The Effects of Spatio-Temporal Heterogeneities on the Emergence and Spread of Dengue Virus

Warren Tennant

Supervised by Mario Recker

Submitted by Warren Tennant, to the University of Exeter as a thesis for the degree of Doctor of Philosophy in Mathematics.

I certify that all material in this thesis which is not my own work has been identified and that any material that has previously been submitted and approved for the award of a degree by this or any other University has been acknowledged.

Signature: .....

Abstract

The dengue virus (DENV) remains a considerable global public health concern. The interactions between the virus, its mosquito vectors and the human host are complex and only partially understood. Dependencies of vector ecology on environmental attributes, such as temperature and rainfall, together with host population density, introduce strong spatio-temporal heterogeneities, resulting in irregular epidemic outbreaks and asynchronous oscillations in serotype prevalence. Human movements across different spatial scales have also been implicated as important drivers of dengue epidemiology across space and time, and further create the conditions for the geographic expansion of dengue into new habitats. Previously proposed transmission models often relied on strong, unrealistic assumptions regarding key epidemiological and ecological interactions to elucidate the effects of these spatio-temporal heterogeneities on the emergence, spread and persistence of dengue. Furthermore, the computational limitations of individual based models have hindered the development of more detailed descriptions of the influence of vector ecology, environment and human mobility on dengue epidemiology.

In order to address these shortcomings, the main aim of this thesis was to rigorously quantify the effects of ecological drivers on dengue epidemiology within a robust and computational efficient framework. The individual based model presented included an explicit spatial structure, vector and human movement, spatio-temporal heterogeneity in population densities, and climate effects. The flexibility of the framework allowed robust assessment of the implications of classical modelling assumptions on the basic reproduction number, R_0, demonstrating that traditional approaches grossly inflate R_0 estimates. The model's more realistic meta-population formulation was then exploited to elucidate the effects of ecological heterogeneities on dengue incidence which showed that sufficient levels of community connectivity are required for the spread and persistence of dengue virus. By fitting the individual based model to empirical data, the influence of climate and on dengue

was quantified, revealing the strong benefits that cross-sectional serological data could bring to more precisely inferring ecological drivers of arboviral epidemiology. Overall, the findings presented here demonstrate the wide epidemiological landscape which ecological drivers induce, forewarning against the strong implications of generalising interpretations from one particular setting across wider spatial contexts. These findings will prove invaluable for the assessment of vector-borne control strategies, such as mosquito elimination or vaccination deployment programs.

Acknowledgements

First and foremost, I would like to thank Dr. Mario Recker for his unrelenting dedication in supervising me throughout my PhD. His unwavering support, amusing sarcastic flair, and persistent belief have been an absolute blessing; I would not have been able to complete this without him. I would also like to thank Dr. Trevelyan McKinley for consistently helping me to develop not only academically, but also personally. This would not have been possible without his incredible belief, wisdom or wit. I would further like to thank Dr. José Lourenço for his continued guidance and for setting the bar so ridiculously high.

Next, I would like to thank my family for their continued love and support. I give thanks in particular to my father and mother, Lord and Lady Tennant, for providing safe refuge and well-cooked meals during times of need. I am incredibly lucky to have such fantastic parents.

I further give thanks to my friends. First, I would like to thank Dr. JJ Valletta for his wonderful friendship throughout my PhD, and for always having my best interests at heart. I would like to thank others who have been part of the cool maths crew over the years: Marina, Tom & Madeleine, Ajit, Caroline & Ben, Sarah & Leon, Kinana & Kevin, and Faryal & Richard. Thank-you for teaching me that there are far greater things in life than working myself to death. Here, I also give my best wishes to some current first-year mathematics PhDs: Dan, Ed and Deepak... Good luck!

I would next like to thank the following people for their continued friendship, support and belief: Katie, Richard, Sara, Louis, Olivia, Jared, Govinder, Ben T., Ben M., Billy, Cat and the rest of past and present members of 3 Norfolk road, Alice, Sophie, Matty, Bekki, Gus, Ellie, Paul and of course, the two Chloes. You've all provided an ear when I needed it, and a distraction when I wanted to get away. I would also like to thank Alex and James for sustaining me over the years with their coffee.

Naturally, honourable mentions go to Beth Roberts and Emily Duncan. I am struggling to find the words to thank you both enough for your continued love and support. Thank-you for always believing in me and helping me to love exactly who I am, thank-you for friendship in which we could talk about anything without judgement, and thank-you for introducing me to yoga with Lucinda.

I would like to thank these wonderful people: Matteo, Louise, Jake, Hamish, Harry, Victor and my beloved Stav. You've all always believed in me, and provided some much needed laughter and love over the years. Here, I also thank Jerry and Tom for their amusing, light-hearted antics. Finalmente, a Manu, te doy las gracias por tu amistad y por enseñarme español a lo largo de mi doctorado.

Contents

List of figures

List of tables

Author declarations

I, Warren Tennant, declare that the entirety of "Chapter 3 Robustness of the reproduction number estimates in vector-borne disease systems" and parts of "Subsection 1.2.4 Transmission potential" in "Chapter 1 Introduction to dengue epidemiology" were published in *PLoS Neglected Tropical Diseases* during 2018. The full citation of the publication is as follows:

Tennant, W. and Recker, M. (2018). Robustness of the reproductive number estimates in vector-borne disease systems. *PLoS Neglected Tropical Diseases*, 12(12):1–20.

Furthermore, I declare that parts of "Section 1.4 Theoretical approaches" in "Chapter 1 Introduction to dengue epidemiology", and "Subsection 4.3.2 Local connectivity of urban centres in the scale-free network" in "Chapter 4 Effects of community structures and vector ecology on dengue epidemiology" were published in *Evolutionary Applications* during 2018. The full citation of the publication is as follows:

Lourenço, J., Tennant, W., Faria, N. R., Walker, A., Gupta, S., and Recker, M. (2018b). Challenges in dengue research: A computational perspective. *Evolutionary Applications*, 11(4):516–533.

Signature: .....

Abbreviations

CPU	Central processing unit
CYD-TDV	Chimeric yellow fever dengue virus tetravalent dengue vaccine
DENV	Dengue virus
DF	Dengue fever
DHF	Dengue haemorrhagic fever
DFT	Discrete Fourier transform
DSS	Dengue shock syndrome
ELISA	Enzyme linked immunosorbent assay
GPS	Global positioning system
GPU	Graphics processing unit
IBM	Individual based model
IDFT	Inverse discrete Fourier transform
LOESS	Locally estimated scatter-plot smoothing
MCMC	Markov chain Monte Carlo
ODE	Ordinary differential equation
SARS	Severe acute respiratory syndrome
SP9	Seroprevalence at age nine
VHTP	Vector to human transmission period
WPS	Wavelet power spectrum
ZIKV	Zika virus

Mathematical notation

α	Linear effect of temperature on the extrinsic incubation period
β	Vector biting rate
$1/\gamma$	Human recovery period
δ	Amplitude of oscillations in extrinsic incubation period
$1/\epsilon_H$	Intrinsic incubation period
$1/\epsilon_V$	Extrinsic incubation period
η	Linear effect of temperature and humidity on vector longevity
η_H	Heterogeneity in human population size
η_V	Heterogeneity in vector population size
θ	Set of unobserved parameters
ι	External introduction rate
κ	Minimum mosquito-to-human ratio
λ	Force of infection
$1/\mu_V$	Vector life expectancy
ν_{I_V}	Vector to human transmission period
π	Probability density function
ρ_R	Non-linear effect of rainfall on vector density
ρ_H	Non-linear effect of humidity on vector longevity
σ	Standard deviation of the local disease dispersion kernel
Φ	Local disease dispersion kernel
ω	Probability of long distance transmission
a_H	Infant host mortality shape
b_H	Infant host mortality scale
c_H	Adult host mortality shape
d_H	Adult host mortality scale
c_V	Adult vector mortality shape

d_V	Adult vector mortality scale
H	Humidity standardised about the mean
$\hat{H}$	Relative humidity in percent
$\bar{H}$	Mean humidity
L	Host mortality location
M	Maximum mosquito-to-human ratio
m	Minimum mosquito-to-human ratio
N_H	Human population size
N_V	Vector population size
p_H	Probability of human infection given contact with an infectious vector
p_V	Probability of vector infection given contact with an infectious human
R	Normalised rainfall
R_0	Basic reproduction number
R_0^{ODE}	Estimate of the basic reproduction number from ODE-derived parameter formula
R_0^{IBM}	Estimate of the basic reproduction number from IBM-derived parameter formula
R_0^{λ}	Estimate of the basic reproduction number from ODE-derived initial growth rate formula
$\hat{R}_0^{\lambda}$	Corrected estimate of the basic reproduction number from the initial growth rate of an outbreak
R_0^*	Estimate of the basic reproduction number from the endemic equilibrium
T	Temperature in degrees Celsius
T_k	Temperature in degrees Kelvin
t	Time in days

Chapter 1

Introduction to dengue epidemiology

Dengue is regarded as the most prevalent mosquito-borne viral disease of our time (Guzman and Harris, 2015). It is now responsible for causing an estimated 20,000 deaths and 50–100 million infections annually (Gubler, 2011; Murray et al., 2013). Due to major range expansions in recent decades, global incidence of dengue has dramatically increased (Gubler, 2006; Vasilakis and Weaver, 2008) with transmission now occuring in at least 128 countries worldwide (Bhatt et al., 2013; Brady et al., 2012). Despite the recent licensure of the first vaccine against dengue, lack of political will, increased urbanisation and climate change are some of the reasons behind why prevention and control have continued to be fairly poor (Gubler, 2002; Morrison et al., 2008; Murray et al., 2013). In this chapter, I provide a brief overview of the virus, the epidemiology of dengue, its ecological drivers and the theoretical approaches that have been taken to explain its epidemiological characteristics.

1.1 The dengue virus

The dengue virus (DENV) belongs to the genus *Flavivirus* of the family *Flaviviradae*. It is a mosquito-borne virus that consists of four antigenically different serotypes: DENV-1, DENV-2, DENV-3, and DENV-4 (De Simone et al., 2004; Raghwani et al., 2011). The genome of DENV is composed of a single positive-sense RNA molecule of approximately 11 kilobases in a single open reading frame (Lindenbach and Rice, 2003). There are seven non-structural proteins which are involved in viral replication, assembly, pathogenesis and immunoinvasion (Fernandez-Garcia et al., 2009), and three structural proteins, including

the virus envelope, capsid and membrane. The viral envelope mediates virus binding to host cell membranes and is the main target of human antibody responses (Weaver and Vasilakis, 2009). Due to high mutation rates, attributed to erroneous self-replication (Holmes, 2003), RNA-based virus evolution is dictated by the time-scales of host immune responses and population dynamics (Grenfell et al., 2004). However, as with many other vector-borne flaviviruses, the virus must survive in both the vertebrate and arthropod host (Bennett et al., 2003; Holmes, 2003; Weaver and Vasilakis, 2009). Therefore, both host and vector act as a bottleneck in the evolution of DENV, resulting in strong purifying selection.

1.1.1 Immunology

Infection with any one of the four dengue serotypes confers lifelong immunity to that serotype (Gibbons et al., 2007). Sabin (1952) found that humans infected with DENV-1 or DENV-2 were protected from clinical illness when challenged with heterologous virus within two months of primary infection. Furthermore, OhAinle et al. (2011) found that waning cross-protection produced the observed epidemiological dynamics of dengue in Managua, Nicaragua. They found that immunity to DENV-1 protected individuals to a DENV-2 outbreak in 2006, however these individuals were at increased risk of DENV-2 infection in subsequent seasons. However, empirical and theoretical studies alike have yet to reach a consensus of the precise time interval of cross-protection between serotypes (Adams et al., 2006; Lourenço and Recker, 2013; OhAinle et al., 2011; Reich et al., 2013; Sabin, 1952).

