIR model. In this case, individuals become exposed to the news when they receive them, but may choose not to share them i.e. they transition directly to the Recovered state. They become infectious only if they decide to act upon receiving the news in a way that allows the information to propagate further. Taking Twitter as an example, users are considered infectious only if they actively respond (by retweeting, replying or sending a related tweet) to a tweet they receive.

2.4 Modelling biological epidemics

Epidemiologists around the world try continuously to come up with mathematical models which target specific biological diseases. As each time an epidemic is formed we only observe a single trace, coming up with tools which capture the dynamics of it in the general case is inherently hard. The task gets more difficult because of the various assumptions that are typically made in order to make the simulation tractable such as homogeneity of the population under study. Furthermore, the quality of public health data used to train the models is questionable as this kind of data is very sensitive to under-reporting and the existence of spikes (when cases are reported late). Nevertheless, the models can still be useful in predicting when the epidemic will reach its peak or when it will cease to exist. Most importantly, they can be used to improve control strategies by evaluating the impact that different kinds of interventions are predicted to have.

The recent 2014 West Africa Ebola outbreak has attracted a lot of attention since it was declared an epidemic, while governments and international organisations still cooperate to prevent a new pandemic. So far the virus has caused severe damage mainly in Guinea, Sierra Leone and Liberia. Figure 2.4 shows the geographical distribution of new and total confirmed cases in these countries until 27 May 2015.

In *Understanding the dynamics of Ebola epidemics* [18], Legrand *et al* proposed a compartmental model specifically for the case of Ebola in Africa, shown in Figure 2.5. It is an extension of the SEIR model that attempts to capture the dynamics of hospitalized infectious individuals and dead individuals who can cause additional infections because of unsafe burial practices. It consists of the following compartments:

S the individuals susceptible to the disease.

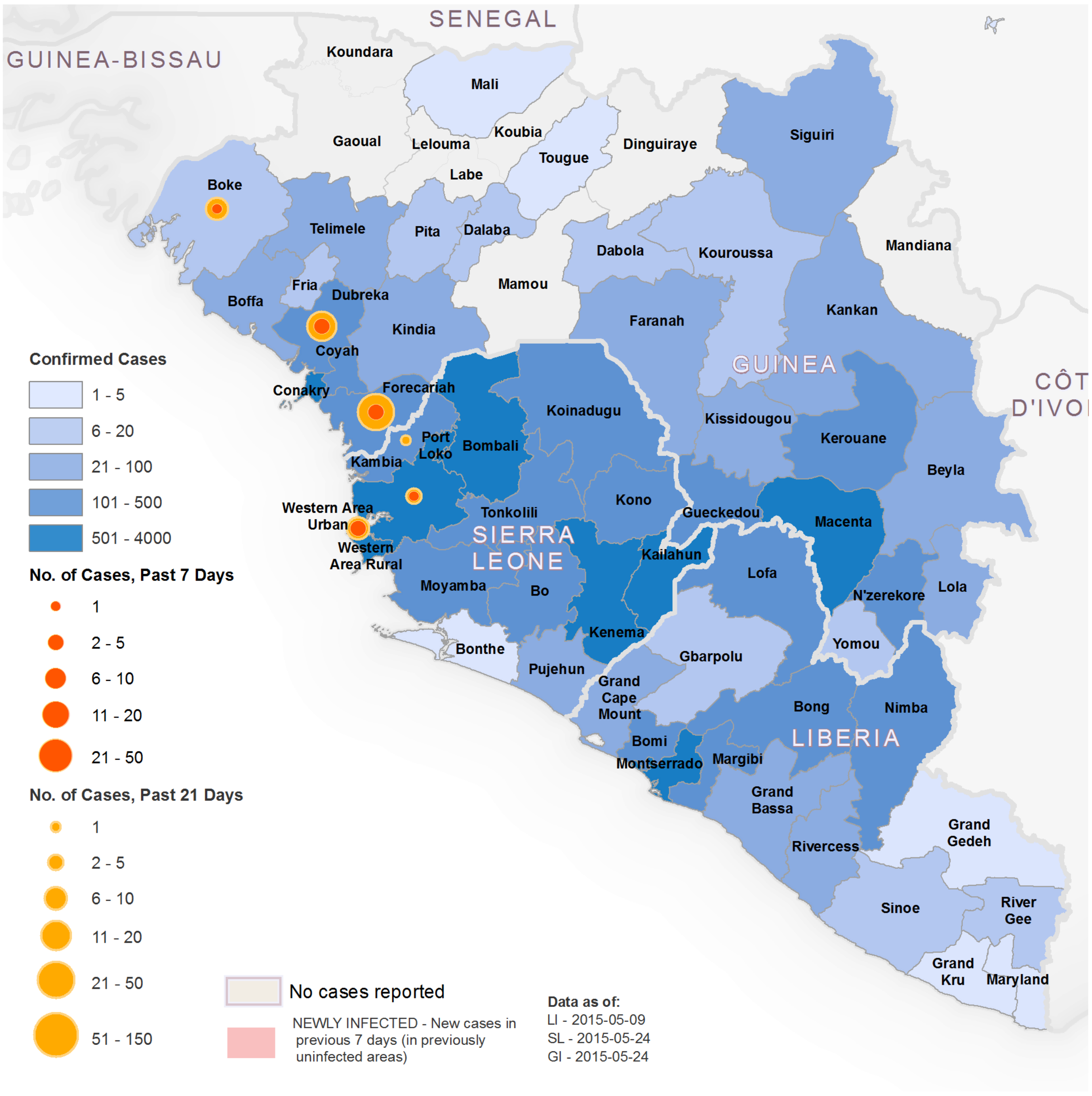


Figure 2.4: Distribution of confirmed cases in Guinea, Sierra Leone and Liberia until 27/05/2015. Source: WHO.

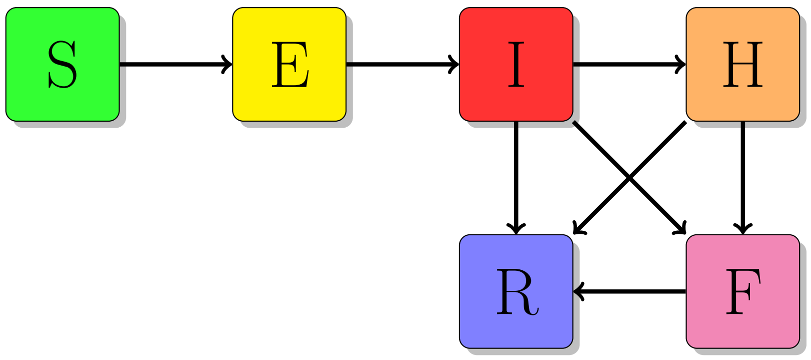


Figure 2.5: Transitions between compartments in Legrand's Ebola model.

E the individuals infected by the disease but not yet infectious.

I the infectious individuals.

H the individuals who are in the hospital.

F the individuals who are dead but not yet buried.

R the individuals who are no longer transmitting the disease.

There exist several studies on the 2014 Ebola epidemic which are based on the above model. In *Predicting the extinction of Ebola spreading in Liberia due to mitigation strategies* [41] the authors use a variant of this model to describe the evolution of the epidemic in Liberia and to make predictions regarding its end. Using 1000 stochastic runs, their model achieved a high goodness of fit when trained with data representing the cumulative cases in Liberia as a whole, as shown in Figure 2.6. Their predictions regarding the arrival time of the disease in each region in Liberia (shown in Figure 2.7) were also compatible with the data from WHO at a 95% confidence interval. By simulating several interventions, they argue that reducing mobility is insufficient to prevent the epidemic from spreading to new areas as it only manages to delay it by a few weeks.

2.5 Modelling socio-technological epidemics

2.5.1 Growth and Spike models

There are several studies that attempt to characterise how information spreads online. An interesting one comes from researchers at Facegroup [9], a "global strategic insight agency that delivers socially intelligent research". They analysed the spread of popularity of four kinds of videos by measuring the degree of sharing activity they caused, and suggested that there are two main patterns of information diffusion. The first one (called "spike") represents a sudden "explosion" of sharing activity which shortly dies out, whereas the second one (called "growth") represents a slower but more organic growth in the popularity of the content.

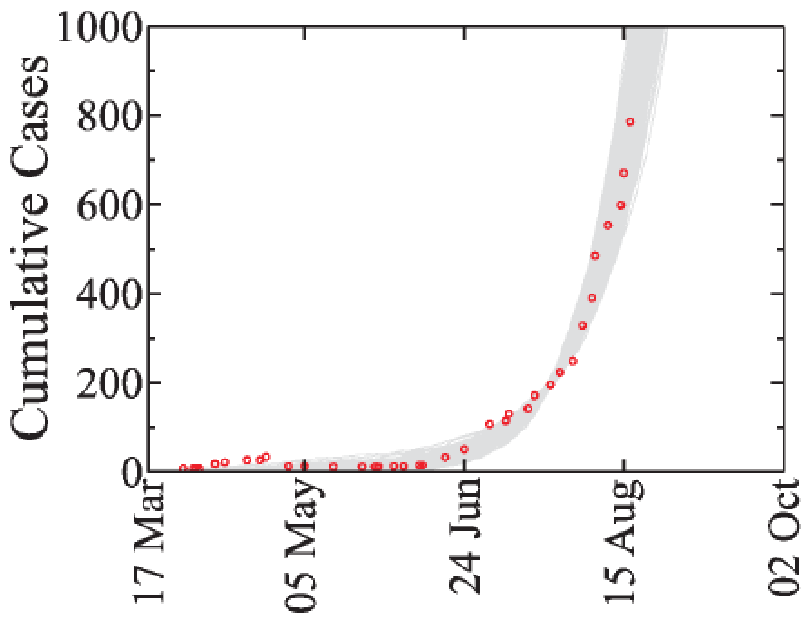


Figure 2.6: Cumulative cases of Ebola in Liberia in 2014. The dots represent the observed cases while the grey lines represent the trajectories predicted by the model. Source: [41]

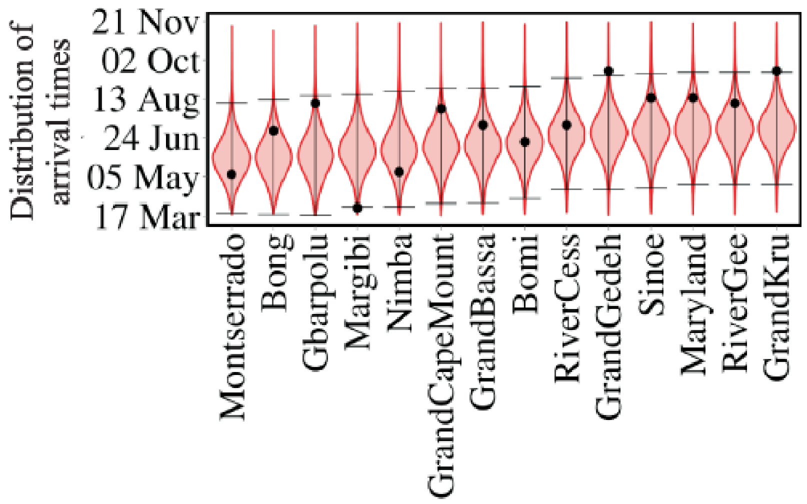
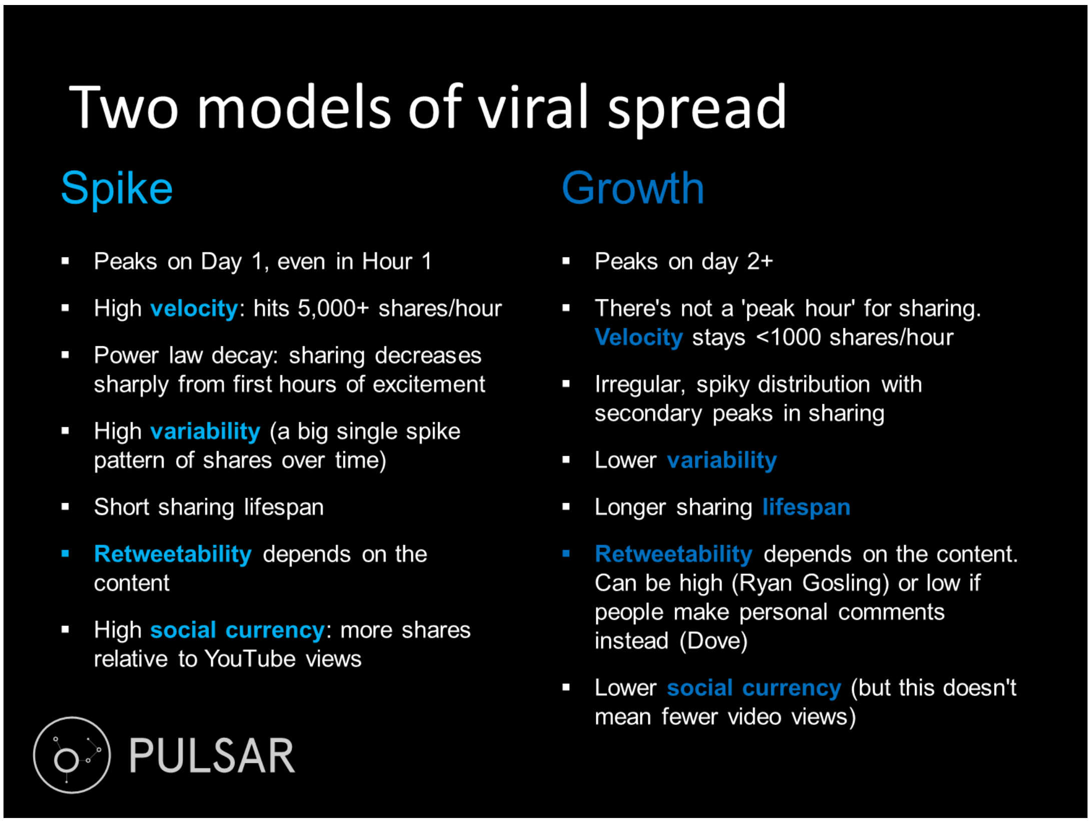


Figure 2.7: Arrival times of Ebola in each region of Liberia. The dots represent the observed times while the Violin plots describe the distribution of the model's predictions. Source: [41]



Two models of viral spread

Spike

- Peaks on Day 1, even in Hour 1
- High **velocity**: hits 5,000+ shares/hour
- Power law decay: sharing decreases sharply from first hours of excitement
- High **variability** (a big single spike pattern of shares over time)
- Short sharing lifespan
- **Retweetability** depends on the content
- High **social currency**: more shares relative to YouTube views

Growth

- Peaks on day 2+
- There's not a 'peak hour' for sharing. **Velocity** stays <1000 shares/hour
- Irregular, spiky distribution with secondary peaks in sharing
- Lower **variability**
- Longer sharing **lifespan**
- **Retweetability** depends on the content. Can be high (Ryan Gosling) or low if people make personal comments instead (Dove)
- Lower **social currency** (but this doesn't mean fewer video views)

PULSAR

Figure 2.8: Facegroup's Spike and Growth models.

The study uses a series of metrics for measuring popularity, such as the number of shares per hour and the number of retweets an original tweet has caused. An interesting one is social currency: to what degree does a user believe that a particular piece of content is relevant to their friends and followers.

In *Synthedemic Modelling and Prediction of Internet-based Phenomena* [28], Marily Nika describes a framework for modelling online phenomena by synthesizing multiple Spike and Growth models. Spike is modelled by an IR process where Growth is modelled by an SIR process.

The Infected-Recovered (IR) model is characterised by the initial number of infected individuals I_0 and the recovery rate γ. The ODE is:

$$\frac{dI}{dt} = -\gamma I$$

There are researchers who say that the SIR family of models is too simple to explain online sharing phenomena. In the *Absence of influential spreaders in rumour dynamics* [4], the authors applied traditional epidemic models to see if they could fit the actual data. They concluded that a key component was missing from these models: influential spreaders. Influential spreaders, such as reputable news reporters and celebrities, tend to have the power to accelerate or quash the rate of information dissemination dramatically. Therefore, according to the authors, their presence should be taken into consideration and modelled appropriately. Such a model was proposed by Avrachenkov et al. in *Information*

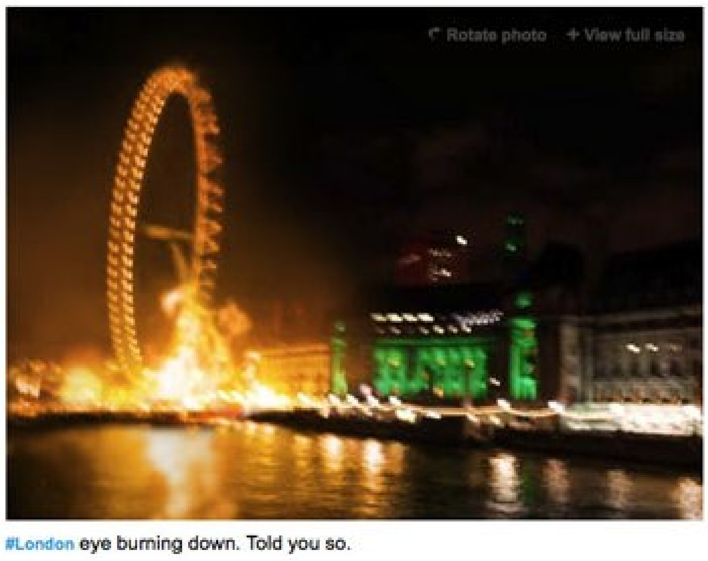


Figure 2.9: A fake photograph circulating on Twitter during the 2011 London riots. Source: Guardian.

dissemination processes in directed social networks [2]. It uses the degree distribution of nodes in a directed network to describe the different levels of influence that users in OSNs have on their connections.

2.5.2 Information diffusion in Twitter

Twitter is a popular online social networking service that enables users to exchange 140-character messages called "tweets". Users can subscribe to other users' tweets; we say that they are "following" them. This kind of relationship is unidirectional as one user may follow another but the second may not follow the first. According to Twitter itself [40], as of May 2015 the social network has more than 500 million users, 302 million of which are considered active. The average number of tweets sent in one day is 500 million.

Twitter users can share information they receive by *retweeting* it, i.e. broadcasting it to all their followers. This makes Twitter a powerful tool for spreading both news and false information. Evidence comes from incidents such as the 2011 London riots. In *How rumours spread on Twitter* [12], analysts at Guardian and the University of Manchester examined the tweets related to false rumours like "rioters attack London zoo and release animals", as well as whether the tweets reinforced the rumours or helped to suppress them. Although they conclude that rioters themselves did not use Twitter to cause more conflicts, they do acknowledge the fact that it is effective at both spreading misinformation at high speeds but also at dispelling it in the presence of strong counter-evidence.

The ease by which information can spread (which often gives rise to trending topics) has called on a lot of researchers to study the network's dynamics. In *Epidemiological Modelling of News and Rumours on Twitter* [14], the authors use the SEIZ model to describe the spread of 2013 news such as the Boston marathon bombings, as well as rumours such as Fidel Castro's death. The SEIZ model has the following compartments:

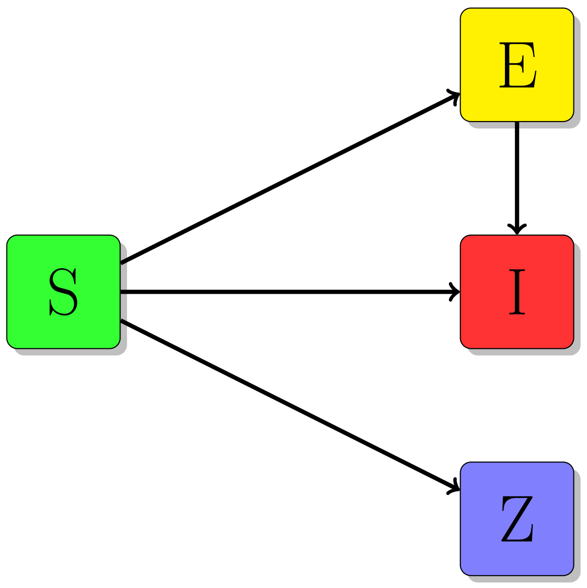


Figure 2.10: Transitions between compartments in the SEIZ model.

Susceptible (S) users who have not yet received the news.

Exposed (E) users who have received the news but take some time before sharing it.

Infected (I) users who have tweeted about the news.

Sceptic (Z) users who have received the news but chose not to share it.

Users transit from the Exposed compartment to the Infected one when they have further contact with an infected individual, or after sufficient time has passed. Note that there is no Recovered sub population, as all infected individuals are assumed to recover immediately after sharing the news. The authors argue that by explicitly modelling the users who do not spread the news or who take some time to believe it, the results are more accurate than using simpler compartmental models.

There exist also studies that use geotagged tweets in order to explore biological diseases. One example is the study described in the New Scientist article titled *AI predicts when you are about to get sick* [33]. In this case the researches were able to predict when people were beginning to show symptoms of flu (and tweet about it) with about 90% accuracy within eight days of being in the same place with people who tweeted that they were sick. Google uses a similar approach to estimate flu activity via its Google Flu Trends service. In this case, data such as how often people search for words related to flu are aggregated to produce a map of high and low activity in real time.

In *Spatio-Temporal Analysis of Topic Popularity in Twitter* [1] the authors consider the question "can the nuances of the temporal evolution of topics be explained by a more thorough study of their spatial evolution?" They consider two notions of the term "spatial": the network topology of *following* relationships among users and the actual geographical location of the users. For the former they conclude that users with a high number of followers have a strong impact on popularity, whereas for the latter they note that popular topics tend to spread to neighbouring regions more frequently and to a greater degree than not so popular ones. This last observation provides motivation for coming up with an epidemic model for activity in Twitter that takes geography into consideration.

In *Event Detection using Twitter: A spatio-temporal approach* [5], the authors suggest the use of space-time scan statistics (STSS) for detecting events. This method examines the data set across both time and space looking for clusters that arise when there is an outbreak of tweeting activity. They used the software SatScan [1] which, among other things, "performs geographical surveillance of disease to detect space-time disease clusters and to see if they are statistically significant." The analysis of their data set is not complete however. Like many other studies and software that attempt to discover and visualise trending topics (such as trendsmap.com), retweets of tweets on these topics are ignored. As retweeting is an essential component in information dissemination on Twitter, we think that any good spatio-temporal analysis of popular topics should analyse the distribution of retweets too.

In *Geography of Twitter Networks* [36] the authors support the general hypothesis that users tend to follow people who are geographically close to them. Specifically, their analysis shows that 39% of social ties are shorter than 100 km and ties up to 1000 km are far less common when compared with a random network. Thus, geography is likely to play some role in the evolution of epidemics across OSNs.

2.6 Agent-based models

Unlike traditional compartmental models which treat the population as homogeneous, agent-based models describe the behaviour at the individual level. This allows them to account for the heterogeneity in the properties of the individuals (such as gender, age and household income) as well as their interactions with others (through their mobility profile). Each "agent" can also have their own set of values for the parameters of the disease model (such as β and γ), making the model more realistic.

Incorporating heterogeneity into a model increases however the computational complexity as the simulation framework needs to keep track of the state of each individual explicitly. For this reason agent-based models are typically used when the population is small.

Agent-based models tend to describe the diffusion of information on an OSN better as there is often a lot of difference between the users' contact networks and activities. On the other hand they are less suited for the case of health surveillance as the process of decomposing aggregated data (such as weekly cases and deaths published by WHO) has high uncertainty [25].

[1]http://www.satscan.org

2.7 Spatio-temporal modelling

Modelling epidemic outbreaks in both time and space allows us to make better observations and draw more informed conclusions. Despite the increase in the complexity of the model and the simulation time, the benefit of being able to make predictions by area makes the use of spatio-temporal models appealing.

Spatio-temporal compartmental models typically make use of a mixing matrix, M. Each entry in this matrix defines the rate at which people move between two regions and it is constructed using a suitable *mobility model.* The population of every region is divided into compartments, and the ODEs of the disease model are adapted to account for the mobility patterns in the population.

Consider to following spatially-enriched SIR model [25]:

$$\frac{dS_i}{dt} = -\frac{\beta S_i \sum_{i=0}^{n} M_{ij} I_j}{\sum_{i=1}^{K} M_{ij} N_j}$$

$$\frac{dI_i}{dt} = \frac{\beta S_i \sum_{i=0}^{n} M_{ij} I_j}{\sum_{i=1}^{K} M_{ij} N_j} - \gamma I_i$$

$$\frac{dR_i}{dt} = \gamma I_i$$

Here K is the number of regions, N_i is the population in region i and M_{ij} is the mixing rate from j to i. Note that $M_{ii} = 1$. The distinguishing characteristic that this version of SIR has is that the number of cases in a region is determined not only by the number of infections that occur between its residents but also by the infections that occur due to contact with people coming from other regions.

2.7.1 Mobility models

Mobility models are used to describe how people move from one area to another in both the short term (commuting) and the long term (migration).

Gravity

The Gravity model [34] is based on Newton's gravitational law. It assumes that the number of people moving between regions is proportional to the product of the populations at the origin and the destination (the *masses*), and inversely proportional to the distance between them. It has been used successfully to describe population movement based on data from numerous sources, such as census and call detail records (CDRs). It is expressed by the following formula:

$$T_{ij} = k\frac{p_i^\alpha p_j^\beta}{d^\gamma}$$

where T_{ij} is the number of individuals that move between regions i and j per unit time, p_i is the population at the origin, p_j is the population at the destination, d is the distance between the two regions and α, β, γ, k are parameters.

Radiation

The Radiation model [35] is used to describe how people commute. It is based on the assumption that a person chooses the closest job to their home which offers higher benefits than the best deal available in their home region. It is expressed by the following formula:

$$T_{ij} = T_i\frac{p_i p_j}{(p_i + s_{ij})(p_i + p_j + s_{ij})}$$

where T_{ij} is the number of individuals that move between regions i and j per unit time, p_i is the population at the origin, p_j is the population at the destination, T_i is the number of commuters from region i and s_{ij} is the total population inside a circle of radius d (the distance between the two regions) centred at i, excluding p_i and p_j.

In contrast with the Gravity model, the Radiation model is parameter-free. This makes it attractive to use in cases where the data is insufficient to estimate the parameters with confidence, or where the fitting process takes a long time.

2.7.2 Graph models

Epidemics on OSNs are typically simulated on networks which have similar characteristics to their own. The space over which the epidemic spreads is defined by a *graph model*. Mobility is accounted for implicitly during the construction of the graph, as nodes can only influence their direct connections. Examples of graphs which provide the logical space for socio-technological epidemics include simple grids, Watts-Strogatz and Barabasi-Albert.

Grid

Nodes arranged in a grid have a maximum of four neighbours. A key characteristic of such topology is the lack of long distance connections (and infections). It is therefore suitable in cases of short-term human mobility or systems with large path lengths. Figure 2.11 shows an epidemic spreading over a 10 by 10 grid.

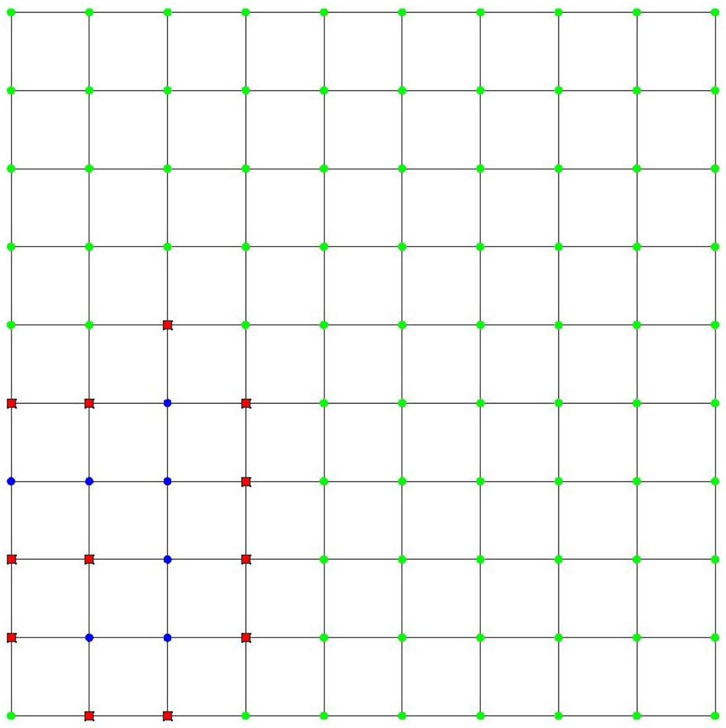


Figure 2.11: An epidemic spreading over a 10 x 10 grid. Green nodes are susceptible, red nodes are infected and blue ones are recovered.

Watts-Strogatz

Watts-Strogatz graphs [42] are random graphs with *small-world* properties, such as high clustering and short mean path lengths. This means that nodes tend to be members of cliques, and are able to reach every other node in a relatively small number of hops. These characteristics are analogous to properties found in typical social networks. Figure 2.12 shows an epidemic spreading over a Watts-Strogatz graph of 100 nodes.

Barabasi-Albert

Barabasi-Albert graphs [3] are scale-free, which means that their degree distribution follows a power law. They have two important characteristics: growth and preferential attachment. The former means that the number of nodes increases over time whereas the latter means that, the more connected a node is, the greater the chance there is to receive new links added to the graph. Such networks are observed in many systems including some social networks: a newcomer is more likely to become friends with a well-connected, reputable person. A Barabasi-Albert graph is therefore suitable to model epidemics over OSNs whose connections obey a power law. Figure 2.13 shows an epidemic spreading over a Barabasi-Albert graph of 100 nodes.

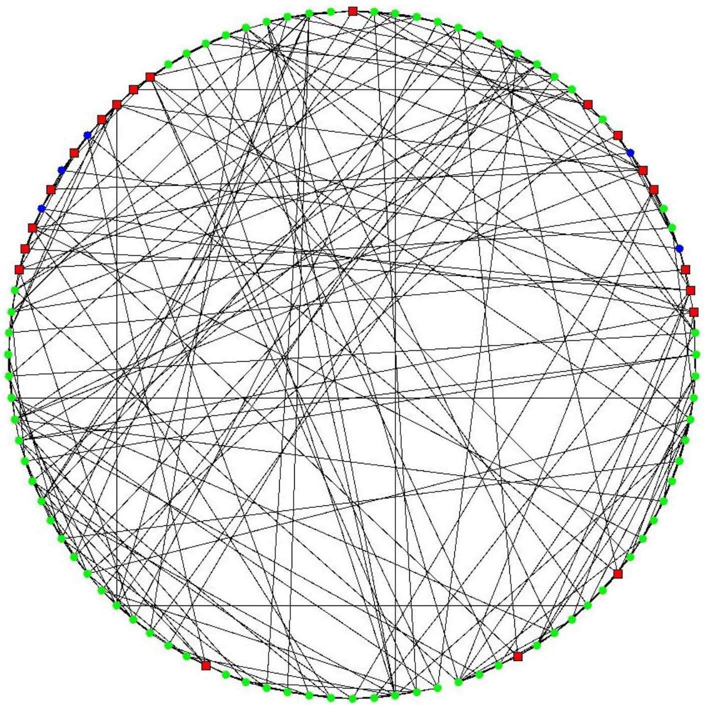


Figure 2.12: An epidemic spreading over a Watts-Strogatz graph of 100 nodes.

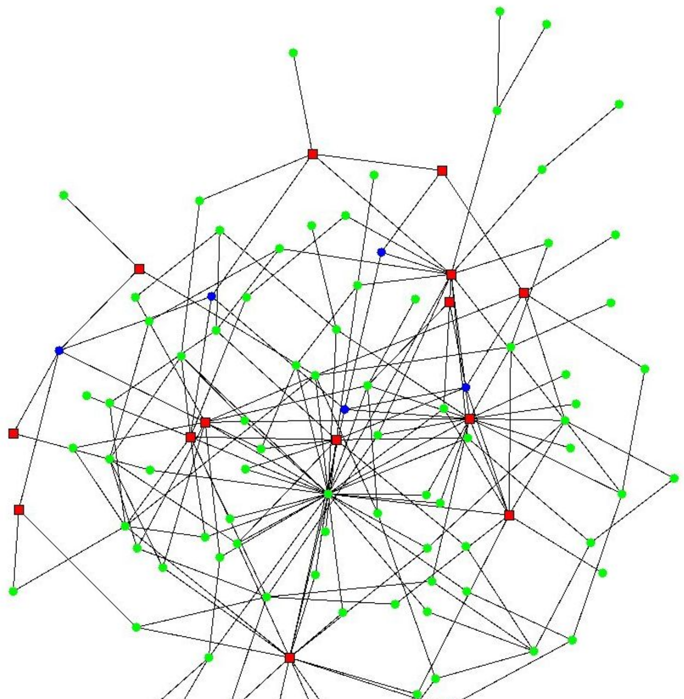


Figure 2.13: An epidemic spreading over a Barabasi-Albert graph of 100 nodes.

2.8 Model fitting

Fitting a model to the real-world observations requires finding the vector of parameters for the model (θ) which produces the best fit to the data. A simple method is to minimise the sum of the squared errors (i.e. the differences between the observed value, y_i, and the value predicted by the model, $f_i(\theta)$). This is described by the following **Least Squares** optimization problem:

$$\arg\min_{\theta} \sum_{i=0}^{n} (y_i - f_i(\theta))^2$$

A popular method for solving such non-linear unconstrained optimization problems is the **Nelder-Mead algorithm** [26]. This works by maintaining a simplex S in $\mathbb{R}^n$ (the convex hull of $n+1$ vertices) as well as the corresponding set of function values at these points. The method tries to decrease the function values at the vertices by performing transformations on S and terminates when the values are close to each other. The fact that it does not require the computation of function derivatives makes it suitable for estimating model parameters where the objective function may not be unimodal or smooth.

We can assess how well the model (with the optimal θ) fits the data by using an appropriate goodness of fit. An example of such statistic is the **coefficient of determination**, R^2. This is the ratio of the variance present in the observations that is explained (i.e. predicted) by the model to the total variation. It is defined as follows:

$$SS_{res} = \sum_{i=0}^{n} (y_i - f_i(\theta))^2$$

$$SS_{tot} = \sum_{i=0}^{n} (y_i - \bar{y})^2$$

$$R^2 = 1 - \frac{SS_{res}}{SS_{tot}}$$

where $\bar{y}$ is the sample mean.

The coefficient of determination quantifies the certainty of making predictions from the model. In the case there is a perfect match between the observed and predicted values, R^2 will be 1. If R^2 is close to zero, or even negative, then the model is not a good fit to the data.

2.9 Gillespie's Stochastic Simulation Algorithm

Dan Gillespie presented a discrete-event simulation algorithm [10] in 1976 capable of producing correct trajectories of a stochastic process. Its main use within the context of mathematical modelling of epidemics it to produce stochastic traces of deterministic systems. Essentially, the algorithm executes two main steps repeatedly. Firstly, it decides on the time at which the next event will occur. It then chooses the type of the event based on the cumulative probability distribution of the rates of all the events.

Consider the case of the simple SIR model. Table 2.1 shows the kinds of events which occur in this model as well as their rates.

Event type	Event rate
Transition from S to I	$\frac{\beta SI}{N}$
Transition from I to R	γI

Table 2.1: Events in the SIR model.

Let σ be the sum of the rates of all possible events in this system. The time to the next event τ is chosen from an exponential distribution with rate parameter σ. The type of the event is chosen at random from the following probability distribution:

$$P(\text{Event} = \text{e}) = \frac{\text{rate of e}}{\sigma}$$

The simulation time is incremented by τ and the sizes of the compartments change according to the event that occurred. The procedure is then repeated.

The algorithm described above refers to Gillespie's *direct* method. Despite being exact, it suffers from high computational cost. There exist several variants which provide optimised performance such as *next reaction* and *tau-leaping*.

2.10 Related work

In this section we present briefly three recent frameworks which were developed to assist epidemiologists with spatio-temporal modelling.

2.10.1 Surveillance

Surveillance is an R package for the modelling of epidemic phenomena using statistical methods. Its features include outbreak detection mechanisms, data visualisation and model fitting. It is also capable of handling location data expressed both as exact coordinates as well as whole geographical regions. Finally, it contains some real-world data sets to help researchers get familiar with it faster such as the reported cases of measles by week and district in the Weser-Ems region of Lower Saxony, Germany, during the years 2001 and 2002.

The package consists of three regression-oriented modelling frameworks; perhaps the most practical one being `hhh4`. This works with epidemic data made up of cases per week and district. Such format is compatible with data published by health organisations. Figure 2.14 shows how the `hhh4` framework visualises epidemic outbreaks. The output is an HTML page demonstrating the spatio-temporal and temporal evolution of the disease simultaneously. The bottom panel is used to control the animation.

The models which are available for simulating epidemics are made up of endemic and epidemic components. The endemic part is used to capture the background risk of infections due to external factors like climate (seasonality) and immigration, and it is therefore independent of the epidemic's history. The epidemic part is autoregressive: its output depends linearly on its previous values. The model can also be given a third component representing random effects as a way of dealing with under-reporting, a common characteristic of public health data sets.

Statistical models are good for preliminary analysis because of their simplicity, allowing us to obtain predictions quickly. Their accuracy is challenged however because of their heavy reliance on the actual data. Missing values, under-reporting and uncertainty greatly affect their performance [25]. Furthermore, Surveillance visualises the spread of epidemics by generating and animating a set of image files. This mechanism does not allow one to investigate in detail the geography of the regions of interest.

2.10.2 SpatioTemporal Epidemiological Modeller (STEM)

The SpatioTemporal Epidemiological Modeller (STEM) is a Java-based tool "designed to help scientists and public health officials create and use spatial and temporal models of emerging infectious diseases." It uses compartmental disease models and offers both deterministic and stochastic simulations. Users can develop their own disease models and share them within its online community as plug-ins. The main form of visualisation (shown in Figure 2.15) consists of shapes defining the borders of the areas of interest, which can be countries, districts or cities. There is also an option to display the state of the epidemic at specific points during the simulation using Google Earth. Human mobility between regions is represented using a graph of transport edges which accounts for commuting, migration and airplane trips.