In contrast to temporary cross-protection, heterotypic infection in a previously exposed individuals results in an increased risk of more severe forms of the disease through a mechanism called anti-body dependent enhancement (Halstead, 2003; Kliks et al., 1989). This happens when serotype-specific antibodies acquired from primary infection fail to neutralise the heterologous serotype and facilitate entry of the pathogen into host cells, which can lead to increased viral replication (Figure 1.1) (Dejnirattisai et al., 2010; Tirado and Yoon, 2003). Recently, Katzelnick et al. (2017b) showed that the risk of developing severe dengue is highest within a narrow range of anti-DENV antibody titres, suggesting that enhancement depends upon pre-existing antibody levels prior to heterotypic infection.

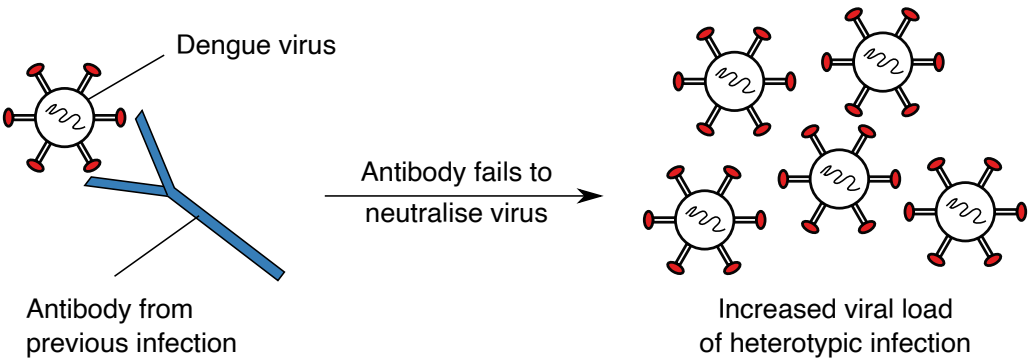

Figure 1.1. Mechanism of antibody dependent enhancement. After acquired immunity to a primary dengue infection, invasion of a second different serotype produces the antibody response of the first infection. The antibodies then fail to bind to the heterotypic virus and facilitate the entry of the pathogen into host cells, increasing overall viral load.

However, the effects of antibody dependent enhancement on the pathogenicity or transmissibility of the virus remain unclear (Katzelnick et al., 2017a). Cross-enhancement between dengue and other arboviruses, such as Zika, has also been suggested to challenge routine diagnosis (Dejnirattisai et al., 2016).

1.1.2 Pathology

Symptomatic dengue infection is diagnosed as Dengue Fever (DF), Dengue Haemorrhagic Fever (DHF) or Dengue Shock Syndrome (DSS) (Halstead, 1980). Dengue fever is symptomatically characterised by flu-like symptoms, such as a fever, headache, joint pain or skin rash (Cobra et al., 1995), whereas individuals suffering from DHF experience internal haemorrhaging which can be fatal (Gubler, 1998). However, a large proportion of dengue infections are asymptomatic, although the ratio of symptomatic to inapparent infections is highly variable from 1:1 to 1:7.5 (Bhatt et al., 2013; Endy et al., 2002; Guzman et al., 2012; Montoya et al., 2013). For example, Endy et al. (2011) showed that in a prospective study of school children in Northern Thailand there was a high spatio-temporal variability in the ratio of symptomatic to inapparent dengue. It has further been shown that clinical outcome are correlated with neutralising antibody titres (Kliks et al., 1989). Recently, Katzelnick et al. (2016) showed that higher neutralising antibody titres correlated with lower probability of symptomatic infection in children in a longitudinal cohort study in Nicaragua.

1.1.3 Serology

Alongside clinical symptoms, serological biomarkers that have been the target for routine diagnosis. This has included isolation of the virus itself and detection of the first non-structural protein of the virus, NS1 (Muller et al., 2017). However, virus clearance by host immune responses and the NS1 protein in post-primary infections make diagnoses difficult (Soler, 1998). Isolation of the virus itself can also take a long time to perform (Lanciotti et al., 1992). An alternative is to measure the presence of host immune response to virus infection through measurement of virus-specific immunoglobulin M (IgM) or immunoglobulin G (IgG). However, as all four serotypes elicit a similar immunological response, IgM and IgG detection is not useful for determining the infecting serotype (Guzmán and Kourí, 2004).

Typically, the IgM antibody appears first, with IgG increasing slowly after the first week of symptom onset. However, in an individual who has previously been exposed to dengue, the IgG antibody appears much more rapidly (Nisalak, 2015). In both cases, the concentrations of both IgM and IgG antibodies slowly decrease during the second week (Muller et al., 2017). Anti-dengue IgM and IgG detection using enzyme-linked immunosorbent assay (ELISA) was one of the most important advances for routine dengue

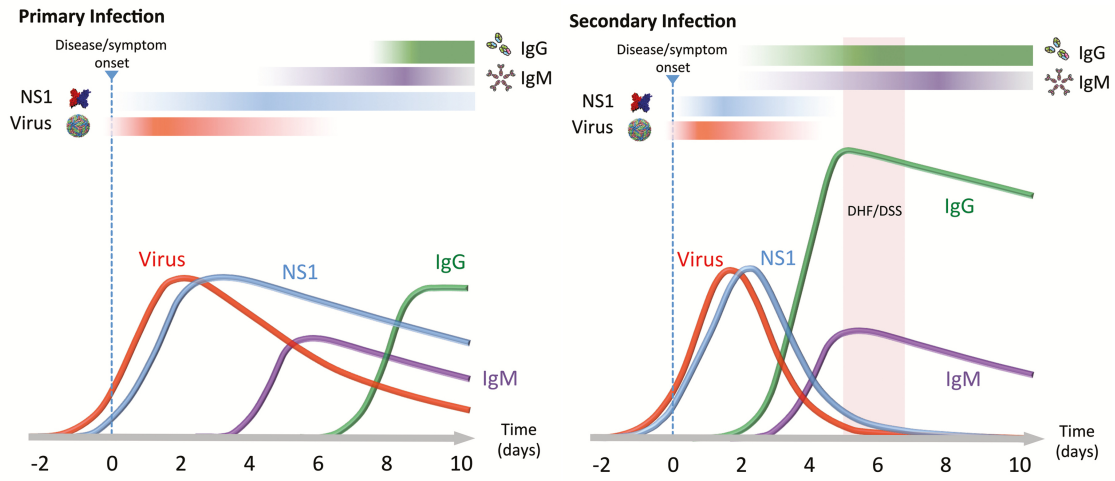

Figure 1.2. Dengue biomarkers used in diagnosis. The timelines of dengue biomarkers in patients of primary and secondary infections differ greatly. Upon primary infection, the virus and non-structural protein 1 (NS1) can be detected after the onset of symptoms, with immunoglobulin M (IgM) appearing well before immunoglobulin G (IgG). Secondary infections are characterised by the rapid emergence of IgG after only a couple of days of disease onset. Figure taken from Muller et al. (2017).

diagnosis (Innis et al., 1989). However, in the case of dengue, ELISAs can result in high false positive rates (Schwartz et al., 2000) due to cross reactivity with other flaviviruses, such as Japanese encephalitis, St. Louis encephalitis and yellow fever (Burke et al., 1982; Vázquez et al., 2003). Furthermore, Felix et al. (2017) found that all anti-dengue ELISAs cross reacted with serum from patients with acute Zika infection, observing a large degree of dengue IgG and IgM seroconversion. Overall, the development of reliable diagnostic tools is required to build a picture of dengue's epidemiology.

1.2 Dengue epidemiology

The earliest record of a dengue-like illness was in a medical encyclopedia from 265AD (Gubler, 2006). The Chinese record described a 'water poison', and noted a possible connection with flying insects. Disease compatible with dengue was then not reported for over a millennia. During the 17th century, reports described outbreaks of a 'break-bone fever' in the French West Indies and Panama (Gubler, 1998). This 'break-bone fever' referred to the characteristic muscle and joint pain of Dengue Fever. From the 1800s, the slave trade enabled the global infestation of dengue's primary vector, *Aedes aegpyti* (Powell et al., 2018). Dengue outbreaks were then large and infrequent until the start of the Second World War (WWII) (Gubler, 1998).

During WWII, population movement spread dengue to Southeast Asia and South America (Gubler, 1998). This established hyper-endemic dengue in countries within these regions, including Venezuela (Barrera et al., 2000; Vincenti-Gonzalez et al., 2017) and the Philippines (Alera et al., 2016; Bravo et al., 2014). With the co-circulation of multiple serotypes, more severe clinical outcomes were observed, including DHF and DSS. For a brief period of time, Central and South America achieved disease control via mosquito elimination. Yet, control was never realised in Asia (Gubler, 2002). By the late 1990s, expanding trade and travel increased the frequency of epidemics globally (Gubler, 2006; Shang et al., 2010).

Since the turn of the millennium, increased international human movement continued to facilitate the spread dengue. This resulted in an unprecedented increase of reported cases in the Americas and Southeast Asia (Dick et al., 2012; Ooi and Gubler, 2009). For example, Lai et al. (2018) reported a large rise in the number of dengue importations to China from 2005–2015. Multiple serotypes began to co-circulate in Brazil (Villabona-Arenas et al., 2014), and there were outbreaks in previously dengue-naïve regions, such as the United States of America: Hawaii in 2001 (Effler et al., 2005), Texas in 2005 (Waterman et al., 2008) and Florida in 2009 (Centers for Disease Control and Prevention (CDC), 2010). More recently, there was an outbreak in Japan during 2014 (Quam et al., 2016), and there were several in Europe, including the south of France in 2015 (Succo et al., 2016), and

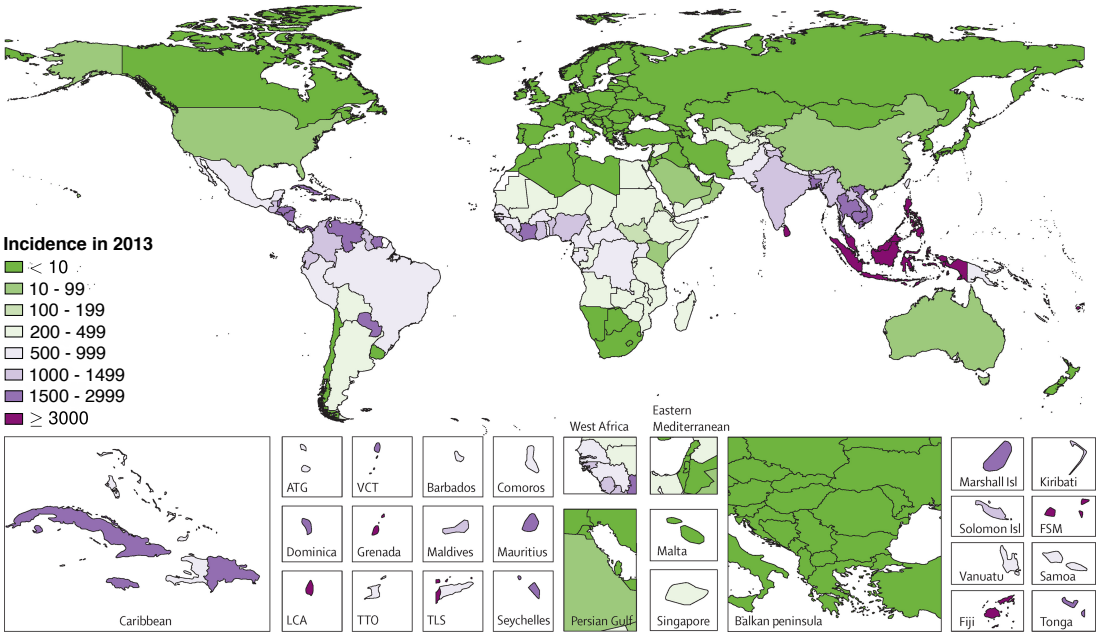

Figure 1.3. Global burden of dengue. Dengue transmission now occurs in over 128 countries worldwide with over 4 billion people at risk. The figure shows age-standardised dengue incidence per 100,000 person-years by country, demonstrating highest burden in South America and South East Asia. Figure taken from Stanaway et al. (2016)

Croatia in 2010 (Gjenero-Margan et al., 2011). During 2012, Madeira experienced a dengue epidemic, sparked by importations from Southeast Asia (Wilder-Smith et al., 2014).

To date, dengue is estimated to infect over 50 to 100 million people annually with 500,000 individuals requiring hospitalisation (Bhatt et al., 2013; Rigau-Pérez et al., 1998). The disease is now endemic in more than 100 countries in Africa, the Americas, the Eastern Mediterranean, Southeast Asia and the Western Pacific (Figure 1.3) (Bhatt et al., 2013; Brady et al., 2012). Persistent high transmission in South America and South East Asia (Fig. 1.3) have placed significant socio-economic costs on these regions (Ladner et al., 2017; Luh et al., 2018; Montibeler and de Oliveira, 2018; Rodrigues et al., 2016; Stanaway et al., 2016). Depending on access to healthcare facilities, the fatality rate of DHF can be as high as 15% (Gubler, 2002). Improvement in healthcare access and treatment has decreased case fatality rates of DHF (AnandaRao et al., 2006; Beckett et al., 2005; Premaratna et al., 2009; Wilder-Smith and Byass, 2016). For example, both Indonesia since 1968 (Karyanti et al., 2014) and Thailand since 1958 (Kalayanarooj, 1999) have reported substantial decreases in dengue-related mortality. In contrast, fatality rates have remained high in India (Chakravarti et al., 2012), and Brazil has reported a sharp rise in case fatality rates from 2001–2011 (Paixão et al., 2015).

1.2.1 Spatio-temporal dynamics

At the population level, dengue dynamics are characterised by irregular epidemic outbreaks, where seasonal oscillations force the dengue virus close to extinction annually, as shown in Figure 1.4. The accumulation of immunity to a particular serotype, and waning immunity to others, drives asynchronous oscillations in dengue's four serotypes. However, within each year there are strong spatio-temporal variations in both incidence and serotype prevalence. This behaviour have been observed across different spatial scales, including at national levels (Figure 1.5) and within urban settings (Jaimes-Dueñez et al., 2015; Teurlai et al., 2012; Yu et al., 2011; Zhu et al., 2019). These patterns are often hidden in spatially aggregated data sets (Figure 1.6). In order to determine the causes behind dengue's epidemiological dynamics, understanding the transmission cycle of the dengue virus has been essential.

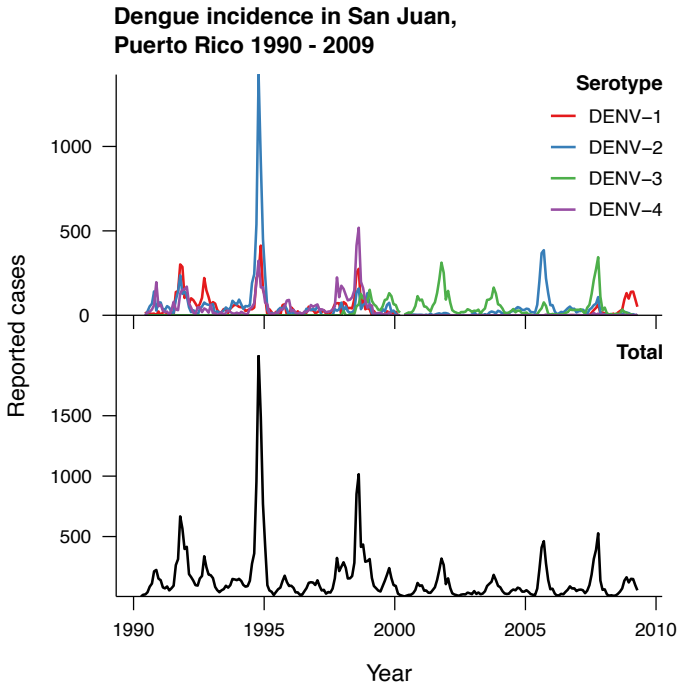

Figure 1.4. Dengue cases in San Juan, Puerto Rico from 1990 to 2009. Reported dengue case data for San Juan, Puerto Rico from 1990 to 2009 was obtained from the Dengue Forecasting project of the National Oceanic and Atmospheric Administration (http://dengueforecasting.noaa.gov), published by the Puerto Rico Department of Health and Centers for Disease Control and Prevention, consisting of laboratory confirmed cases. The frequency of serotype-specific tests has varied over time, so the serotype-specific laboratory confirmed cases were adjusted to match the total laboratory confirmed cases for each month.

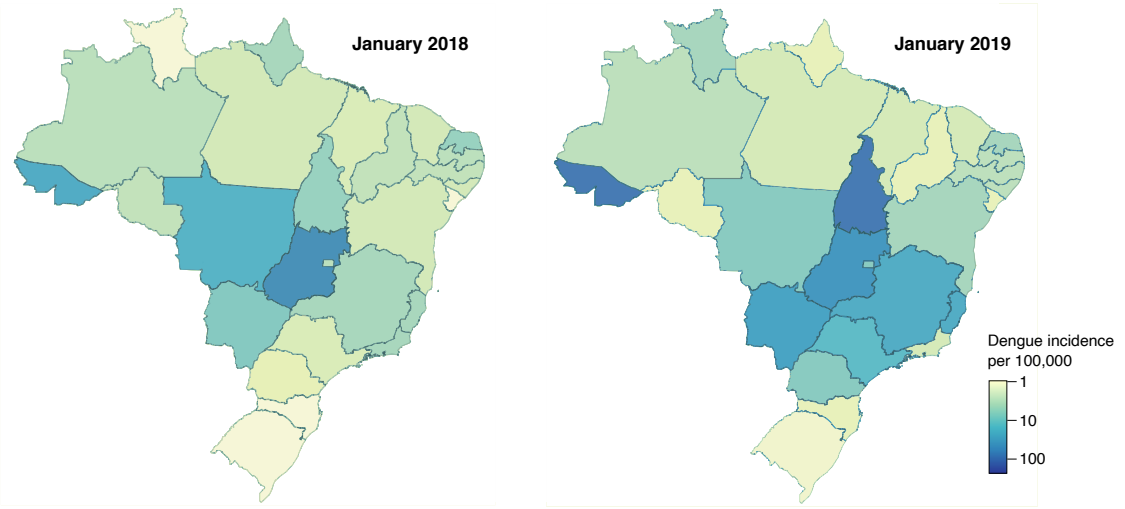

Figure 1.5. Spatio-temporal heterogeneity of dengue incidence across Brazil. Dengue incidence per 100,000 individuals per state in Brazil in January 2018 and 2019. Probable case data for each state was obtained from the Ministério da Saúde, Brasil (Ministério da Saúde, 2019), and incidence calculated from population estimates from the Instituto Brasileiro de Geografia e Estatistica (Instituto Brasileiro de Geografia e Estatistica, 2018).

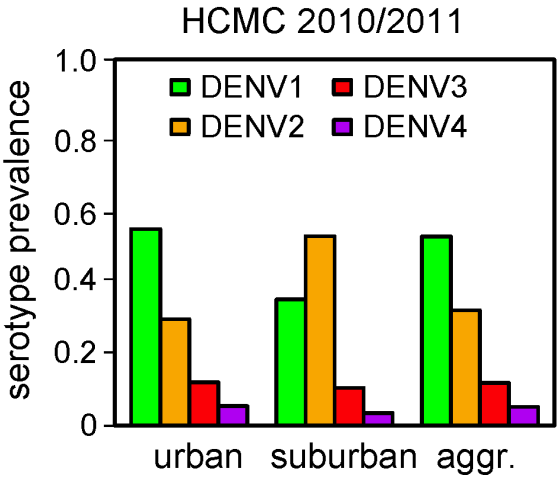

Figure 1.6. Spatio-temporal heterogeneity of dengue seroprevalence in Ho Chi Minh City, Vietnam. There were significant spatial differences in incidence and serotype prevalence in Ho Chi Minh City, Vietnam during the 2010/11 season, which would not have been seen in aggregated data. Taken from Lourenço and Recker (2013).

1.2.2 Transmission

There are two transmission cycles that have been fundamental in the characterising dengue's epidemiological dynamics: the endemic and sylvatic transmission cycles.

Endemic dengue

The endemic transmission cycle mains dengue transmission between human and mosquito populations (Figure 1.7) (Nisalak et al., 2003). The dengue virus is spread primarily by *Aedes aegypti* and partly by *Aedes albopictus* adult mosquitoes. Transmission occurs when an infected female mosquito takes a blood meal from an infected human via insertion of her proboscis (a protruding appendage) into the human's bloodstream for a blood meal necessary for the development of her eggs, a procedure known as probing. The virus replicates in the epithelial cell lining of the mosquito midgut and then travels to the

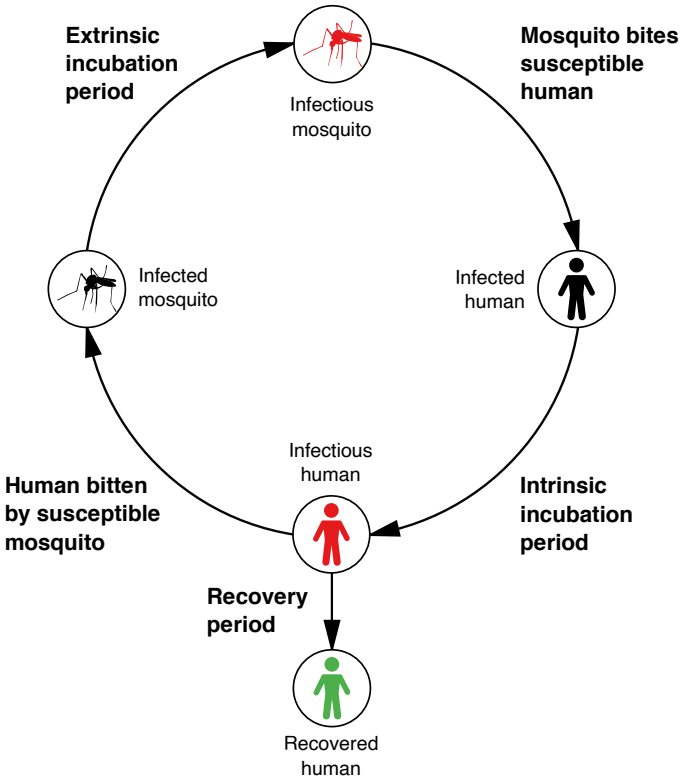

Figure 1.7. Endemic dengue transmission cycle. Endemic dengue is maintained through a human to mosquito-to-human transmission cycle. An infectious mosquito bites a susceptible human individual, who becomes infectious after a period of time known as the intrinsic incubation period. A susceptible mosquito is infected by biting an infectious human, and becomes infectious after the time-period called the extrinsic incubation period has elapsed.

salivary glands. The duration of time for the virus to infect the saliva in the mosquito after taking an infected blood meal is referred to as the extrinsic incubation period. The extrinsic incubation period lasts from eight to twelve days and the mosquito remains infected for rest of life (Salazar et al., 2007). Once the virus has entered the saliva, infection of another human is caused during probing (Salazar et al., 2007). The virus replicates in the human, eventually causing the onset of fever, which lasts 2–10 days (Gubler, 2011). Known as the intrinsic incubation period, the time between infection and the onset of fever is approximately five to seven days (Chan and Johansson, 2012).