One of the great benefits of STEM is the in-house availability of geographic, demographic and transport data for numerous countries. This includes common borders, populations, rainfall, temperature, road networks and airport links. Another important feature is the ability to simulate intervention strategies such as school closures and mass vaccinations programs. This allows epidemiologists to propose measures for containing the disease to the authorities.

STEM can be used to simulate epidemics involving humans or animals, but it is restricted to a physical spatial model. This makes it unsuitable for simulating socio-technological epidemics as there is no support for constructing logical networks of individuals and using them to transmit information.

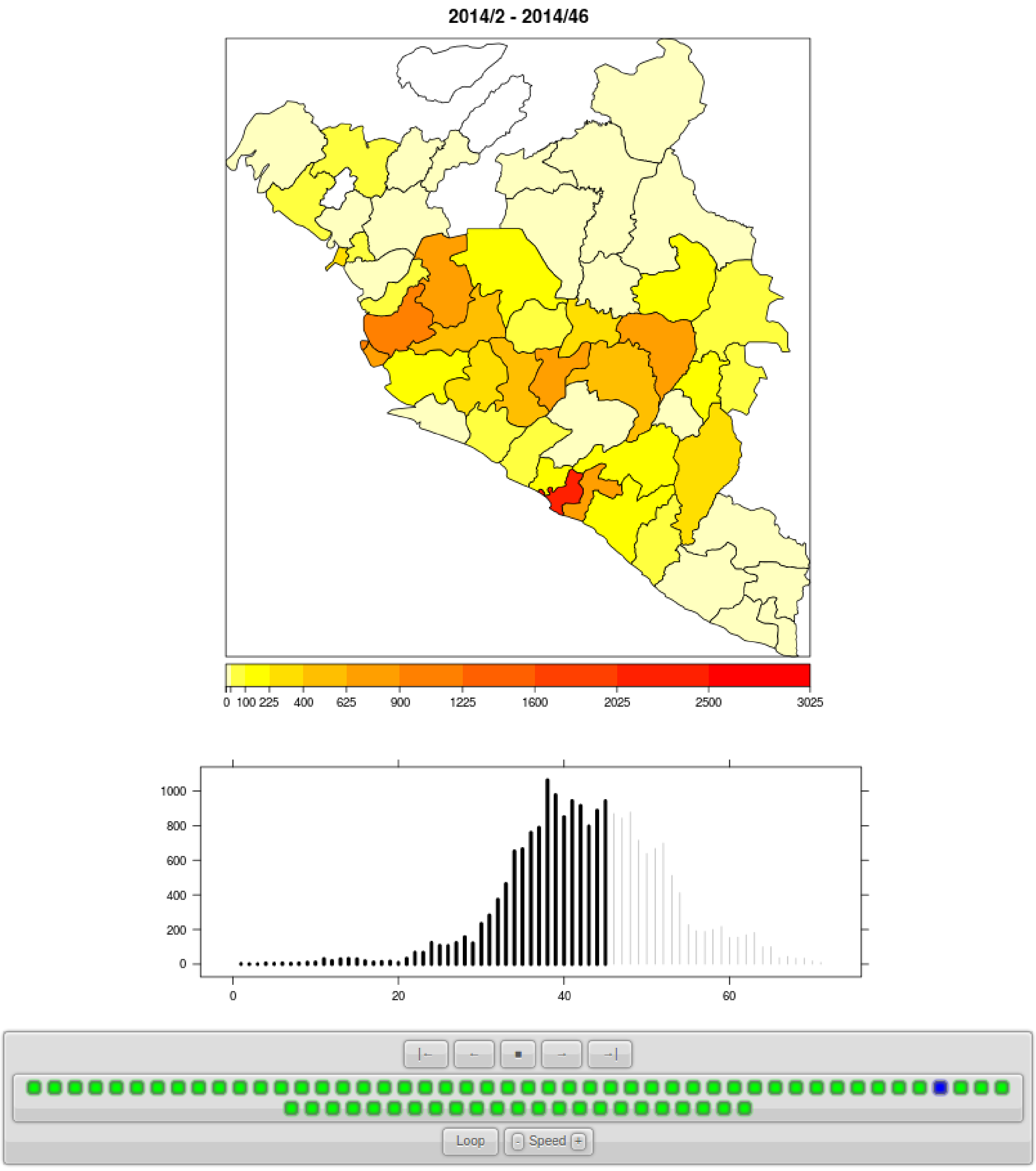


Figure 2.14: Spatio-temporal visualisation of the 2014 West Africa Ebola outbreak using Surveillance.

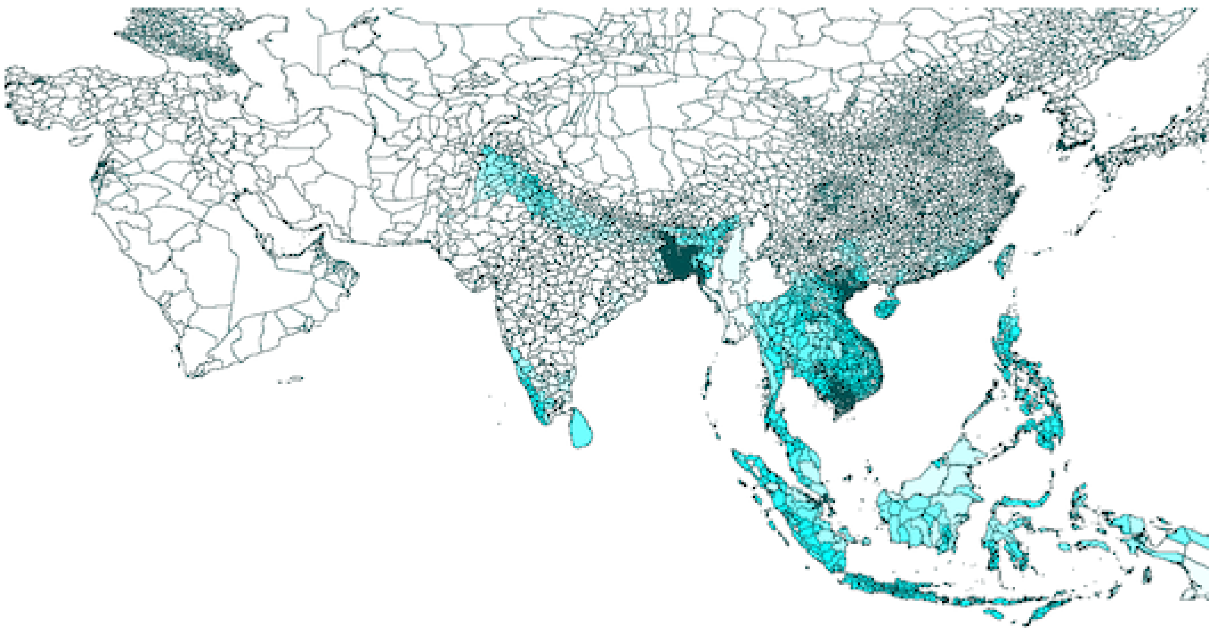


Figure 2.15: Simulation of seasonal mosquito density in Asia using STEM.

2.10.3 GLEAMviz

GLEAMviz is a pandemic modelling framework based on the GLobal Epidemic and Mobility model. The model combines real-world demographic and mobility data with stochastic disease models to help forecast the impact of epidemics. The framework consists of the client application and the GLEAM server. The client allows one to develop a compartmental disease model, request for it to be simulated remotely on the server and evaluate the produced results. The results can be visualised on a zoomable 2D map or an interactive 3D globe as shown in Figure 2.16.

One important advantage of GLEAMviz is that it maintains an up-to-date database of mobility patterns. Its primary focus is on air traffic for which it uses world-wide booking data sets. These come from the Official Airline Guide (OAG) which maintains a database containing more than 4,000 airports in 230 countries.

This framework is most often used to simulate pandemics. For example, it was recently used to assess the international risk of the 2014 Ebola outbreak [11]. Such global simulations require great processing power, which is kindly offered by the team through their servers. This client-server architecture however, along with the closed-source nature of the project, limits the flexibility of the framework. Additional mobility models cannot be implemented, and there is no control over the model fitting procedure. Also, the only kind of epidemic which can be simulated is the one where the host population is human. Furthermore, like STEM, the framework is restricted to a spatial model which is unsuitable for simulating socio-technological epidemics.

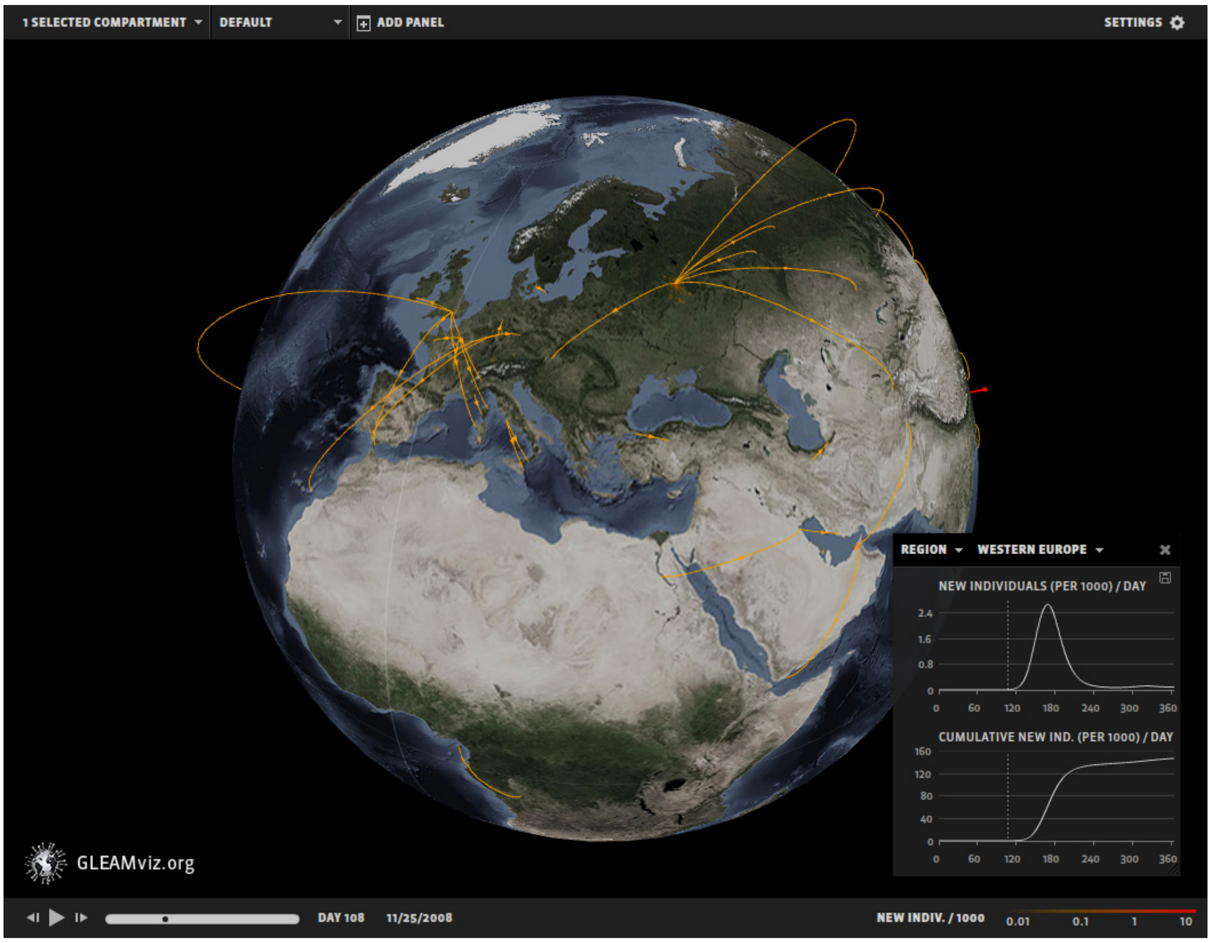


Figure 2.16: 3D visualisation of a pandemic using GLEAMviz.

Chapter 3

STAGE: A framework for visualising and simulating general epidemics

This chapter describes the objectives, design, implementation and functionality of STAGE, our Spatio-Temporal Analyser for General Epidemics. The end-product is a Java-based framework which is suitable for visualising, modelling, simulating and analysing the spreading of both biological and socio-technological epidemic phenomena. It consists of 65 top-level classes and around 4600 executable lines of code.

3.1 Aims

The main goal of the framework is to provide a great level of flexibility regarding the kinds of spatio-temporal models one can simulate and visualise. This would enable researchers to test scenarios made up of various disease and mobility models on either physical or logical space. The high level aims, derived from the limitations of existing frameworks like Surveillance, STEM and GLEAMviz, are as follows:

- Visualise and simulate both biological and socio-technological epidemics.
- Support several kinds of models and simulations.
- Provide quick, visual feedback on the model's goodness of fit.
- Provide the means to investigate the geography of areas under study.
- Have a flexible architecture which allows new models to be added easily.
- Provide a simple user interface which allows for visualising data sets, constructing simulation scenarios and fitting models to data sets quickly.

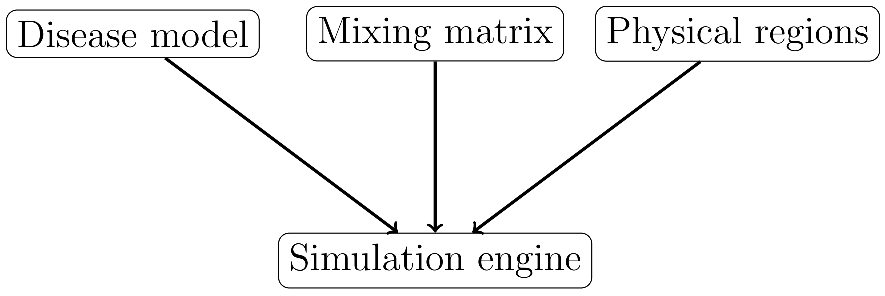


Figure 3.1: The standard approach in spatio-temporal modelling.

3.2 Approach

The typical spatio-temporal model which is used in simulations of biological epidemics consists of three main components:

1. A disease model defining the rates at which people transition between the various compartments.
2. A mixing matrix defining the rates at which people in one region move into another (and thus mix with people in the destination).
3. The set of physical regions (towns, districts, countries, etc.) over which we want to track the epidemic.

This model constitutes the input to the simulation engine as shown in Figure 3.1. The problem with this approach is that the space over which the epidemic is simulated is restricted to physical areas. Moreover, it makes the assumption that transmission between different areas depends on physical movement, and that a matrix of mixing rates is a sufficiently good representation to model the average case. In the case of socio-technological epidemics however, these do not apply because of the existence of influential spreaders: members of logical networks with thousands or even millions of connections. The decision of an influential spreader to share or not some information with their acquaintances can have a massive impact on the spreading of the epidemic, so we cannot rely on modelling just the average user behaviour. Instead, it is preferable to use agent-based modelling to keep track of the actions and state of each person in the simulation explicitly. By constructing a logical network of the individuals involved with a realistic degree distribution we acknowledge the part played by influential spreaders and therefore we can simulate information diffusion more accurately.

Figures 3.2 and 3.3 show the granularity at which transmission can be simulated. For biological epidemics the physical space is often adequate, but for socio-technological ones the logical space is preferable. Using the latter, however, means that we lose the ability to visualise these epidemics over physical areas, a space which is far more interesting and useful. To deal with this issue we propose to make use of both kinds of space simultaneously: the logical one to transmit the information among the individuals and the physical one to visualise the effect of the epidemic using the physical locations of these individuals, as shown in Figure 3.4. We thus propose to replace the physical space

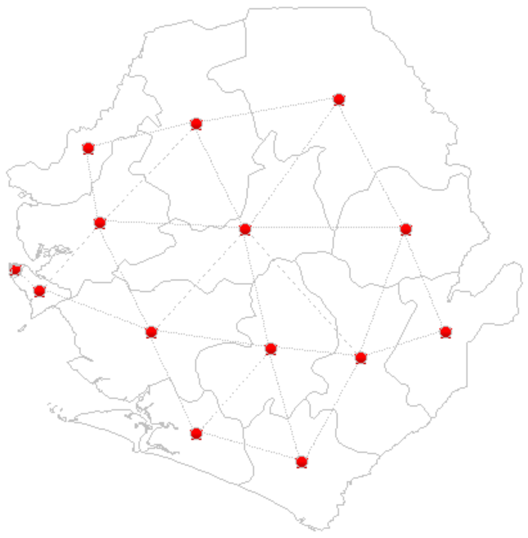


Figure 3.2: Transmission over physical space is modelled at the level of the population by determining the mixing rates between regions.

Figure 3.3: Transmission over logical space is modelled at the level of the individual by using the relationships between people.

component in the spatio-temporal model with a more flexible *space model*, thereby making the notion of space a key element of the framework. The choice of space (physical, logical or both) in simulation and visualisation should be made by the framework's user.

The remaining problem concerns how to create the links between the nodes in the logical network. This process cannot be random, as for example [36] showed that users on Twitter are more likely to be friends with people who live close to them. Our idea is to replace the mixing matrix (which is redundant in the socio-technological case) in our spatio-temporal model with a more general *mobility model* which would be capable of characterising both physical movement and information diffusion. The idea behind this model is to use the same formula which estimates the amount of movement between regions based on geographical factors like distance and population to compute both the mixing matrix and another structure which can be used to determine the endpoints of the network's links. This enables us to use existing mathematical models used in the modelling of biological epidemics to account for the role of geography in socio-technological ones too. Figure 3.5 illustrates the components of our spatio-temporal model.

To visualise an epidemic we need a component which will accept input from either the simulation engine (for predictions) or an external file (for observations) and render the appropriate space. If we design the visualiser to be able to accept data from multiple sources (simulators, external files) and display them in lock step then we have an elegant way of comparing them directly. By visualising a model's predicted values and the epidemic's observed values side-by-side we immediately get the intuition regarding the goodness of fit. Moreover, by visualising the predictions of multiple models altogether we point out their differences. Figure 3.6 illustrates this approach.

To provide support for a wide range of models and modes of simulation we make use of abstractions. These abstractions define clean interfaces for each of the components

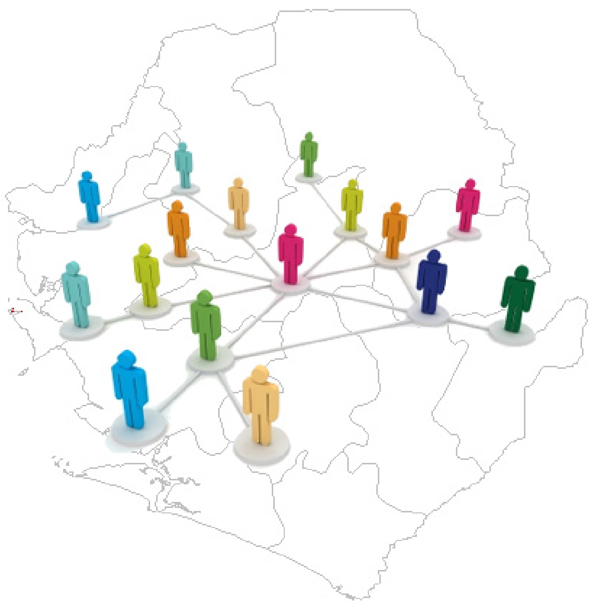


Figure 3.4: The proposed way to simulate the spread of a general epidemic: transmission occurs based on a logical network of the individuals but the impact of the epidemic is monitored separately for each region, using their physical locations.

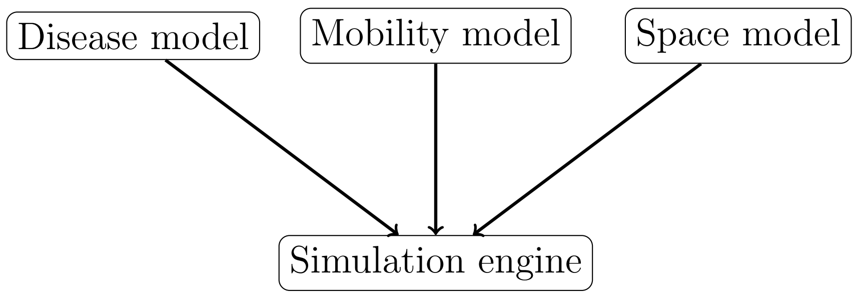


Figure 3.5: The proposed, more flexible approach in spatio-temporal modelling.

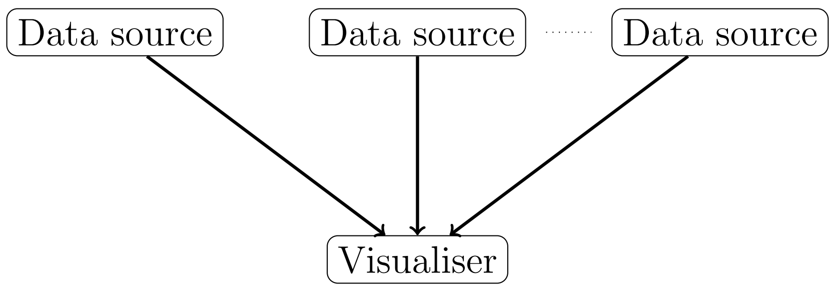


Figure 3.6: Visualising data from multiple sources simultaneously.

of our spatio-temporal model. For example, the simulation engine should not care if it uses a deterministic or stochastic disease model, as long as that model is able to compute the number of new cases that occur at each time point. Similarly, the visualiser should not care if its input data comes from a simulator using population-based or agent-based models as long as it can respond to its requests in the expected manner.

The framework can assist the researchers in investigating the geography of specific regions when they want to understand the spatial spread of an epidemic by rendering real-world maps instead of shapes. Users should be able to interact with the map (via e.g. zooming, panning and clicking) so that they can inspect the morphology and the infrastructure of an area of interest. To display the cumulative number of cases in each region we can overlay the map with a set of bounding polygons for the regions and set their colour intensity appropriately.

Finally, to make the framework easily extensible with new models we adopt a modular approach. By dividing the software into a collection of components whose responsibility is clearly defined using interfaces, the task of adding new kinds of models consists of simply implementing the required methods.

3.3 Objectives

Based on the approach presented above, for the framework to fulfil its aims while also providing much of the functionality available in existing software it should be able to:

- Simulate and visualise epidemics over physical and/or logical space.
- Simulate both population-based and agent-based compartmental disease models.
- Simulate both deterministic and stochastic models.
- Fit models to data using optimisation and provide a goodness of fit.
- Visualise the model's predictions and the observed cases side-by-side.
- Visualise the predictions of two models side-by-side.
- Simulate the effect of various kinds of human intervention.
- Generate synthetic data using stochastic simulation.
- Visualise physical regions using real-world, interactive maps.
- Provide an extensible interface so that additional disease, mobility and graph models can be integrated easily.
- Provide a simple graphical user interface that requires minimal amount of clicks to visualise a data set, simulate a model and fit a model to the data.

3.4 Programming language

The decision regarding the programming language to be used in order to implement the proposed framework was critical. Not only such decision would affect the development process but it would also determine to a high degree the performance and maintainability of the end-product.

Matlab and R offer extensive support for data analysis and numeric computation. R in particular has many packages which were developed for epidemiology such as `EpiModel`, `Surveillance` and `GillespieSSA`. The fact that it is also open-source makes it more appealing than Matlab because users would not be expected to have purchased any licenses. However, the performance and flexibility offered by the two languages are inferior to those provided by general-purpose languages such as Java and C++.

Java and C++ were both good candidates as they are both well-documented and they both enable the development of flexible, modular and extensible code using abstractions. The fact that they are object-oriented is also important as it makes the design more intuitive. On the negative side they have fewer packages for analysing data and fitting models, meaning that some algorithms would have to be implemented from scratch.

Considering the benefits and weaknesses of each of the four languages, we decided to go with Java as it offers a mixture of good performance, portability, memory abstraction and clean interfaces.

3.5 Architecture

STAGE was designed by following a modular approach so that it would be easy to extend in order to cater for additional features or models. It consists of 15 components (organised in 10 packages), as shown in Table 3.1. The class `Framework` - which does not belong to any package - constitutes the entry point.

Figure 3.7 shows how the main components are connected to form the system. When the class `Framework` is loaded, it starts the graphical user interface (GUI) to capture input from the user. Depending on the user's actions it can start the visualisation engine, model fitting or synthetic data generation. In the first case, the visualisation window is constructed and either a map or a graph is rendered, depending on the given spatial model. The visualisation uses one or more data sources as input, which can be file readers or simulators. Simulators consist of four components: a spatial model which describes the space over which the epidemic is simulated, a disease model which captures the dynamics of the disease, a mobility model which describes how individuals move between regions and a solver to integrate the disease model's ODEs. Model fitting uses a file reader and a simulator to fit the overall spatio-temporal model to the data whereas synthetic data generation uses a simulator to produce random traces of the epidemic.

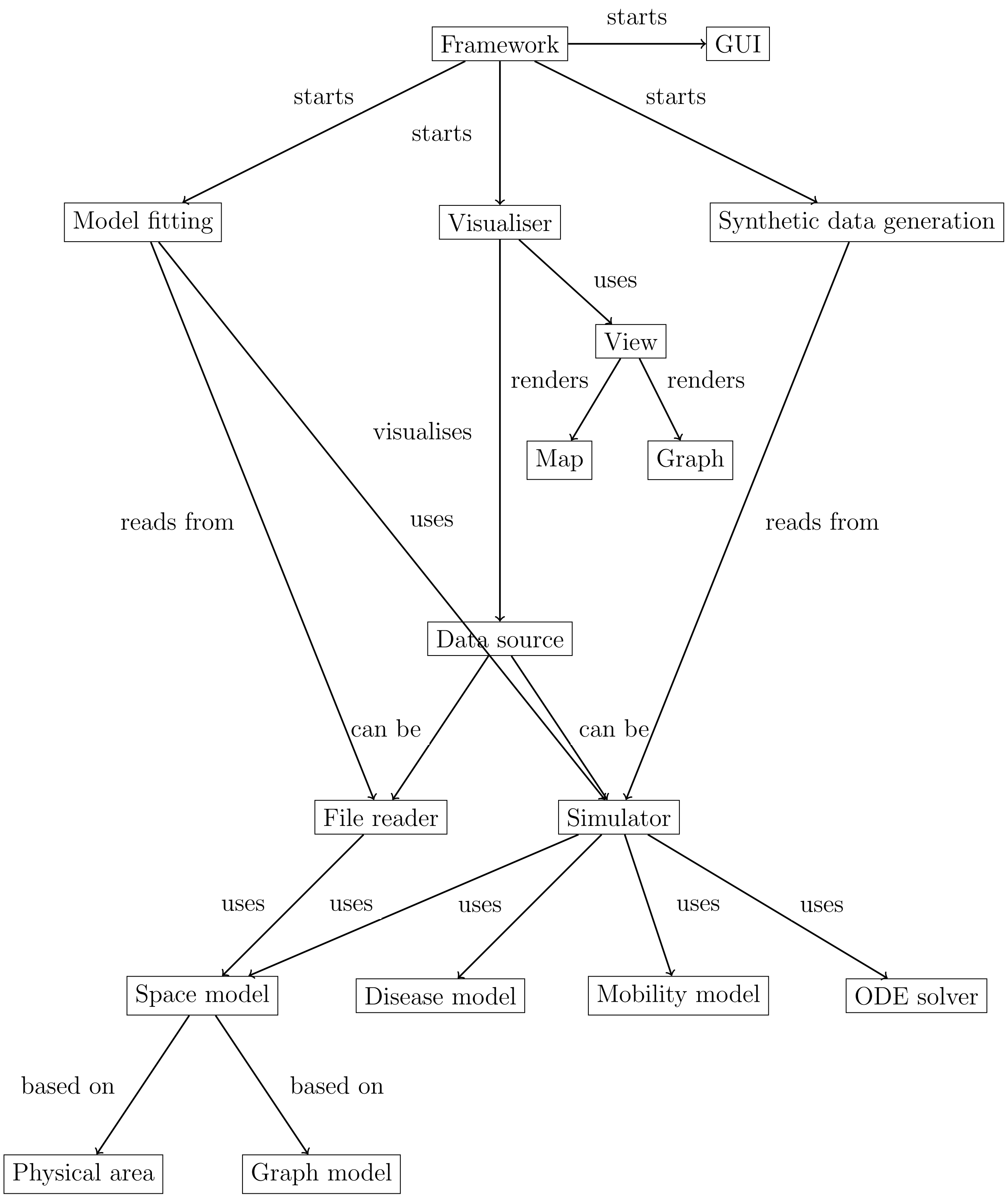


Figure 3.7: System architecture.

Package	Components
analysis	model fitting and synthetic data generation
core	visualization engine and abstract models
datasource	file readers and simulators
disease	compartmental disease models
space	spatial models
graphs	graph models
mobility	mobility models
solvers	ODE solvers
ui	GUI and map rendering
utils	loggers and CSV handling

Table 3.1: Packages and components of STAGE.

3.6 Components

Each component of the framework was designed and implemented in its own sprint using Agile Software Development.

3.6.1 Abstract models

Figure 3.8 shows the abstractions at the top of the pyramid of models. This pyramid consists of the disease models, mobility models and graph models which are described in separate sections. Implementations of these interfaces extend the class `AbstractModel`. This implements the methods defined in the interface `ParameterisedModel` and has a single additional abstract method which forces its subclasses to construct an array of element type `Parameter`, which it then stores as a field. `Parameter` is a simple class that stores the name, symbol and current value of a model's parameter. It also stores information on whether it should be optimised during model fitting.

The purpose of `Parameter`, `ParameterisedModel` and `AbstractModel` is to provide a uniform interface to the storage and manipulation of parameter values. This allows the model fitting procedure to optimize the parameters of <u>any</u> kind of model, without worrying about calling class-specific methods.

3.6.2 Disease models

Figure 3.9 shows the hierarchy of disease models. These models implement the interface `DiseaseModel` and extend the abstract class `AbstractModel`. They specify their parameters and define a method which uses the given mixing matrix or graph to predict the new cases at the next step of the simulation.

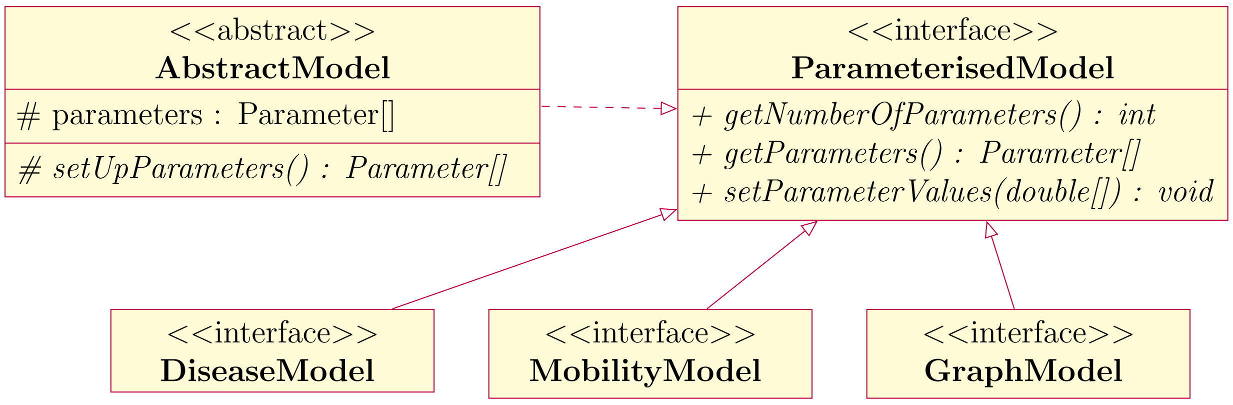

<<abstract>>
AbstractModel
# parameters : Parameter[]
# setUpParameters() : Parameter[]
<<interface>>
ParameterisedModel
+ getNumberOfParameters() : int
+ getParameters() : Parameter[]
+ setParameterValues(double[]) : void
<<interface>>
DiseaseModel
<<interface>>
MobilityModel
<<interface>>
GraphModel

Figure 3.8: UML diagram of the abstract classes in the hierarchy of models.

The disease models use the **Decorator design pattern**[1] to store the number of individuals in each compartment for each region. An inner class named `RegionWrapper` is defined which holds a `Region` object as well as counters for the compartments. The pattern helps distinguish between the two layers of data (geographical properties and compartment memberships), making the two classes compatible with the *Single Responsibility Principle*[2].

A disease model is initialised with the identifier of the region in which the first infection occurs. The `Region` with that identifier has its infected population counter set to one while the rest start clean. The number of susceptible individuals is initialised to the difference between the total population and the number of infections in that region. Throughout the simulation the number of cases in a region is determined by the total population minus the susceptible population. By working at the level of cases instead of the number of people in the infected compartment, we make our models suitable for the type of data which is published by health organisations (cases and deaths). This avoids accuracy errors which can occur when converting from one format to the other.

The most important method of a disease model is getNewCases(). This is used by the simulator to get the number of new cases that have occurred in each region during the last step of the simulation. The method uses the given ODE solver to integrate its set of equations and thus obtain new counts for each compartment. Note that by using the interface `OdeSolver` in the class (via *dependency injection*[3]) the model is unaware of how the equations are solved - deterministically or stochastically - and is therefore not dependent on any specific algorithm. The actual dynamics of the model are captured in an inner class which implements the interface `CompartmentalEquations`. This class is used to specify what transitions occur, what their rates are and what the ODEs are. For example, the spatially-enriched SIR model is defined using the following equations:

$$\frac{dS_i}{dt} = -\frac{\beta S_i \sum_{i=0}^{n} M_{ij} I_j}{\sum_{i=1}^{K} M_{ij} N_j}$$

[1]http://en.wikipedia.org/wiki/Decorator_pattern
[2]http://en.wikipedia.org/wiki/Single_responsibility_principle
[3]http://en.wikipedia.org/wiki/Dependency_injection

$$\frac{dI_i}{dt} = \frac{\beta S_i \sum_{i=0}^{n} M_{ij} I_j}{\sum_{i=1}^{K} M_{ij} N_j} - \gamma I_i$$
$$\frac{dR_i}{dt} = \gamma I_i$$

These are split into three parts:

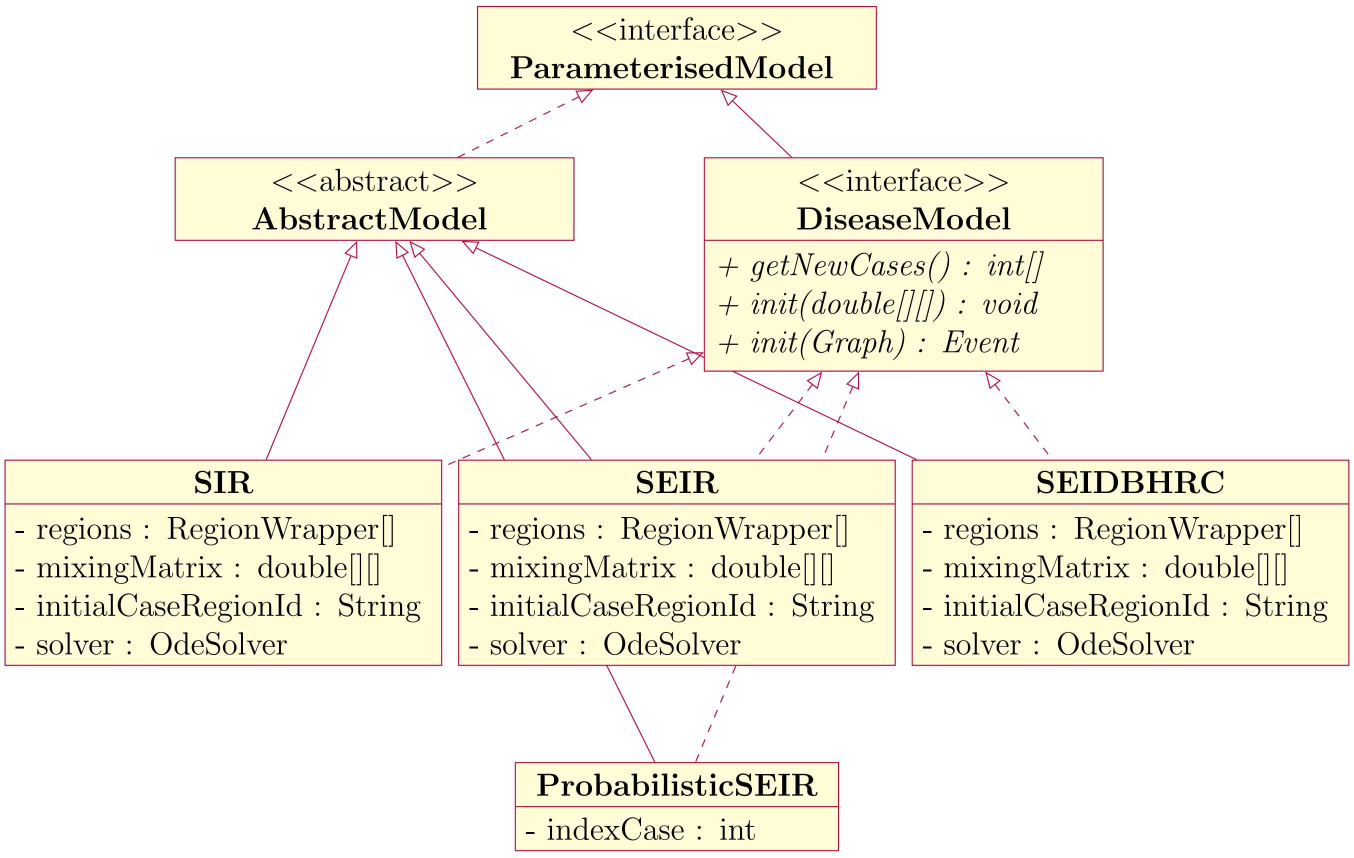
1. What are the transitions in this model? $S \to I$ and $I \to R$.
2. What are their rates? $\frac{\beta S_i \sum_{i=0}^{n} M_{ij} I_j}{\sum_{i=1}^{K} M_{ij} N_j}$ and γI_i.
3. How are the ODEs expressed in terms of these transitions?

$$\frac{dS_i}{dt} = -(S \to I)$$
$$\frac{dI_i}{dt} = (S \to I) - (I \to R)$$
$$\frac{dR_i}{dt} = I \to R$$

This decomposition enables both deterministic and stochastic solvers to be used. Deterministic solvers make use of components 2 and 3 while stochastic ones make use of components 1 and 2.