Sylvatic dengue

Sylvatic dengue transmission is maintained by a transmission cycle between non-human primates and arboreal *Aedes* mosquitoes in the forests of Southeast Asia and West Africa (Vasilakis et al., 2011; Wolfe et al., 2001). Sylvatic DENV has the potential to spill over into human populations living in close proximity to the regions where they circulate (Carey, 1971; Franco et al., 2011; Vasilakis and Weaver, 2008; Young et al., 2017), implicating the forest-dwelling *Aedes furcifer* and *Aedes albopictus* as bridge vectors of dengue into peri-urban human populations. However, the low susceptibility of *Aedes aegypti* to sylvatic strains (Diallo et al., 2005, 2008) and the lack of evidence for their replication in humans suggest that in order for sylvatic strains to become integrated into the endemic transmission cycle, the strains need to evolve such that they can also replicate in both humans and *Aedes aegypti* (Vasilakis et al., 2007). Indeed, phylogenetic analysis has shown that endemic DENV strains have their ancestry in the sylvatic viruses (Wang et al., 2000). Therefore, tt has previously been suggested that the sylvatic cycle may be a source for dengue re-emergence Vasilakis et al. (2007). However, due to the strong purifying selection of DENV evolution, it is unlikely that sylvatic dengue strains can spill over into the endemic transmission cycle.

1.2.3 Control

The endemic transmission cycle has been the main target for dengue control (Gubler, 1998). However, successful disease prevention has been hindered by the lack of anti-virals against dengue and that there is only a single licensed vaccine available, which has limited efficacy (Wilder-Smith et al., 2010; World Health Organization, 2018b). For these reasons, current control intervention strategies are mainly based on controlling the vector population.

Mosquito elimination

The most effective method for dengue prevention is to reduce the population of dengue's primary vector, *Aedes aegypti* (World Health Organization, 2011). Vector control programs using ultra-low volume application of insecticides have been successfully implemented in the past, leading to severely reduced dengue transmission (Gubler and Clark, 1996). Community-based control programs have been the focus of vector control, aimed at reducing the population of mosquito larvae by targetting standing water in which *Aedes* vectors lay their eggs (Espinoza-Gómez et al., 2002; Lin et al., 2016; Lloyd et al., 1992; Vanlerberghe et al., 2009). Placement of temephos larvicides into local water sources and household water storage containers have been demonstrated to reduce larvae populations with varying degrees of success (George et al., 2015). Furthermore, more environmentally friendly strategies, including introducing biological agents, such as *Mesocyclops* and *Micronecta*, into water storage units, and implementation of community mobilisation strategies, involving the provision of vector breeding sties through community education are possible. These methods have been shown to dramatically reduce *Aedes* larvae abundance, although the resulting success on reducing dengue transmission has been highly variable (Andersson et al., 2015, 2017; Lazaro et al., 2015; Nam et al., 2000; Vu et al., 2005).

Vaccination

Until December 2015, there was no licensed vaccine against dengue. The main challenge in vaccine development is to ensure that immunisation would not prime individuals to more severe infection through antibody dependent enhancement. Pharmaceutical agencies thus sought a vaccine which would provide protection against all four dengue serotypes

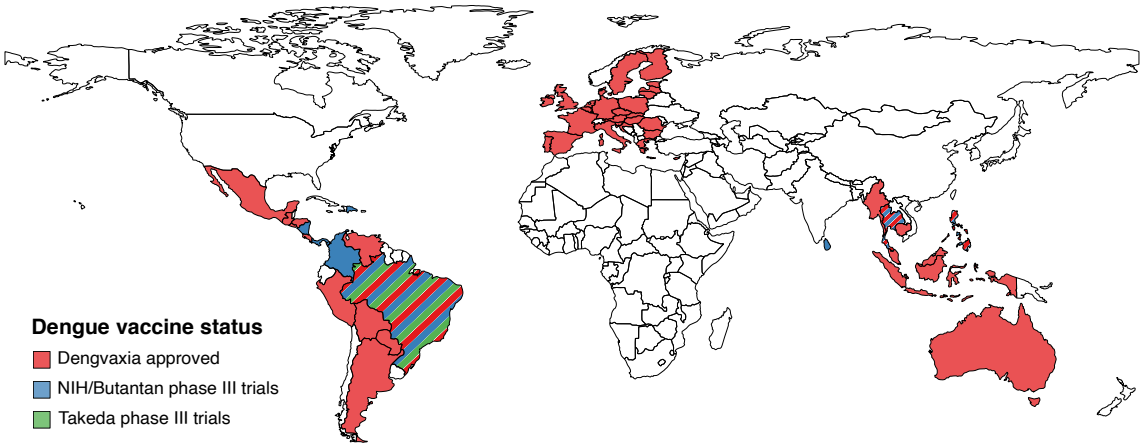

Figure 1.8. Status of dengue vaccines. As of the end of 2018, only one dengue vaccine, CYD-TDV, has been approved for use in the Americas, Southeast Asia and Europe, with NIH/Butantan and Takeda vaccines undergoing phase II clinical trials. Data for countries listed in phase III clinical trials were obtained from ClinicalTrials.gov (https://clinicaltrials.gov), maintained by the National Library of Medicine at the National Institutes of Health, and countries which have approved CYD-TDV were obtained from the World Health Organization (2018b) and European Medicines Agency (2018).

simultaneously (Bhamarapravati and Sutee, 2000; Webster et al., 2009; Whitehead et al., 2007). At the end of 2015, the first licensed vaccine, the live-attenuated tetravalent vaccine, CYD-TDV, also known as Dengvaxia, became commercially available. The vaccine has now been licenced in over 20 countries and authorised for use within the European Union (European Medicines Agency, 2018). Several other vaccines are under development with two vaccines, one developed by NIH/Butantan and the other by Takeda, now undergoing phase III clinical trials (Figure 1.8) (World Health Organization, 2018a). Since licensure, evidence has come to light that Dengvaxia has the potential to prime seronegative individuals to more severe infection (Aguiar et al., 2016, 2017; Flasche et al., 2016), resulting in the Philipines suspending their vaccination programme and the World Health Organization recommending that the vaccine should only be used in highly endemic regions, and on individuals who have been pre-screened for previous dengue infection (Fatima and Syed, 2018; World Health Organization, 2018a).

1.2.4 Transmission potential

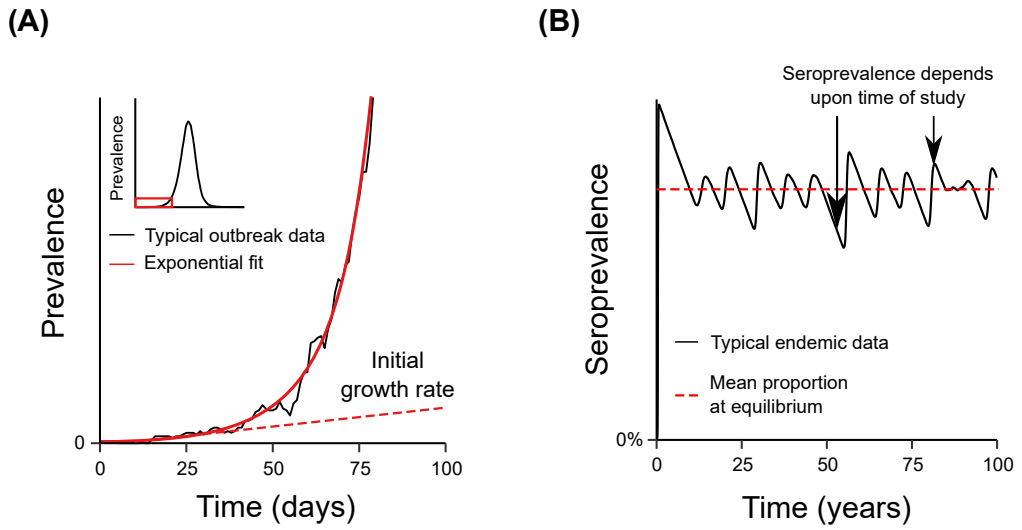

Figure 1.9. Estimating the basic reproduction number from empirical data. **(A)** The reproduction number, R_0, can be estimated from epidemic outbreak data assuming an initially exponential growth rate. **(B)** Seroprevalence levels within a population can also be used to estimate R_0.

A common method to quantify outbreak risk is the basic reproduction, R_0, defined as the expected number of secondary infections arising from a single infection in a fully susceptible population (Heesterbeek and Dietz, 1996). From the R_0 estimate, the risk of infection to susceptible individuals, and the conditions to prevent disease outbreaks, such as through mosquito control or vaccination, can be calculated. A lot of emphasis is put upon the estimation of R_0, which can be done retrospectively, for example, using the initial growth rate of an outbreak (Figure 1.9A), or serological surveys (Figure 1.9B) (Dietz, 1993). But, in its simplest form, it can described as the ratio of infected to susceptible individuals provided the disease is at endemic equilibrium. However, in the case of vector-borne disease there is the added complication of the vector-to-host transmission period (VHTP).

A fundamental component of basic reproduction number estimates, the VHTP is defined as the mean time that an infected vector is able to transmit the pathogen to a host (Mendes Luz et al., 2003). Traditional derivations of R_0 place strong assumptions on key demographical and epidemiological factors, such as the rate at which vectors die, or the rate at which they become infectious. Almost always, constant (age-independent) mosquito mortality rates are assumed, which directly influences the VHTP. However, several field

studies have shown that the mortality rates of *Aedes aegypti* are strongly age-dependent (Harrington et al., 2008; Hugo et al., 2014; Styer et al., 2007). The effects of relaxing these demographical and epidemiological assumptions on R_0 estimates have only been partly explored (Bellan, 2010).

In the context of DENV, the strong spatio-temporal heterogeneities in dengue incidence can induce considerable variation in R_0 estimates across space and time as well. For example, the reproduction number for dengue was estimated to be between 2 and 103 from nine outbreaks across Brazil from 1996–2003 (Favier et al., 2006). The timing and location of dengue introduction will shape the course of an outbreak. That is, an introduction into a highly populated urban area will behave differently to one imported into a rural region. A disease brought in at the start of the transmission season will also react differently to one introduced at the end of the transmission season. Understanding how these ecological factors influence dengue epidemiology is therefore key in implementing effective control strategies and quantifying robust basic reproduction number estimates.

1.3 Ecological drivers of dengue epidemiology

The main drivers of dengue's population level dynamics are still debated, and to some degree, the subject of this thesis. Here, we highlight four factors that are known to influence dengue's spatio-temporal epidemiology: heterogeneity in human and mosquito population density, vector ecology, environmental factors, such as temperature, rainfall and humidity, and movement of both humans are mosquitoes.

1.3.1 Mosquito ecology

There are two main species of mosquito that spread dengue between humans: *Aedes aegpyti* and *Aedes albopictus*. Both vectors have high vectorial competency (susceptibility to the virus), however variation in their ecology induce differences in their ability to spread the disease (Yang et al., 2014).

Aedes aegypti is indigenous to the forests of Africa. The species adapted to the peridomestic environment by breeding in water storage containers. Between the 17th and 19th century, slave trade and commerce introduced *Ae. aegypti* into large tropical coastal cities of Southeast Asia and the Americas. Invasion of countries via river systems during World War II provided a mechanism for the species to penetrate inland (Gubler, 2006). Through urbanization, increased transport, and drinking water supply proliferation in rural areas, the species is now present in both urban and rural areas in most parts of world (Guha-Sapir et al., 2005; Kyle and Harris, 2008). *Ae. aegypti* is a nervous feeder, requiring more than one host to complete a blood meal and more than one blood meal for completion of the gonotrophic cycle (Ponlawat and Harrington, 2005). Combining this timid behaviour with its high domestication levels and strong affinity for human blood (Harrington et al., 2001), results in a highly efficient transmission system for the dengue virus.

Aedes albopictus originates from South East Asia and the islands of the Western Pacific and Indian ocean. Over the last few decades, the species has spread to Africa, West Asia, Europe and the Americas via passive introduction of dormant eggs on international shipments of used tyres (World Health Organization, 2011). The species aggressively

feeds on both humans and animals, and is a concoradant species, meaning that it only requires one blood meal for the completion of the gonotrophic cycle (Delatte et al., 2010; Ponlawat and Harrington, 2005). There exists considerable concern that *Ae. albopictus* would cause serious outbreaks since it is a competent vector of at least 22 arboviruses (notably dengue) (Gratz, 2004). It is projected to have increasing range expansion due to climate change providing more suitable environments in which it can thrive (Yang et al., 2014), including Catalonia, Spain in 2015 (Aranda et al., 2018). However, *Ae. albopictus* generally prefers rural areas, only partly invading peripheral areas of urban cities. For these reasons, it is generally considered a less important vector for dengue.

1.3.2 Climate

Seasonal oscillations in temperature, precipitation and humidity further induce annual fluctuations in vector suitability (Caminade et al., 2017; Johansson et al., 2009; Li et al., 2019; Sharma et al., 2005; Strickman and Kittayapong, 2002). Climate factors have also been shown to influence the transmissibility of the virus (Mordecai et al., 2017). Therefore, differences in climate across space and time drive the marked spatio-temporal heterogeneity in dengue incidence. However, the exact relationships of temperature, rainfall and humidity with the intrinsic factors of dengue are not well-established.

Temperature

Higher temperatures in both *Aedes aegypti* and *Aedes albopictus* are associated with shorter extrinsic incubation periods and faster virus replication rates (Mordecai et al., 2017; Xiao et al., 2014). Additionally, higher temperatures increase the transmissibility of the virus from humans to mosquitoes and vice versa (Lambrechts et al., 2011; Mordecai et al., 2017). The mortality rates of *Aedes* mosquitoes are generally negatively correlated with temperature (Alto and Bettinardi, 2013; Mordecai et al., 2017), although very high temperatures have been associated with shortened mosquito life expectancies (Alto and Juliano, 2001). Moreover, increased temperatures hasten the life cycle of the vector, resulting in smaller-sized mosquitoes, in turn forcing more frequently taking blood meals by the vector in order to obtain enough protein for egg production (Kuno, 1995).

Rainfall

Rainfall is well-documented as a determinant of mosquito-borne disease transmission with increased precipitation creating additional breeding sites for vectors (Chen et al., 2012; Harris et al., 2018; Hu et al., 2006; Li et al., 1985; Messina et al., 2016; Scott et al., 2000b). Therefore, there is generally an increase in dengue transmission during the wet season. For example, Sumi et al. (2017) showed that dengue fever incidence was moderately positively correlated with precipitation levels in Manila, the Philippines 2013–2014. However, at extreme precipitation levels, larvae are easily washed away, decreasing mosquito population density (Koenraadt and Harrington, 2008; Paaijmans et al., 2007).

It is important to note however that during the wet season both temperature and humidity are also favourable for virus propagation (Wearing and Rohani, 2006). The relative contribution of each climate factor to the timing and magnitude of dengue outbreaks is thus unclear and is likely different within each spatial region.

Humidity

Humidity has also been demonstrated to correlate with dengue infection outbreaks (Descloux et al., 2012; Naish et al., 2014; Sumi et al., 2017). However, the effects on vector suitability are disputed (Alto and Juliano, 2001; Canyon et al., 2013; Da Cruz Ferreira et al., 2017). The time delay between humidity and disease incidence is also unclear (Naish et al., 2014). For example Descloux et al. (2012) demonstrated that the highest relative humidity was in phase with the epidemic peak in Noumea, New Caledonia, whereas Depradine and Lovell (2004) demonstrate a 1–2 month delay between relative humidity and dengue incidence.

1.3.3 Population density

Together, the influence of climate on vector suitability and the ecology of dengue's two vectors introduces significant spatio-temporal heterogeneity in mosquito population density. Increased risk of dengue infection has been affiliated with greater mosquito density (Morrison et al., 1998; Sang et al., 2014), suggesting that the habitat preference of *Aedes aegypti* and *Aedes albopictus* may induce heterogeneity in dengue incidence across space. Theoretical approaches have also demonstrated that heterogeneous vector exposure is fun-

damental in generating the observed spatio-temporal incidence patterns of mosquito-borne disease(Manore et al., 2014; Perkins et al., 2013; Romeo-Aznar et al., 2018), whereby increased heterogeneity in vector abundance induces strong spatio-temporal heterogeneity in disease transmission rates, and in turn facilitates disease persistence (Acevedo et al., 2015). However, Cromwell et al. (2017) found there to be no connection between *Aedes aegypti* abundance and DENV seroconversion in a cross-sectional study in Iquitos, Peru. This indicates that the general role which mosquito population density plays in dengue's epidemiological dynamics is uncertain.

To further complicate matters, the relationship between human population density and spatio-temporal dengue incidence is currently not well established, even though the risk of dengue importation has clearly been demonstrated to increase with population expansion and urbanisation (Carbajo et al., 2001; Gubler, 2011; Murray et al., 2013; Ooi, 2015; Pang et al., 2017). Several empirical studies have shown a positive correlation between population density and dengue incidence across different spatial scales (Díaz-Quijano and Waldman, 2012; Ko, 1989; Qi et al., 2015; Sirisena et al., 2017), whereas others established no relationship between human population size and dengue incidence (Kong et al., 2018; Lin and Wen, 2011; Siqueira et al., 2004). Others have also noted a negative association between host population density and dengue incidence (Schmidt et al., 2011), although this may have been confounded by access to an adequate water supply. These conflicting findings suggest that human population density alone is not sufficient to explain the observed spatio-temporal patterns of dengue incidence.

1.3.4 Host mobility

Human movement has long been recognised as a key underlying driver in the dynamics of directly transmitted diseases (Bharti et al., 2011; Riley, 2007). Vector-borne diseases add an additional layer of complexity, whereby the dispersal of vectors can induce further heterogeneity in transmission intensity (Carter et al., 2000). However, due to the limited flight range of *Aedes aegypti* mosquitoes (Kuno, 1995), humans are implicated for the dissemination of the dengue virus across different spatial scales (Figure 1.10) (Adams and Kapan, 2009; Harrington et al., 2005; Stoddard et al., 2009; Wilder-Smith and Gubler, 2008).

At local scales, congregation at schools, hospitals, and religious institutions result in high-levels of dengue transmission, and more long-range human movements spread infection to other parts of the city (World Health Organization, 2011). This has been suggested to induce spatio-temporal heterogeneity in transmission rates and in turn generating the underlying spatio-temporal dynamics of dengue (Perkins et al., 2014; Stoddard et al., 2013). Furthermore, Wen et al. (2012) showed that non-commuters had the propensity to locally spread the virus, with long-distance commuters carrying the virus to geographically distant

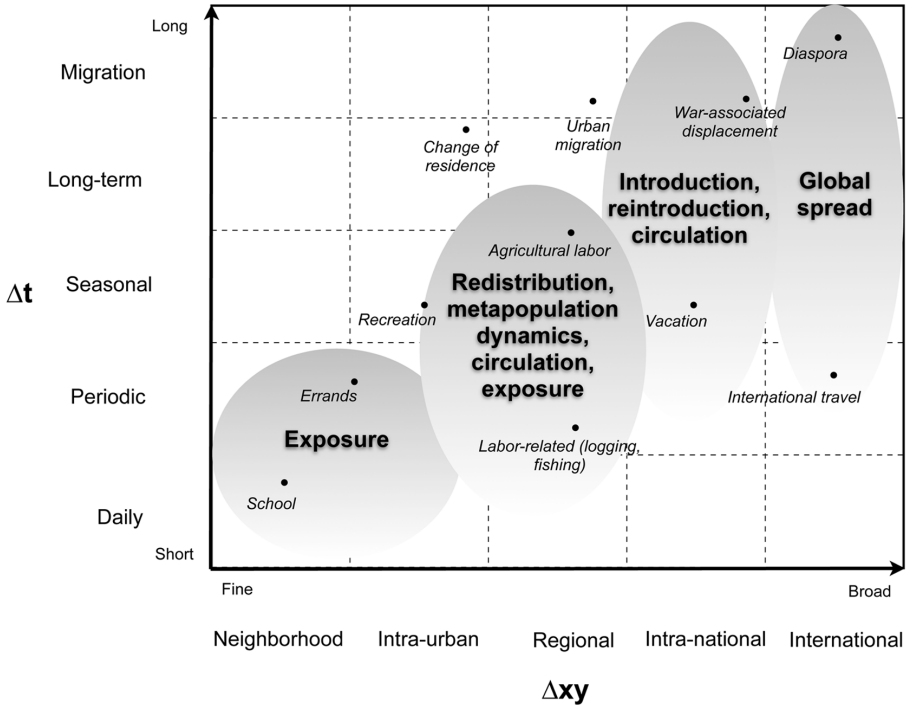

Figure 1.10. Human mobility across different spatio-temporal scales. Human movement allows the dissemination of the dengue virus across different spatial ($\mathbf{\Delta xy}$) and temporal ($\mathbf{\Delta t}$) scales. Figure taken from Stoddard et al. (2009).

areas, increasing the risk of epidemic outbreaks. Within the same household, the mobility patterns of infected and uninfected individuals have been shown to differ (Falcón-Lezama et al., 2017), with Perkins et al. (2016) demonstrating that febrile illness reduces human mobility. However, it is not clear if and how this might influence transmission.

Human movement has further been shown to provide a major role in the spread of dengue on a national scale (Teurlai et al., 2012). Wesolowski et al. (2015) showed that long-distance human movement, based on mobile phone data, was necessary in capturing the spread and timing of dengue outbreaks across Pakistan in 2013. However, work has not been done to show how human movement across large geographical areas influences dengue transmission in hyper-endemic regions.

At an international scale, modern travel is well-documented in providing an efficient mechanism by which the virus can be introduced into dengue-naïve regions, potentially sparking large epidemic outbreaks (Wilder-Smith, 2012; Wilder-Smith and Gubler, 2008). For example, human movement has been implicated as playing an essential role in the recent Madeira 2012 outbreak (Rezza, 2014). In dengue endemic regions, air travel has further been demonstrated to contribute to seeding annual dengue outbreaks in addition to facilitating multiple serotype co-circulation (Nunes et al., 2014; Tian et al., 2017).

1.3.5 Summary

From empirical studies alone, it is not clear how the different ecology of dengue's two vectors dictate the observed spatio-temporal heterogeneity in dengue incidence. Furthermore, there is a lot of uncertainty over how heterogeneity in human, alongside vector, population density drives dengue's epidemiological dynamics. Local, national and international human movement clearly influence dengue epidemiology, yet no work has been done to investigate how human movement across these different scales come together to enable the persistence of dengue in (hyper-)endemic regions. Finally, although we know that climate is strongly associated with dengue incidence, the relationships of temperature, rainfall, and humidity with vector suitability and virus transmissibility are currently not well quantified. Theoretical approaches have therefore been used to further elucidate how these ecological factors drive dengue epidemiology.

1.4 Theoretical approaches

In order to improve our understanding of the effects of climate, vector ecology and host demography on the emergence, spread and persistence of dengue, theoretical transmission models are often employed (Figure 1.11). The vast majority of existing epidemiological models for dengue are based on deterministic systems of ordinary differential equations, which describe the change in the number of individuals experiencing each stage of disease (e.g. susceptible, infected, recovered) over time. However, increasingly complex individual (or agent) based approaches are also being used to capture the inherently stochastic dynamics of dengue epidemiology. Here, we briefly outline two modelling approaches used over the past few decades to better understand the epidemiological drivers of dengue. We further discuss how fitting these epidemiological models to empirical data have been used to further quantify dengue epidemiology.