There are currently three models which use a mixing matrix to account for mobility in the framework: `SIR`, `SEIR` and `SEIDBHRC`. The first class implements the simple SIR model described above, the second implements the similar SEIR model described in Section 2.3.2 and the third implements the so-called Ebola model which is described in detail in Chapter 4. Additional models can be constructed by building classes which implement the `DiseaseModel` and `CompartmentalEquations` interfaces.

Disease models for graphs can also be constructed. These are stochastic in nature as they rely on the structure of the underlying graph for the transmission of the disease, which most of the time involves an element of randomness in the way the edges are assigned. The framework includes the class `ProbabilisticSEIR` which is the graph-based equivalent model of the one using a mixing matrix and ODEs. Its parameters characterise the time taken for an individual to infect their connections (either all at once or individually), the time taken to recover as well as the probability of actually infecting the connections. The dynamics of the disease model are captured by the construction of inner classes which represent events such as infections and recoveries. These events are processed by the `GraphBasedSimulator` using discrete-event-simulation. During the processing of an event future events can be generated and added to the simulation queue. For example an `InfectionEvent` generates another `InfectionEvent` for each individual they choose to infect. Figure 3.10 shows the types of events in `ProbabilisticSEIR`.



<<interface>>
ParameterisedModel
<<abstract>>
AbstractModel
<<interface>>
DiseaseModel
+ getNewCases() : int[]
+ init(double[][]) : void
+ init(Graph) : Event
SIR
- regions : RegionWrapper[]
- mixingMatrix : double[][]
- initialCaseRegionId : String
- solver : OdeSolver
SEIR
- regions : RegionWrapper[]
- mixingMatrix : double[][]
- initialCaseRegionId : String
- solver : OdeSolver
SEIDBHRC
- regions : RegionWrapper[]
- mixingMatrix : double[][]
- initialCaseRegionId : String
- solver : OdeSolver
ProbabilisticSEIR
- indexCase : int

Figure 3.9: UML diagram of the hierarchy of disease models.

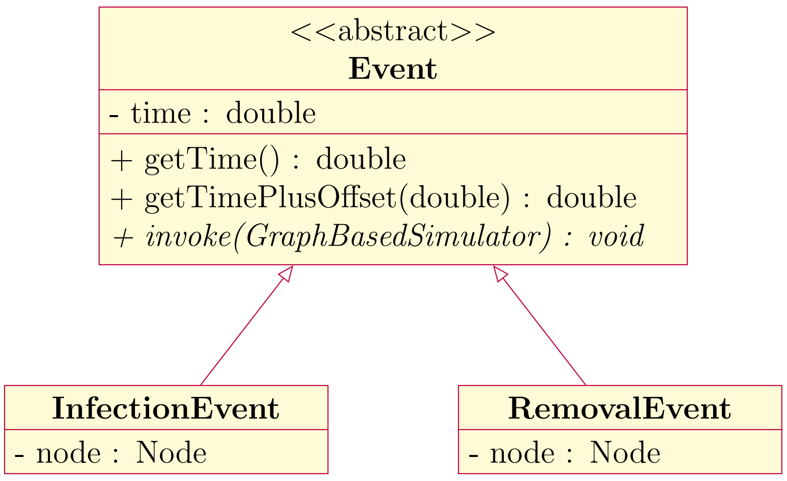


Figure 3.10: UML diagram of event types used in ProbabilisticSEIR.

3.6.3 ODE solvers

Figure 3.11 shows the relationships between the classes in the `solvers` package. These solvers are used in the simulation of epidemics to compute the number of individuals in each compartment in the next time unit. In the deterministic case, the ODEs defining the transitions between the compartments are integrated numerically using a small step size to produce the results. In the case where stochastic results are preferred, the equations are solved using a stochastic simulation algorithm.

Whether the ODE solvers should be first-class citizens in STAGE was a non-trivial design question. We could decide on a single solver to use under the scene in the simulation of deterministic compartmental models. Regarding stochastic models, these could be implemented in their own classes using an appropriate simulation algorithm. The benefit of this approach would be that the complexity of the program would decrease as the entire `solvers` package would not exist. Moreover the user would not be expected to be familiar with ODE solvers.

The decision to build the `solvers` package was based on the objective of increasing the program's extensibility. Not only new solvers can be created by just implementing the interface `OdeSolver` (which specifies a single method), but disease models can be added by just creating one class every time. This new class needs to implement the interface `CompartmentalEquations` by defining what transitions are possible in that model and how their are rates computed. Also, as `CompartmentalEquations` extends the `ode` interface `FirstOrderDifferentialEquations`, the actual ODEs need to be implemented too in the method computeDerivatives(). The new disease model is then suitable for both deterministic and stochastic simulation using any of the available solvers, without needing to distinguish between the two anywhere in its implementation. The separation of model and algorithm allows us to avoid code duplication. Furthermore, this design provides much better encapsulation as the disease models are unaware of the internals of the solvers. This means that we can make changes to the algorithms without the need to update the models too.

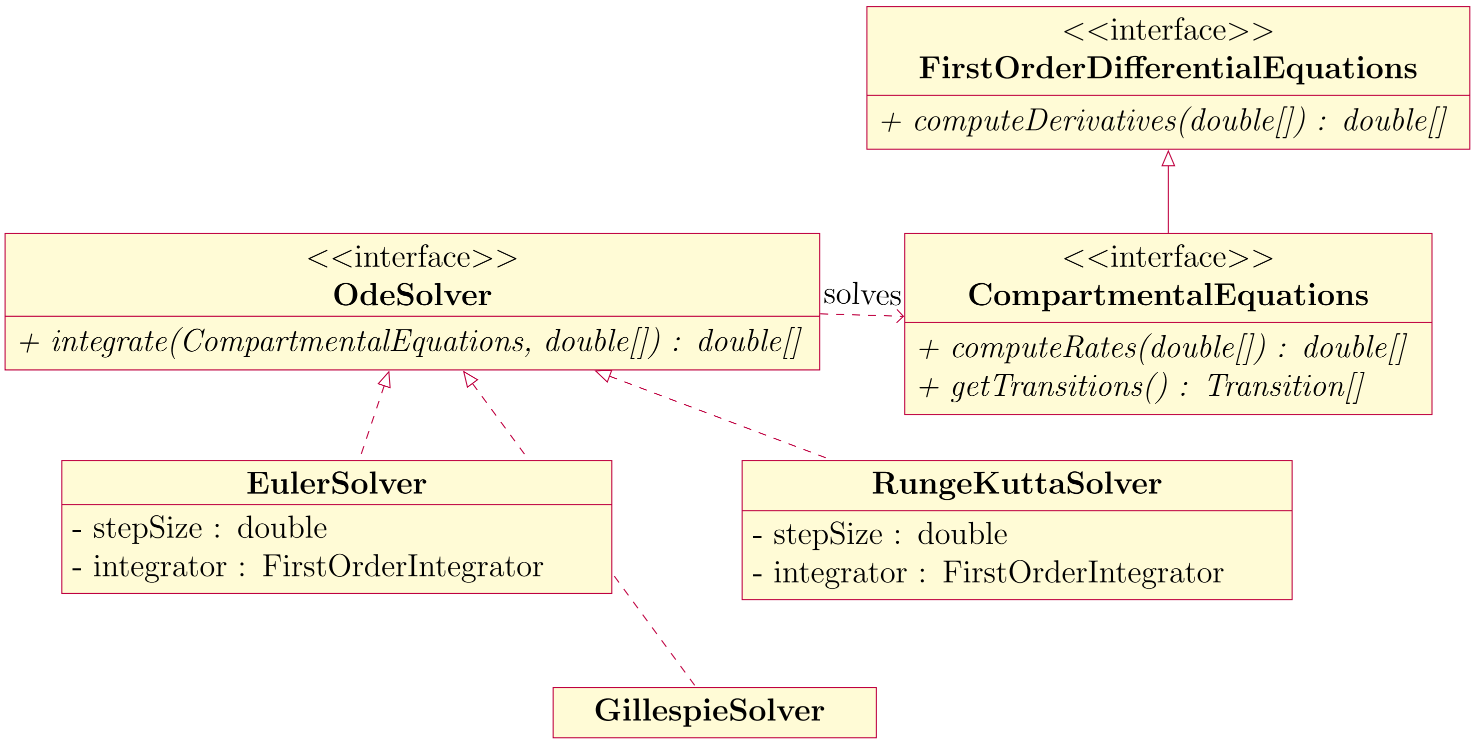
The following sections describe the three solvers available in the framework.

Euler method

`EulerSolver` uses the Euler method for solving ODEs given some initial values. Its implementation uses the functionality of the Apache Commons Math `ode` package to compute the result after one time unit.

4th order Runge-Kutta method

`RungeKuttaSolver` uses the classical 4th order Runge-Kutta method for solving ODEs. This is computationally more efficient than the Euler method as it does not need such small step sizes to achieve good accuracy. The implementation uses the `ode` package to perform the actual integration.



<<interface>>
FirstOrderDifferentialEquations
+ computeDerivatives(double[]) : double[]
<<interface>>
OdeSolver
+ integrate(CompartmentalEquations, double[]) : double[]
solves
<<interface>>
CompartmentalEquations
+ computeRates(double[]) : double[]
+ getTransitions() : Transition[]
EulerSolver
- stepSize : double
- integrator : FirstOrderIntegrator
RungeKuttaSolver
- stepSize : double
- integrator : FirstOrderIntegrator
GillespieSolver

Figure 3.11: UML diagram of the hierarchy of ODE solvers.

Gillespie's Stochastic Simulation Algorithm

Unlike the other two solvers, `GillespieSolver` gives non-deterministic results. In this case the ODEs are solved using Gillespie's Stochastic Simulation Algorithm described in Section 2.9. This solver is particularly useful in the generation of synthetic data.

3.6.4 Mobility models

Figure 3.12 shows the family of models of human mobility. There are currently five classes which implement the interface `MobilityModel` by defining the two necessary methods. These methods are used to construct either a mixing matrix M or a rank matrix R which is then given to the chosen disease model. Each entry in M (m_{ij}) specifies the rate at which people move between two regions. Each row in R (r_i) contains the indices of the regions in descending order of mixing rate with i. M is typically used in the modelling of biological epidemics whereas R is used in the case of socio-technological epidemics.

The following sections describe the built-in mobility models. Additional models can be created by simply implementing the `MobilityModel` interface.

Zero mobility

`ZeroMobility` is used in cases where we have just one region to monitor. This may be a small town, a district, a country or event a collection of countries. It removes the spatial component of the modelling allowing one to get quick results using a space-free compartmental disease model.

The mixing matrix is constructed as follows:

$$m_{ij} = \begin{cases} 1 & \text{if } i = j \\ 0 & \text{otherwise} \end{cases}$$

The rank matrix is constructed as follows:

$$r_{ij} = i$$

Common borders

`CommonBorders` assigns a constant mixing rate k between all regions which are adjacent to each other. This rate is a parameter which can be optimised in model fitting. The inability of this model however to account for long-distance movement limits its applicability.

The mixing matrix is constructed as follows:

$$m_{ij} = \begin{cases} 1 & \text{if } i = j \\ k & \text{if } A_{ij} = \mathit{true} \\ 0 & \text{otherwise} \end{cases}$$

where A_{ij} is the adjacency matrix.

Neighbour order

`NeighbourOrder` defines the mixing rate between two regions to be inversely proportional to the power k of their order of neighbourhood. The order is the smaller number of intermediate regions that need to be traversed to go from one region to another. As in the previous model the parameter k can be optimised.

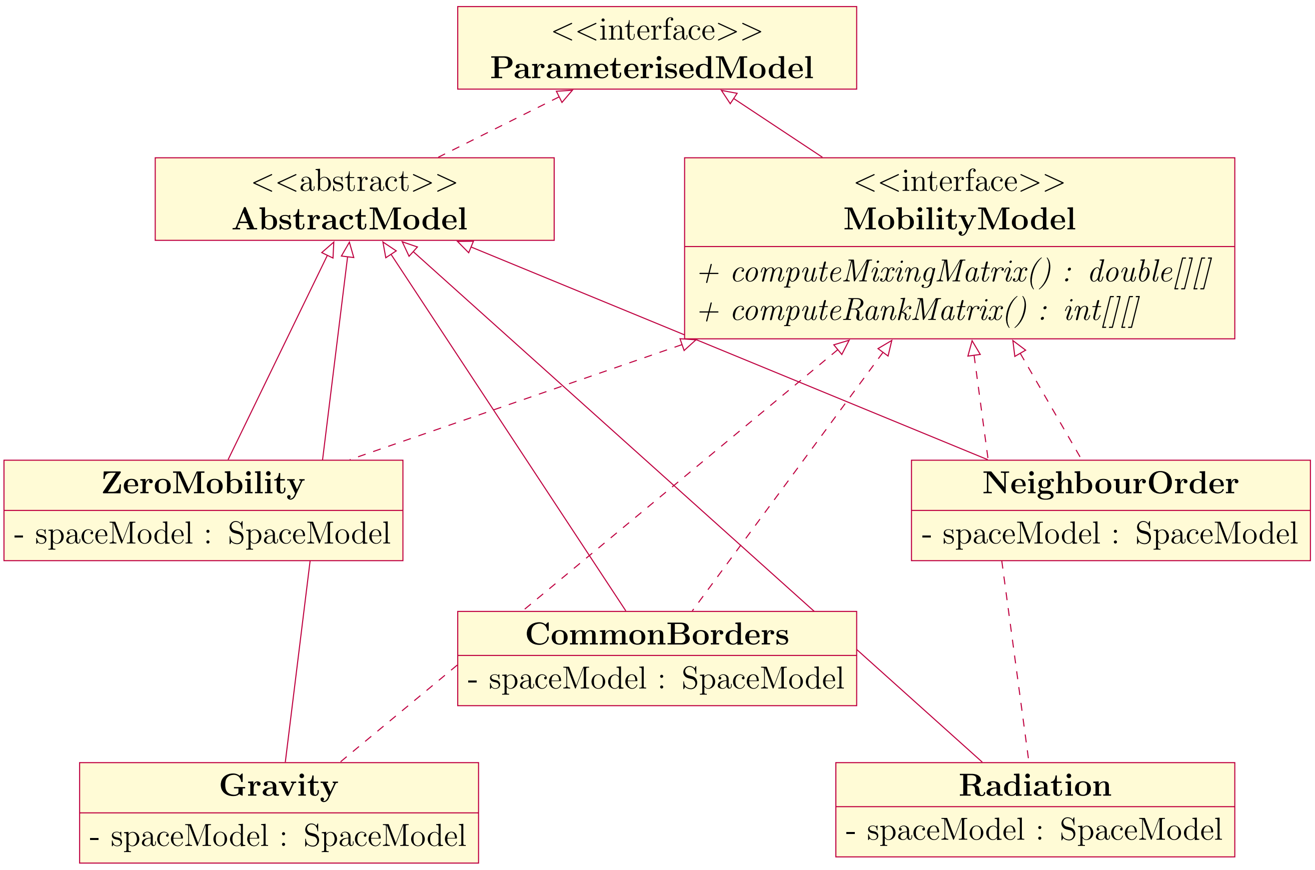
The mixing matrix is constructed as follows:

$$m_{ij} = \begin{cases} 1 & \text{if } i = j \\ (N_{ij} + 1)^{-k} & \text{otherwise} \end{cases}$$

where N_{ij} is the matrix defining the neighbour order.

Gravity

`Gravity` implements the mobility model based on Newton's gravitational law described in Section 2.7.1.



<<interface>>
ParameterisedModel
<<abstract>>
AbstractModel
<<interface>>
MobilityModel
+ computeMixingMatrix() : double[][]
+ computeRankMatrix() : int[][]
ZeroMobility
- spaceModel : SpaceModel
NeighbourOrder
- spaceModel : SpaceModel
CommonBorders
- spaceModel : SpaceModel
Gravity
- spaceModel : SpaceModel
Radiation
- spaceModel : SpaceModel

Figure 3.12: UML diagram of the hierarchy of mobility models.

The mixing matrix is constructed as follows:

$$m_{ij} = \begin{cases} 1 & \text{if } i = j \\ k\frac{p_i^\alpha p_j^\beta}{d^\gamma} & \text{otherwise} \end{cases}$$

where p_i is the population at the origin, p_j is the population at the destination, d is the distance between the two regions and α, β, γ, k are parameters.

Radiation

`Radiation` implements the Radiation model of commuting described in Section 2.7.1.

The mixing matrix is constructed as follows:

$$m_{ij} = \begin{cases} 1 & \text{if } i = j \\ k\frac{p_i p_j}{(p_i+s_{ij})(p_i+p_j+s_{ij})} & \text{otherwise} \end{cases}$$

where p_i is the population at the origin, p_j is the population at the destination, s_{ij} is the total population inside a circle of radius d centred at i (excluding p_i and p_j) and k is a coefficient.

3.6.5 Space models

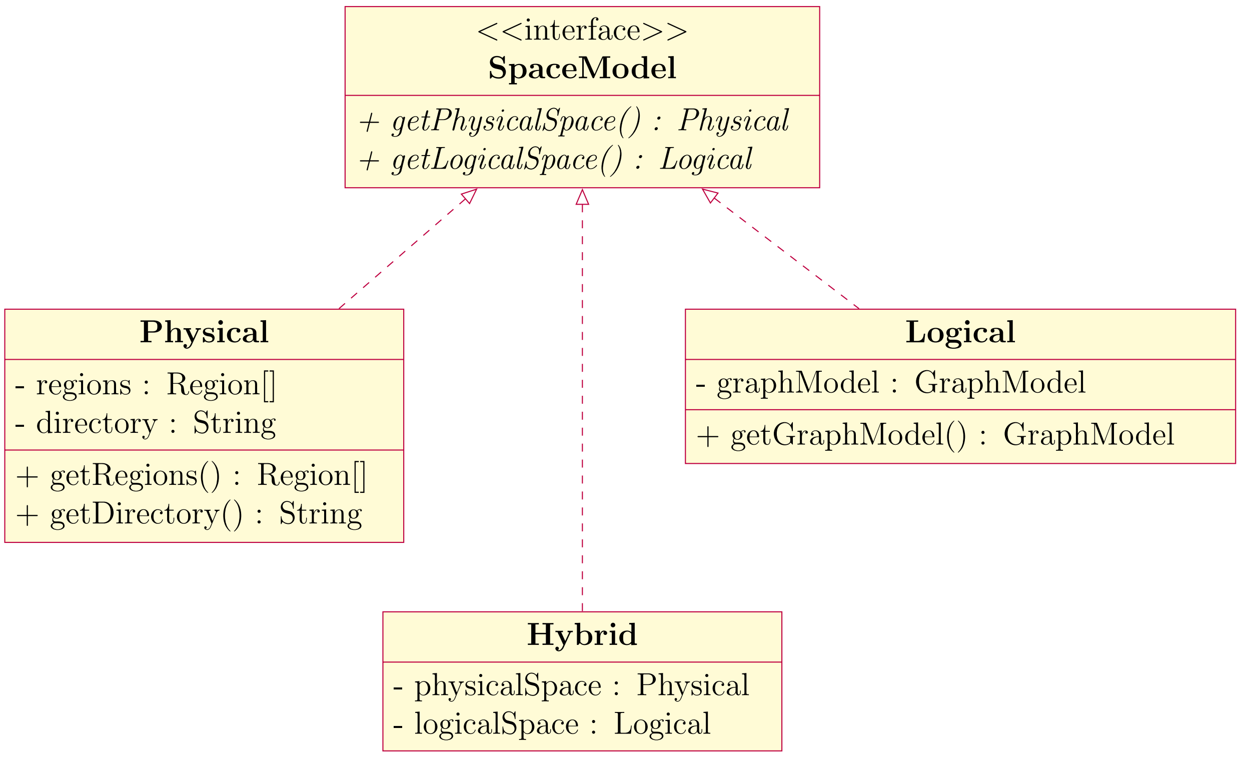
Figure 3.13 shows the models which define the space in which epidemics are simulated. There are currently three classes which implement the interface `SpaceModel`:

1. `Physical`. This represents a physical area of interest. It holds an array of `Region` objects which contain information regarding the individual sub-areas whose behaviour during an epidemic we want to track, such as population and area. These objects are created by reading a CSV file named "demographics.csv" inside the given *directory.*
2. `Logical`. This represents a space defined by a graph. The type of the graph is determined by the actual type of the enclosing `GraphModel`.
3. `Hybrid`. This represents a physical area over which the epidemic spreads, but the transmission between regions is determined on the basis of the logical contact network of the population rather than a general population-based model of human mobility. It is therefore made up of both a `Physical` and a `Logical` objects. It is mainly used in the spatio-temporal modelling and analysis of socio-technological epidemics, but it can also be used to simulate biological epidemics using agent-based modelling. In this case, mobility is accounted for by infecting people who belong in the agent's constant network - and who may be located anywhere - instead of changing their location. This speeds up the simulation as the cost of incorporating mobility is paid only in the beginning by constructing the extra logical space.

STAGE can currently simulate epidemics in West Africa and London. Additional geographical areas can be incorporated using the following procedure:

1. Download the Shapefile[4] describing the administrative regions of interest. DIVA-GIS[5] provides such data for any country in the world for free.
2. Convert the Shapefile into the GeoJSON[6] format. Ogre[7] is a web service that uses the ogr2ogr command line tool to execute such conversions. Name the new file "regions.geo.json".
3. Create a CSV file named "demographics.csv" containing the administrative identifier, administrative name, population and area for each region. A possible data source is GeoHive[8].
4. Place the two files in a new directory under "data/".
5. Pass the name of the directory to `Physical`'s constructor.

[4]A geospatial vector data format for geographic information system software. Source: Wikipedia
[5]http://www.diva-gis.org/gdata
[6]An open standard format for encoding collections of simple geographical features along with their non-spatial attributes using JavaScript Object Notation. Source: Wikipedia
[7]http://ogre.adc4gis.com
[8]http://www.geohive.com



<<interface>>
SpaceModel
+ getPhysicalSpace() : Physical
+ getLogicalSpace() : Logical
Physical
- regions : Region[]
- directory : String
+ getRegions() : Region[]
+ getDirectory() : String
Logical
- graphModel : GraphModel
+ getGraphModel() : GraphModel
Hybrid
- physicalSpace : Physical
- logicalSpace : Logical

Figure 3.13: UML diagram of the hierarchy of space models.

3.6.6 Graph models

Figure 3.14 shows the main fields and methods of the graph models family. These represent graphs of specific types and characteristics. Implementations of the `GraphModel` interface define methods to set up their parameters, construct graphs using the `Graphs` utility class and create the UI element that displays their graphs. Table 3.2 shows the models which are currently implemented in STAGE. Figure 3.22 shows how they are rendered by the framework.

Graph	Parameters
Grid	Side length
Crossed Grid	Side length
Random	Number of nodes, average degree
Watts-Strogatz	Number of nodes, number of edges per node, rewiring probability
Barabasi-Albert	Number of nodes, maximum number of edges to add in each step

Table 3.2: Types of graphs implemented in the framework.

The framework can easily be extended with additional types of graphs by creating more classes which implement the `GraphModel` interface. The GraphStream library provides built-in algorithms to build a wide variety of graphs such as banana tree, Chvatal, Dorogovtsev-Mendes and Petersen generators[9]. These can be used directly in the im-

[9]http://graphstream-project.org/doc/Generators/Overview-of-generators_1.0/

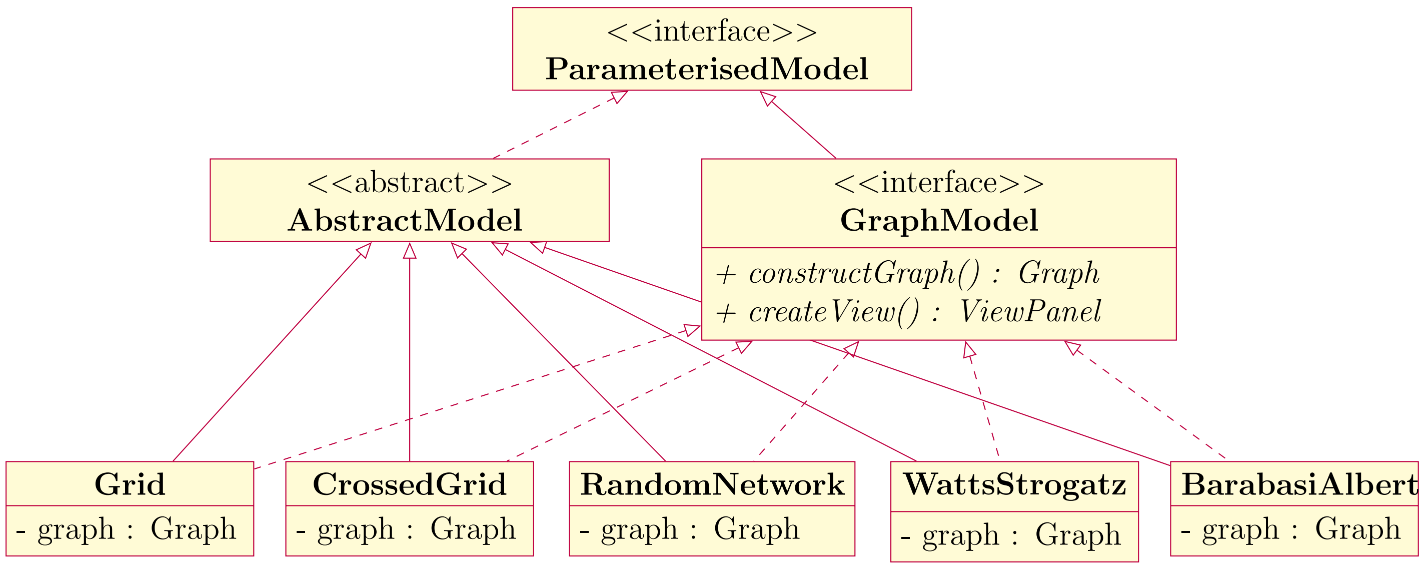


Figure 3.14: UML diagram of the hierarchy of graph models.

plementation of the new model's constructGraph() method. Furthermore, the library makes it easy for someone to build their own graph from scratch.

3.6.7 Data sources

Figure 3.15 shows the main fields and methods of the classes which provide data to the visualisation engine. The two abstract classes `FileReader` and `Simulator` define the two major types of data sources.

The key method specified in the `DataSource` interface is next(), which returns an object of type `NewCasesEntry`. This is a simple class used to store the timestamp and the new cases that have been observed at the current point in visualization. The interface promises to deliver this data in the same format irrespectively of the underlying implementation, allowing its clients to handle all sources of information uniformly. By depending on the interface clients are also decoupled from any of its implementing classes. This makes it easy to extend the program with new data sources.

CasesFileReader

The `CasesFileReader` data source is responsible for reading the content of a CSV file containing per region and per time point counts of new cases. Its next() method uses an iterator to deliver the next line in the file.

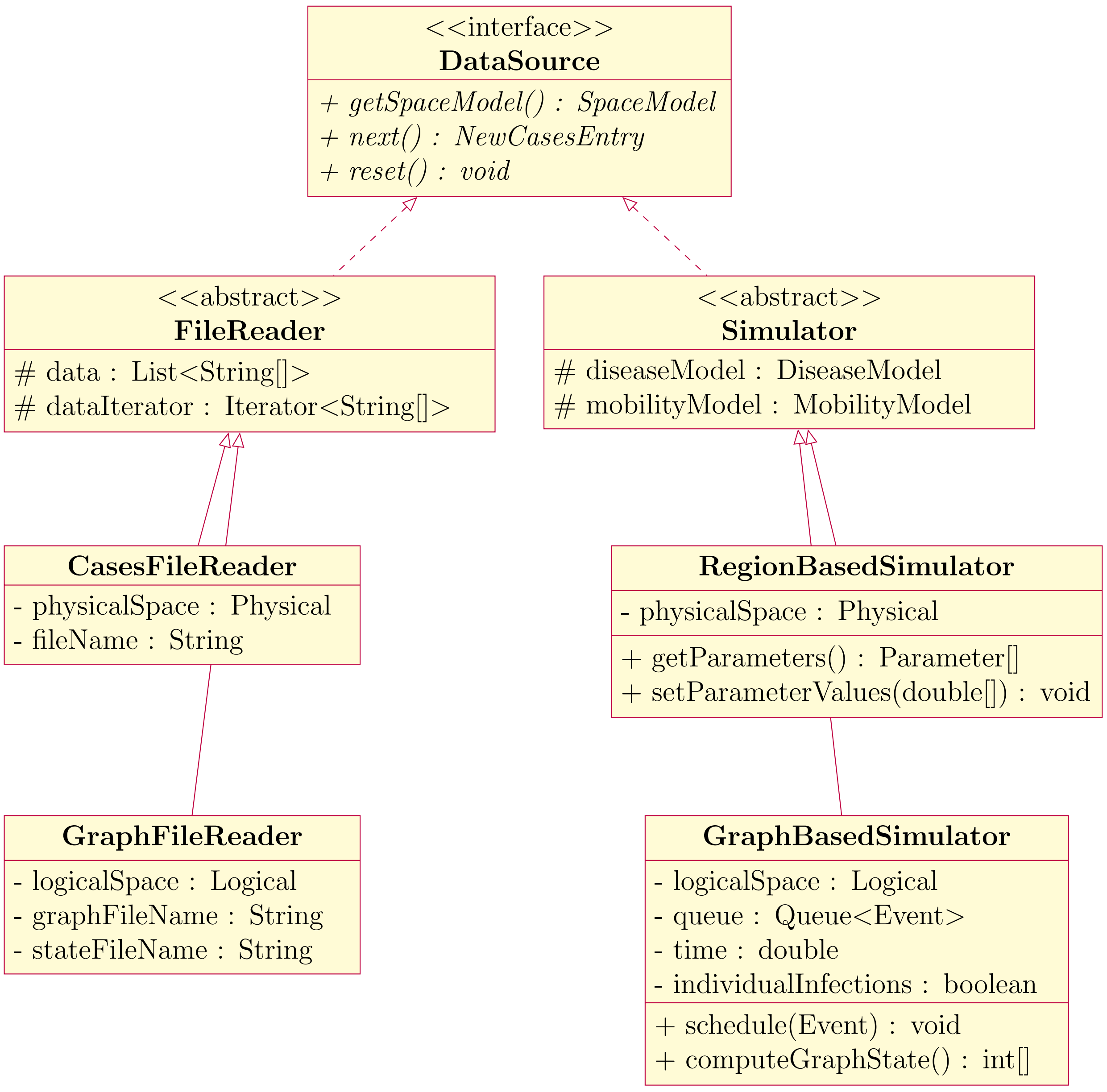


Figure 3.15: UML diagram of the hierarchy of data sources.

GraphFileReader

The `GraphFileReader` data source is responsible for reading two files, a DGS[10] file containing the structure of the graph and a CSV file containing the state of each node per time point. In the beginning it reads the first file and reconstructs the graph. On each invocation of next(), it reads the next line of the second file.

RegionBasedSimulator

`RegionBasedSimulator` simulates an epidemic spreading across a physical area using a given disease model and mobility model. The mobility model is used in the beginning to initialise the disease model with a mixing matrix M which specifies the rate at which people move between regions in a single time unit. The next() method delegates control to the disease model which is responsible for computing the number of individuals in the compartments in each region and determining the new cases. The getParameters() and setParameterValues() methods are convenient methods to get and set the parameters of both models altogether respectively.

GraphBasedSimulator

`GraphBasedSimulator` simulates an epidemic on a graph using a given disease model. Its first action is to construct the graph based on the graph model wrapped inside the given space model.

The discrete-event-simulation[11] paradigm is used in order to maintain simplicity and enhance readability. The simulator maintains a queue of future events, sorted in ascending order in time. In the beginning the nodes are initialised in the susceptible state and the first event (the initial infection) is scheduled. This will in turn schedule further events when occurred and so on. On each next() call the simulator processes the events that were scheduled to happen in the next time unit. It then increments the simulation's virtual time and instructs the disease model to determine the new cases that have occurred.

The simulator also has a method that computes an array of integers representing the state of each node, which is used to export results of the simulation. It also allows for both simultaneous and individual infections along the edges of the nodes so that it can simulate one-to-all and one-to-one message passing in OSNs.

3.6.8 Visualisation engine

Figure 3.16 shows the main fields and methods of the class `Visualiser`, the *model* in our **Model-View-Controller (MVC) design pattern**[12] for visualisation. The key

[10] http://graphstream-project.org/doc/Advanced-Concepts/The-DGS-File-Format/
[11] http://en.wikipedia.org/wiki/Discrete_event_simulation
[12] http://en.wikipedia.org/wiki/Model-view-controller

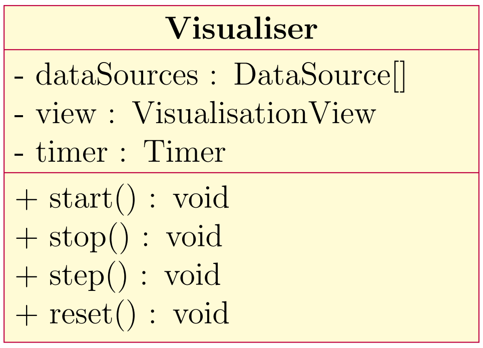


Figure 3.16: UML diagram of Visualiser.

logic resides in the implementation of method step() which executes a single step in the visualisation (and simulation). Specifically, it requests data regarding new cases from a file reader or simulator and instructs the `VisualisationView` object (the *view* in the MVC) to update the heatmap of cumulative cases for all regions, or in the case of a graph to update the state of the nodes. It also tells the view to show the latest timestamp and total cases counter. The method start() uses a `Timer` object to call step() at fixed intervals, while stop() and reset() are self-explanatory.

The class was designed to be flexible in terms of what data it can visualise. By using the interface `DataSource` it can request new data from either a file or a model; its actions are the same in any case. This avoids the need to have separate classes for file visualisation and model simulation, which would mean a lot of duplicated code.

But the real power of `Visualiser` is that it executes step() for an arbitrary number of `DataSource` objects, which are passed in the constructor using Java's varargs. This enables the framework to display the data obtained from multiple data sources in lock step, side-by-side. As a result, direct comparisons can be made between any kind of data source. Epidemiologists can therefore get fast, visual feedback on whether their model matches the data or whether a change in the value of a parameter has the desired effect.

The graphical component of the visualisation engine is the class `VisualisationView`. This uses the Swing framework to generate UI elements. Figure 3.17 shows its main fields and methods.

Upon construction, `VisualisationView` determines whether it needs to render and embed a map or a graph based on the current space model. It then constructs an instance of the inner class `Controller` which is responsible for listening to user input and calling the four `Visualiser`'s methods shown in Figure 3.16. This is the last component of the MVC model which is shown in Figure 3.18. The benefit of such design is that coupling between the model and the view is reduced, allowing each class to have a single responsibility, which is one of the five SOLID principles[13] of good object-oriented design.

Furthermore, `VisualisationView` provides the means to manually change the parameter values of the spatio-temporal model during simulation. This allows us to account for the impact of various containment measures which are taken by the authorities at certain points in time. For example, by reducing the coefficient of the Gravity or Radiation

[13]http://en.wikipedia.org/wiki/SOLID_%28object-oriented_design%29

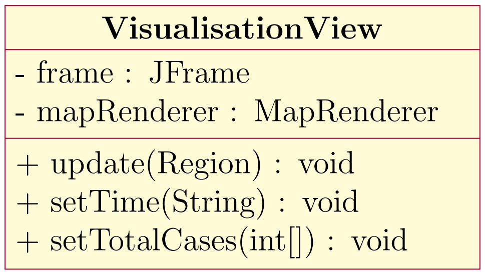


Figure 3.17: UML diagram of VisualisationView.

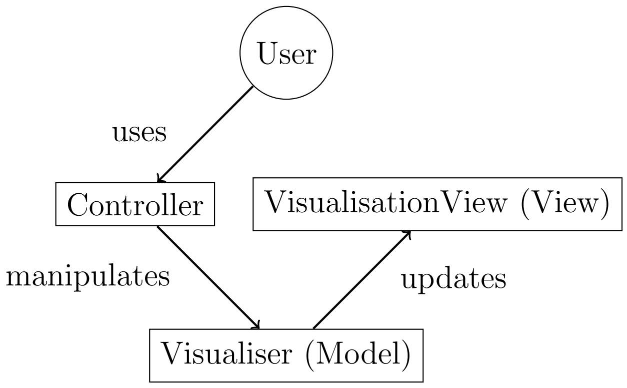


Figure 3.18: The MVC design pattern in practice.

model by half we simulate an intervention which resulted in halving the rates of movement between regions.

Map rendering

The ability to produce nice and helpful visualisations was a key objective and therefore the choice of a map rendering library was important. The essential requirement was to be able to render real-world maps and not just a collection of shapes as in Surveillance and STEM. This would result in a more visually pleasing product, which could also provide the means to explore specific areas in detail from the program itself. There are many Java libraries offering such functionality such as Unfolding Maps and Google Maps Client. We chose Unfolding Maps because of the following features:

- Interactive maps: the maps are rendered with built-in support for panning and zooming. They are also capable of listening to mouse clicks and other input events generated by the users.
- Multiple map providers: the tiles making up the maps can be obtained from numerous built-in providers such as OpenStreetMaps, Google Maps and Microsoft Aerial. Each one focuses on showing different aspects of a physical area (e.g. terrain, roads, airports, rivers, services) which are potentially helpful to epidemiologists.
- GeoJSON support: the library has a simple mechanism to read shapes from a GeoJSON file and render them as markers on the map. This allows us to specify

the borders of each region of interest in a simple format and then add them on the map so that we can show the number of cases per region.

- Multiple maps: the library supports the construction and interaction with multiple maps simultaneously. This is critical in enabling side-by-side comparisons between the various data sources to be made.

Figure 3.20 shows the main fields and methods of the class `MapRenderer`, which uses the library to construct and render maps with OpenGL. The setup phase involves reading the polygon shapes that characterise the borders of the regions of interest and creating the associated `Marker` objects. These associations between the `Region` and `Marker` objects are stored in a hash table.

When the number of cases changes in a region, `VisualisationView` instructs `MapRenderer` to reflect on this change via the update(Region) method. This is an application of the **Observer design pattern**[14] and it is used to avoid checking constantly for changes (polling), a process which wastes resources. The method finds the marker corresponding to that region using the previously constructed hash table and updates the intensity of its colour. The intensity value is calculated using linear interpolation between zero and the maximum number of cases that occur in a region in the data set.

Users can change the type of the map rendered by choosing another tiles provider. There are currently five options for this matter in STAGE:

- OpenStreetMap-Grey. This is the default map provider. Its colours are ideal for emphasizing which regions are hit more than others during an epidemic.
- Google Maps. This is useful for examining the road network and the services provided in each region.
- Microsoft Aerial. This provides a satellite view of the area highlighting the differences of the terrain among regions.
- OpenStreetMap. This highlights the transport links between regions.
- Stamen Water-colour. This gives emphasis on natural resources like rivers and forests.

Note that the Unfolding Maps API makes it straight-forward to extend this list.

The side-by-side rendering of two (or more) maps is made possible by creating and storing not just one `UnfodlingMap` but a list of them. Similarly, the class stores a list of lists of region markers and a list of hash tables, where the size of these lists equals the number of maps created. The setup() method, which is responsible for constructing these objects, divides the rendering area evenly among the supplied `Physical` instances and draws one map for each one. Figure 3.21 how maps are rendered in STAGE.

[14]http://en.wikipedia.org/wiki/Observer_pattern

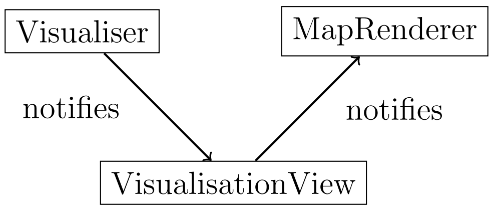
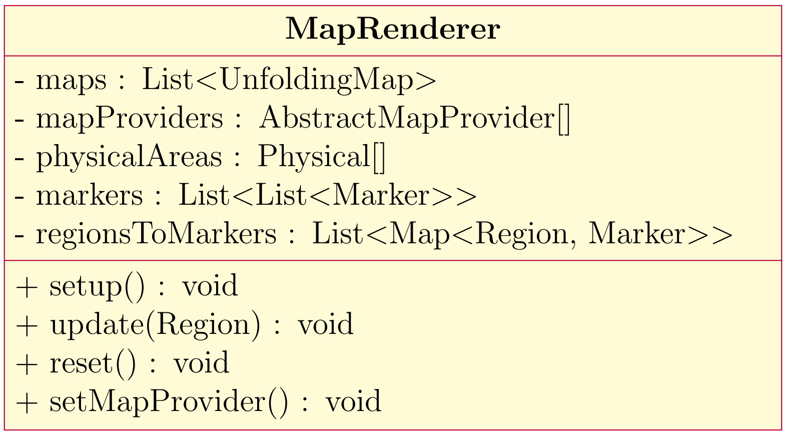


Figure 3.19: The Observer design pattern in practice.



MapRenderer

- maps : List<UnfoldingMap>
- mapProviders : AbstractMapProvider[]
- physicalAreas : Physical[]
- markers : List<List<Marker>>
- regionsToMarkers : List<Map<Region, Marker>>

+ setup() : void
+ update(Region) : void
+ reset() : void
+ setMapProvider() : void

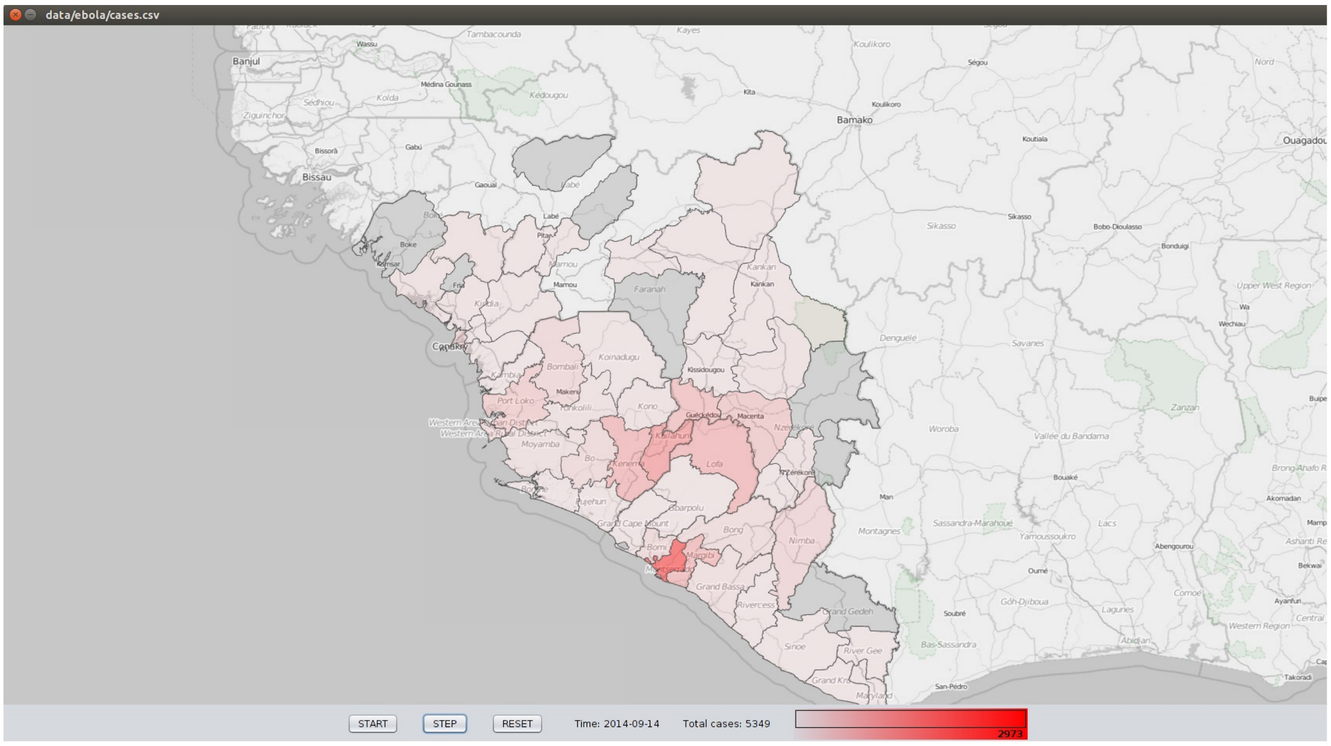
Figure 3.20: UML diagram of MapRenderer.

Graph rendering

STAGE is also capable of visualising epidemics on logical space using graphs. Graphs can be represented natively in Java using adjacency matrices, adjacency lists or just objects with references. However when it comes to rendering these on the screen there is a lot of complexity which must be dealt with. Where should each node be placed so that it is not too far or too close to others? To what degree can Swing elements be constructed in a way that they are visually pleasing? How does the output adapt when new nodes and edges are added dynamically? To avoid such issues it was decided to use a library for the construction and visualisation of graphs.

We decided to use the library GraphStream because it offers the following benefits:

- Dynamic graphs: the structure of the graphs can change dynamically. The display is adapted automatically on each change.
- Graph generators: the library provides several built-in algorithms to construct well-known graphs like Dorogovtsev-Mendes and Barabasi-Albert easily.
- Storage: the graphs can be exported to files so that they can be rebuilt later.
- Nice visualisations: the nodes can be rendered using automatic layout, which are optimised for clarity. They can be easily styled using CSS-like rules.
- User interaction: the display can listen for mouse clicks and deliver events to the rest of the system.



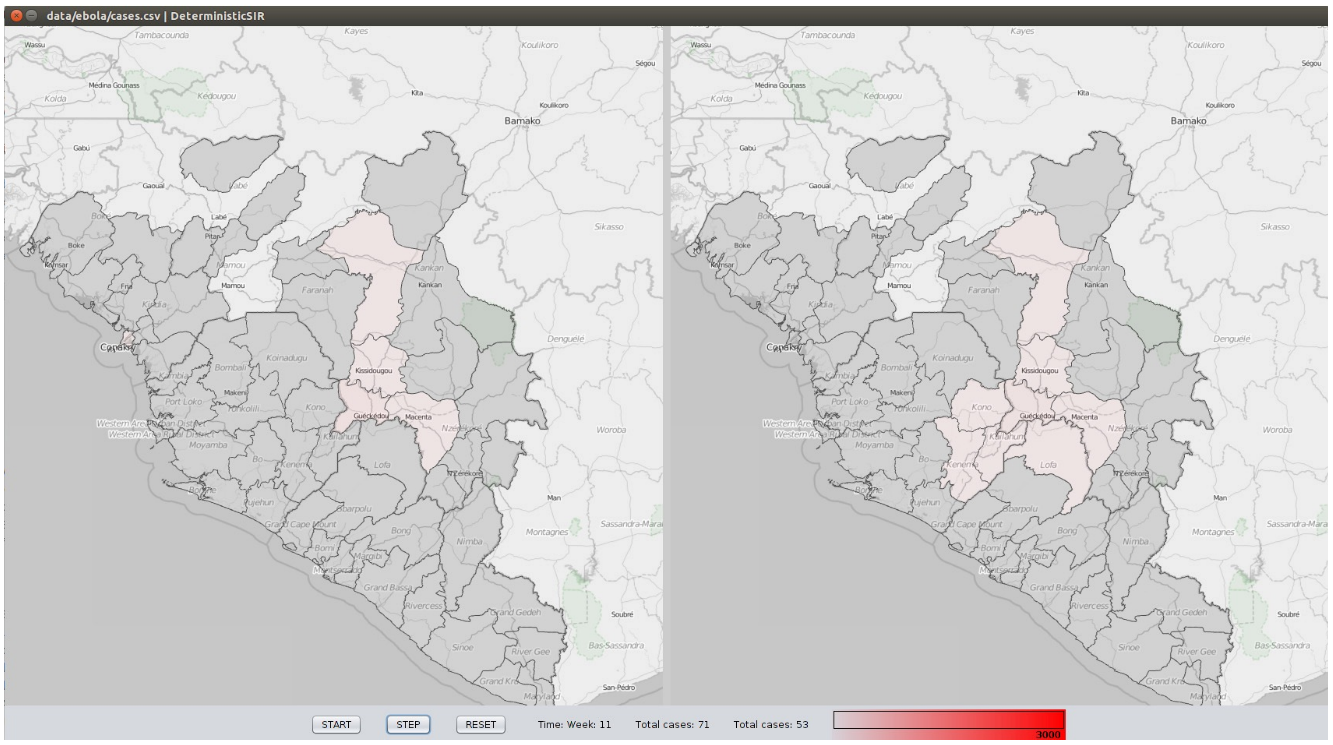


Figure 3.21: Top: single map rendering in STAGE. Bottom: side-by-side map rendering for direct comparisons between data and model or between different models.

- Attributes: the nodes can store any number of attributes, of any type. We make use of this feature by storing the state of each node (susceptible, infected, recovered, etc) as an enum as well as the region associated with that node as a `Region` object.

The class `Graphs` provides a set of static methods for generating graphs using this library. Note that unlike maps there is no need to update the display when the state of a node changes as the library takes care of this by itself. Figure 3.22 shows how the logical space is rendered by the framework.

3.6.9 Model fitting

Figure 3.23 describes the process to fit a model to data. It is implemented by the classes `ModelFitter`, `NelderMeadOptimization` and `SumOfSquaredErrors`. The output consists of the best parameter vector found, the error at that point and the coefficient of determination, R^2.

The class `SumOfSquaredErrors` defines a method that computes the sum of squared errors between the observations (y) and the model's predictions (f) for each region and time point, evaluated with the current parameter vector. Formally the optimization problem is the following:

$$\arg\min_{\theta} \sum_{i=0}^{r} \sum_{j=0}^{n} (y_{ij} - f_{ij}(\theta))^2$$

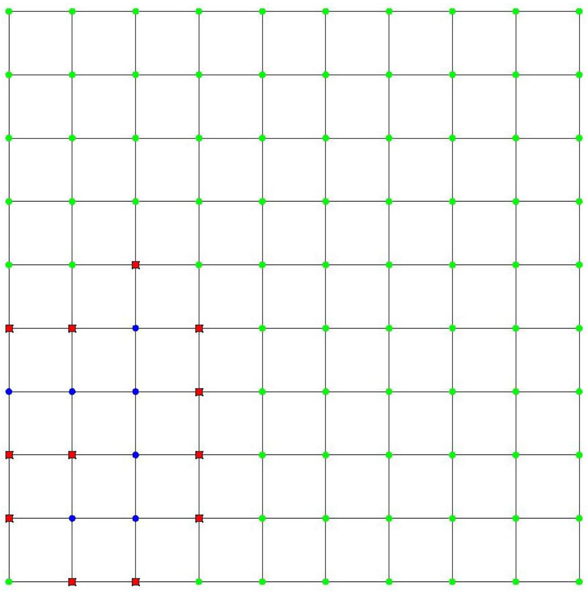
where θ is the parameter vector, r is the number of geographical regions and n is the number of time points. Note that the logarithms of the parameter values are used throughout the process in order to avoid negative values. The optimization problem is solved in `NelderMeadOptimization` using the Nelder-Mean simplex algorithm available in the Apache Commons Math `optim` package.

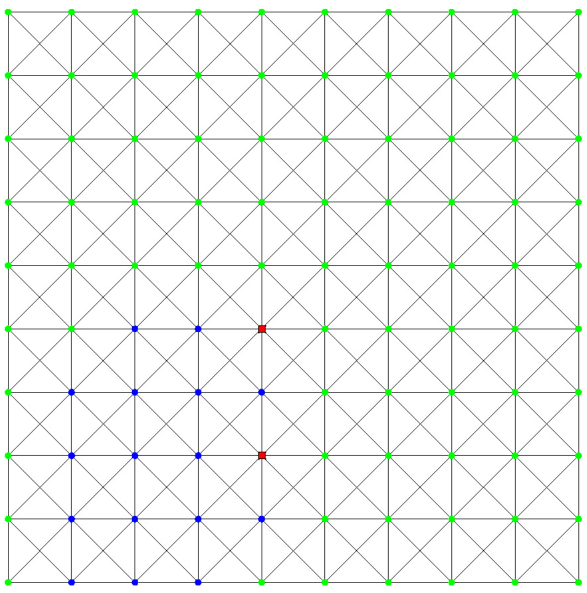
Intermediate and final results are logged using a concrete class that implements the interface `Logger`, passed to `ModelFitter` in the constructor. The use of dependency injection allows us to choose at runtime whether we want to see the results on the console or in a text box in the UI.

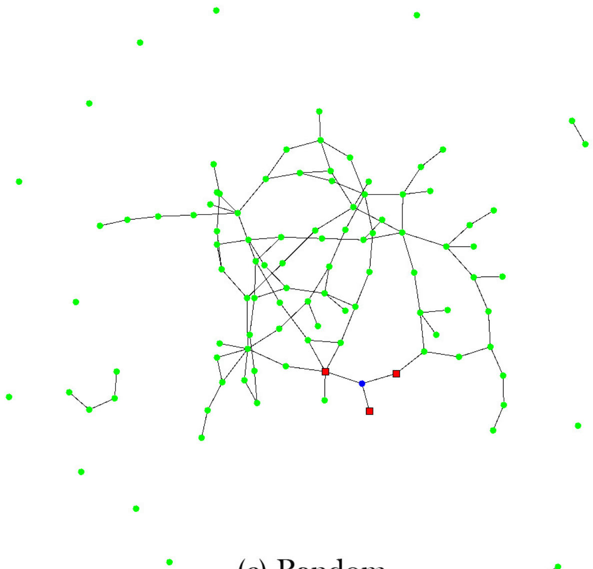
3.6.10 Synthetic data generation

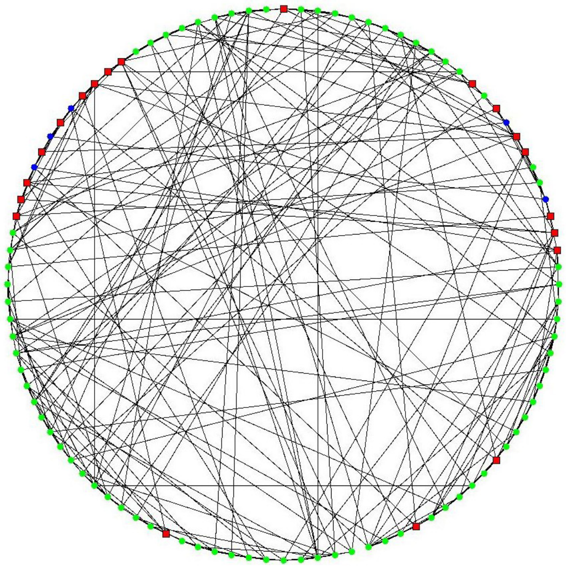
Synthetic data is data generated by the program using stochastic simulation of models with known parameters. It is used as input to the program to test how well the models can be fitted back to it.

Figure 3.24 shows the main fields and methods of the class `SyntheticDataGenerator` which is used to construct this kind of data. Any class which implements the `Simulator` interface (and thus consists of space, disease and mobility models) can be passed as input. In the case of a `RegionBasedSimulator` the generator creates case data for physical areas. It uses the simulator to create a matrix of counts regarding new cases per region per time


(a) Grid


(b) Crossed Grid


(c) Random


(d) Watts-Strogatz

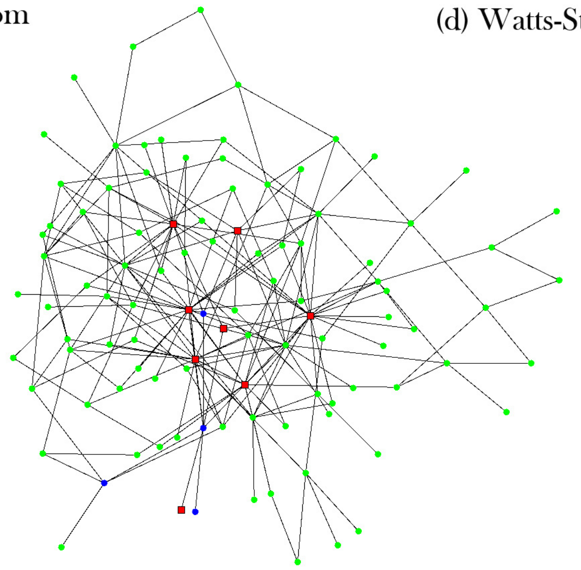

(e) Barabasi-Albert

Figure 3.22: Graphs of 100 nodes as rendered by STAGE.

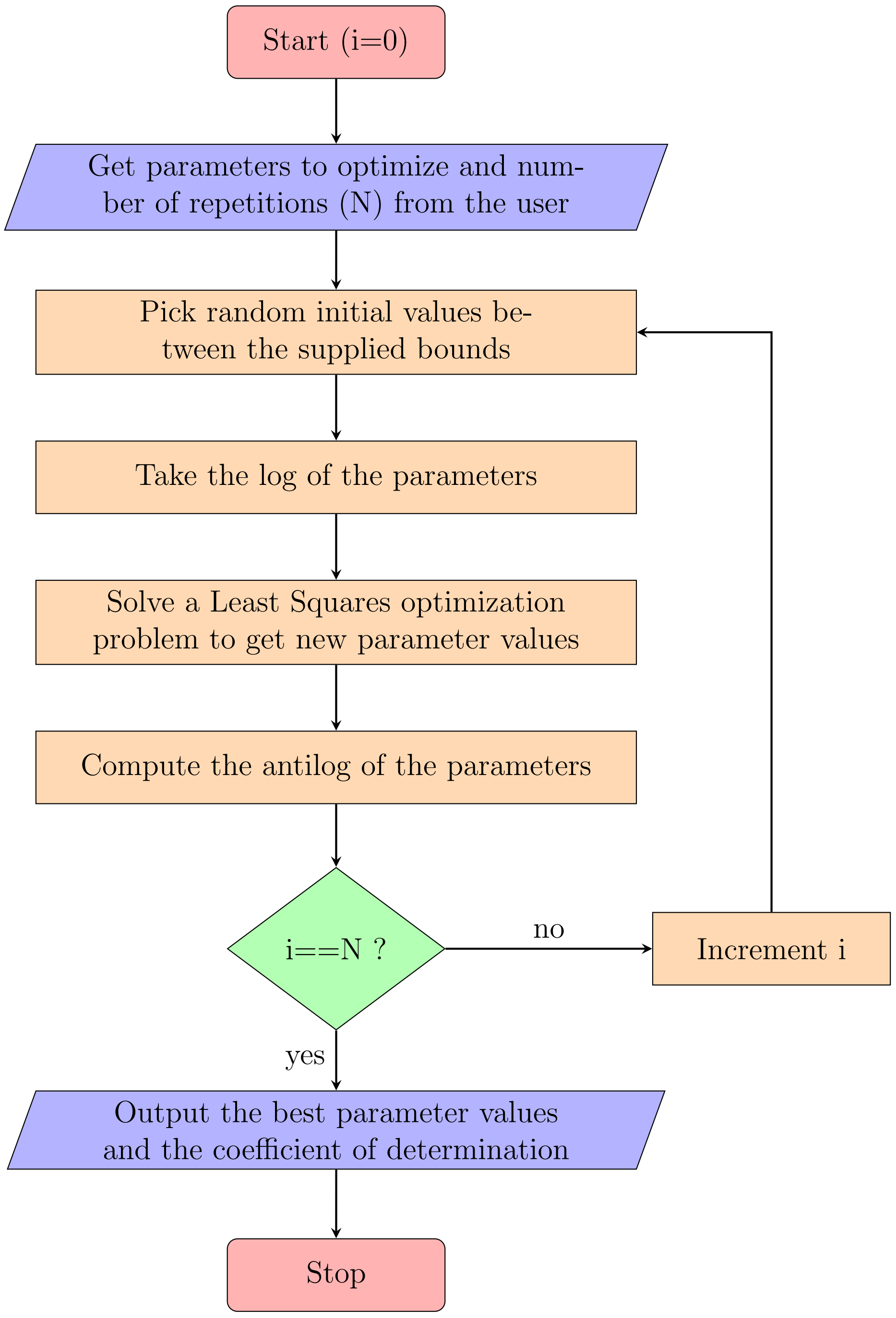

Start (i=0)
Get parameters to optimize and number of repetitions (N) from the user
Pick random initial values between the supplied bounds
Take the log of the parameters
Solve a Least Squares optimization problem to get new parameter values
Compute the antilog of the parameters
i==N ?
no
Increment i
yes
Output the best parameter values and the coefficient of determination
Stop

Figure 3.23: Flowchart describing the process of model fitting in STAGE.

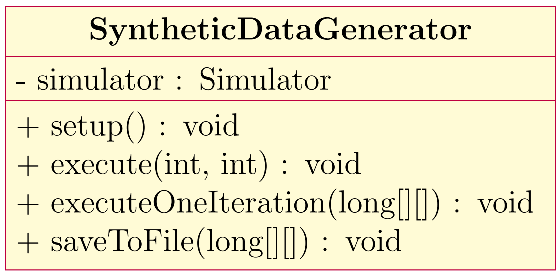

SyntheticDataGenerator
- simulator : Simulator
+ setup() : void
+ execute(int, int) : void
+ executeOneIteration(long[][]) : void
+ saveToFile(long[][]) : void

Figure 3.24: UML diagram of SyntheticDataGenerator.

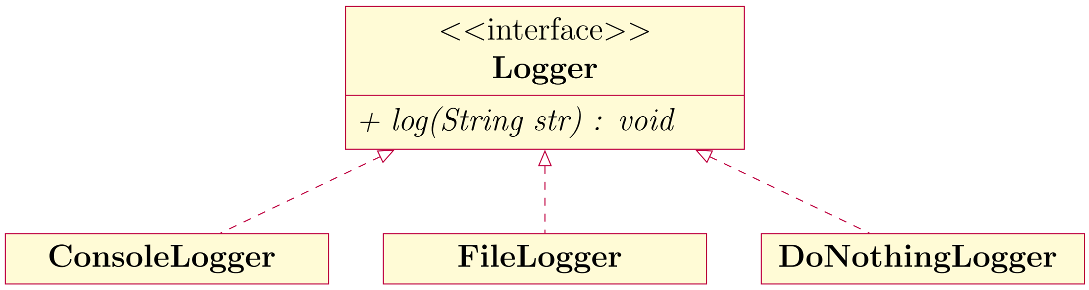


Figure 3.25: UML diagram of Logger hierarchy.

point, and averages it over multiple runs before exporting it as a CSV file. Gillespie's Stochastic Simulation Algorithm - used by the simulator to compute random walks of a deterministic model - makes sure that each run is different than the previous.

The generator can also deal with graphs by giving it an instance of `GraphBasedSimulator`. It starts by exporting the graph into a DGS file so that it can be reconstructed and visualised later by the framework. This is necessary as graphs like Watts-Strogatz and Barabasi-Albert are created randomly. Next it uses the simulator to obtain the integers corresponding to the state of each node in each time point. This state data is then averaged over multiple runs and it is finally exported as a CSV file.

3.6.11 Utilities

Loggers

Loggers are used in the system to output results, informative messages and errors. There are currently three implementations of the `Logger` interface: `ConsoleLogger` which writes to standard output, `FileLogger` which writes to a file and `DoNothingLogger` which does not write anything. The latter is used in model fitting when initiated from the user interface as the results are then displayed directly in a text box. The use of an interface makes it easy to add new kinds of loggers in the future.

CSV handling

The class `CSVHandler` defines static methods to read and write data in the CSV format. It provides a more convenient interface to the Opencsv library which is used for doing the actual I/O operations.

3.7 Dependencies

Table 3.3 summarises the dependencies of the framework on external libraries.

Library	Description
Unfolding Maps	Used to create interactive maps and geovisualizations
Apache Commons Math	Mathematics and statistics packages
GraphStream	Used to construct, model and analyse dynamic graphs
Opencsv	A simple comma-separated values (CSV) file parser
Json-java	JSON encoders and decoders

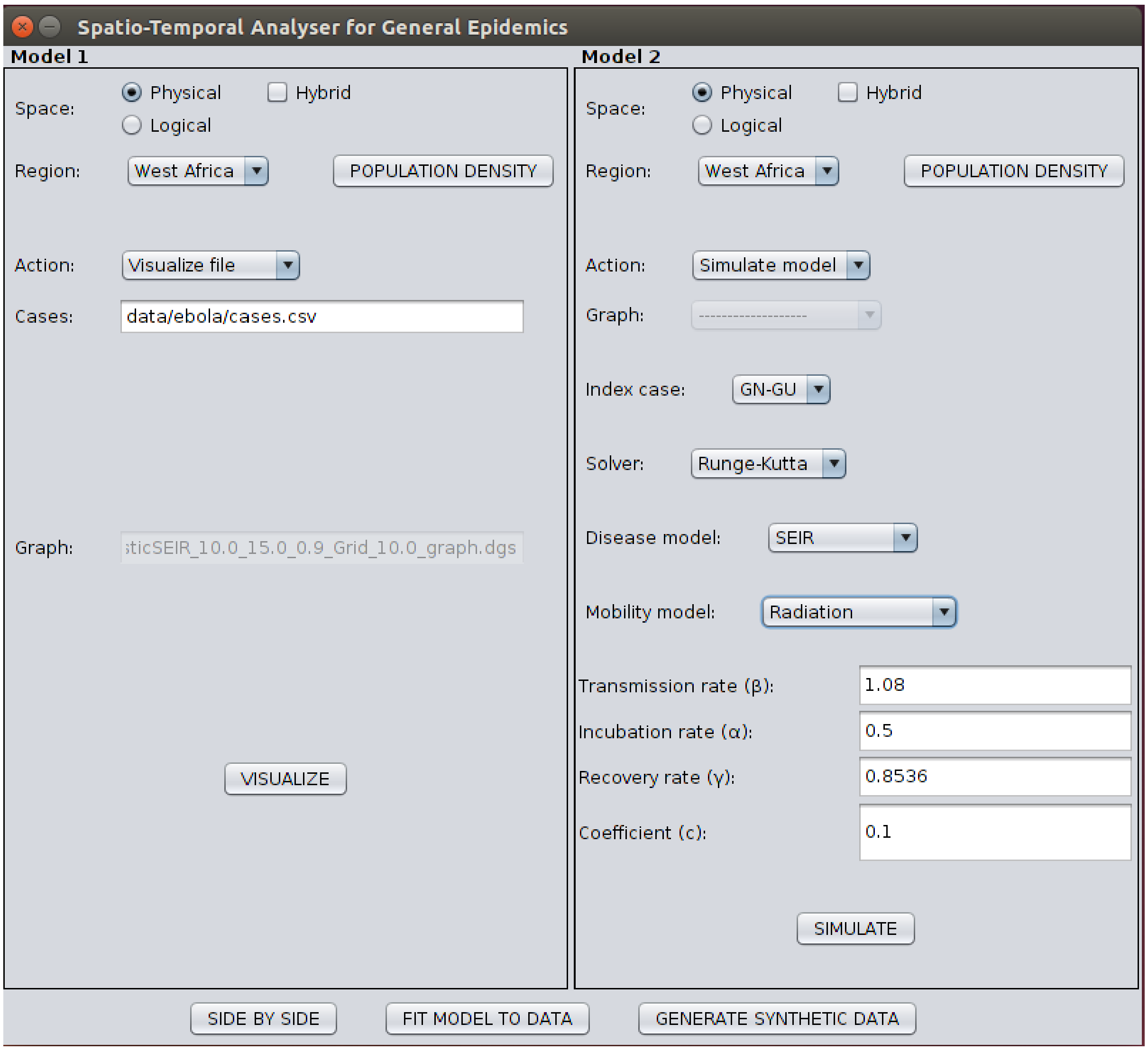
Table 3.3: Dependencies of STAGE on external libraries.

3.8 User interface

The main component that captures user input regarding the settings of the simulation is implemented by the class `Controls`. This Swing-based class provides a simple user interface consisting of radio buttons, checkboxes, comboboxes, text fields and buttons, enabling the user to select data source (file, simulator), models (space, disease, mobility, graph) and action (visualise, simulate, fit model, generate synthetic data). The window is split into two panels, each responsible for configuring its own model. This provides the means to set up the side-by-side visualisation. Furthermore, by storing all Swing component in lists, `Controls` can be easily extended to generate more than two sets of configuration options at a time. Figure 3.26 shows the main window of STAGE.

User requests are delegated to the `Framework` class which takes care of constructing the models and data sources, as well as starting the visualiser, model fitter and synthetic data generator. Figure 3.27 shows the control flow between the user interface and the core program.

The model fitting window is shown in Figure 3.28. It allows the user to specify what parameters need to be optimised in the model in order to make the predictions as close as possible to the real data. To avoid freezing the user interface until this computationally expensive procedure completes, it was decided to use the utility class `SwingWorker`. This provides a mechanism to carry out time-consuming tasks on the background Worker thread leaving the GUI thread free to respond to user interactions. It also allows the two



Spatio-Temporal Analyser for General Epidemics

Model 1

Space: [✓ Physical] Hybrid Logical

Region: West Africa — POPULATION DENSITY

Action: Visualize file

Cases: data/ebola/cases.csv

Graph: ;ticSEIR_10.0_15.0_0.9_Grid_10.0_graph.dgs

VISUALIZE

Model 2

Space: [✓ Physical] Hybrid Logical

Region: West Africa — POPULATION DENSITY

Action: Simulate model

Graph: -------------------

Index case: GN-GU

Solver: Runge-Kutta

Disease model: SEIR

Mobility model: Radiation

Transmission rate (β):	1.08
Incubation rate (α):	0.5
Recovery rate (γ):	0.8536
Coefficient (c):	0.1

SIMULATE

SIDE BY SIDE — FIT MODEL TO DATA — GENERATE SYNTHETIC DATA

Figure 3.26: Main window of STAGE.

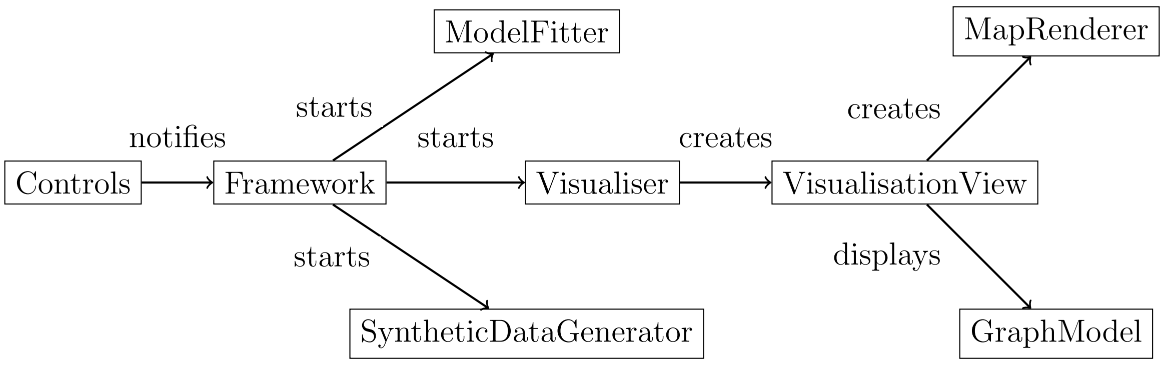
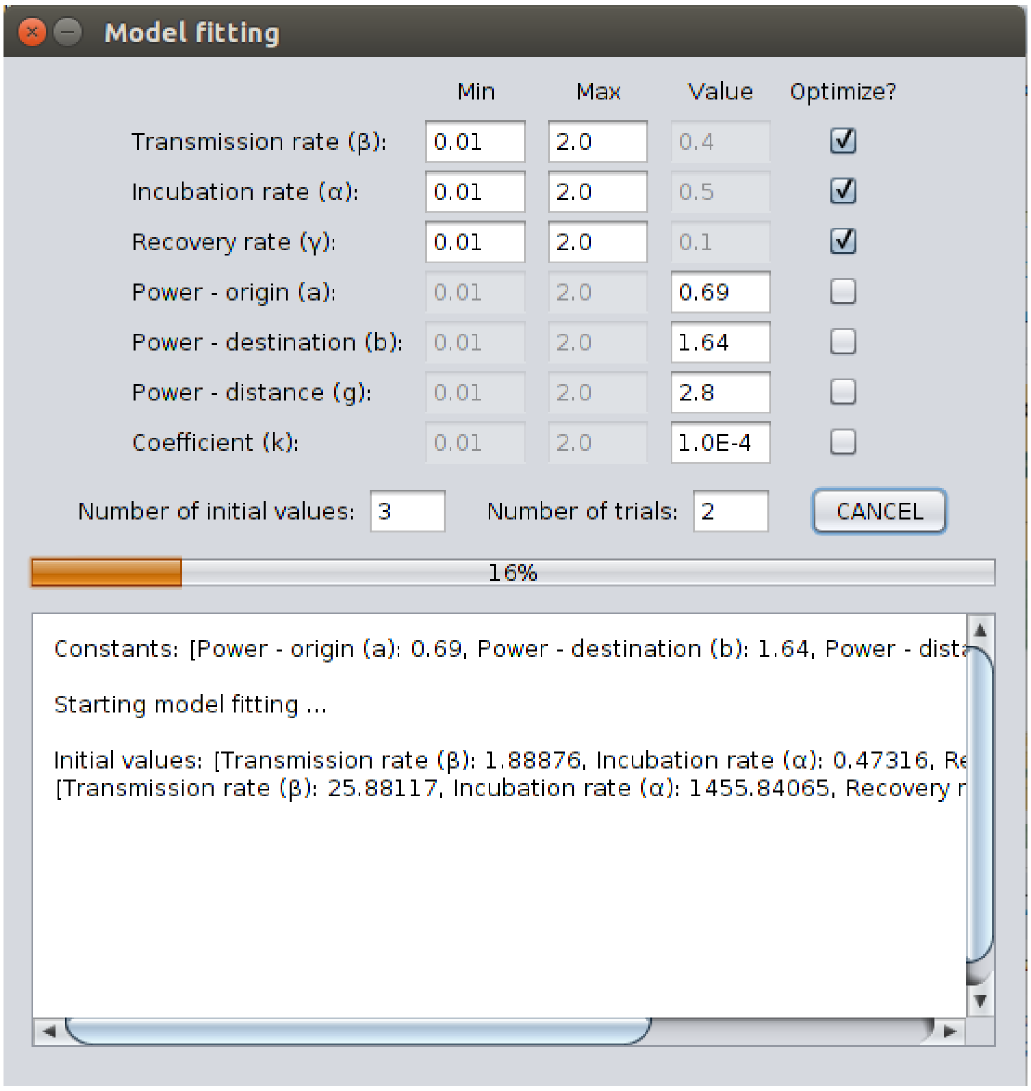


Figure 3.27: Main control flow in the program.