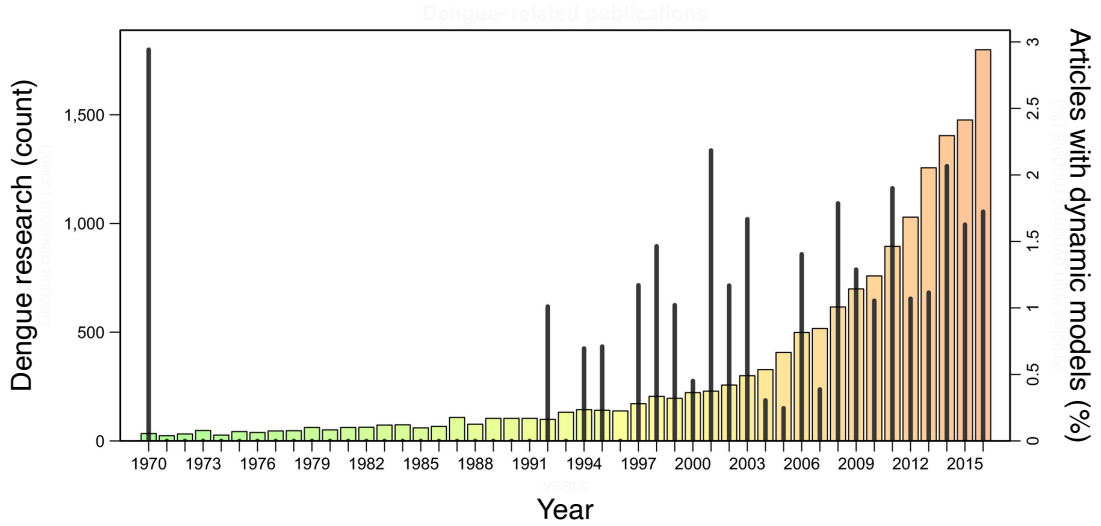

Figure 1.11. Dynamic models in dengue research. Dengue publication over the last five decades. Total number of dengue articles per year (bars) and the percentage of those with a computational focus (spikes). Between 1970 and 2016, a total of 15,267 dengue articles were published, including 190 modelling studies. Figure in Lourenço et al. (2018b)

1.4.1 Systems of ordinary differential equations

Many traditional approaches in theoretical epidemiology rely on mass-action principles whereby individuals are grouped into epidemiological compartments, such as susceptible, infectious or recovered, and the rates of change between each class is described. Here, the rate at which a disease spreads through a population is directly proportional to either the number or proportion of infected and susceptible individuals within that population. Under these assumptions, every individual has the same probability of getting infected, contributes equally to disease transmission and recovers at the same rate as everyone else. Their low computational footprint and analytical tractability make these models an attractive choice for investigating population-level dynamical behaviours, especially when homogeneity in time and space can safely be assumed.

In the context of dengue, ordinary differential equation frameworks have focused upon capturing the irregular epidemic outbreaks of dengue, in addition to the sequential dominance of dengue's four serotypes. One of the first epidemiological models to successfully capture these features was by Ferguson et al. (1999a). They modelled antibody dependent enhancement (ADE) within a two-serotype, vector-host system by increasing the transmission probability to mosquitoes from humans experiencing their secondary infection. By including ADE, serotype prevalence destabilised at the population level. This 2-strain model was then generalised by Schwartz et al. (2005) and Cummings et al. (2005), where the latter showed that ADE was most advantageous in regions where multiple serotypes co-circulate and there are enough hosts to avoid dengue extinction.

However, Wearing and Rohani (2006) then demonstrated that ADE alone was not enough to produce results consistent with empirical data. They further showed that a 1–2 month period of temporary cross immunity was necessary to reproduce the data. Including cross immunity further lowered the extinction risk of dengue outside of the transmission season. These findings were then supported by Adams et al. (2006), who found that only temporary cross-immunity, alongside seasonal oscillations of mosquito density, was required to reproduce the alternating outbreaks of dengue serotypes in Bangkok from 1977–2000.

Until Recker et al. (2009), models required cross-immunity or seasonal forcing, on

top of extreme ADE effects, to avoid annual virus extinction and capture the dynamics of empirical data. But, Recker et al. (2009) demonstrated that extending the model for ADE was enough to reproduce dengue's epidemiological dynamics in the absence of temporary cross-protection and/or seasonal forcing. By increasing susceptibility to secondary infections, in addition to enhancement of human to vector transmissibility, the observed periodic behaviour of dengue serotypes was captured at much more realistic ADE-related effects than previously studied.

As shown above, theoretical approaches first focused on immunological interactions to capture dengue's epidemiological dynamics (Adams et al., 2006; Cummings et al., 2005; Ferguson et al., 1999a; Nagao and Koelle, 2008; Recker et al., 2009; Schwartz et al., 2005; Wearing and Rohani, 2006). However, there is little empirical data showing how ADE influences virus transmissibility and susceptibility (Katzelnick et al., 2017a) and the precise time interval of cross-protection between serotypes has yet to be established (Adams et al., 2006; Lourenço and Recker, 2013; OhAinle et al., 2011; Reich et al., 2013; Sabin, 1952). Additionally, ordinary differential equations implicitly place strong assumptions on key demographical and epidemiological parameters, often prescribing constant transition rates between each epidemiological class. In many cases, these assumptions are unrealistic (Hugo et al., 2014; Mordecai et al., 2017) as homogeneity across space and time cannot be guaranteed. Individual variations, arising through the stochastic nature of infection events or environmental, ecological and demographic heterogeneities, also cannot be captured by these models. Therefore, a different approach is required altogether, such as an individual based model.

1.4.2 Individual based models

Individual or agent-based models offer a more prescriptive way to account for relaxed assumptions on probabilistic infection events and individual-level variation by keeping track of the demographic and epidemiological processes of both humans and mosquito vectors. These frameworks have been implemented to various degrees of realism and permit the inclusion of different spatial details by dividing the population into smaller subpopulations, or communities, typically arranged into a rectangular grid (Figure 1.12). The spatial segregation of individuals in this manner can induce the stochastic local extinction and re-invasion of DENV (Lourenço and Recker, 2013). In turn, this produces the irregular epidemic outbreaks and asynchronous oscillations of dengue's four serotypes, without the need to include immunological interactions. Barmak et al. (2016) similarly showed that random human movement across an urban region can generate the marked spatio-temporal heterogeneity of dengue.

Further spatial detail can also be introduced by adding more realistic spatial arrangements by means of complex networks with nodes representing villages or cities and edges representing their connecting trade or commuting routes. Subpopulations can further be divided to take into account individual households, work places or schools, as well as the human movement patterns between them (Barmak et al., 2016; Chao et al., 2012; Karl et al., 2014; Perkins et al., 2014). Many existing individual based approaches for dengue

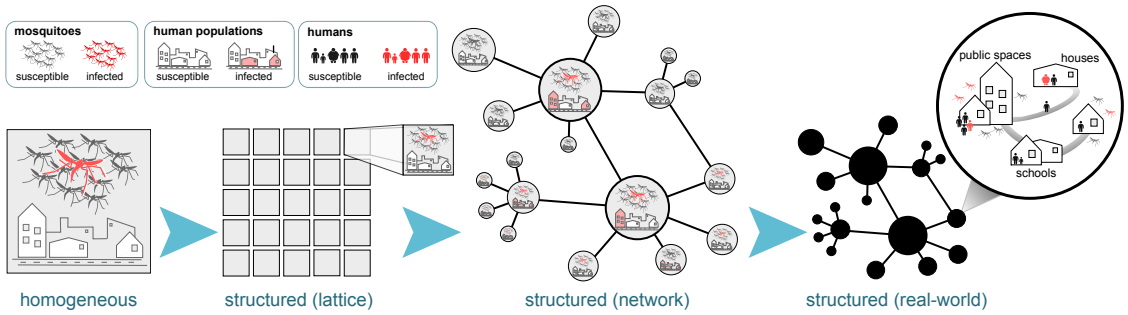

Figure 1.12. Increasing model complexity demands higher computational power. Model detail can be added by dividing a well-mixed population into separate sub-populations, arranged in a regular spatial grid or by means of complex networks to represent geographic distribution of villages, towns and cities, with edges corresponding to major human movement patterns. Depending on data availability, more spatial and demographic detail can be added by considering individual households, places of work or schools. However, the computational demands increase significantly with more detailed information to keep track of, making the model very setting-specific and impractical for sensitivity analyses and model fitting to empirical data. Figure in Lourenço et al. (2018b)

have focused upon this fine spatial scale in order to improve our understanding of the influence of local human movement on dengue epidemiology.

One of the first epidemiological models to adopt this approach was by Chao et al. (2013). They were interested in finding about the effects of relaxing classical model assumptions of homogeneous distribution of vectors and homogeneous mixing between hosts and vectors. In this framework, they explicitly modelled homes, schools and work places, and represented humans and mosquitoes explicitly in a spatial environment. They found there to be no difference in dengue epidemiology between heterogeneous and homogeneous distributions of mosquitoes. But, in line with empirical findings, they found that limited flight range of the vector greatly reduced its ability to transmit dengue among humans, implicating human movement as the main driver of dengue spread.

Work by Karl et al. (2014) then found that human movement only had a small effect on the spread of the dengue virus during the 2008–2009 outbreak in Cairns. Within their framework, they also included explicit relationships between temperature, rainfall and mosquito population dynamics. Their approach indicated that these relationships had little influence on the outbreak. However, they found that compared to the 2003 outbreak, a shorter extrinsic incubation period was necessary to capture the observed dynamics of the 2008–2009 epidemic. This implicated higher than average temperatures during the 2008–2009 period in driving the explosive outbreak.

Common to both above approaches were two models for human movement: one driven by daily commuting behaviour and the other by some distance-dependent infrequent movement, such as to a shopping store, or visiting friends or family. However, socially driven movement had been shown by some studies to be independent of distance (at least over relatively short scales) (Stoddard et al., 2013). This motivated Reiner et al. (2014) to instead investigate the effects of socially-driven human movement on dengue epidemiology. Within their individual based model, all individuals were put into different social groups, where individuals would then visit each other based on their social group. Using this approach, they were able to recreate the observed epidemiological dynamics of a dengue outbreak in north-eastern Peru. This implicated that social proximity was more crucial in driving dengue epidemiology across small spatial scales.

As demonstrated by these frameworks, the explicit description of each individual, together with their ecological and socio-demographic interactions, allows a near limitless level of spatio-temporal detail to be incorporated. However, the inclusion of every minute detail comes at the cost of computational feasibility and generalizability. The increased computational demands imposed by higher model complexity, due to the incorporation of more, and more detailed information of individual-level behaviours, can quickly exceed the capabilities of modern-day personal computers and require either very long run-times or implementation onto high-performance computer clusters. Furthermore, the limited availability of fine-scaled data necessary to parametrise these models often restricts their use to a single spatial setting and/or the theoretical investigation of certain aspects on dengue epidemiology. Additionally, the increased difficulty of interpreting results from the use of more complex models is not compensating for by the increased availability of genetic (Faria et al., 2017; Woolhouse et al., 2015), mobility (Kraemer et al., 2015; Lemey et al., 2014; Wesolowski et al., 2015) or social (Salathé et al., 2012) data sets required to validate model output. Results obtained from these studies, although highly informative for the particular research question, are therefore not easily transferable to other epidemiological or geographical settings, implying that for research questions of a more general nature, a balance needs to be struck between a model's biological and ecological realism and computational feasibility.

Therefore, the spatial resolution of the research question should dictate the complexity level of community structuring required in a model. Non-spatial deterministic approaches provide a natural entry point in understanding the epidemiology of pathogens strictly across time (Ferguson et al., 2016; Nagao and Koelle, 2008; Rodríguez-Barraquer et al., 2014). Epidemiological questions directed over smaller geographical areas, such as investigating the effects of vector distribution, or vector and host movement between an individual's home and workplace, require fine-scale spatial models (Chao et al., 2013; Hladish et al., 2016; Perkins et al., 2016; Reiner et al., 2014). Coarser-scaled network models are applied when individual movement between households becomes redundant, such as exploring the effects of national or international human movement, or environmental heterogeneities over large spatial regions, on dengue epidemiology.