	Min	Max	Value	Optimize?
Transmission rate (β):	0.01	2.0	0.4	✓
Incubation rate (α):	0.01	2.0	0.5	✓
Recovery rate (γ):	0.01	2.0	0.1	✓
Power - origin (a):	0.01	2.0	0.69	☐
Power - destination (b):	0.01	2.0	1.64	☐
Power - distance (g):	0.01	2.0	2.8	☐
Coefficient (k):	0.01	2.0	1.0E-4	☐

Number of initial values: 3 Number of trials: 2 CANCEL

16%

```
Constants: [Power - origin (a): 0.69, Power - destination (b): 1.64, Power - dist

Starting model fitting ...

Initial values: [Transmission rate (β): 1.88876, Incubation rate (α): 0.47316, Re
[Transmission rate (β): 25.88117, Incubation rate (α): 1455.84065, Recovery r
```

Figure 3.28: Model fitting window.

threads to communicate so that intermediate and final results can be presented on the screen. Figure 3.29 shows this communication.

Synthetic data generation is also an expensive process so it makes use of a `SwingWorker` too. The window used to control this process is shown in Figure 3.30. The user can specify the number of time points to generate data for, as well as the number of runs to be made before taking their average in the end.

The main window also has an option to start the visualiser in a mode which shows the distribution of population density across the geographical regions of interest. This may help researchers assess the role of this factor when deciding which mobility model to use in their simulations.

$$\begin{tikzcd}
\text{Worker thread} \arrow[r, bend left=20, "\text{sends intermediate results}"] \arrow[r, bend right=20, "\text{sets progress level}"'] \arrow[r, bend right=50, "\text{sends final result}"'] & \text{GUI thread} \arrow[l, bend right=50, "\text{submits task}"']
\end{tikzcd}$$

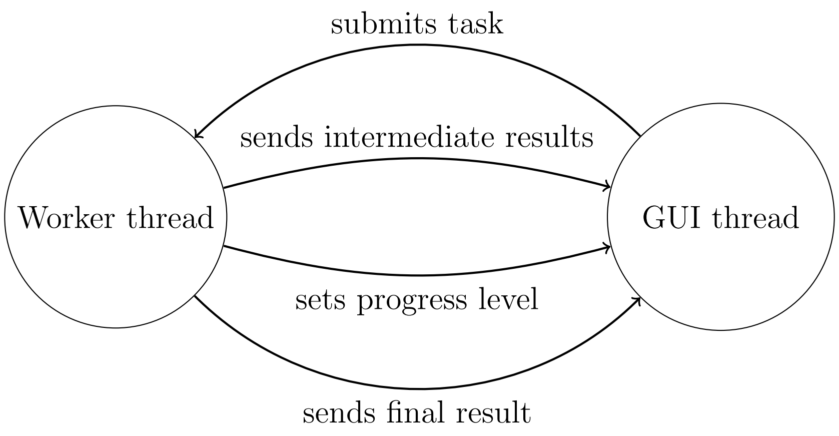
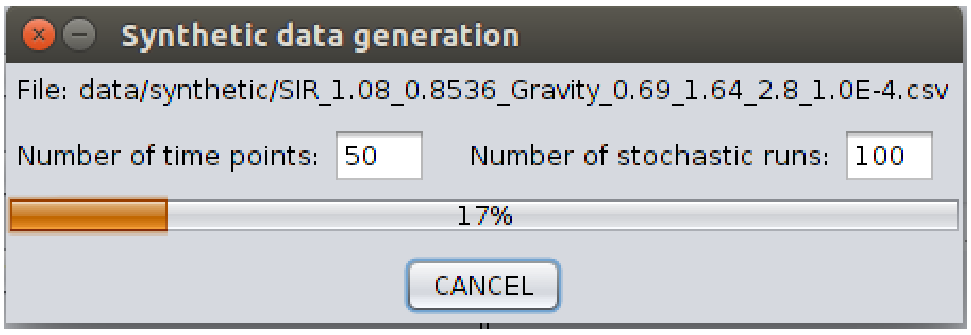


Figure 3.29: Communication between Worker and GUI threads in model fitting and synthetic data generation.



File: data/synthetic/SIR_1.08_0.8536_Gravity_0.69_1.64_2.8_1.0E-4.csv

Number of time points: 50 Number of stochastic runs: 100

17%

Figure 3.30: Synthetic data generation window.

3.9 Features

The components described earlier give STAGE the necessary functionality to visualise, model, simulate and analyse the spatio-temporal spread of biological and socio-technological epidemics. Its main features are summarised below:

- Flexible space model: epidemics can be simulated and visualised over physical and/or logical space.
- Flexible disease model: compartmental disease models may be population-based, agent-based, deterministic or stochastic.
- Flexible mobility model: any mathematical model of human mobility can be used in simulations.
- Interactive visualisation: several kinds of real-world maps and graphs are available to use to display the results. User interactions such as zooming, panning and clicking are supported.
- Side-by-side visualisation: two independent models or a model and the real data set can be visualised side-by-side in lock step. This allows us to receive immediate feedback about the models under test.
- Intervention policies: the model's parameters can be changed during simulation to account for the application of certain containment measures. This allows us to predict the outcomes of these measures.
- Flexible model fitting procedure: the subset of parameters to optimise as well as the number of trials are specified by the user. The coefficient of determination is used to describe the goodness of fit.
- Synthetic data generation: stochastic trajectories of deterministic models can be generated to test the recoverability of the models' parameters.
- Extensibility: additional space, disease, mobility and graph models can be constructed by implementing the appropriate interfaces.
- Usability: the number of clicks needed to visualise data sets, simulate models and fit models to data is minimal.

Chapter 4

Case study: 2014 West Africa Ebola outbreak

The ongoing Ebola Virus Disease (EVD) epidemic in West Africa is the largest of its kind in history. Epidemiologists believe that the index case was a one-year-old boy in a village in Gueckedou, a prefecture of Guinea, who died in December 2013. Due to the fact that Gueckedou is near the Sierra Leone and Liberian borders, the virus spread quickly among the three countries. Although there have been a few cases in other countries like Nigeria and the USA, the transmission of the disease has only been intense in Guinea, Sierra Leone and Liberia, where the fatality rate is estimated to be as high as 71% [46]. The total number of deaths by 3 June 2015 was 11,147 [45].

4.1 Data collection

We constructed our data set based on the cases and deaths per week and region (administrative level 2) in Guinea, Sierra Leone and Liberia, published by WHO[1]. It contains the number of confirmed and probable cases reported in each region every week from the beginning of the outbreak until 13 May 2015. We also obtained data about the population and area of each region from GeoHive[2]. Furthermore, the GeoJSON file containing the borders of the 55 regions in the data set was constructed from the Shapefile data available from WHO.

4.2 Visualisation

Figure 4.1 shows the spatio-temporal spread of Ebola as visualised by STAGE. We observe that the disease initially spread to adjacent regions of Gueckedou. By epidemic week 20 it had moved to more distant but densely populated regions such as Siguiri and

[1]http://apps.who.int/gho/data/node.ebola-sitrep.ebola-country?lang=en

[2]http://www.geohive.com

Margibi. The three capital cities - Conakry (Guinea), Freetown (Sierra Leone) and Monrovia (Liberia) - were also hit, due to people visiting them for the services they provide [30]. By week 30, the regions close to Gueckedou and the capitals had experienced a lot of cases. This pattern continued until the final week of the data set.

4.3 Analysis

We used SPSS[3] to plot scatter diagrams in order to check for linear dependence between two variables. The statistic we used is the Pearson product-moment correlation coefficient, r. In each case we are using a two-tailed significance test to assess the result.

4.3.1 Cases against distance

Figure 4.2 shows the number of cases per region against the distance between the region and Gueckedou, the epidemic epicentre. In this case r is -0.187, which suggests that there exists weak evidence of negative linear correlation between the two variables.

Figure 4.3 shows the number of cases per region against the distance between the region and the closest of Gueckedou, Monronia, Conakry and Freetown (i.e. the epicentre and the capitals of the three countries). In this case r is -0.480, which suggests that there exists moderate to strong negative linear correlation between the two variables, significant at the 1% level. The result verifies what we observed visually: the regions close to the epicentre or the capitals are hit more.

4.3.2 Cases against population

Figure 4.4 shows the number of cases per region against the population of that region. In this case r is 0.493, which suggests that there exists moderate to strong positive linear correlation between the two variables, significant at the 1% level. More populous places are therefore hit more.

4.3.3 Cases against population density

Figure 4.5 shows the number of cases per region against the population of that region. In this case r is 0.568, which suggests that there exists strong positive linear correlation between the two variables, significant at the 1% level. This suggests that population density may be better in helping us explain the evolution of the epidemic across West Africa. In fact, the heat maps of population density across the regions and cumulative cases have a lot of similarities, as shown in Figure 4.6.

[3]http://www-01.ibm.com/software/uk/analytics/spss/

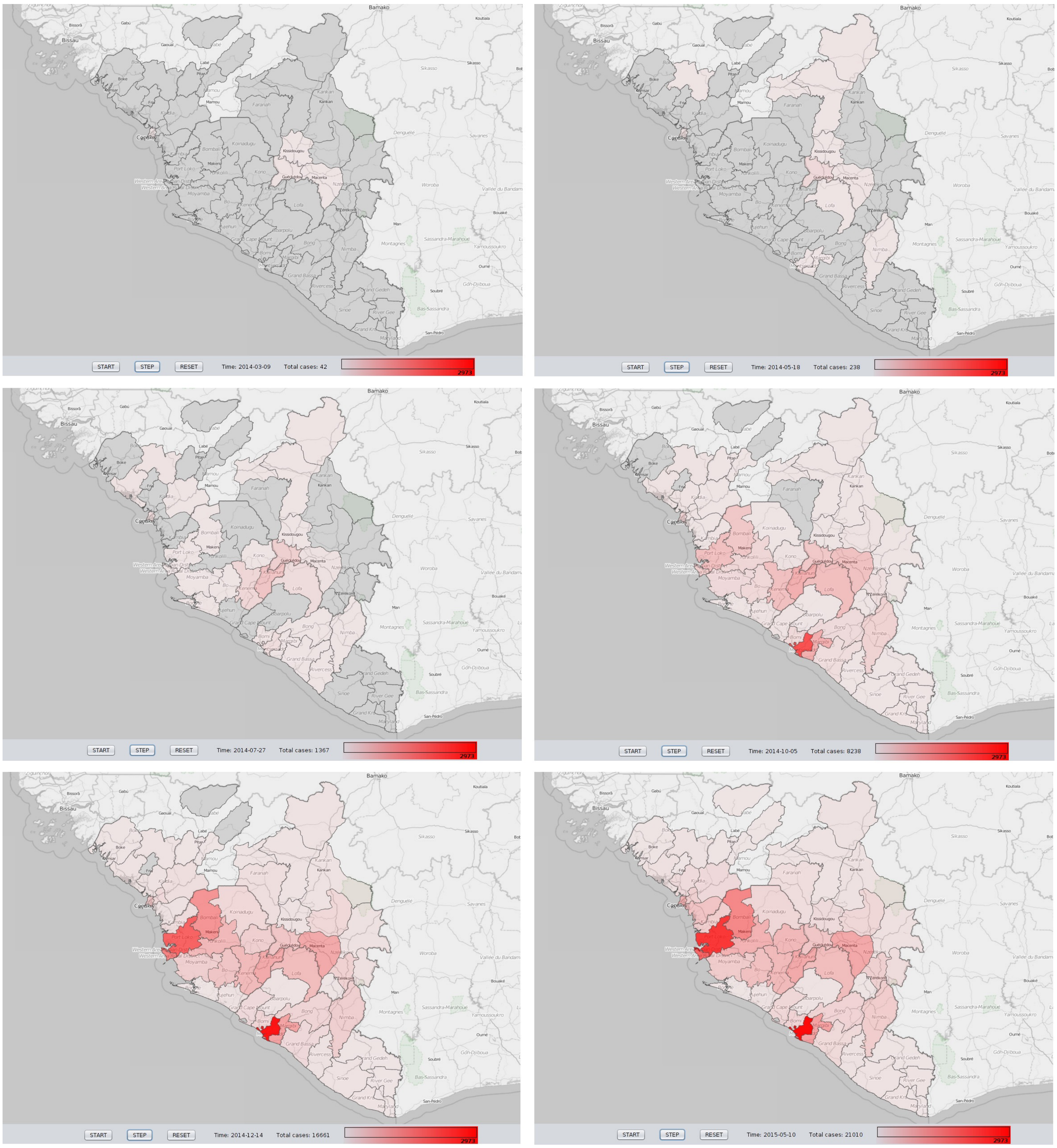

START STEP RESET Time: 2014-03-09 Total cases: 42 2973
START STEP RESET Time: 2014-05-18 Total cases: 238 2973
START STEP RESET Time: 2014-07-27 Total cases: 1367 2973
START STEP RESET Time: 2014-10-05 Total cases: 8238 2973
START STEP RESET Time: 2014-12-14 Total cases: 16661 2973
START STEP RESET Time: 2015-05-10 Total cases: 21010 2973

Figure 4.1: Left-to-right, row by row: spatio-temporal spread of Ebola in Guinea, Sierra Leone and Liberia after 10, 20, 30, 40, 50 and 70 weeks, as visualised by STAGE. The red colour represents the cumulative cases.

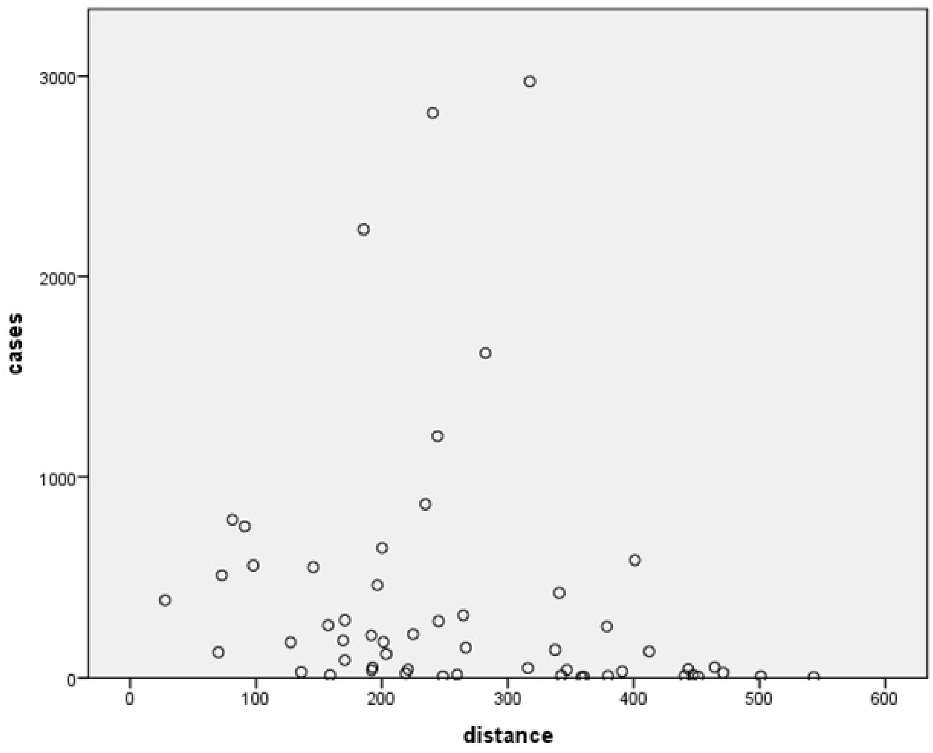


Figure 4.2: Cases against distance from epicentre, per region.

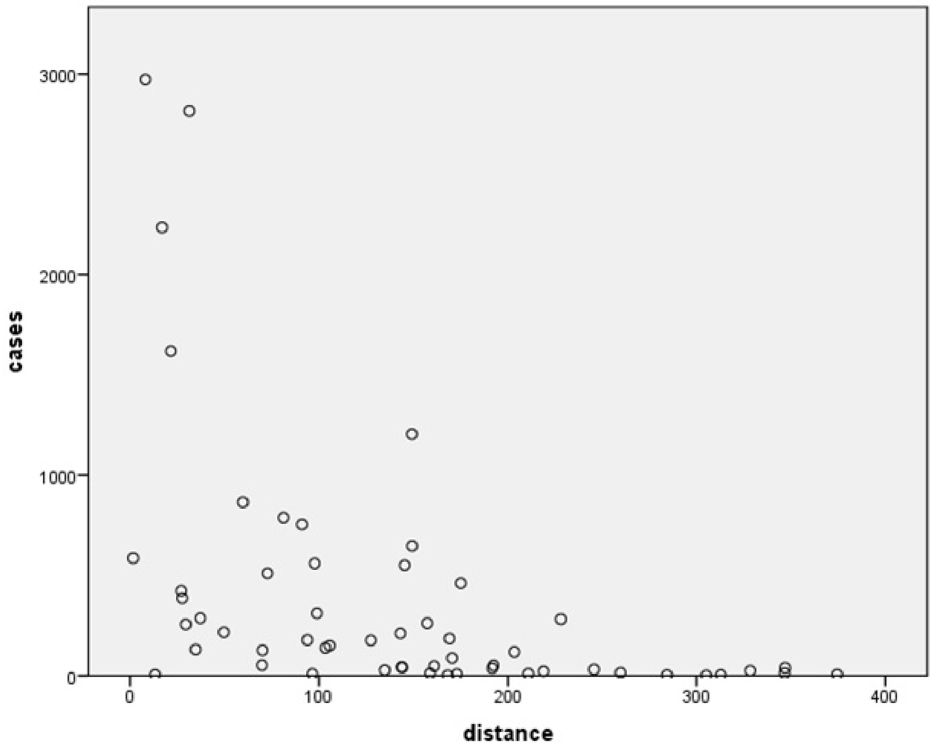


Figure 4.3: Cases against distance from epicentre or closest capital, per region.

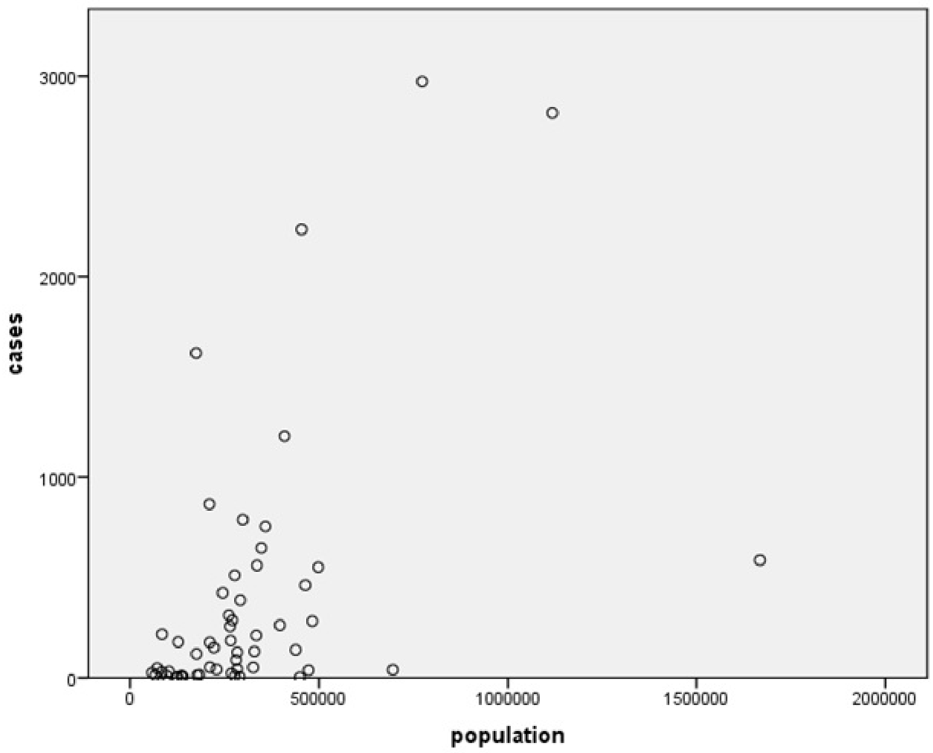


Figure 4.4: Cases against population, per region.

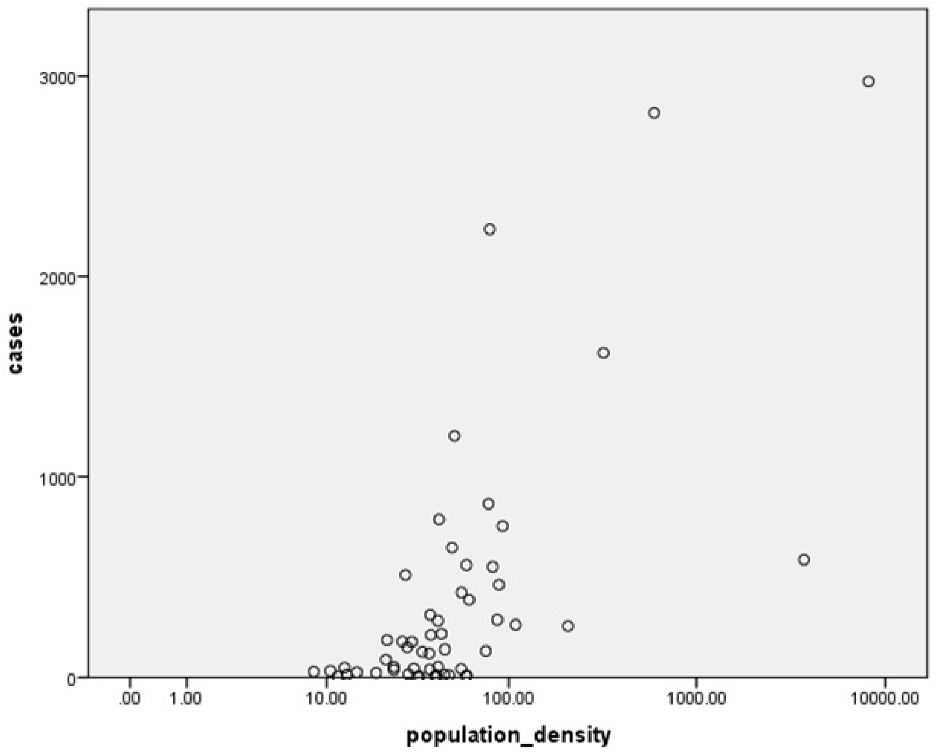
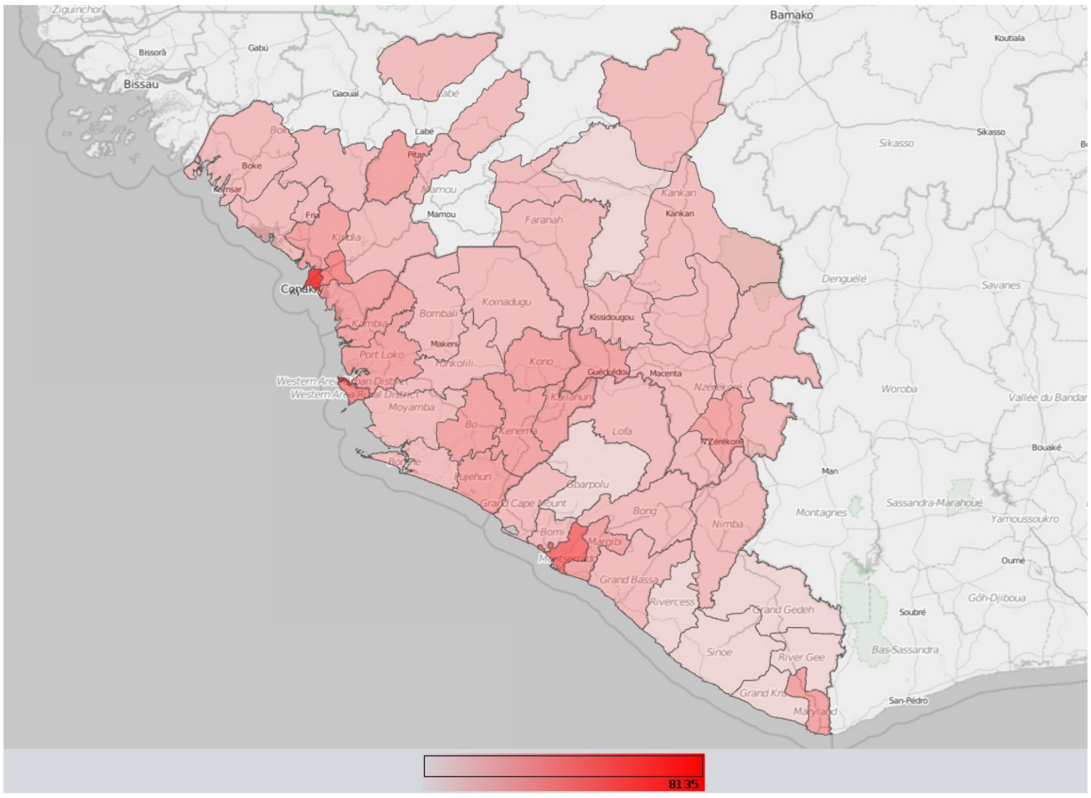


Figure 4.5: Cases against population density, per region.



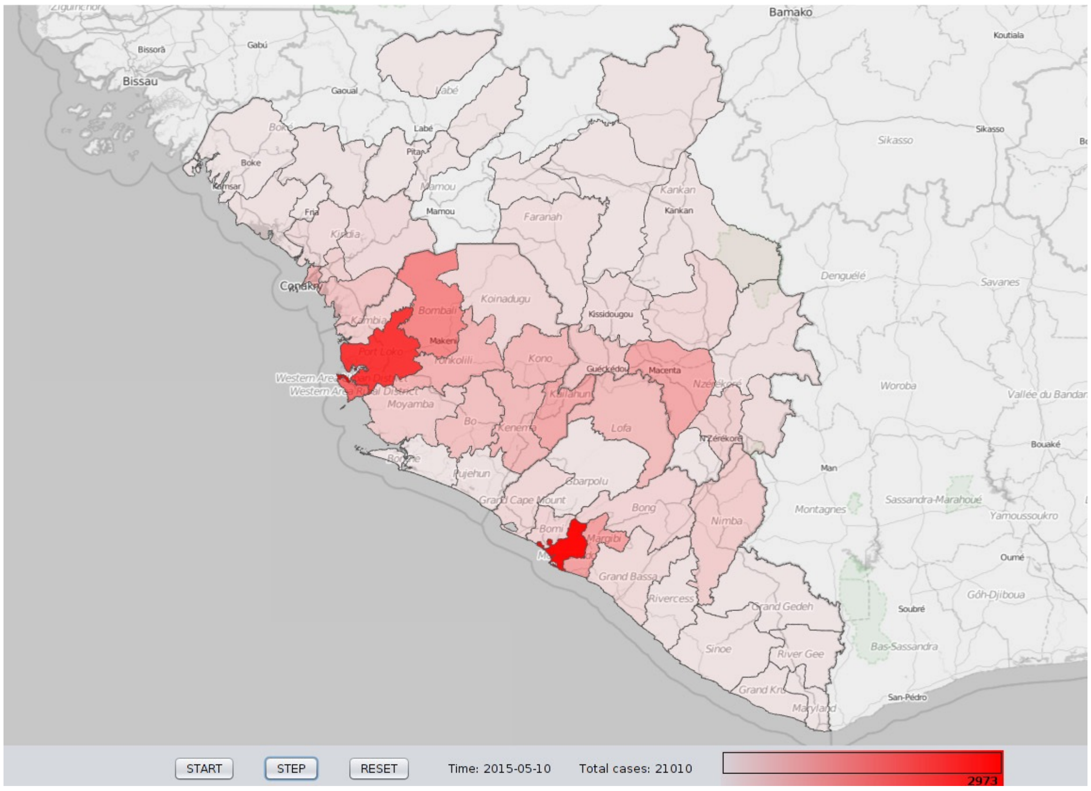


Figure 4.6: Top: population density across Guinea, Sierra Leone and Liberia. Bottom: cumulative Ebola cases until 13/05/2015.

4.4 Spatio-temporal modelling

4.4.1 Mobility model

Based on our observations so far, a suitable mobility model should take into account at least the proximity and the populations (or population densities) between any two regions. The Gravity and Radiation models described in Section 2.7.1 are therefore candidates.

Researchers from the Flowminder Foundation[4] have been working on modelling human movement in Africa since the beginning of the outbreak in order to help control the epidemic. In *West Africa Human Mobility Models* [44] they provide parameter values for the Gravity model for several countries. These were obtained by analysing their census migration data as well as mobile phone call detail records for Senegal, Cote d'Ivoire and Kenya. The same data for Cote d'Ivoire was used in *Approaching the Limit of Predictability in Human Mobility* [19], where the researchers managed to achieve a prediction accuracy of 87%. Despite the fact that this kind of data does not represent movement in the event of epidemic outbreaks, it is still useful in indicating how people tend to move in general.

Simple spatial models such as Gravity and Radiation succeed in describing the general observation that people prefer to move to regions which are either very close to them or very populous. Still, for the current Ebola outbreak, more complex models are needed to describe the spread accurately, and so researchers are working on coming up with more suitable models. The Spatial Ecology and Epidemiology Group in the University of Oxford is currently working on a project to develop a global and regional model of Ebola importation risk and provide predictions on maps through their website[5].

4.4.2 Disease model

In Section 2.4 we described briefly a compartmental model by Legrand *et al* which was created specifically for the case of Ebola in Africa. It uses six compartments and tries to capture explicitly the transmission paths from hospitalized infectious individuals and dead individuals who cause additional infections due to unsafe burial practices. A further extension, and perhaps the most sophisticated Ebola model to-date, is the one created by the developers of STEM [38]. It has eight compartments:

S the individuals susceptible to the disease.

E the individuals infected by the disease but not yet infectious.

I the infectious individuals.

D the individuals who are dead but not buried yet.

[4] http://www.flowminder.org

[5] http://seeg-oxford.github.io/ebola-spread/

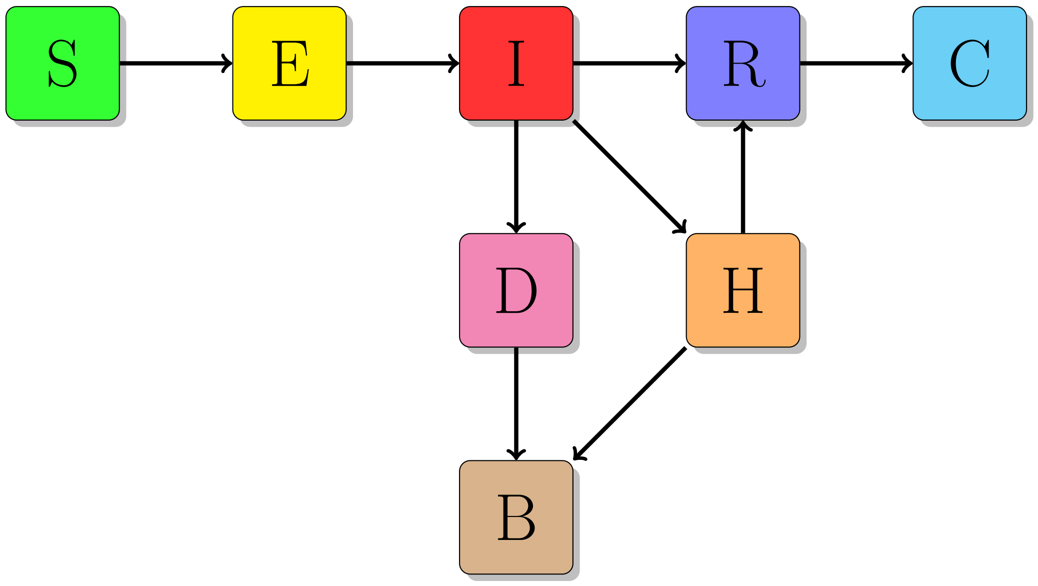


Figure 4.7: Transitions between compartments in the STEM Ebola model.

B the individuals who are buried.

H the individuals who are in the hospital.

R the individuals who have recovered, but can still transmit the disease as an STD.

C the individuals who have completely recovered.

Unsafe burial practices play a crucial role in the transmission of the disease in Africa. In August 2014, Guinea's Ministry of Health reported that 60% of the cases in the country could be linked to contact by an infectious corpse [31]. In November of the same year WHO estimated that 80% of the cases in Sierra Leone were also linked to these practices. It is therefore essential to model the transmission on these occasions explicitly.

The H compartment is used to model the different recovery and death rates the hospitalised patients have compared to those who do not visit a clinic, as well as the transmission between patients and health care workers.

Furthermore, in contrast to traditional compartmental models, the R compartment represents an intermediate stage. According to WHO, the disease can remain in the semen of men who have been declared recovered for up to three months [32]. Recently, there has been a case where the virus was still present there after six months the survivor was declared clean [7]. WHO warns that although transmission through sex has not been proved, it is probable. The model accounts for this transmission path by having the C compartment, which consists of the individuals who have recovered completely.

4.5 Simulation

We implemented the STEM Ebola model described in the previous section in the class `SEIDBHRC` under the framework's `disease` package. We then obtained estimates for the model's parameter values from multiple sources including WHO and STEM. We used the model fitting procedure to obtain estimates for two parameters - β_d and β_r - as we could not find reliable sources for them. Table 4.1 shows the values we used in the simulation.

Parameter	Symbol	Value	Source
Transmission rate	β_i	0.224 per day	[38]
Transmission rate - in funerals	β_d	0.028 per day	Optimisation
Transmission rate - in hospitals	β_h	0.33 per day	[21]
Transmission rate - as STD	β_r	0.0014 per day	Optimisation
Mean duration of incubation period	$1/\alpha$	11.4 days	[39]
Mean time for hospital admission	$1/\tau$	2.5 days	[45]
Mean infectious time	$1/\gamma_i$	8.2 days	[38]
Mean infectious time - hospital	$1/\gamma_h$	4.6 days	[21]
Mean time to bury	$1/\delta$	2 days	[21]
Mean time to recover completely	$1/\epsilon$	5 weeks	[38]
Case fatality ratio	d_i	0.708	[39]
Case fatality ratio - hospital	d_h	0.53	[45]
Proportion of cases hospitalised	h	0.5	[41]

Table 4.1: SEIDBHRC parameter values used in the simulation.

We also obtained estimates for the parameters of the Gravity model of human mobility from [44]. Specifically, we used the values derived from the CDR data of 150,000 randomly sampled individuals in Senegal during the entire 2013, as suggested by FlowMinder (it is the most recent CDR data regarding countries in West Africa). Despite the fact that this data describes the mobility patterns in a country other than Guinea, Sierra Leone or Liberia, it can still be used to cover the lack of recent data for these three countries. This eligibility comes from the observation that the parameter estimates when the model is trained with CDR data for each of Senegal, Cote d'Ivoire and Kenya are similar [43]. Table 4.2 shows the values we used in the simulation.

Parameter	Symbol	Value
Origin population exponent	α	0.47
Destination population exponent	β	0.46
Distance exponent	γ	1.78
Coefficient	k	3.93

Table 4.2: Gravity parameter values used in the simulation.

The remaining bits of the configuration are given in Table 4.3. The ‘time unit’ is used in the computation of the rate parameters.

Option	Value
Space	Physical
Solver	Runge-Kutta
Index case	Gueckedou
Time unit	Week

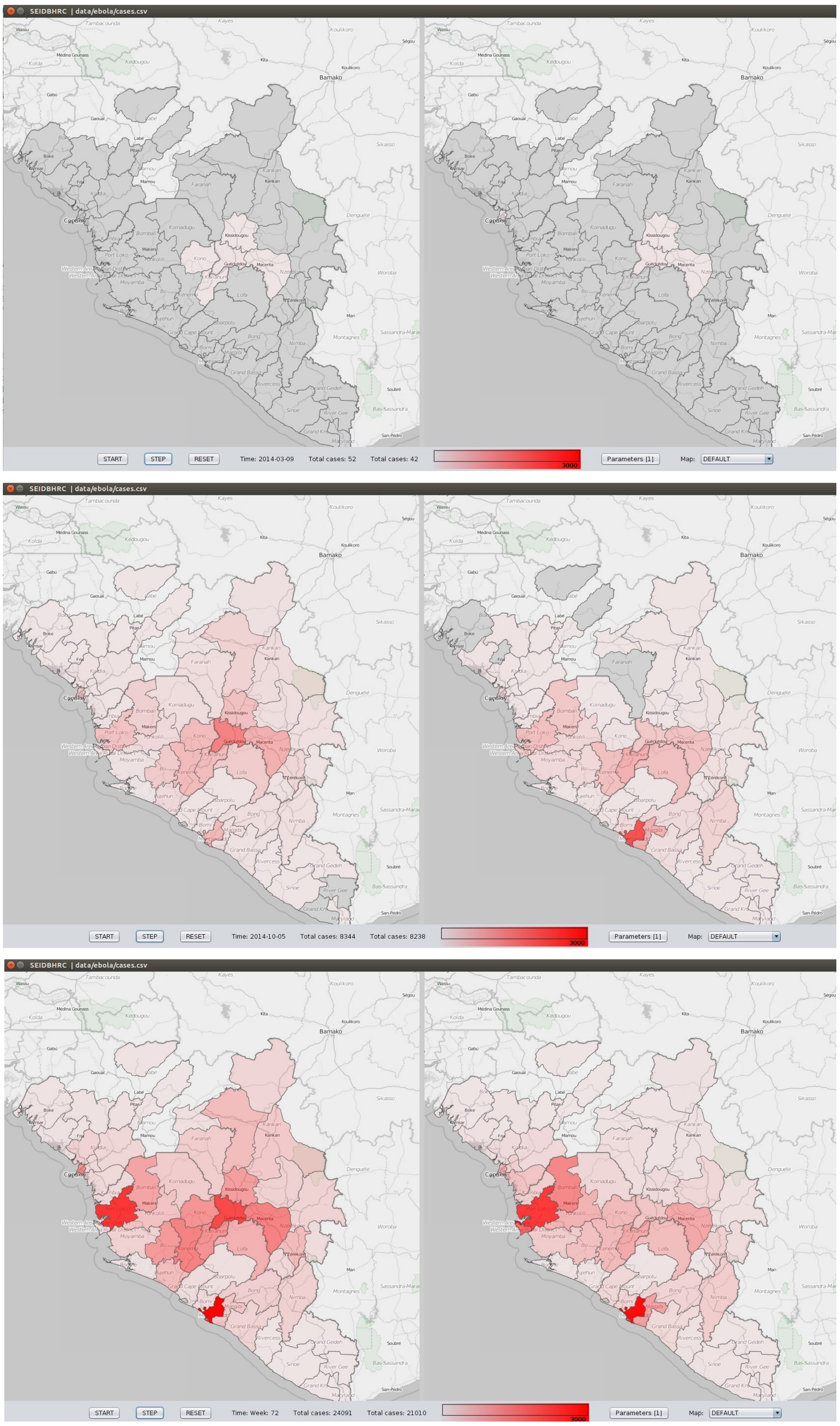
Table 4.3: Remaining options used in the simulation.