Existing individual based models have only focused on very fine spatial scales to better understand ecological drivers of dengue. There is therefore a clear need for these modelling approaches to be adapted to coarser scales. Furthermore, current individual based modelling approaches are calibrated to very rich empirical data sets, which is often not available. Epidemiological models can instead be fit to much more sparse data sets in order to elucidate drivers of communicable diseases such as dengue.

1.4.3 Fitting epidemiological models to empirical data

Fitting epidemiological models to empirical data provides a useful way to quantify the relationships between epidemiological drivers and the spread of disease. In model fitting, model parameters that may not otherwise be easily, or ethically, measured, such as the probability of dengue infection after a bite from an infected mosquito, are estimated. This is done in a way such that model output reproduces the given empirical data as best as statistically possible. Typically, these parameters are usually inferred by fitting to relatively sparse data sets, such as disease incidence data, and are often done within a maximum likelihood or Bayesian framework.

Maximum likelihood

With maximum likelihood approaches, only single-value estimates of each parameter of interest are calculated. Model parameters are calculated such that the likelihood of observing the empirical data given the unobserved parameters is maximised.

One of the first epidemiological models for dengue to fit to empirical data was by Ferguson et al. (1999b). They fit a system of ordinary differential equations (ODEs) to cross-sectional serological data, which allowed them to estimate changes in serotype-specific transmission rates over time. The results further provided evidence that ADE influences dengue transmission dynamics. Similarly, Chowell et al. (2007) fit to disease incidence over time in order to question the effects of model assumptions on the reproduction number. They found that the strong simplifying assumptions of ODEs may inflate number estimates.

In both of these methods, only the parameter values of best-fit can be calculated (in addition to small confidence intervals around them). Furthermore, Chowell et al. (2007)

highlighted that prior knowledge about key epidemiological parameters is an important consideration in epidemiological modelling. In contrast, the Bayesian paradigm can take prior knowledge about the model parameters into account.

Bayesian frameworks

Bayesian frameworks permit the uncertainty around parameter estimates to be inferred by modelling our knowledge about the parameters of interest through a probability distribution, known as the posterior distribution, in combination with prior knowledge about the parameters of interest. Theoretical determination of the posterior distribution is often impractical in the context of epidemiological models, and so instead the model is executed several tens of thousands of times with different sets of parameters using some variant of a Monte Carlo Markov chain (MCMC) algorithm in order to approximate the posterior distribution. Provided the model is constructed appropriately, the posterior distribution can then shed light upon the epidemiological drivers of disease.

In the context of DENV, only systems of ordinary differential equations have been fit within a Bayesian framework. For example, Pandey et al. (2013) estimated dengue transmission rates by fitting ODEs to hospitalisation data from a dengue outbreak in Thailand during 1984. They surprisingly found that explicitly including mosquitoes within the ODE framework was unnecessary. However, fitting to long-term data, or regions where dengue isn't endemic, requires consideration of mosquito population dynamics.

Lourenço and Recker (2014) considered the influence of temperature and rainfall on mosquito demography, in addition to dengue transmissibility, during the 2012 Madeira outbreak. They fit a climate-dependent ODE framework to reported dengue cases, and quantified the relationships of climate with mosquito life expectancy and the extrinsic incubation period. This allowed them to conclude that there was a high potential for future dengue outbreaks between May and August, when temperature and rainfall were sufficiently high, informing the ideal time-period during which to increase disease surveillance.

Fitting systems of ordinary differential equations can therefore be useful in better quantifying ecological drivers and ultimately informing control strategies. However, as mentioned earlier, these systems of ordinary differential equations place strong assumptions

on model parameters and assume homogeneity across space. Therefore, in order to better understand the relationships between dengue epidemiology and other ecological factors, fitting to individual based models is necessary. However, due to the computational costs of the individual based model itself, fitting within a Bayesian framework, where the model needs to be simulated hundreds of thousands of times, is impractical. Because of this, many individual based models fit within maximum likelihood frameworks or Approximate Bayesian Computation (ABC) (Irvine and Hollingsworth, 2018).

Individual based model fitting

Approximate Bayesian Computation bypasses the need to estimate the posterior distribution from some likelihood function, bypassing the need for the simulation to be executed an extreme number of times. Instead, the posterior distribution is estimated from some summary statistics, such as the timing of the epidemic, or outbreak magnitude. One of the first individual based models for dengue to be fit using this approach was by Hladish et al. (2016). They used the fine-scale spatial model previously developed by Chao et al. (2012) over a much larger geographical region: Yucatan, Mexico. The model was fit to both surveillance and serological data, approximating dengue introduction and reporting rates. However, their focus was more on understanding the benefit of different vaccine deployment strategies, rather than the ecological drivers of dengue.

Recently, Soda et al. (2018) fit an individual based model within a maximum likelihood framework to two consecutive dengue outbreaks in San Juan, Puerto Rico from 2007 to 2008. Over 15 model parameters were inferred from total and sero-specific incidence data, the age distribution of cases, and mosquito trap data. They noted a correlation between rainfall and vector population dynamics, although this relationship appeared to be constrained within the model itself. Even though the model was able capture the dynamics of the epidemiological data, systems of ordinary differential equations have been shown to be more than capable of this. Therefore, the goal of model fitting is not to determine if dengue's epidemiological dynamics can be captured or not, but to find out more about the parameters of interest, and ultimately, the drivers of dengue epidemiology.

Overall, transmission models for dengue are now reaching capacity to be fit to particular

geographical settings in order to simulate real-world scenarios in terms of transmission and control. This has already been done to assess the impact of dengue vaccination on clinical outcomes (Chao et al., 2012; Coudeville and Garnett, 2012; Flasche et al., 2016; Hladish et al., 2016; Lourenço and Recker, 2016; Perkins et al., 2018). These modelling frameworks typically fit systems of ordinary differential equations to empirical data (Chowell et al., 2007; Lourenço et al., 2017; O'Reilly et al., 2018; Tuncer et al., 2018), which, as previously mentioned, fail to capture the inherent stochasticity of dengue dynamics and exhibit strong simplifying assumptions on key epidemiological parameters. Due to computational inefficiency, agent based frameworks have only previously been fit to surveillance data within maximum likelihood or ABC frameworks. However, with the amelioration of computational costs, individual based models could be fit within fully Bayesian frameworks in order to elucidate the ecological drivers of dengue epidemiology.

Thesis outline

In this thesis, I seek to better understand the ecological drivers of dengue epidemiology within a mathematical framework. I use an individual based model as this permitted the inclusion of detailed socio-ecological interactions under much more relaxed assumptions than systems of ordinary differential equations. This approach allows for me to quantify the effects of human movement, climate and heterogeneity in mosquito and human population density on the emergence, spread and persistence of dengue.

In Chapter 2, the spatially-explicit individual based model for dengue that will be used throughout this body of work is presented. Here, I describe the demographic processes of humans and mosquitoes, in addition to the different stages of infection. Communities of humans and mosquitoes are arranged into a spatial structure, and I outline how infections are passed between these communities. The suitability of the framework for modelling dengue is then assessed by comparing model output with empirical data and previously developed theoretical models. Findings agree with existing studies, and I perform several sensitivity analyses in order to gain a better understanding of the influence of human mobility within the model. Importantly, the model is newly implemented within a computationally efficient GPU-accelerated framework, which reduces computational run-times by a factor of 100 compared to classical implementations. The amelioration of computational costs demonstrates the potential for this framework to be used to more deeply explore the effects of spatio-temporal heterogeneities on dengue epidemiology.

In Chapter 3, I assess the impacts of classical model assumptions on the basic reproduction number, R_0. A commonly used R_0 formula assuming constant mosquito mortality rates is compared to an R_0 estimate derived from first principles under relaxed assumptions about mosquito survivorship. Under different assumptions of vector mortality, I directly measure the average number of secondary cases arising from a single infection in a fully susceptible population from the stochastic individual-based simulation model presented in

Chapter 2. I find that by assuming constant mosquito death rates, R_0 estimates are over twice that when assuming most age-dependent death rates. These results are consistent with R_0 estimates based on epidemic growth curves, and I present a correction factor to alleviate these R_0-inflations. Additionally, I show that R_0 estimates based on cross-sectional serological surveys in endemic regions are generally robust.

Many theoretical modelling approaches have focused exclusively on the effects of human mobility on dengue transmission at very fine spatial scales. In Chapter 4, I investigate the effects of community structure, in addition to socio-ecological heterogeneities, on dengue epidemiology across larger geographical regions. Three different networks describing the local mobility of humans are presented, and the critical community connectivity necessary for the spread and persistence of dengue is identified. I show that introducing differences in the connectivity of each community induces the marked spatio-temporal heterogeneity of dengue incidence, with outbreaks synchronised across different regions of the network. Additionally, the influence of spatio-temporal heterogeneity in the mosquito-to-human ratio on dengue epidemiology is assessed. I demonstrate that heterogeneity in mosquito population density can significantly affect the epidemiological dynamics of dengue.

In Chapter 5, the individual based model (IBM) presented in Chapter 2 is extended to include flexible relationships of climate with vector demography and dengue transmissibility. Due to the amelioration of computational costs through GPU parallelisation, the individual based model is able to be fit within a fully Bayesian framework. The model is first fit to simulated incidence data from the IBM itself, correctly inferring key epidemiological parameters used to generate the simulated data. After, I fit the model to empirical Zika incidence data in Feira de Santana, Brazil, from 2015–2017, in order to better understand the importance of environment on arboviral epidemiology. I find that humidity had a large role in dictating the two outbreaks in Feira de Santana, whereas temperature was not as crucial. In addition, motivated by my findings in Chapter 3, the effects of relaxing classical ordinary differential equation model assumptions on inference are explored. I discover that assuming constant and age-dependent vector mortality rates can capture the outbreak data equally well, but inferred mosquito life span is much shorter under constant mosquito death rates. The importance of inter and intra-urban human mobility on the disease

outbreak is then explored. I conclude that the pathogen needed to be introduced into multiple spatially-segregated foci in order to reproduce the observed (spatially-aggregated) empirical data. Furthermore, the forecasting capabilities of this framework are discussed, highlighting its potential usage as a real-time analysis tool for epidemiological outbreaks.

Together, the results presented in Chapters 3–5 enhance the understanding of human mobility, climate and heterogeneity in mosquito and human population density on dengue epidemiology. These findings further re-emphasise the limitations of the strong simplifying assumptions of ordinary differential equation modelling approaches on quantifying ecological drivers of disease.

Chapter 2

A GPU-accelerated, spatially-explicit individual based model

2.1 Introduction

In order to more deeply investigate the effects of environmental, ecological and immunological heterogeneities on dengue epidemiology, while continuing to capture the stochastic epidemiological dynamics of dengue, increasingly complex transmission models are required (Katzelnick et al., 2017a; Lourenço et al., 2018b). Deterministic systems of differential equations dominate the epidemiological modelling literature. In order to capture persistent irregular epidemic outbreaks, in addition to the sequential dominance of dengue's four serotypes, these models often rely on temporary cross-immunity or antibody dependent enhancement to desynchronise serotype-specific immunity at the population level (Adams et al., 2006; Cummings et al., 2005; Ferguson et al., 1999a; Recker et al., 2009; Wearing and Rohani, 2006). However, there is little detail on how these phenomena influence the virus pathogenicity and transmissibility (Katzelnick et al., 2017a). Ordinary differential equations also implicitly place strong simplifying, often unrealistic, assumptions on demographical and epidemiological features, such as mosquito mortality rates or incubation periods. Individual based models offer a more prescriptive approach, allowing for greater flexibility in epidemiological assumptions and do not require immunological interactions to capture the spatio-temporal dynamics of dengue (Lourenço and Recker, 2013). However, individual based models are notoriously computationally expensive as at every time step, the demography of host and vector populations need to be updated. Their long computational run times therefore greatly hinder deep exploratory analyses, and so focus should be

placed upon improving the computational efficiency of these models.