Figure 4.8 shows the results of the simulation. We observe that after ten weeks the disease is predicted to have spread in four regions which are adjacent to the epicentre, having a total of 52 cases. At the same time our WHO data set contains 42 observations over a subset of the predicted regions. After forty weeks the model predicts 8,344 cases compared to 8,238 observed ones. The spatial patterns are also similar, with the main differences being that the model (1) infects more regions than in reality and (2) suggests that Gueckedou is hit to a greater extent. By week 70, the model predicts 24,091 cases compared with 21,010 observations. The difference in the total number of cases is quite significant, but the spatial patterns between the two are not too dissimilar.

4.6 Discussion

For our spatio-temporal model the coefficient of determination, R^2, in the above simulation setting was 0.212. This suggests that it is not a very good fit to the data. It comes at no surprise as in the end the model predicts around 3,000 more cases in total than reality. It features a greater concentration of cases around Gueckedou too. However, we have to acknowledge that our data set is not necessarily representative of the ground truth as a portion of cases remains undetected e.g. when people are treated at home or not at all. The Centres for Disease Control and Prevention estimate that the true number of Ebola cases is around 2.5 times higher than the recorded number [20]. There are also spikes in the data set as some cases are reported late, especially in remote areas [27]. Thus, predicting a higher number of cases makes sense both in total and around Gueckedou. Furthermore, we have used a model for human mobility which may be too simple to capture movement between regions sufficiently well. This model was also trained using data about a country which did not experience intense transmission as there is currently a shortage of up-to-date mobility data regarding Guinea, Sierra Leone and Liberia. Nevertheless, the simulation succeeded in conveying the general spatio-temporal spread of the disease. This makes it useful for making short-term predictions on a high level.

To illustrate the importance of analysing the spatial component in addition to the temporal one we compare the results of the Gravity model with the ones obtained using the Common Borders one. This is a far simpler model as it only allows the disease to spread to adjacent regions. It has a single parameter, the mixing rate k, which was set to 0.02 following an optimisation process. Table 4.4 shows the sum of squared errors and the


SEIDBHRC | data/ebola/cases.csv

Time: 2014-03-09   Total cases: 52   Total cases: 42

Parameters [1]   Map: DEFAULT

SEIDBHRC | data/ebola/cases.csv

Time: 2014-10-05   Total cases: 8344   Total cases: 8238

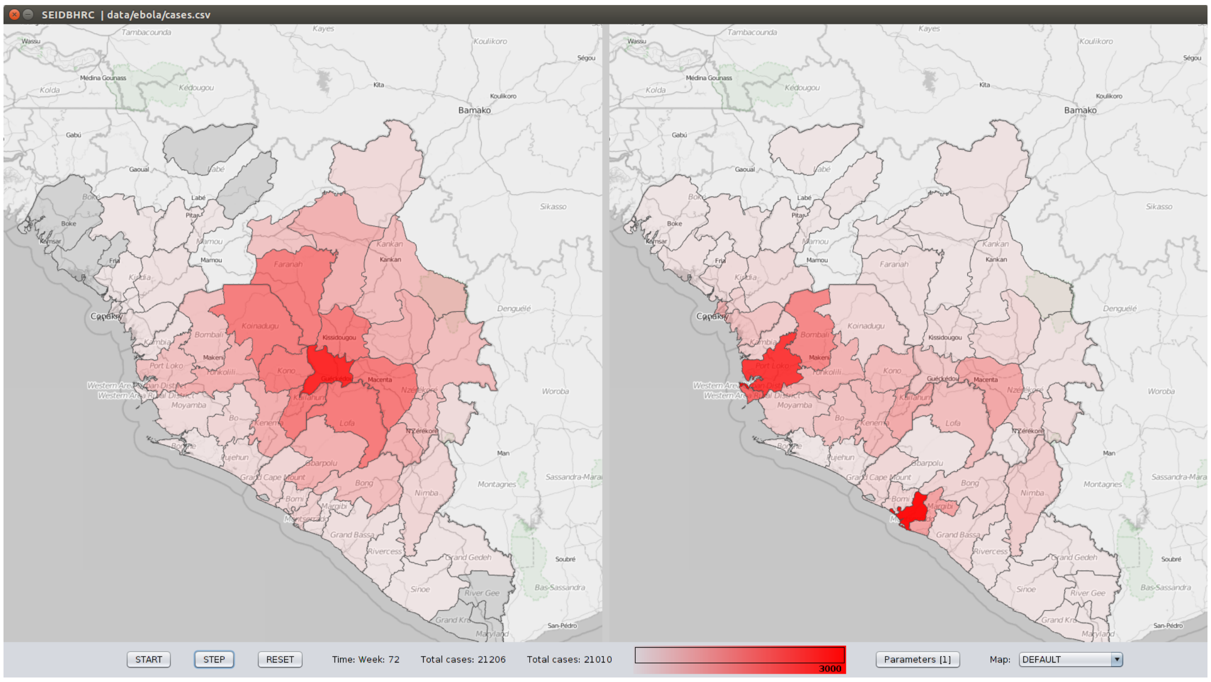
Parameters [1]   Map: DEFAULT

SEIDBHRC | data/ebola/cases.csv

Time: Week: 72   Total cases: 24091   Total cases: 21010

Parameters [1]   Map: DEFAULT

Figure 4.8: Spatio-temporal spread of Ebola in Guinea, Sierra Leone and Liberia after 10, 40 and 70 weeks, visualised by STAGE side-by-side, using the Gravity mobility model. The model's predictions are on the left, the observations are on the right.



Time: Week: 72 Total cases: 21206 Total cases: 21010

Map: DEFAULT

Figure 4.9: Spatio-temporal spread of Ebola in Guinea, Sierra Leone and Liberia after 70 weeks, visualised by STAGE side-by-side, using the Common Borders mobility model. The model's predictions are on the left, the observations are on the right.

coefficient of determination for each case, while Figure 4.9 shows the final simulation step in the Common Borders case. We observe that using Common Borders the difference between the total number of cases is tiny - the model predicts 21,206 whereas in reality there were 21,010 observations. On the spatial axis however the two sets of data show massive differences. The coefficient of determination is negative, reflecting this discrepancy.

Mobility model	SSE	R^2
Gravity	1.16×10^6	0.212
Common Borders	1.54×10^6	-0.040

Table 4.4: Comparison between the Gravity and the Common Borders model.

The results verify that the spreading of a complex disease like Ebola is not homogeneous between regions. Proximity, population, infrastructure and other geographical factors are important indeed and they should be taken into account by using explicit models of human mobility. By performing spatio-temporal analysis and modelling we account for these factors, and we are more likely to build realistic models.

Regarding the main objective of the project, we have demonstrated in this chapter how STAGE can be used to visualise and simulate biological epidemics. The STEM's Ebola model and the Gravity mobility model were easily implemented due to the framework's modular architecture. Furthermore, the side-by-side feature was useful as it allowed us to perform and evaluate experiments quickly.

Chapter 5

Case study: Information diffusion in Twitter

5.1 Geolocation in Twitter

The use of Twitter data for spatio-temporal analysis of online phenomena is common nowadays. One of the reasons for this is the availability of location data in tweets or user profiles. This data comes in three forms:

Exact location a set of (latitude, longitude) coordinates provided by GPS or cellular triangulation at the time the tweet is sent. The consent of the user is required.

Place a description of the current location given by the user in terms of address, city and/or country.

Profile location a free-form, self-declared location on the user's profile page.

Latest statistics [17] say that just over 3% of all tweets include native geolocation information (exact location or place). For this reason researchers often include the profile location information in their studies too.

5.2 Data collection

We collected and analysed the geotagged tweets that were posted in London when the following events were taking place: the London marathon (26/04/2015), the birth of the royal baby Princess Charlotte (02/05/2015) and the UK General Elections (07/05/2015). We chose London because of its high volume of posted tweets (it is ranked third out of 20 cities world-wide [13]). Table 5.1 describes the data sets.

The data sets were first constructed by querying Twitter's Streaming API using the Java library `hbc`. This returns 1% of all the tweets posted in real-time. We also complemented

Event name	Keywords	Hashtags	Geotagged tweets
London marathon	“london marathon”	#LondonMarathon	2906
Royal baby	“royal baby”	#RoyalBaby	497
General elections	elections	#UKElections2015, #GE2015, #IVoted	4453

Table 5.1: Twitter data sets used in this case study.

the data sets with tweets collected by querying Twitter’s Search API using the Java library `Twitter4J`. This API returns a sample of tweets sent in the last 6-9 days. Our requests aimed to retrieve tweets coming from a circle of radius 30km around the centre of London.

We chose to use MongoDB, a document-oriented database system, to store the tweets. This was convenient to use because of the direct manipulative operations it can perform over JSON objects, Twitter’s data format.

We converted the tweets from the JSON format to GeoJSON, an extension suitable for encoding geographical information. For tweets with an exact location we stored the coordinates present in the original format. For tweets with a Place attribute we stored the coordinates of the centre of a bounding box around the specified area. For the rest we used the GeoNames[1] API to map the profile location description to a pair of coordinates through a process called *geocoding*.

To perform spatio-temporal analysis on the spread of tweets the collected, we had to have an indication of where each user’s followers are located. This is a difficult problem in general as there is no easy way to obtain their exact location (they have to post a geotagged tweet themselves in which case they will be in the data set anyway). We used the following method to get the coordinates:

1. Query Twitter’s Search API to obtain profile descriptions of each user’s followers.
2. Extract the profile location of each follower and ask the GeoNames API to return the corresponding coordinates, provided that the location matches a specific area in London.
3. Append the coordinates to the tweet objects in the data set.

The bottleneck in the above methodology were the limits imposed by the APIs of Twitter and GeoNames. Twitter allows an application to issue a maximum of 30 requests per 15 minutes, and each request can return a maximum of 200 user profiles. GeoNames has an hourly limit of 2,000 requests, as well as a daily limit of 30,000. Given that some users in our data sets contained more than 120,000 followers, it was important to cut down the waiting time. Our solution was to use caching and concurrent application instances.

[1]http://www.geonames.org

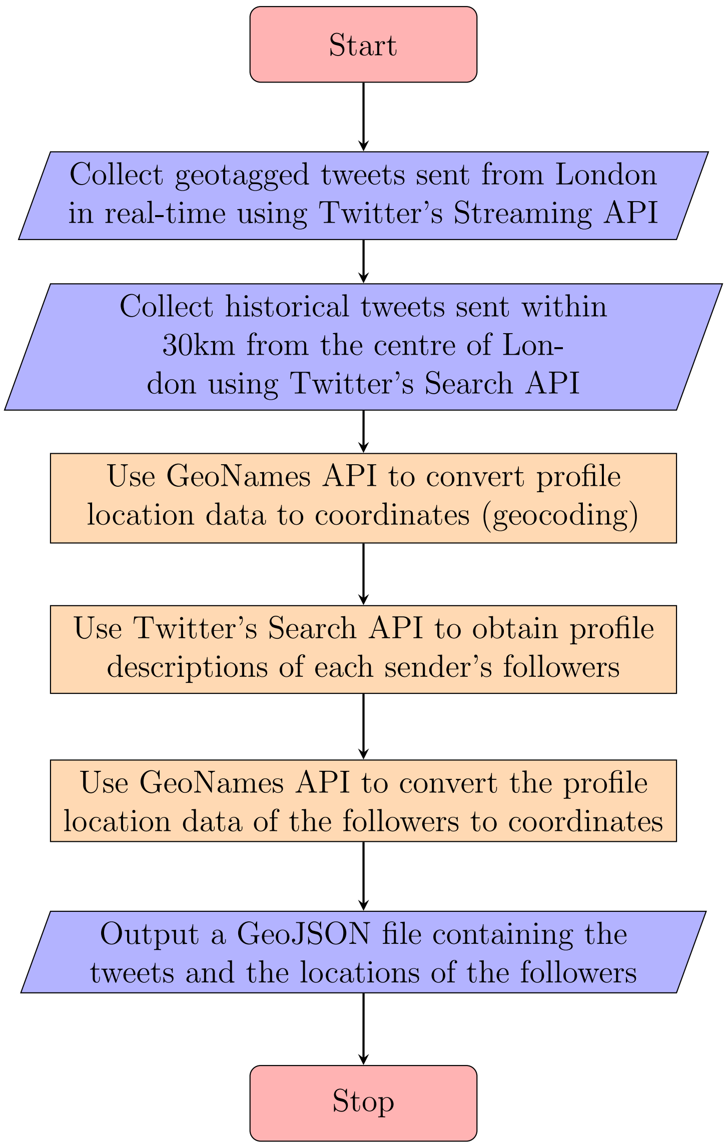

Start
Collect geotagged tweets sent from London in real-time using Twitter's Streaming API
Collect historical tweets sent within 30km from the centre of London using Twitter's Search API
Use GeoNames API to convert profile location data to coordinates (geocoding)
Use Twitter's Search API to obtain profile descriptions of each sender's followers
Use GeoNames API to convert the profile location data of the followers to coordinates
Output a GeoJSON file containing the tweets and the locations of the followers
Stop

Figure 5.1: Flowchart describing the process of constructing a data set.

Figure 5.1 summarises the process of constructing the data sets. The final data set for each event contains the tweets sent from London that had some sort of valid geographical reference, as well as the locations of each sender's followers which the geocoding process was able to map to valid areas in London.

5.3 Visualisation

We used the library Unfolding Maps to create a Java program that visualizes tweets in the GeoJSON format. It draws each tweet object as a simple point marker on a map. Clicking on the marker shows the tweet's message, as shown in Figure 5.2. It is also possible to switch between several map providers such as OpenStreet, Google and Microsoft. This program was integrated into STAGE.

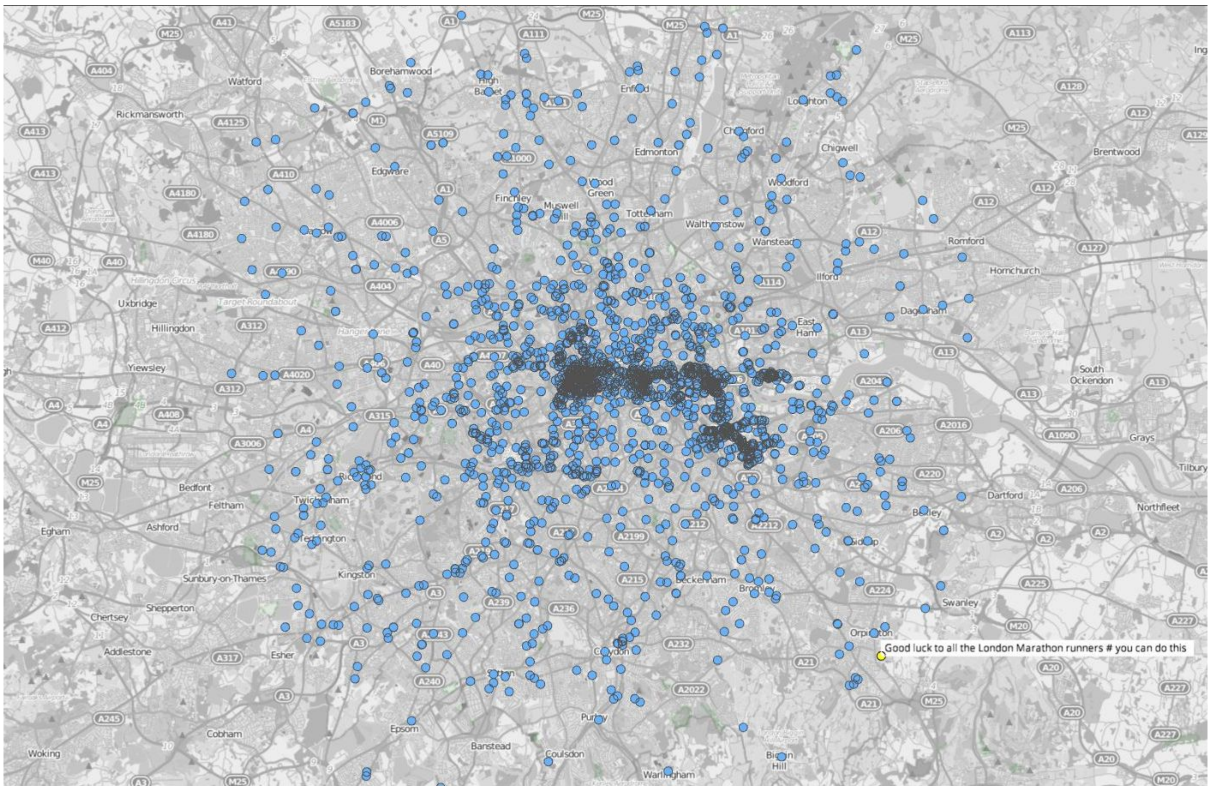

Good luck to all the London Marathon runners # you can do this

Figure 5.2: London Marathon tweets on 26/04/2015 (OpenStreetMap view in STAGE)

Figure 5.2 shows the spatial spread of tweets regarding the London marathon. We observe that the majority of them came from the area around river Thames, where the race took place. The place with the biggest concentration is Saint James's Park, the finish point.

Figure 5.3 shows the spatial spread of tweets regarding the birth of the royal baby. Most of them appear to have come from the area around Buckingham Palace. As in the case of the marathon, *eyewitnesses* generated a big portion of Twitter activity.

Figure 5.4 shows the spatial spread of tweets regarding the general elections. Since this was a country-wide event the tweets are spread almost evenly throughout the city. There is still some significant concentration around Downing Street though.

5.4 Analysis

5.4.1 Tweets over time

Figures 5.5 and 5.6 show the distribution of tweets posted over time, excluding retweets. We observe that in the marathon case a simple SIR-like model is sufficient to describe the data, whereas in the case of the elections we need a multi-modal model. The synthedemic framework [28] is a good candidate for the latter.

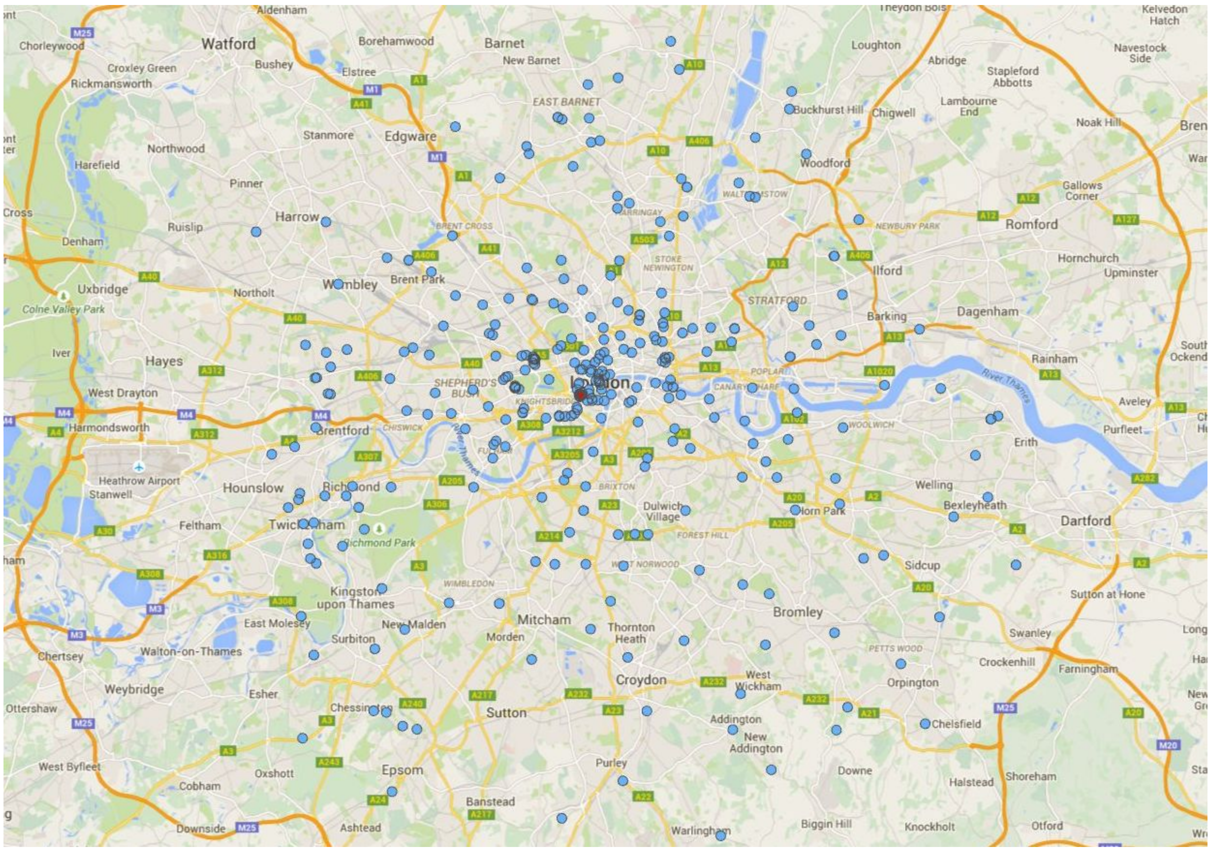


Figure 5.3: Royal Baby tweets on 2/05/2015 (Google Maps view in STAGE)

5.4.2 Distance from the event's location

Figures 5.7 and 5.8 show the number of tweets sent against the distance from a specific point, excluding retweets. In the royal baby case we observe that the number of tweets decays at an almost exponential rate with increasing distance from Buckingham Palace. Regarding the elections we see a decline of the number of tweets with increasing distance from Downing Street but this is a lot smoother. In fact, this may be simply due to the higher concentration of people in the centre of London. It is therefore necessary for any Twitter model to account for both location-based events and country-wide phenomena.

5.4.3 Time to retweet

Figures 5.9 and 5.10 show the time taken for a tweet sender's follower to retweet the message they just received. In both the marathon and elections cases we observe that the vast majority of retweets are made during the first three hours. After this time the number of retweets decays exponentially. The model for retweets should therefore favour small response times.

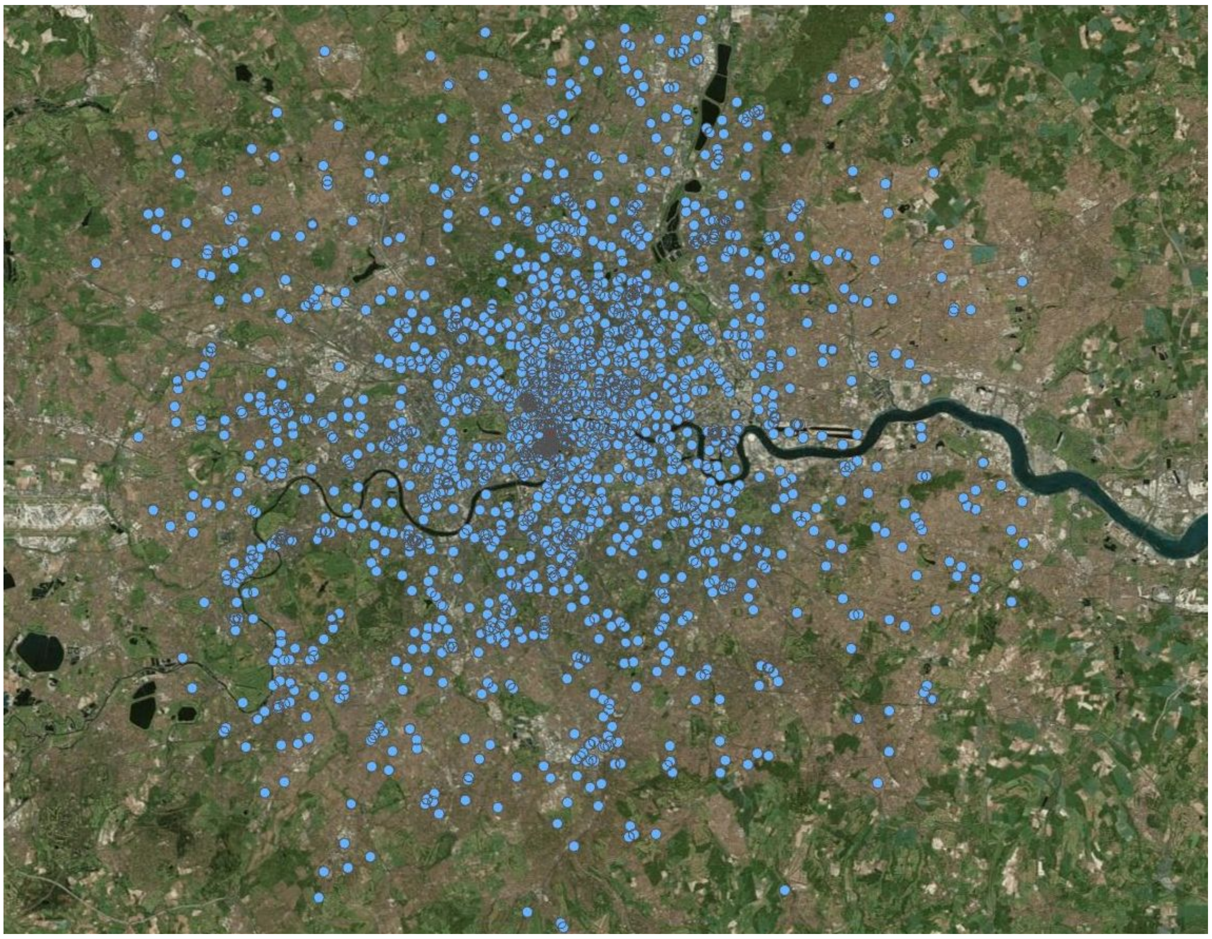


Figure 5.4: General Elections tweets on 7/05/2015 (Satellite view in STAGE)

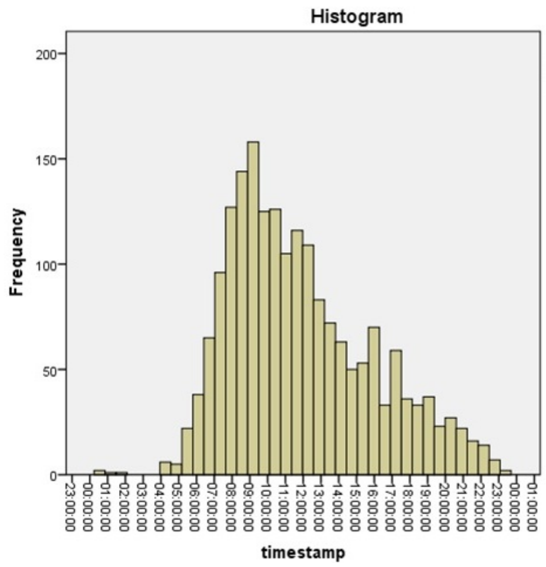


Figure 5.5: Number of tweets about the marathon during 26/04/2015

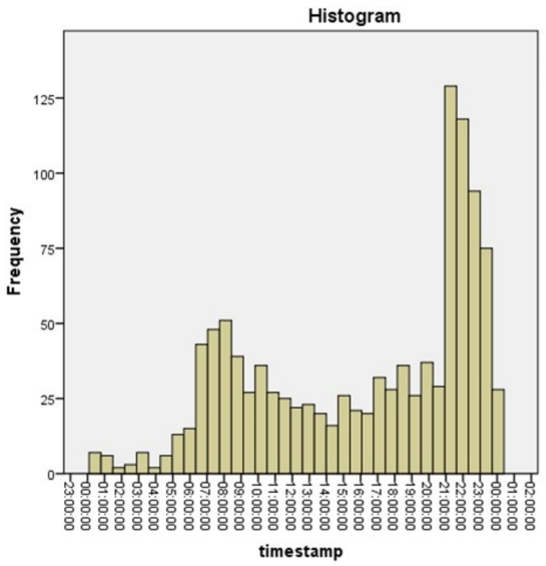


Figure 5.6: Number of tweets about the elections during 7/05/2015

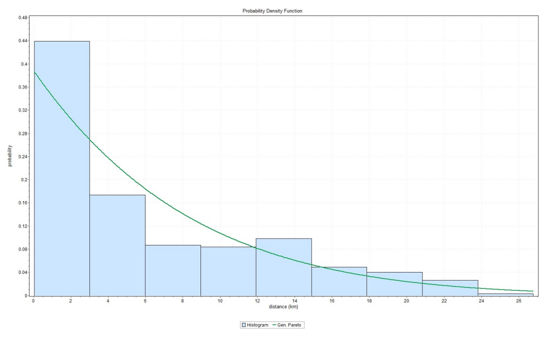


Figure 5.7: Distance between senders and Buckingham Palace (royal baby). Approximated by a Pareto distribution with $k = -0.10$, $\sigma = 7.52$, $\mu = -0.14$

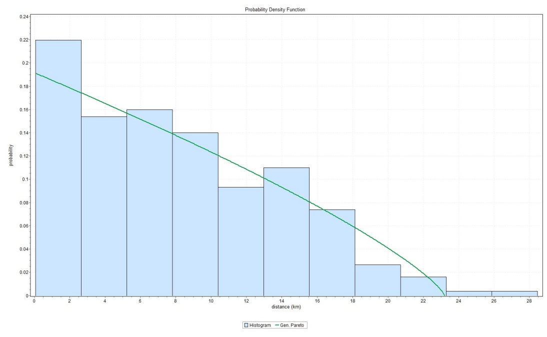


Figure 5.8: Distance between senders and Downing Street (elections). Approximated by a Pareto distribution with $k = -0.56$, $\sigma = 13.27$, $\mu = -0.43$

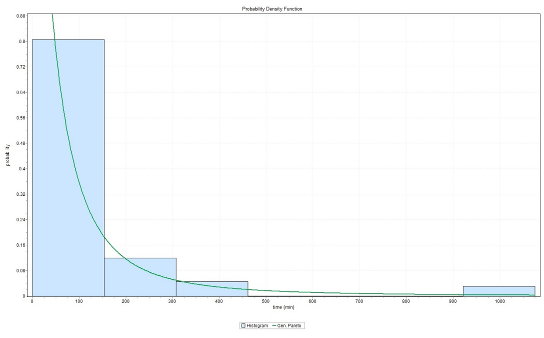


Figure 5.9: Time taken to retweet (marathon). Approximated by a Pareto distribution with $k = -0.50$, $\sigma = 55.25$, $\mu = -8.24$

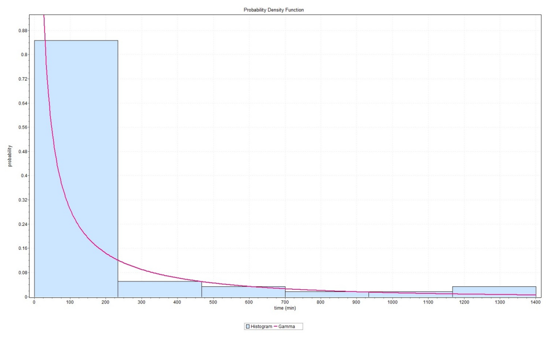


Figure 5.10: Time taken to retweet (elections). Approximated by a Gamma distribution with $\alpha = 0.0.19$, $\beta = 712.89$, $\gamma = 0$

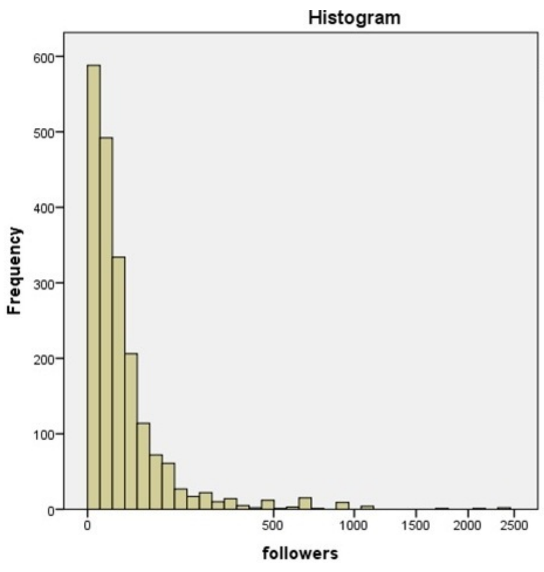


Figure 5.11: Number of followers in London per sender (marathon)

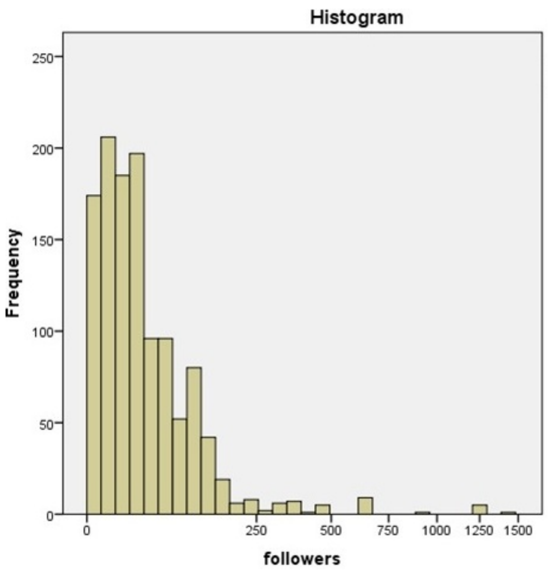


Figure 5.12: Number of followers in London per sender (elections)

5.4.4 Number of followers

Figures 5.11 and 5.12 show the distribution of the number of followers the senders have in London. We observe that our data sets are consistent with the general observation that most users in OSNs have a few contacts while a small minority tends to have an abnormally large number of them. A scale-free graph like the Barabasi-Albert one can thus be used to represent the contact network.

5.4.5 Distance between senders and their followers

Figures 5.13 and 5.14 show a key result of our analysis. We observe that **the majority of one's followers are located close to them**. This is in line with the conclusions of Takhteyeva *et al.* [36]. Moreover, the tendency to follow people close to you is now shown in the much smaller scale of a single city. This is a significant finding as it verifies our assumption that a geographical model which takes physical distance into account (such as the Gravity or Radiation mobility models) is suitable to use in the construction of the location-based network between users on Twitter.

5.5 Proposed spatio-temporal model

Based on the conclusions of the data analysis, we propose a stochastic spatio-temporal model made up of the following components:

1. Mobility model: any mobility model that takes into account the proximity between regions can be used. This is in line with our observations in Section 5.4.5. The

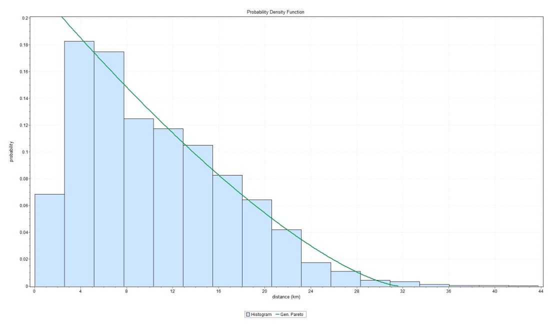


Figure 5.13: Distance between senders and their followers (marathon). Approximated by a Pareto distribution with $k = -0.42$, $\sigma = 12.4$, $\mu = -1.7$

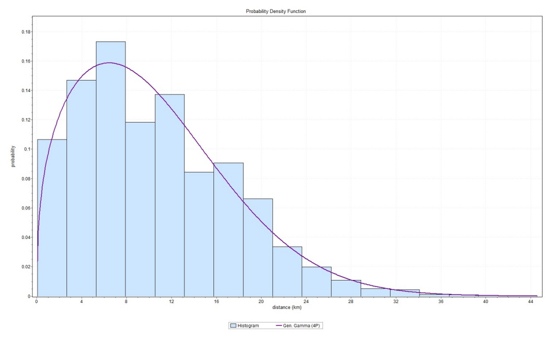


Figure 5.14: Distance between senders and their followers (elections). Approximated by a Gamma distribution with $\alpha = 0.67$, $\beta = 15.22$, $\gamma = 0.03$

Radiation model is the first option as it is parameter-free.

2. Graph model: a Barabasi-Albert like network where the nodes represent users and edges represent *followed by* relationships. This is in line with Section 5.4.4.

3. Node allocation scheme: an algorithm to assign nodes to physical regions and connect them based on the chosen mobility model.

4. Disease models: an SIR process to describe the level of enthusiasm that causes people to tweet about it (without receiving any related tweets first) as well as a SEIR process to describe the retweeting activity.

5.5.1 Network construction

The following procedure is used to construct the logical network of individuals susceptible to a socio-technological epidemic:

1. A set of n nodes are created and allocated to regions based on the population distribution across the area of interest.

2. A *rank matrix R* is constructed by invoking the corresponding method on the mobility model in STAGE. This is a square matrix of size equal to the number of regions, k. For each region i, R_{ir} gives the index of the region upon which it has a (k - r) degree of influence. For example, the Gravity model gives a high rank index between regions which are close to each other or have a large population. Figures 5.15 - 5.17 demonstrate the impact of the choice of mobility model in the structure of a network.

3. For each node, we determine the number of outgoing edges to add by sampling from a zipf distribution, which is a discrete power law distribution.

4. For each edge to be added, we select a rank index r by sampling from a geometric distribution. The destination region is then given by R_{ir}. We then select a node from that region at random to create the link.

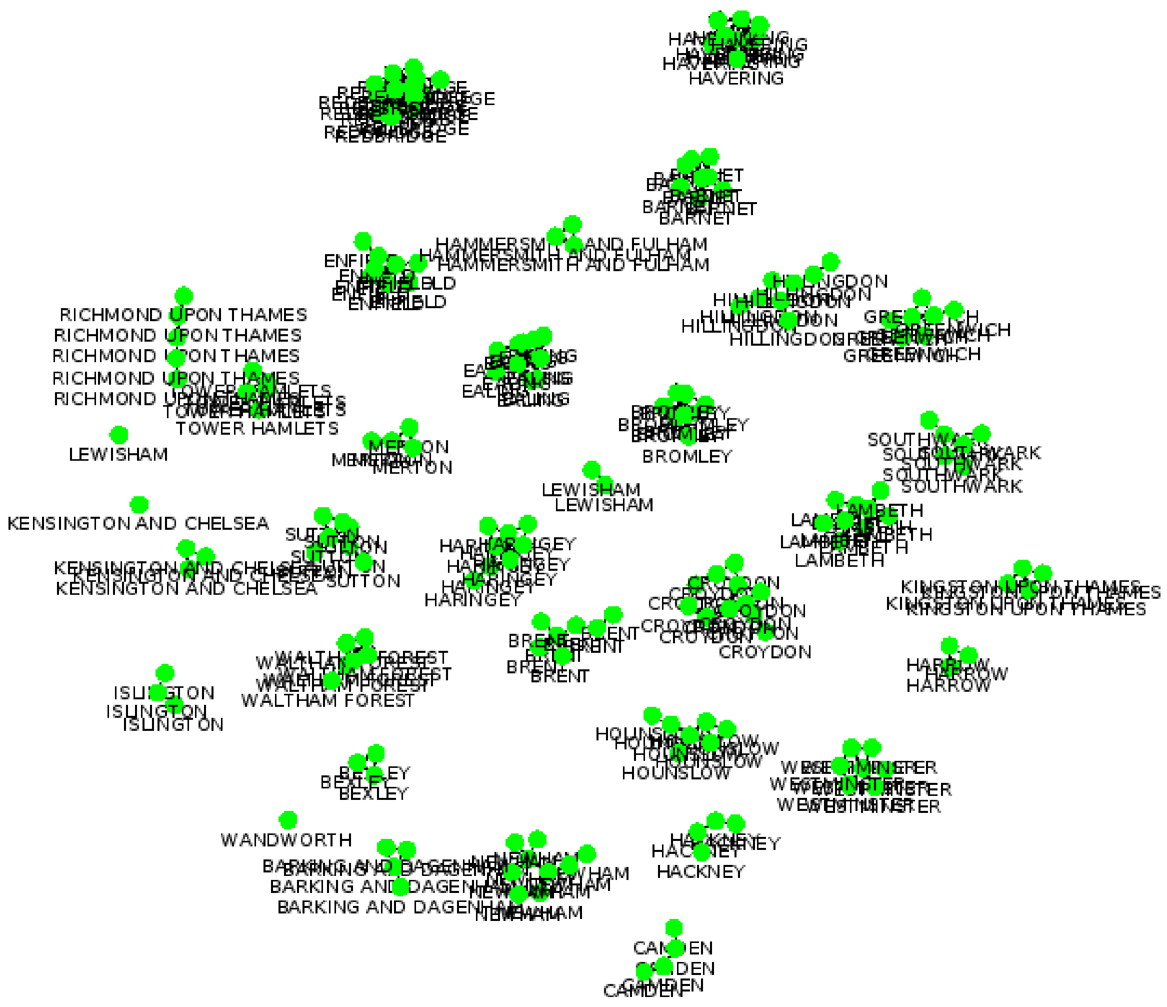


Figure 5.15: A network of 200 nodes constructed with the Zero Mobility model using STAGE. Each node is allocated to a London borough at random. The use of Zero Mobility ensures that no links are created between nodes in different regions.

The resultant network is directed, small-world and scale-free.

Table 5.2 shows the parameters of the network:

Parameter	Symbol	Description
Number of nodes	n	The number of users (senders and their followers)
Max edges per node	e	The maximum number of followers a user has
Degree exponent	d	Zipf parameter for the distribution of node degree
Rank exponent	r	Geometric parameter for the effect of the mobility model

Table 5.2: Parameters of the network.

Depending on the choice of the mobility model there may be additional parameters.

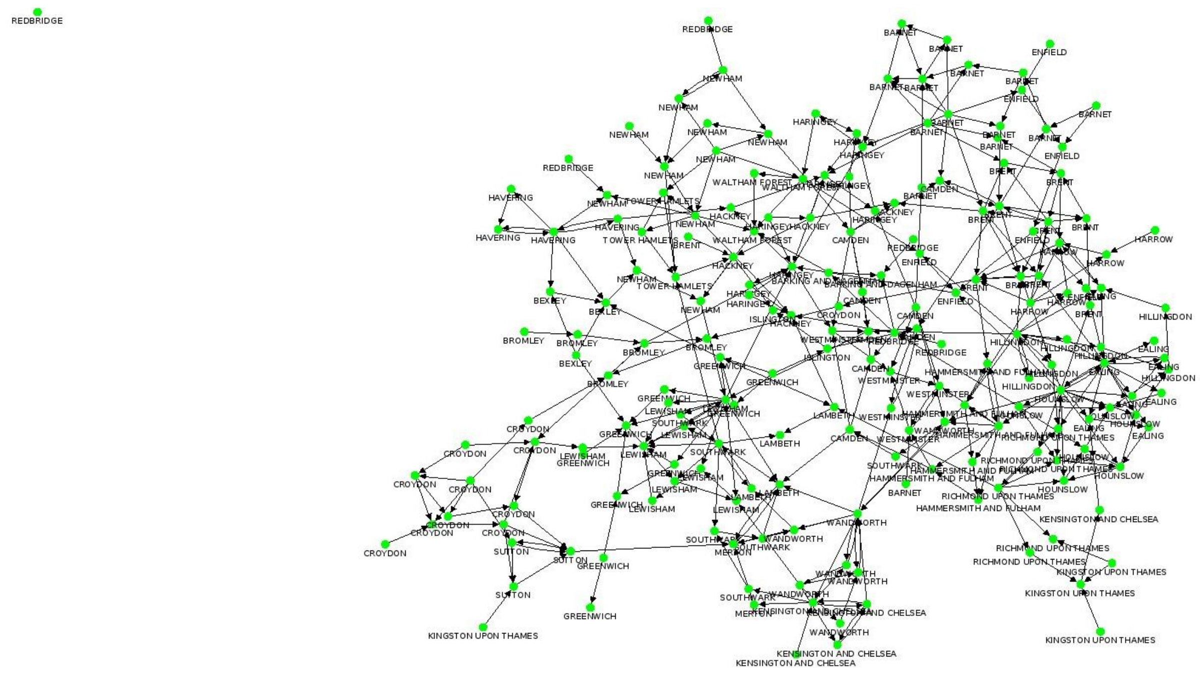


Figure 5.16: A network of 200 nodes constructed with the Gravity model using STAGE. This favours links towards regions which are close-by or have a large population.

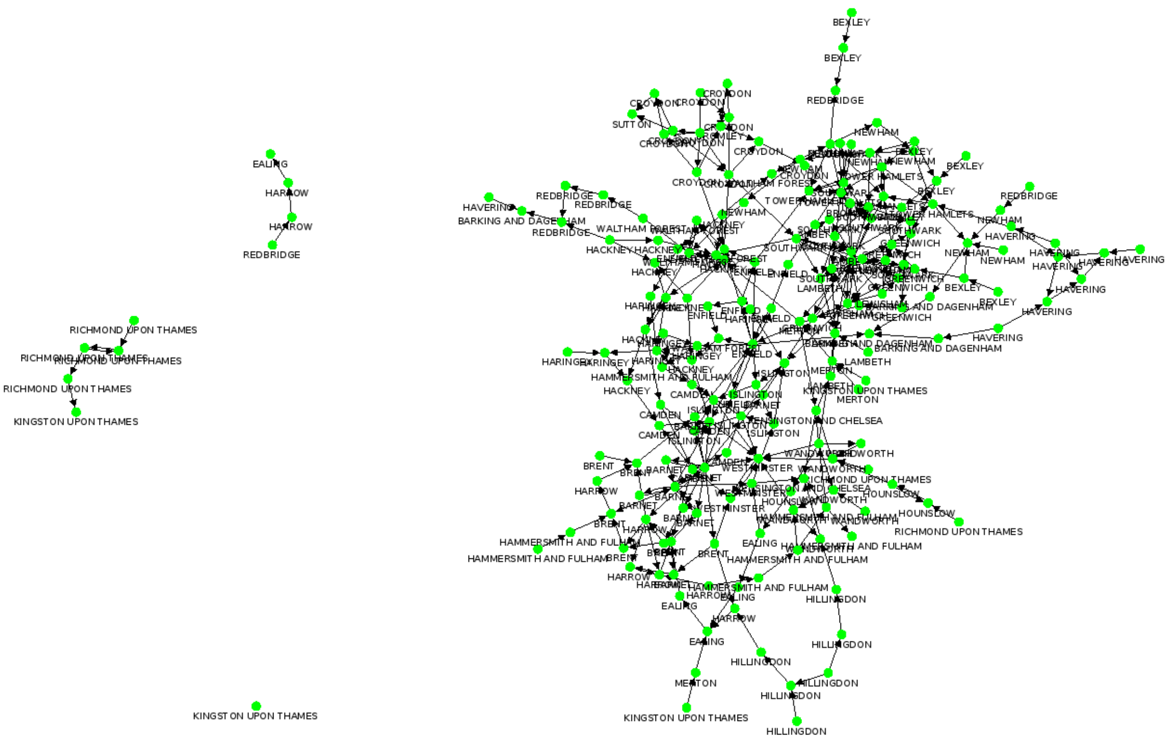


Figure 5.17: A network of 200 nodes constructed with the Radiation model using STAGE. The links are created based on the model's dynamics (proximity and population).

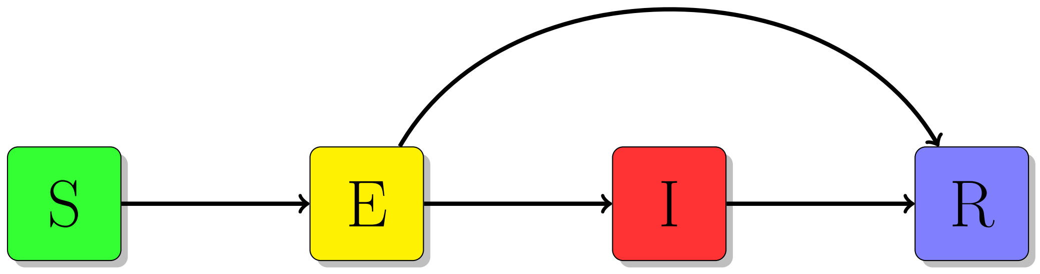


Figure 5.18: Transitions between compartments in the Retweet SEIR model.

5.5.2 Disease models

SIR

The SIR model is used to describe the rate at which users post tweets (i.e. get infected) over time without having any contact with other users. The actual nodes that get infected are chosen as follows:

1. In the beginning, the nodes are sorted in ascending order of distance from the location of the event.
2. At each infection point, a susceptible node with index sampled from a geometric distribution is selected. The parameter of this distribution lets us adjust the level of "bias" we have in infecting nodes which are allocated to regions close to the event.

Table 5.3 shows the parameters of the model:

Parameter	Symbol	Description
Infection rate	β	The rate at which users post tweets
Recovery rate	γ	The rate at which users lose interest in the event
Proximity exponent	c	Geometric parameter for the importance of proximity

Table 5.3: Parameters of the SIR model.

SEIR

The SEIR model illustrated in Figure 5.18 is used to describe the rate at which users respond to an incoming tweet (via retweeting or replying) over time. Once a susceptible user in the network receives a tweet, they become exposed. They then have a probability p to infect all their followers (and recover) after some time τ and a probability (1 - p) to recover after (1 / γ) without infecting anyone.

Table 5.4 shows the parameters of the model:

The time to infect, τ, is exponentially distributed with rate (1 / α).

Parameter	Symbol	Description
Incubation rate	α	The rate at which users retweet
Probability to infect	p	The probability to respond to a tweet

Table 5.4: Parameters of the SEIR model.

5.6 Simulation

We implemented our proposed spatio-temporal model for tweets and retweets into STAGE. We also added support for visualisations over London by specifying the borders and properties of its 33 boroughs.

We estimated the values of the model's parameters from the data. Specifically, we obtained them by fitting probability distributions to the histograms in section 5.4. Note that in places where the model uses a discrete distribution such as Geometric and Zipf their parameters were estimated from the corresponding Exponential or Pareto continuous distributions fitted to the histograms. Table 5.5 shows the parameter values used in the simulation of London Marathon 2015.

Parameter	Symbol	Value	Estimated from
Number of nodes	n	98805	Number of senders + their followers
Max edges per node	e	2416	Histogram of number of followers
Degree exponent	d	1.44	Geometric(0.02)
Rank exponent	r	0.19	Pareto(-0.42,12.4,-1.7)
Infection rate	β	81	SIR fit to tweets over time
Recovery rate	γ	9	SIR fit to tweets over time
Proximity exponent	c	0.23	Pareto(-0.41251,10.107,-0.00286)
Incubation rate	α	0.01	Exponential(0.01)
Probability to infect	p	0.0007	Number of retweets / number of followers

Table 5.5: Parameter values used in the simulation of London Marathon 2015.

We used our model to construct a network consisting of tweet senders and receivers (i. e. their followers) in London, based on our data set. Table 5.6 shows the remaining parts of our configuration.

Option	Value
Space	Hybrid
Index case	Westminster
Time unit	Hour

Table 5.6: Remaining options used in the simulation.

Figure 5.19 shows the spatio-temporal spread of tweets regarding the marathon on the day of the event in 8-hour intervals. It is the average of 10 stochastic runs. The model's predictions are displayed side-by-side with the observations. We observe that the model predicts much fewer cases than in reality, but is manages to capture at least the correlation between the number of tweets sent and the distance from the finish line.

The other two data sets generated similar results, showing a greater activity at boroughs near the location of the event. Figures 5.20 and 5.21 shows the spread of Royal Baby and general elections tweets at the end of the day respectively.

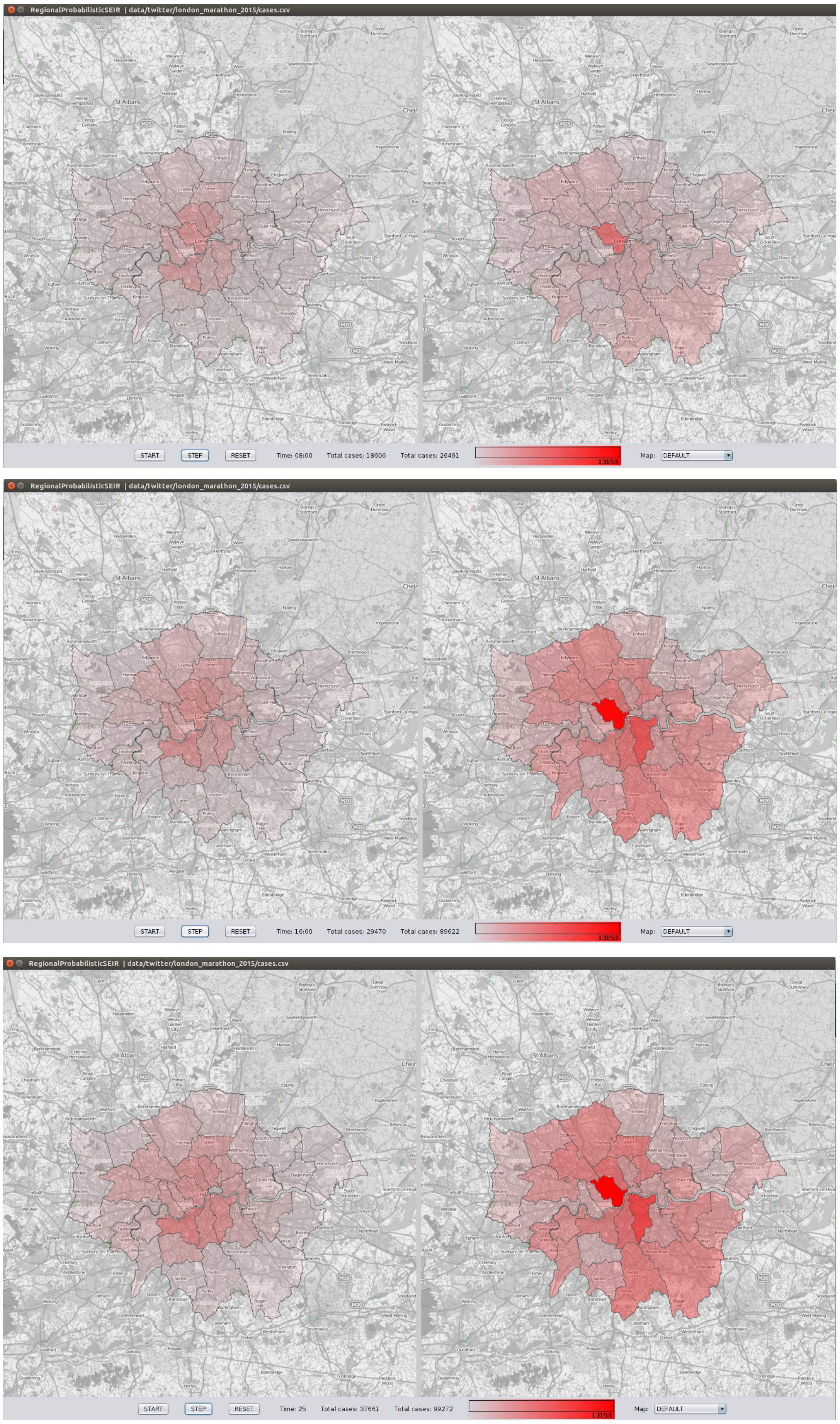
5.7 Discussion

To recap, we tested a spatio-temporal model for the spread of tweets and retweets which accounts for geographical factors in three parts:

- The nodes in the network of London users are allocated to boroughs based on the city's population distribution.
- The connections between the nodes are constructed using a mobility model which favours links between people in near-by areas and links towards people in more populous areas.
- There is a bias towards infecting people who are close to the source of the event.

Regarding the primary objective of the project, we demonstrated how STAGE can be used to test hypotheses regarding the role of geography in socio-technological epidemics. It allowed us to form a spatio-temporal model featuring the above geographical properties, and then simulate it in both physical and logical space. Furthermore, the availability of real-world maps helped us examine the areas which experienced high Twitter activity.

Regarding the specific experiment described in this chapter, the proposed model did not achieve a good fit to the three data sets, but its form was not disproved either. A significant portion of the error may be due to the small sample of geo-tagged tweets. The SEIR model for retweets in particular is greatly influenced by this since there is a tiny amount of retweets in the data sets. For example the marathon one has only 67. Retweets are essential in studies of information diffusion as they suggest that a tweet is important enough to be shared with every follower a user has. The reason we did not have many is due to a limitation of the Twitter API: retweets of geotagged-tweets do not carry any geolocation metadata. So the only piece of information which is obtainable in this case is the user-defined location in their profile. The chances of this being an accurate description of an administrative area in London are too low. For example, over 80% of the users in our data sets specified “London” in their profiles, which was too general to be used in our analysis. Moreover, geocoding is another source of error which reduces the quality of our data sets. For example, looking up “Surrey” on GeoNames API returns a match for Surrey Quays in South East London instead of the city. Some of these issues were addressed by making the search parameters stricter, but still there may be additional inaccuracies which remained undetected.



RegionalProbabilisticSEIR | data/twitter/london_marathon_2015/cases.csv

Time: 08:00 Total cases: 18606 Total cases: 26491 Map: DEFAULT

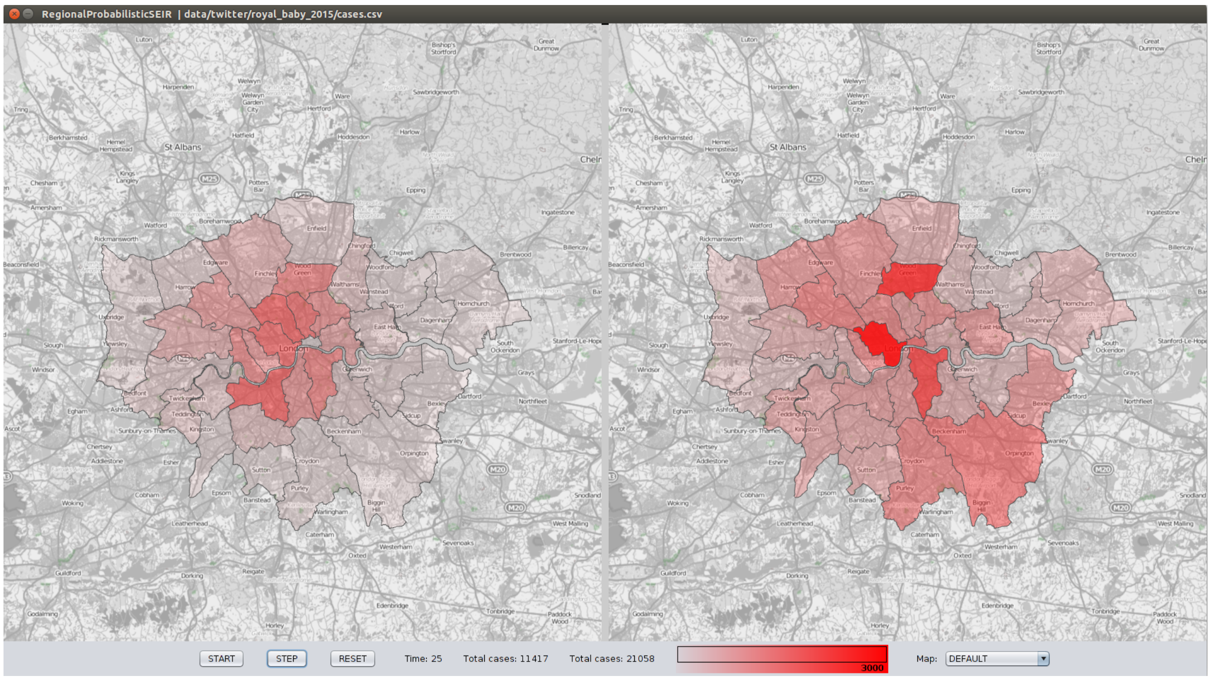
RegionalProbabilisticSEIR | data/twitter/london_marathon_2015/cases.csv

Time: 16:00 Total cases: 29470 Total cases: 89622 Map: DEFAULT

RegionalProbabilisticSEIR | data/twitter/london_marathon_2015/cases.csv

Time: 25 Total cases: 37661 Total cases: 99272 Map: DEFAULT

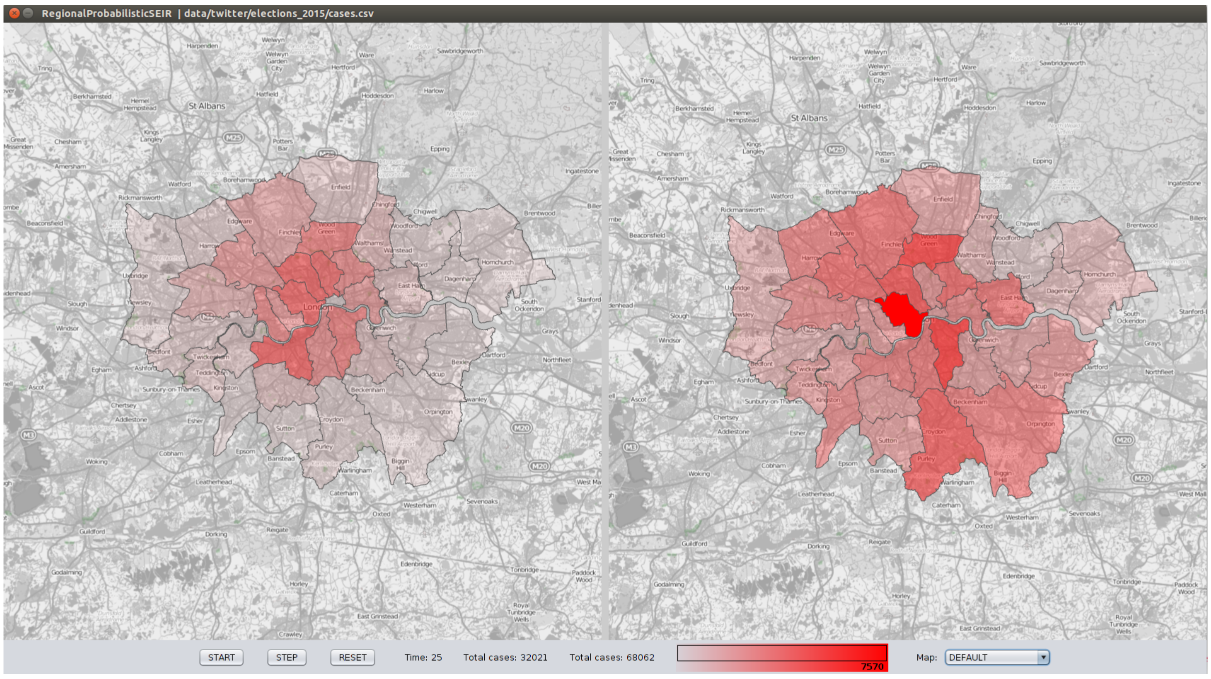
Figure 5.19: Spatio-temporal spread of London Marathon tweets by 08:00, 16:00 and 24:00 on 26/04/2015, visualised side-by-side using STAGE. The model's predictions are on the left, the observations are on the right.


RegionalProbabilisticSEIR | data/twitter/royal_baby_2015/cases.csv

Time: 25 Total cases: 11417 Total cases: 21058

Map: DEFAULT

Figure 5.20: Spatio-temporal spread of Royal Baby tweets by midnight of 02/05/2015, visualised side-by-side using STAGE. The model's predictions are on the left, the observations are on the right.


RegionalProbabilisticSEIR | data/twitter/elections_2015/cases.csv

Time: 25 Total cases: 32021 Total cases: 68062

Map: DEFAULT

Figure 5.21: Spatio-temporal spread of general elections tweets by midnight of 07/05/2015, visualised side-by-side using STAGE. The model's predictions are on the left, the observations are on the right.

Despite the fact that the data sets are too small to assess the model precisely, the model succeeded in reflecting the observation that the majority of tweets comes from people who are close to the location of the event. Its results are closer to the ground truth than a model which ignores geography by examining just the temporal component of the epidemic, as in this case we expect to see an even distribution of cases along the boroughs. As the actual spreading is not even or random, but it is influenced by geographical factors like distance to the event or distance between friends, it is worth performing spatio-temporal analysis for this kind of epidemics indeed.

Chapter 6

Evaluation

This chapter evaluates our aims and objectives concerning STAGE. We start by providing a qualitative and quantitative evaluation of our framework in terms of functionality, performance, design, extensibility and usability. We then compare it with two existing epidemiological frameworks and discuss its strengths and weaknesses.

6.1 Functionality and performance

6.1.1 Simulation and modelling

Our main aim was to build a framework which would be suitable for examining both biological and socio-technological epidemics. To accomplish this goal it was necessary to support both physical and logical space, population-based and agent-based disease models, as well as deterministic and stochastic disease models. We could then simulate a biological epidemic using a deterministic, population-based disease model along with a mobility model. For socio-technological epidemics we could visualise a physical area but simulate transmission over logical space using a stochastic, agent-based model. In any case the program would have to be able to make forecasts by area based on a single trace. The following sections reflect on these objectives.

Forecasts by area and time based on a single trace

All classes which implement the interface `DataSource` provide the method getNewCases() whose return type is an array of integers. In `RegionBasedSimulator` these integers represent counts of new cases in each of the regions in the area under study after one step of the simulation. Successive invocations of the method are used to add a temporal axis on top of the spatial data, constructing a 2D matrix of fine-grained predictions.

It it noteworthy to mention that the time unit of the simulation is determined implicitly by our choice of parameter values. For example, if the transmission rate β for a particular disease is 0.2 infections per day, setting this parameter to 7 * 0.2 = 1.4 tells the simulator to output results on a weekly basis. This approach has the benefit that it is simple as the correct parameter values can be determined once and used continuously without requiring the user to specify the time unit on each run. A limitation is that if someone wants to change this interval in the middle of the simulation they have to update all the parameters of the disease and mobility models. Since the interval at which cases are reported by health organisations rarely changes in practice, the simplification made in the program seems legitimate.

The spatio-temporal model used to make predictions is initially trained with whatever data we have at that point. This data represents a single trace as every time an outbreak occurs there is only one set of observations to be made. As the program's model fitting procedure uses a single `FileReader` instance (that reads data from one file only) the predictions made by the model will be based on a single trace.

Furthermore, the program also supports the option of reducing the dimensions of the problem from two to one. Specifically, one can run simulations and get results in time only, in a single region of interest. This is achieved by choosing the `ZeroMobility` model which makes sure that the disease is contained within that single region (which may be defined on any administrative level such as town, district and country). This feature was not part of the original set of objectives but it was developed later to accommodate quick modelling and testing of candidate disease models on their own, without worrying about getting the mobility model right at that time. It is worth mentioning that because of the modular structure of the framework we just needed to add this single and extremely simple class to create an entirely new feature.

Physical and logical space

Simulation and visualisation can be done using physical areas, abstract graphs or both kinds of space simultaneously. The classes that provide this behaviour are `Physical`, `Logical` and `Hybrid` respectively. Unlike existing frameworks, such as Surveillance and STEM, the choice of space is left to the user. Additional options can be added in the future by constructing new implementations of the interface `SpaceModel`. For example, one could create a model consisting of the physical aspect and a collection of graphs, with each graph representing a separate transmission channel.

Support for population-based and agent-based compartmental disease models

The program is capable of simulating the interactions between people at the level of the entire population in a region as well as at the level of the individuals. For the first case, a compartmental disease model is used to monitor the population in each region and a suitable mobility model is used to spread the disease further. Both kinds of models describe the interactions between groups in the population using rates of certain events happening such as transmission, recovery and movement. These models assume that the population is homogeneous. By tracking how many individuals are in each compartment in each region the simulation is able to produce results which are based on the average behaviour of the entire population. `RegionBasedSimulator` solves the ODEs which describe the rates at which people transition between the compartments to enable this kind of simulation.

Agent-based compartmental models also divide the population in different groups but the behaviour of each individual is simulated explicitly. This mode of simulation is enabled by using the `GraphBasedSimulator`. This uses the discrete-event-simulation paradigm to simulate the events of transmission, recovery and so on by scheduling and processing each one independently. The disease spreads along the connections of an infected individual in a graph, which replaces the mobility model on this occasion. The simulator tracks the state of each individual providing fine-grained results.

The `SEIR` and `ProbabilisticSEIR` classes are examples of population-based and agent-based compartmental models respectively. Note that, although in the second type of models the interactions between people are stochastic, they do not account for heterogeneity among the characteristics of the population. All individuals have exactly the same age, gender and social class. The only thing which is unique for each one is the people with whom they have contact with, which is modelled using a graph. While this is an important differentiation, it is insufficient to capture the level of heterogeneity in the behavioural patterns of a population. This simplification was made to keep the computational complexity of the simulation to a reasonable level.

Support for deterministic and stochastic models

Both kinds of disease models have been implemented in the program. For example, the model defined in the class `SIR` is deterministic as it involves rates of events happening while the one defined in the class `ProbabilisticSEIR` is stochastic as it involves probabilities. Both types implement the same interface by defining the method getNewCases(). The `Simulator` depends on the interface only and so it is unaware of the actual kind of the model it uses to get the results.

Furthermore, the implementation of Gillespie's Stochastic Simulation Algorithm in the class `GillespieSolver` allows us to perform stochastic simulations using deterministic models. The user can specify the nature of the simulation in this case by selecting an appropriate ODE solver from the `solvers` package. The architecture of the program allows them to be used interchangeably.

6.1.2 Visualisation

The following sections reflect on the objectives set regarding visualisation.

Support for physical areas and logical space

STAGE is capable of visualising epidemics over the physical space by rendering real-world maps and adding on top of them a layer of shapes, the colour intensity of which represents the number of cumulative cases in the various regions. The Java library Unfolding Maps was used to provide the basis on which the required functionality was created.

To visualise logical space we chose to use the library GraphStream. This proved to be a valuable resource as it allowed us to create interactive graphs. The built-in algorithms for generating well-known graphs were also helpful in creating and visualising quickly different kinds. Figure 3.22 shows some examples of how graphs are visualised in STAGE.

Interactive maps

One of the main reasons for choosing Unfolding Maps was that it provides the means to offer good user experience. It allows STAGE not only to respond to user actions such as panning and zooming but also to switch between various map providers on the spot. As each provider highlights different aspects of geography, the program gives the users the means to explore possible causes of the spatial spread of an epidemic. For example, Figure 6.1 shows two views of Liberia's capital. The top one emphasizes the morphology of the area while the second one emphasizes the transport network. This area contains two airports - which may play an important role in the transmission of the disease over long distances - but they are easily recognisable in the bottom view only.

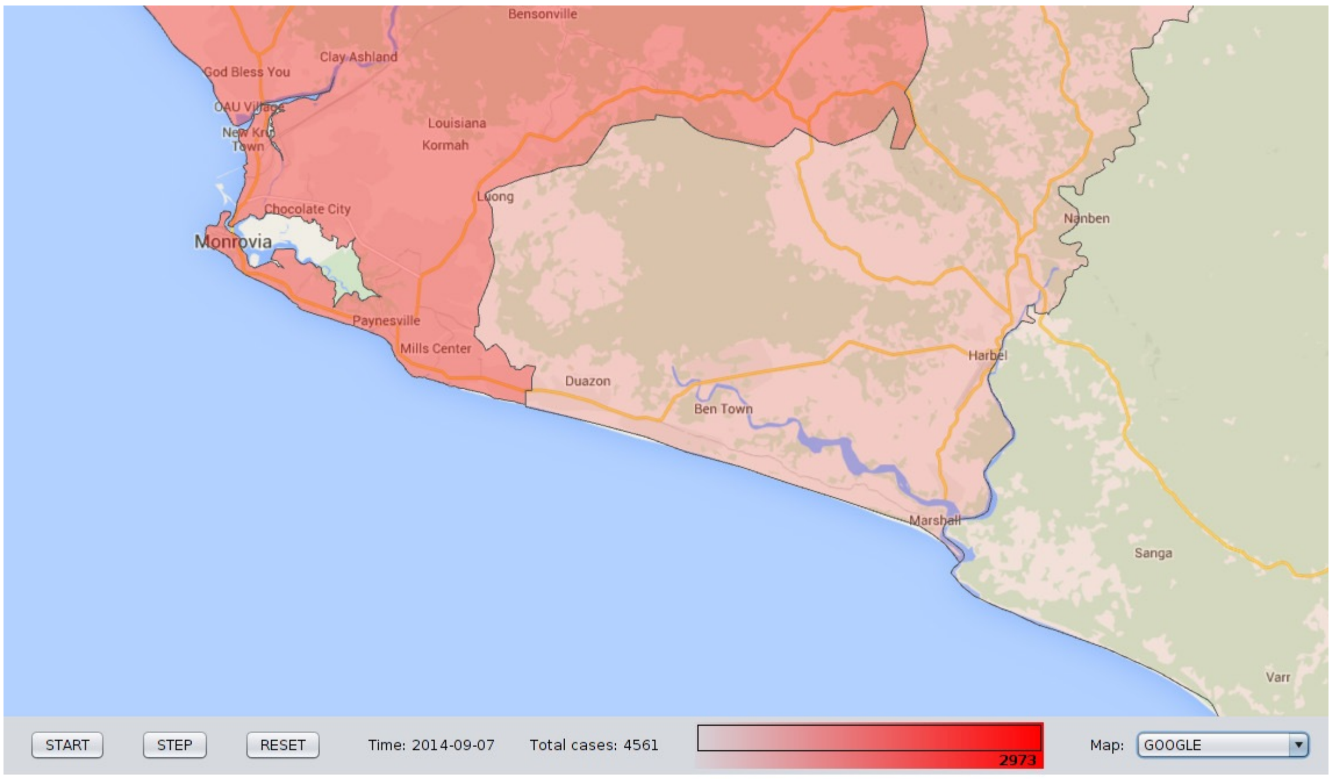
Side-by-side comparisons

Another benefit of Unfolding Maps was the support of visualising and updating multiple independent maps simultaneously. This allowed us to structure the visualisation engine in a way that any two data sources (files readers or simulators) could generate output side-by-side, as shown in Figure 3.21. This is perhaps the program's most powerful feature as it provides <u>immediate feedback</u> on the goodness of fit of a model to the data. It also allows us to compare quickly different scenarios such as two spatio-temporal models which share the same disease model but have different mobility models, or even just different parameter values for the same exact models.

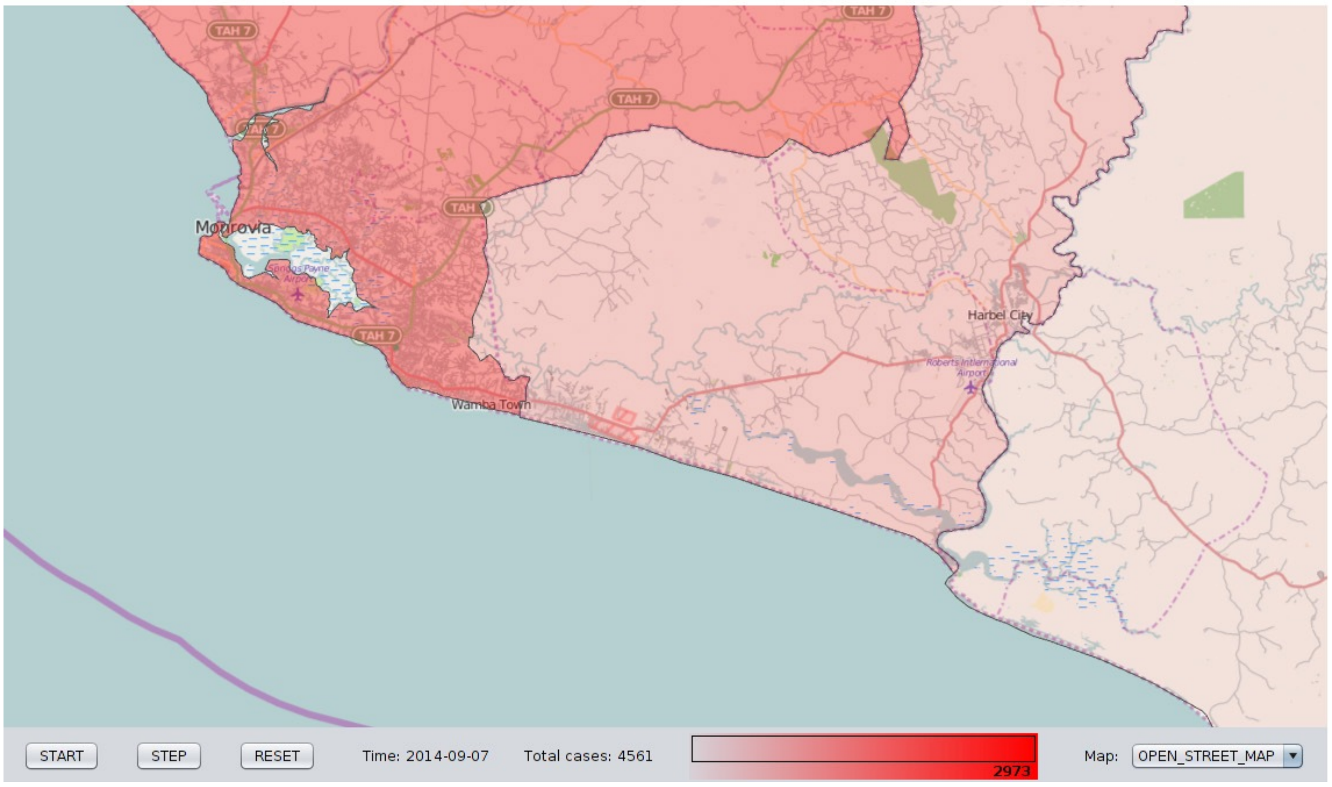
6.1.3 Synthetic data generation

STAGE is capable of generating synthetic data from deterministic models using Gillespie's Stochastic Simulation Algorithm (SSA). This data can then be fed back to the program to compute the goodness of fit between the data and the model. The aim here is to be able to determine whether the program can recover the parameters of the model from a single trace when the ground truth is known before attempting to use it against biological and socio-technological epidemic data.

To evaluate this component we generated synthetic data for the SIR and SEIR disease models with parameter vectors [$\beta = 1.08$, $\gamma = 0.8536$] and [$\alpha = 0.8$, $\beta = 1.20$, $\gamma = 0.8536$] respectively. The space is the 55 regions in West Africa which were used to simulate Ebola in Guinea, Sierra Leone and Liberia, with Gueckedou having the index case. The



Time: 2014-09-07 Total cases: 4561 Map: GOOGLE



Time: 2014-09-07 Total cases: 4561 Map: OPEN_STREET_MAP

Figure 6.1: Monrovia, Liberia. Top: Google Maps view highlighting the terrain's morphology. Bottom: OpenStreetMaps view highlighting transport links (such as the two airports).

generated data represents the mean trajectory of 100 stochastic runs. Table 6.1 shows the time taken to produce the synthetic data on an Intel i7 2.5GHz machine with 8GB of RAM.

Disease	Mobility	Parameters	Time (m:ss)
SIR	ZeroMobility	-	1:41
SIR	CommonBorders	$k = 0.25$	3:14
SIR	NeighbourOrder	$k = 2.0$	3:49
SIR	Gravity	$\alpha = 0.69$, $\beta = 1.64$, $\gamma = 2.8$, $k = 0.0001$	8:32
SIR	Radiation	$k = 0.1$	6:22
SEIR	ZeroMobility	-	3:09
SEIR	CommonBorders	$k = 0.25$	3:57
SEIR	NeighbourOrder	$k = 2.0$	4:29
SEIR	Gravity	$\alpha = 0.69$, $\beta = 1.64$, $\gamma = 2.8$, $k = 0.0001$	7:30
SEIR	Radiation	$k = 0.1$	5:47

Table 6.1: Performance of synthetic data generation.

We observe that the time taken to compute the data sets increases as the complexity of the chosen mobility model rises. Overall the time taken for 100 runs looks acceptable. However, if we wanted to execute a much greater number of simulations (to make the data set more accurate) the computation time would be a limiting factor. For example, scaling the time taken in the Gravity model case by 10 we estimate the time for 1000 runs to be around 85 minutes. This can be improved by replacing the *direct* method used in SSA with a faster one such as *next reaction* or *tau-leaping*. Such modifications, however, can compromise the precision of the algorithm.

6.1.4 Model fitting

To evaluate our model fitting component and simulation engine we used the synthetic data sets discussed above. We set up the program to perform the optimisation with 5 initial parameter vectors chosen randomly within an interval close to the ground truth, with 5 repetitions for each one. Table 6.2 presents the results when the program was asked to optimise all parameters. These consist of the final sum of squared errors (SSE) between the model and the data, the coefficient of determination R^2 which characterises the goodness of fit and the time taken for the process to complete.

The results demonstrate the difficulty of achieving a high goodness of fit using a spatio-temporal model with a large number of unknowns. The non-linear optimisation problem contains a large number of local minima and therefore the initial estimates for the parameters are very important to obtain a solution which is globally optimal. Furthermore, the process takes a long time even when the initial values are close to the ground truth.

Disease	Mobility	Parameters	SSE	R^2	Time (mm:ss)
SIR	ZeroMobility	-	1.6×10^4	0.984	26:47
SIR	CommonBorders	$k = 0.25$	2.9×10^4	0.927	34:13
SIR	NeighbourOrder	$k = 2.0$	1.8×10^5	0.473	20:34
SIR	Gravity	$\alpha = 0.69$, $\beta = 1.64$, $\gamma = 2.8$, $k = 0.0001$	4.5×10^6	0.443	18:23
SIR	Radiation	$k = 0.1$	3.1×10^6	0.569	38:01

Table 6.2: Performance of model fitting with all parameters unknown.

We also used the program to fit the model to the synthetic data sets when a single parameter was unknown. We chose the transmission rate β for this role as the rest can be estimated fairly easily from the data. For example, the recovery rate γ is the inverse of the average infectious period which can be computed using public health data. Table 6.3 presents the results when β was the only unknown parameter.

Disease	Mobility	Parameters	SSE	R^2	Time (mm:ss)
SIR	ZeroMobility	-	1.4×10^3	0.990	20:51
SIR	CommonBorders	$k = 0.25$	5.8×10^3	0.988	25:56
SIR	NeighbourOrder	$k = 2.0$	6.3×10^3	0.987	16:17
SIR	Gravity	$\alpha = 0.69$, $\beta = 1.64$, $\gamma = 2.8$, $k = 0.0001$	2.7×10^4	0.982	21:05
SIR	Radiation	$k = 0.1$	1.3×10^3	0.992	19:22

Table 6.3: Performance of model fitting with β unknown.

We observe that in this case the goodness of fit is much higher. The process also takes less time to complete. It is therefore essential to estimate as many parameters as possible from the data and the literature.

A limitation of the model fitting component is that it does not currently plot the number of cases in time per region of interest as predicted by the fitted model against the actual observations. To create such charts one has to use an external program. This feature has not been implemented yet because of our decision to focus on providing immediate feedback regarding the goodness of fit by developing the side-by-side element.

6.1.5 Simulating human intervention

An important aspect of any spatio-temporal epidemiological framework is the ability to simulate the effect of certain kinds of measures such as closing schools and airports. Our program uses a simple mechanism to provide this feature. Specifically, it allows one to change the parameters of the models while the simulation is in progress. For example, to observe the outcome of reducing mobility by 80% three weeks after the first case was detected, we just reduce the coefficient of the mobility model by that percentage at the correct point. The benefit of this approach is that it provides an abstraction over what kinds of interventions can take place. This enables us to simulate everything, provided of course that we manage to represent the intervention as a numerical change in our variables. We acknowledge that this may not be easy to do but the alternative would overcomplicate the process as the users would need to provide input in some form regarding the services which are available in the area of interest.

A further limitation of the program is that currently the parameters must be changed manually. This can be relaxed in future versions by providing the means to the users to schedule events at certain time points or after certain predicates become true (e.g. as soon as the total number of cases reaches a hundred).

6.2 Application

The main objective of the project was to build a framework which could be used in studies of both biological and socio-technological epidemics. In Chapter 4, we examined the spatio-temporal evolution of the ongoing Ebola outbreak in West Africa, providing practical evidence that our program is capable of performing simulations of diseases. In Chapter 5, we showed how a network of a sample of Twitter users in London can be constructed using STAGE to enable simulations of information diffusion. **We also found evidence in support of the fundamental assumption made by the program that geographical factors, represented in mathematical models of human mobility, influence both kinds of epidemics.** This allowed us to use the same spatio-temporal model (consisting of space, disease and mobility models) to elegantly support both types.

6.3 Extensibility

The main challenge in the development of the program concerned its architecture. Specifically, it was about making the program flexible enough so that it could be extended easily with additional space, disease, mobility and graph models. The overall aim was to make it easy for a researcher to implement and test a hypothetical scenario.

The program was designed to accommodate for this requirement. The code was organised in packages with each one having a clear focus, modules with single responsibilities were developed and standardised design patterns were implemented. In addition, dependency injection was used to reduce coupling between modules and increase extensibility via the use of interfaces and abstract classes (which account for 20% of the classes in STAGE). The greatest piece of evidence regarding the flexibility of the design comes from our own experience in implementing models tailored to the cases of Ebola and Twitter in Chapters 4 and 5 respectively.

Sections 3.6.2, 3.6.4, 3.6.5 and 3.6.6 describe the process for adding a new disease, mobility, space and graph model respectively. Table 6.4 summarises the work which needs to be done to add new components in terms of the number of classes (including inner ones) and the number of methods that need to be implemented, as well as the average number of lines to be written. The latter was computed based on the sizes of existing classes.

Component	Number of classes to write	Number of methods to implement	Average number of lines
Disease model (deterministic)	3	7	192
Disease model (stochastic)	1 + number of events	3 + number of events	113
Graph model	1	3	58
Mobility model	1	3	91
ODE solver	1	1	28
Error function	1	1	47

Table 6.4: Amount of work needed to extend the components.

The table above shows that the number of lines which need to be written to support additional components is estimated to be low. Furthermore, the code is split among a small number of well-defined methods and classes. Moreover, to add new geographical regions of interest one has to create two files (a GeoJSON and a CSV file) and simply place them under the "data/" directory following the process outlined in section 3.6.5.

6.4 Usability

To evaluate the usability of STAGE we quantified the effort needed to perform the main operations. Table 6.5 presents the results. 'P' represents the number of parameters in the spatio-temporal model. The last column shows the maximum number of mouse clicks that need to be made in order to execute tasks from the framework's initial state.

Operation	Space	Number of settings	Max number of clicks
Data visualisation	Physical	4	4
Data visualisation	Logical	4	6
Simulation	Physical	7 + P	13 + P
Simulation	Logical	5 + P	10 + P
Simulation	Hybrid	6 + P	12 + P
Model fitting	Physical	2 + P	3 + 3P

Table 6.5: Amount of work needed to perform the main operations.

The table above demonstrates the simplicity of the user interface as all three operations can be executed with little input from the user. Data visualisation in physical space, in particular, is very quick to initiate (it requires at most 4 clicks) because the default settings in STAGE are optimised for this. In contrast, STEM requires the user to configure first almost the entire simulation environment, whereas GLEAMviz does not support data visualisation from files at all. The remaining operations in STAGE have more options but these can be filled in less than half a minute. We thus consider the objective of providing an easy-to-use interface to be fulfilled.

6.5 Comparison with other frameworks

6.5.1 Surveillance

We used [22] as a guide to visualise and fit a simple model to our Ebola data set. Table 6.6 compares STAGE with Surveillance based on their unique features.

Feature	STAGE	Surveillance
Interactive map visualisation	✓	✗
Side-by-side comparisons	✓	✗
Parameter modifications during simulation	✓	✗
Graphical user interface	✓	✗
Graph-based simulations	✓	✗
HTML animations	✗	✓
Analytical comparisons between models	✗	✓
Charts showing cases over time for each district	✗	✓
Charts showing observed and predicted values	✗	✓

Table 6.6: STAGE vs Surveillance.

Overall, STAGE lacks the capability to generate charts and animations during data visualisation and model fitting. On the other hand, it gives greater emphasis in user interaction during simulations. Furthermore, Surveillance's lack of graph modelling makes it significantly harder to analyse the spatio-temporal spread of socio-technological epidemics.

Figure 6.2 shows the spatio-temporal spread of the Ebola outbreak as visualised by Surveillance. This form of visualisation looks very simple when compared to the real-world, interactive maps rendered by STAGE. The regions are not even labelled. Moreover, the fact that the output consists of external image files does not permit the examination of the geographical properties of the area at all.

Both frameworks require the same number of files in order to visualise an epidemic: one describing the physical space, one containing information regarding the demographics in each region and one containing the epidemic data. Regarding the first file, Surveillance works directly with the Shapefile format while STAGE uses the GeoJSON format. This is a minor difference however as it is very simple to covert between the two formats (see Section 3.6.5).

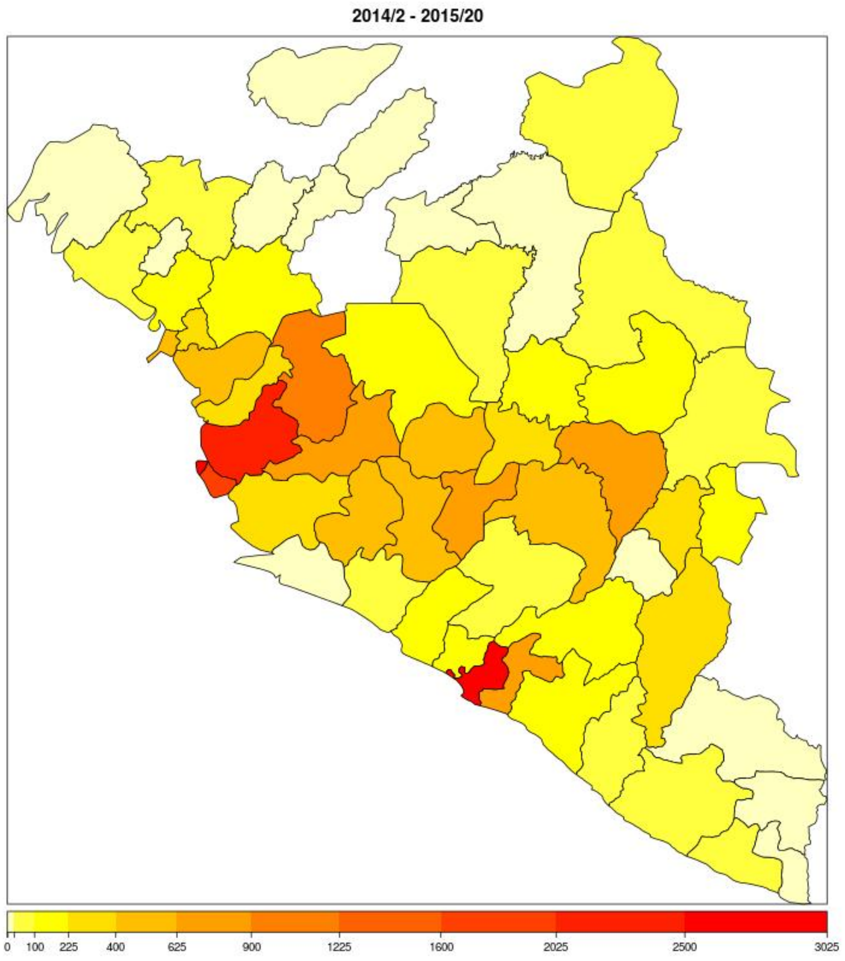


Figure 6.2: Visualization of the 2014 West Africa Ebola outbreak using Surveillance.

6.5.2 STEM

We used STEM to visualise our Ebola data set and simulate their suggested model for this disease. Table 6.7 compares STAGE with STEM based on their unique features.

Feature	STAGE	STEM
Built-in detailed map visualisation	✓	✗
Mathematical mobility models	✓	✗
Support for agent-based simulation	✓	✗
Synthetic data generation	✓	✗
Suitable for analysing online epidemics	✓	✗
Built-in geographic and mobility data	✗	✓
Population segmentation	✗	✓
Simulations in batch mode	✗	✓
Automatic parameter modifications in simulations	✗	✓
Plots of compartments' sizes over time	✗	✓

Table 6.7: STAGE vs STEM.

Figure 6.3 shows the default form of visualisation in STEM. Like Surveillance, this is simply a collection of shapes which does not tell us anything about the underlying geography (such as locations of capitals or airports). There is also a plug-in that uses Google Earth for visualisation, but this can only be used to display the current state in the simulation i.e. it does not feature animations.

Side-by-side visualisation is also available in STEM, with each scenario having its own set of controls. This has the benefit of allowing us to start and pause each one independently, but it requires a greater amount of user input to visualise the data sets synchronously. STAGE follows the simpler approach of always performing the visualisation in lock step, using a single set of controls.

Regarding the simulation of human intervention, both frameworks follow the same approach of allowing the model's parameters to be modified during simulation. STEM however provides the means to automate these changes, making the process more convenient to the user.

Mobility in STEM is modelled using transportation edges between regions describing the rates at which people move from one area to another by various means. To perform changes to this model one has to use a built-in graphical editor to add or remove edges manually, or construct a graph of such edges in a file and import it into STEM. STAGE on the other hand uses mobility models defined mathematically; users can use one of the existing models by providing the values for the parameters or they can make their own by implementing a simple interface. Moreover, mobility models in STAGE are not tied to any specific physical area whereas in the case of STEM the transportation graphs need to be defined for a particular set of regions.

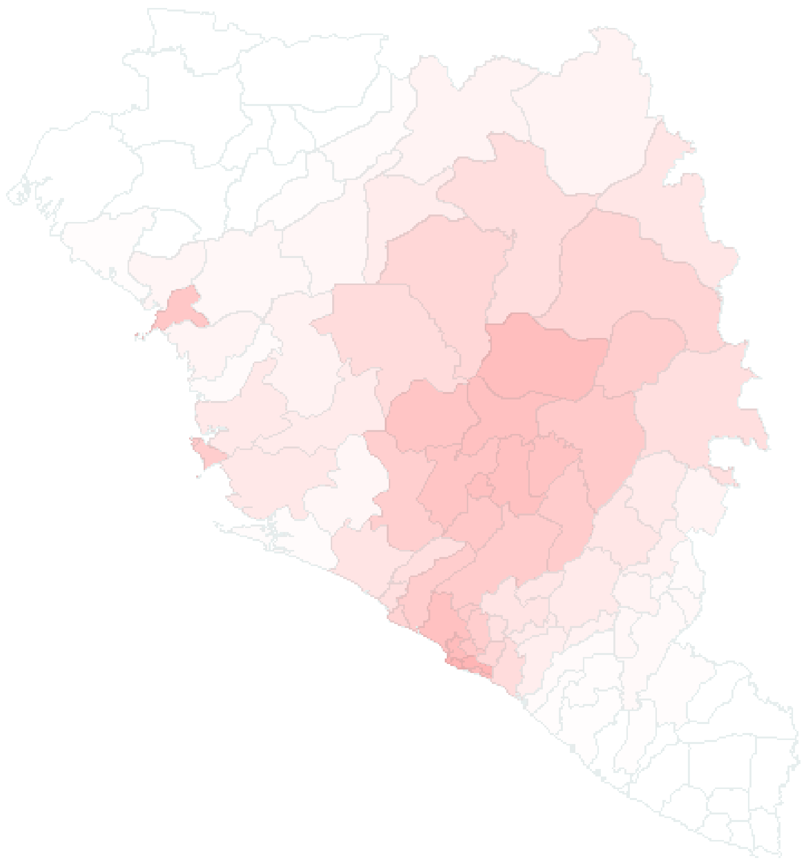


Figure 6.3: Visualization of the 2014 West Africa Ebola outbreak using STEM.

Like Surveillance, STEM is unsuitable for simulating socio-technological epidemics as there is no support for constructing logical networks of individuals and using them to transmit information. On the other hand STAGE features a flexible spatial model which enables it to analyse a wider range of epidemic phenomena.

STEM has built-in data regarding the geography and demography of numerous countries. This means that in theory the user is not required to provide a Shapefile or GeoJSON regarding the regions to be visualised, nor a CSV file containing population sizes. However, the available data is outdated so external resources are needed to produce accurate results. For example, the population sizes for Guinea, Sierra Leone and Liberia come from 2006. Some of their regions, including Conakry (Guinea's capital) are also missing from the geography database.

STAGE makes a clear distinction between visualisation and simulation, offering the ability to visualise data from an external file by specifying just the file name and the physical space. In STEM, one has to set up an entire simulation scenario (requiring multiple components to be configured), choosing a special disease model which replays data from external files. The format of such files is a 2D matrix of cases; the dimensions being space and time. A weakness of STEM's model is that the identifiers of the regions are STEM-specific (and hard to find), whereas STAGE uses the ISO 3166[1] standard.

[1]http://en.wikipedia.org/wiki/ISO_3166

Chapter 7

Conclusion

7.1 Contributions

The main contribution of this project is the development of a spatio-temporal framework for visualising, modelling, simulating and analysing epidemics. Unlike existing frameworks, this is not restricted to physical space, making it suitable to use in the study of socio-technological epidemic phenomena in addition to the biological ones. Its main features include:

- Flexible spatial model: epidemics can be simulated and visualised over physical and/or logical space.
- Flexible disease model: compartmental disease models may be population-based, agent-based, deterministic or stochastic.
- Flexible mobility model: any mathematical model of human mobility can be used in simulations.
- Interactive, detailed visualisation: several kinds of real-world maps and graphs are available to use to display the results. User interactions such as zooming, panning and clicking are supported.
- Side-by-side visualisation: two independent models or a model and the real data set can be visualised side-by-side in lock step. This allows us to receive immediate feedback about the models under test.
- Flexible model fitting procedure: the subset of parameters to optimise as well as the number of iterations are specified by the user. A metric describing the goodness of fit is provided.
- Intervention policies: the model's parameters can be changed during the simulation to reflect the application of certain containment measures. This allows us to predict the effect that the measures have.

- Synthetic data generation: stochastic trajectories of deterministic models can be generated to test the recoverability of the models' parameters.
- Usability: the number of clicks needed to visualise data sets, simulate models and fit models to data is minimal.
- Extensibility: additional space, disease, mobility and graph models can be constructed by implementing the appropriate interfaces.

We demonstrated that our framework is suitable for simulating biological epidemics by examining the 2014 West Africa Ebola outbreak. Specifically, we showed that the spreading is affected by geographical factors like distance, population and population density. We then formed a spatio-temporal model consisting of a sophisticated disease model made up of 8 compartments and the popular *Gravity* model of human mobility. By simulating the model we were able to explain many of the real-world observations regarding the epidemic such as the fact that urban or nearby places are hit more.

We also demonstrated that our framework is applicable for modelling and simulating socio-technological epidemics by examining the tweets and retweets in London on three occasions. Specifically, we collected and visualised the geo-tagged tweets related to the London Marathon (26/04/2015), the birth of Princess Charlotte (02/05/2015) and the UK General Elections (07/05/2015). By analysing the activity on the day of each event we investigated whether the use of a spatio-temporal model was justified. The results indicated that the majority of the tweets were sent by users who were close to the location of the event, and that the majority of one's followers in London were located close to them. We then proposed a spatio-temporal model for tweeting and retweeting which accounts for these geographical properties. We also presented a novel way of using existing models which describe the physical movement of people to construct networks of users in which the connections are weighted by the geographical characteristics of these models. By fitting probability distributions to the histograms generated during the analysis, we obtained estimates for the distributions and possible values of the model's parameters. We noted that the used model succeeded in reflecting the observation that the majority of tweets come from people close to the events.

7.2 Main remarks

Modelling and analysing epidemics is a challenging task. Apart from randomness in each outbreak, the data which is available for analysis may not represent the ground truth with sufficient accuracy. For biological epidemics the quality of the data is often poor due to cases being aggregated, reported late or not reported at all. Location data on the Internet is also hard to obtain due to privacy considerations as well as restrictions imposed by the companies behind popular social networks. We should therefore try to make the visualisation, modelling and simulation aspects as convenient and flexible as possible by providing better tools which are capable of working with general epidemics. Having one framework for both biological and socio-technological epidemics allows us to compare instances of each other directly, using the same models. This allows us to assess better the significance of geography is in the two kinds of interest.

The importance of performing spatio-temporal analysis and modelling for epidemic diseases has been sufficiently demonstrated by the results obtained in Chapter 4. By looking at the 2014 West Africa Ebola outbreak we verified that transmission is not homogeneous between regions as it is affected by certain geographical factors like distance and population, resulting in different mixing rates between them. A suitable mobility model capable of capturing the role of such factors is therefore a necessary component of our overall spatio-temporal model. By incorporating the mobility model in the simulations we attempt to make our predictions more accurate and obtain estimates on where the disease is likely to spread next.

Geography may be important to socio-technological epidemics too. In the case of Twitter we showed that even at the scale of a single city the contact network among Twitter users is influenced by the physical distance between them, and that the tweets regarding an event are concentrated around its location. Such factors may help explain better the spatio-temporal spread of these epidemics, resulting in better marketing strategies. Also, on a research level, a spatio-temporal analysis is worth conducting in order to see if the Internet is so well-connected that the physical locations of users are not important at all.

7.3 Future work

STAGE offers in many ways greater flexibility than existing frameworks like Surveillance and STEM regarding what a researcher can model, simulate and visualise. Nevertheless, there is a plethora of possible extensions which can improve its functionality such as:

Interacting epidemics So far we have been looking at the case where the population under study is affected by a single epidemic. This assumption simplifies both the modelling and the simulation but it does not allow us to examine the dynamics of multiple interacting epidemics. On such occasions it is possible that the combined impact differs from the impact they would have if the epidemics occurred in succession. For example, the risk of developing tuberculosis can be 12-20 times higher when the subject is infected with HIV [16]. On the other hand, there are studies which argue that the replication of HIV can be slowed down by a measles infection [24]. The same phenomena also exist in the context of technology. For example, the tweets of two influential spreaders may reinforce each other and have a greater influence on the recipients, or they may come in conflict, making the epidemic die faster. A useful study in this area would be about finding the best time to share the truth so that a false rumour is brought to a halt. By supporting interacting epidemics STAGE can assist the analysis of such phenomena.

Heterogeneous populations The compartmental models for disease propagation which are supported by the framework assume that the underlying population is homogeneous, in the sense that they all have the same parameter values. We can make these models more realistic by providing the means to users to divide the population into segments based on age, gender, social class, etc. and then provide a different set of parameter values for each segment. This may be a good compromise between increasing the sophistication of the models and keeping the computational complexity of the simulation and fitting at reasonable levels.

Automation Some tasks can be automated to make the process of performing simulations more convenient. For example, we can let the users specify in advance when a parameter should be modified to account for an intervention. They can then run simulations without the need to manually make this change at the correct point each time. One way of implementing this extension would be to provide the means to the users to schedule events at certain time points or when a predicate becomes true e.g. as soon as the number of cases surpasses a certain level. Another idea is to enable the users to run simulations in batch mode, by specifying how the parameters change during each run e.g. incrementing β by 0.05 each time.

Advanced model fitting The model fitting procedure can be enhanced in a number of ways. For example, it can create charts illustrating how the observed and predicted values differ over time in each region. Another idea is to perform such experiments *on-the-fly* given regular observations in time and space regarding the number of new cases. This would allow the model to progressively improve its predictive ability.

Bibliography

[1] Sebastien Ardon et al. “Spatio-temporal analysis of topic popularity in twitter”. In: *arXiv preprint arXiv:1111.2904* (2011).

[2] Konstantin Avrachenkov et al. “Information dissemination processes in directed social networks”. In: *MMB & DFT 2014* (2013), p. 35.

[3] Albert-László Barabási and Réka Albert. “Emergence of scaling in random networks”. In: *science* 286.5439 (1999), pp. 509–512.

[4] Javier Borge-Holthoefer and Yamir Moreno. “Absence of influential spreaders in rumor dynamics”. In: *Physical Review E* 85.2 (2012), p. 026116.

[5] Tao Cheng and Thomas Wicks. “Event detection using Twitter: a spatio-temporal approach”. In: (2014).

[6] CNN. *South Korea grapples to contain MERS as 1,369 in quarantine.* June 4, 2015. URL: `http://edition.cnn.com/2015/06/03/world/south-korea-mers/` (visited on June 13, 2015).

[7] Medical Daily. *Ebola Virus Found In Survivor’s Semen 6 Months After Negative Blood Test.* Apr. 17, 2015. URL: `http://www.medicaldaily.com/ebola-virus-found-survivors-semen-6-months-after-negative-blood-test-329676` (visited on June 13, 2015).

[8] R. Dolgoarshinnykh. “Law of large numbers for the SIRS epidemic”. In: (2013).

[9] Facegroup. *How Stuff Spreads.* Nov. 2013. URL: `http://www.facegroup.com/blog/category/studies/how-stuff-spreads` (visited on June 13, 2015).

[10] Daniel T Gillespie. “Exact stochastic simulation of coupled chemical reactions”. In: *The journal of physical chemistry* 81.25 (1977), pp. 2340–2361.

[11] Marcelo FC Gomes et al. “Assessing the international spreading risk associated with the 2014 West African Ebola outbreak”. In: *PLoS currents* 6 (2014).

[12] Guardian. *How riot rumours spread on Twitter.* Dec. 2011. URL: `http://www.theguardian.com/uk/interactive/2011/dec/07/london-riots-twitter` (visited on June 13, 2015).

[13] Barbara Hofer, Thomas J Lampoltshammer, and Mariana Belgiu. “Demography of Twitter Users in the City of London: An Exploratory Spatial Data Analysis Approach”. In: *Modern Trends in Cartography.* Springer, 2015, pp. 199–211.

[14] Fang Jin et al. “Epidemiological modeling of news and rumors on twitter”. In: *Proceedings of the 7th Workshop on Social Network Mining and Analysis.* ACM. 2013, p. 8.

[15] William O Kermack and Anderson G McKendrick. “A contribution to the mathematical theory of epidemics”. In: *Proceedings of the Royal Society of London A: Mathematical, Physical and Engineering Sciences.* Vol. 115. 772. The Royal Society. 1927, pp. 700–721.

[16] Candice K Kwan and Joel D Ernst. “HIV and tuberculosis: a deadly human syndemic”. In: *Clinical microbiology reviews* 24.2 (2011), pp. 351–376.

[17] Kalev Leetaru et al. “Mapping the global Twitter heartbeat: The geography of Twitter”. In: *First Monday* 18.5 (2013).

[18] Judith Legrand et al. “Understanding the dynamics of Ebola epidemics”. In: *Epidemiology and infection* 135.04 (2007), pp. 610–621.

[19] Xin Lu et al. “Approaching the limit of predictability in human mobility”. In: *Scientific reports* 3 (2013).

[20] Martin I Meltzer et al. “Estimating the future number of cases in the Ebola epidemic—Liberia and Sierra Leone, 2014–2015”. In: *MMWR Surveill Summ* 63.suppl 3 (2014), pp. 1–14.

[21] Stefano Merler et al. “Spatiotemporal spread of the 2014 outbreak of Ebola virus disease in Liberia and the effectiveness of non-pharmaceutical interventions: a computational modelling analysis”. In: *The Lancet Infectious Diseases* 15.2 (2015), pp. 204–211.

[22] Sebastian Meyer, Leonhard Held, and Michael Höhle. “Spatio-temporal analysis of epidemic phenomena using the R package surveillance”. In: *arXiv:1411.0416* (2014).

[23] David Moore and Colleen Shannon. *The Spread of the Code-Red Worm (CRv2).* 2011. URL: `http://www.caida.org/research/security/code-red/coderedv2_analysis.xml` (visited on June 13, 2015).

[24] William J Moss et al. “Suppression of human immunodeficiency virus type 1 viral load during acute measles”. In: *The Pediatric infectious disease journal* 28.1 (2009), p. 63.

[25] Bernard Moulin et al. *Analyzing and Modeling Spatial and Temporal Dynamics of Infectious Diseases.* Wiley Online Library, 2015, pp. 341–370. ISBN: 9781118629932.

[26] John A Nelder and Roger Mead. “A simplex method for function minimization”. In: *The computer journal* 7.4 (1965), pp. 308–313.

[27] Newsweek. *Sierra Leone Grapples With Spike in Ebola Numbers, Fears of Underreporting.* Dec. 12, 2014. URL: `http://www.newsweek.com/sierra-leone-grapples-spike-ebola-numbers-fears-underreporting-291424` (visited on June 13, 2015).

[28] Marily Nika. “Synthedemic Modelling and Prediction of Internet-based Phenomena”. PhD thesis. Imperial College London, 2014.

[29] Marily Nika, Gergana Ivanova, and William J Knottenbelt. “On celebrity, epidemiology and the internet”. In: *Proceedings of the 7th International Conference on Performance Evaluation Methodologies and Tools.* ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering). 2013, pp. 175–183.

[30] World Health Organisation. *Ebola Response Roadmap Situation Report 1*. Aug. 29, 2014. URL: http://www.who.int/csr/disease/ebola/evd-sitrep1-20140828.pdf (visited on June 13, 2015).

[31] World Health Organisation. *Factors that contributed to undetected spread of the Ebola virus and impeded rapid containment*. Jan. 2015. URL: http://www.who.int/csr/disease/ebola/one-year-report/factors/en/ (visited on June 13, 2015).

[32] World Health Organisation. *Interim advice on the sexual transmission of the Ebola virus disease*. May 8, 2015. URL: http://www.who.int/reproductivehealth/topics/rtis/ebola-virus-semen/en/ (visited on June 13, 2015).

[33] Michael Reilly. *AI predicts when you're about to get sick*. July 2012. URL: http://www.newscientist.com/blogs/onepercent/2012/07/ai-predicts-when-youre-about-t.html (visited on June 13, 2015).

[34] Jean-Paul Rodrigue. *Spatial Interactions and the Gravity Model*. URL: hhttp://people.hofstra.edu/geotrans/eng/methods/ch5m1en.html (visited on June 13, 2015).

[35] Filippo Simini et al. "A universal model for mobility and migration patterns". In: *Nature* 484.7392 (2012), pp. 96–100.

[36] Yuri Takhteyev, Anatoliy Gruzd, and Barry Wellman. "Geography of Twitter networks". In: *Social networks* 34.1 (2012), pp. 73–81.

[37] Jeffery K Taubenberger and David M Morens. "1918 Influenza: the mother of all pandemics". In: *Rev Biomed* 17 (2006), pp. 69–79.

[38] SpatioTemporal Epidemiological Modeller team. *Ebola Models*. 2014. URL: http://wiki.eclipse.org/Ebola_Models (visited on June 13, 2015).

[39] WHO Ebola Response Team et al. "Ebola virus disease in West Africa—the first 9 months of the epidemic and forward projections". In: *N Engl J Med* 371.16 (2014), pp. 1481–95.

[40] Twitter. *Company facts*. URL: http://about.twitter.com/company (visited on June 13, 2015).

[41] LD Valdez et al. "Predicting the extinction of Ebola spreading in Liberia due to mitigation strategies". In: *arXiv preprint arXiv:1502.01326* (2015).

[42] Duncan J Watts and Steven H Strogatz. "Collective dynamics of 'small-world'networks". In: *nature* 393.6684 (1998), pp. 440–442.

[43] Amy Wesolowski et al. "Commentary: containing the Ebola outbreak-the potential and challenge of mobile network data". In: *PLoS currents* 6 (2014).

[44] A. Wesolowski et al. "West Africa human mobility models". In: (2014).

[45] WHO. *Ebola Situation Report - 3 June 2015*. June 3, 2015. URL: http://apps.who.int/ebola/en/current-situation/ebola-situation-report-3-june-2015 (visited on June 13, 2015).

[46] WHO. *Ebola Situation Report - 31 December 2014*. Dec. 31, 2014. URL: http://apps.who.int/ebola/en/status-outbreak/situation-reports/ebola-situation-report-31-december-2014 (visited on June 13, 2015).

[47] Cliff Changchun Zou, Weibo Gong, and Don Towsley. “Code red worm propagation modeling and analysis”. In: *Proceedings of the 9th ACM conference on Computer and communications security*. ACM. 2002, pp. 138–147.