Individual based models are typically implemented on the central processing unit (CPU) of a computer, which, on standard architectures, usually permit up to eight tasks to be processed in parallel. The graphics processing unit (GPU) of a graphics card, an optional component of desktop machine, offers increased computational power, as, unlike the CPU, the GPU is specialised in carrying out thousands of arithmetic operations concurrently. GPU-accelerated computing has already been widely used to enhance general purpose programming: from simulating molecular dynamics (Le Grand et al., 2013; Yang et al., 2007), to machine learning algorithms (Li et al., 2015), improving medical software, such as 3D modelling soft tissue dissection (Wu and Heng, 2004), or simulating medical ultrasounds from CT images (Kutter et al., 2009), to agent-based models for fish schooling (Li et al., 2009) and bird flocking (Hidayat et al., 2016). However, up until now, only the GPU-acceleration of a simple SIR individual based model has been explored (Galvão Filho et al., 2016).

In this chapter, we present the spatially-explicit individual based model for dengue that will be used throughout this thesis, which tracks individual humans and mosquitoes and defines seasonal fluctuations in the extrinsic incubation period and mosquito density. A spatial structure of human and mosquito communities is further included, and we describe how infections are locally and globally passed between these communities. The model is compared with existing findings from empirical data and theoretical models in order to verify the suitability of the framework in capturing the epidemiological dynamics of dengue. Importantly, we implement the model within a GPU-accelerated environment and compare computational run-times with a CPU-exclusive implementation of the model, highlighting the potential of this framework to be used to more deeply explore the effects of spatio-temporal heterogeneities on the emergence, spread and persistence of dengue.

2.2 Methods

2.2.1 Model overview

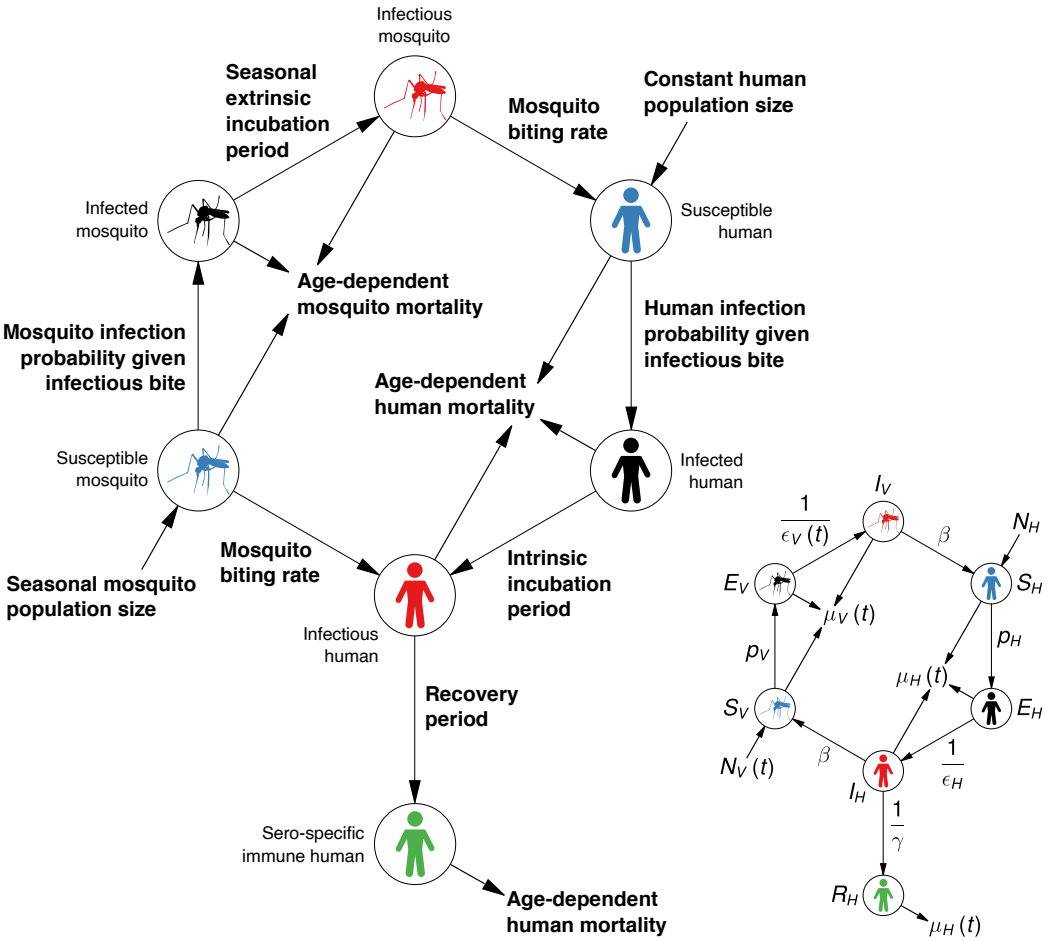

Figure 2.1. Stochastic dengue individual based model. An overview of our dengue individual based model motivated by Lourenço and Recker (2013), which included human and mosquito individuals, an infection process between humans and mosquitoes, and seasonality in the extrinsic incubation period and mosquito population density. In the bottom right corner are the mathematical notations used to describe each process.

In order to capture the socio-ecological drivers of dengue epidemiology, we developed a spatially-explicit stochastic individual based model to describe dengue transmission, motivated by the one previously proposed by Lourenço and Recker (2013). Within our individual based model, we included:

(i) explicit human and mosquito demography,

(ii) a daily infection process describing how individual humans and mosquitoes become infected and transmit dengue,

(iii) seasonality in both mosquito population density and the extrinsic incubation period of dengue and

(iv) a spatial structure of human and mosquito communities,

with points (i)–(iii) exemplified by Figure 2.1. Here, we describe each of the processes (i)–(iv) in full detail.

Human demography

Time, t, was discretised into days and the human population was assumed to be constant over time. Individual humans and mosquitoes aged each day, and died once they'd reached life expectancy. In order to implement infant mortality and flexible adult mortality in the human population, a continuous five-parameter bi-Weibull distribution (Forbes et al., 2010) was considered:

$$\mu_H(t) = \begin{cases} a_H b_H^{a_H} t^{a_H - 1} \exp\left[-(tb_H)^{a_H}\right], & 0 < t < L, \\ \left(a_H b_H^{a_H} t^{a_H - 1} + \dfrac{c_H}{d_H^{c_H}} (t - L)^{c_H - 1}\right) \exp\left\{-\left[(tb_H)^{a_H} + \left(\dfrac{t - L}{d_H}\right)^{c_H}\right]\right\}, & L \leq t < \infty, \end{cases} \tag{2.1}$$

where t is the individual's age in days, L is the location parameter of where infant and adult mortality coincide, and $\{a_h, b_h\}$ and $\{c_h, d_h\}$ are the shape and scale parameters of

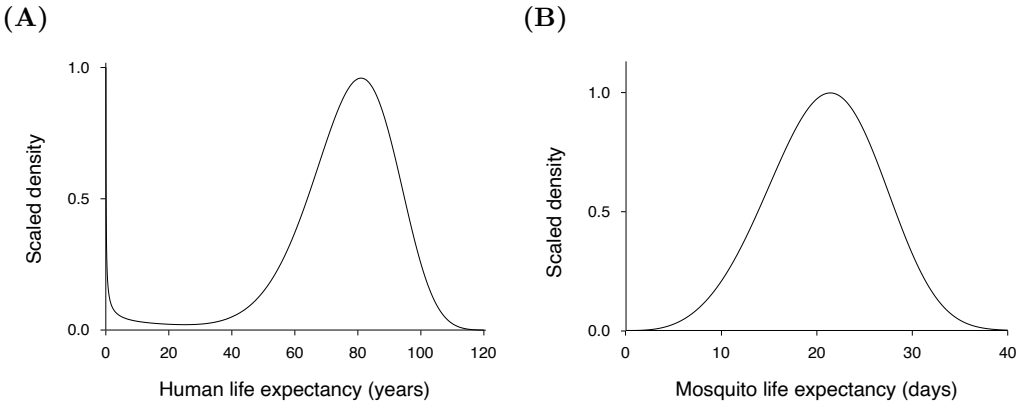

Figure 2.2. Human and mosquito life expectancy. (A) The human life expectancy probability density function was realised as a five-parameter bi-Weibull distribution with $(L, a_H, b_H, c_H, d_H) = \left(8 \times 365, 0.4, 3.65 \times 10^{-7}, 6, 75 \times 365\right)$. Here, the mean life expectancy of a human is approximately 75 years, and the under-five mortality rate is 18 per 1000. **(B)** The mosquito life expectancy probability density function was realised as a two-parameter Weibull distribution with $(a, b) = (4, 23)$. Here, the mean life expectancy is 21 days.

infant and adult mortality, respectively (Figure 2.2A).

Mosquito demography

Only the adult stage of the mosquito life cycle was considered, since only adult vectors acquire and transmit the virus. By utilizing the standard two-parameter Weibull distribution (Forbes et al., 2010), the relative likelihood of an individual mosquito having a life expectancy of age t was:

$$\mu_V(t) = \frac{c_V}{d_V^{c_V}} t^{c_V - 1} \exp\left[-\left(\frac{t}{d_V} \right)^{c_V} \right], \quad 0 < t < \infty, \tag{2.2}$$

where c_V and d_V are the shape and scale parameters respectively (Figure 2.2B).

Infection

The dengue virus was introduced into the system by allowing every human individual to be either susceptible (S_H), exposed (E_H), infected with (I_H), or have recovered from (R_H) each of the four serotypes. Similarly, individual mosquitoes were set to be either susceptible (S_V), exposed (E_V) or infected with (I_V) each serotype. Individuals were only exposed to or infected by one dengue serotype at a time, therefore individuals were removed from the susceptible compartments of all four serotypes upon successful exposure, and only being put back in once recovered. Upon recovery of a specific serotype, human individuals were immune to that serotype for the duration of their life. For a particular serotype, susceptible human individuals were infected at a rate λ_H, also known as the force of infection. Infected humans became infectious after $1/\epsilon_H$ days and recovered after $1/\gamma$ days. To account for the introduction of the disease into the system, humans were also infected at an external infection rate, ι. Susceptible mosquito individuals became exposed to the virus at a rate λ_V, and became infectious after $1/\epsilon_V$ days (see Figure 2.1). The duration of the extrinsic incubation period, $1/\epsilon_V$, intrinsic incubation period, $1/\epsilon_H$, and recovery period for humans, $1/\gamma$, were assumed to be independent of serotype.

The force of infection λ_H, was dependent on the number of infected vectors in the system, and the force of infection on susceptible mosquitoes, λ_V, was dependent on the number of infected humans in the system. All mosquitoes were assumed to bite at a

constant per day rate β. Thus the forces of infection on susceptible humans and mosquitoes for each serotype s were:

$$\lambda_V^{(s)}(t) = p_V \beta \frac{I_H^{(s)}(t)}{N_H},$$
$$\lambda_H^{(s)}(t) = p_H \beta \frac{I_V^{(s)}(t)}{N_H},$$

where p_V and p_H are the probabilities of successfully transmitting the virus from human to mosquito and mosquito to human, respectively, and $I_H^{(s)}(t)$ and $I_V^{(s)}(t)$ are the number of humans and mosquitoes infected with serotype s at time t, respectively.

Seasonality

We first considered annual variations in vector abundance due to seasonality in rainfall, modelled as:

$$N_V(t) = \frac{N_V(0)}{2}\left[\left(1 - \frac{m}{M}\right)\cos\left(\frac{2\pi t}{365}\right) + 1 + \frac{m}{M}\right], \quad t > 0,$$
$$N_V(0) = M N_H(0).$$

where M and m are mosquito to human population size ratios in the middle of the heavy rainfall season and middle of the dry season respectively, $N_V(t)$ is the mosquito population size at time t, and $N_H(t)$ is the human population size at time t (Figure 2.3).

We also included annual, temperature-dependent fluctuations in the extrinsic incubation period, such that

$$\frac{1}{\epsilon_V(t)} = \frac{1}{\epsilon_V(0)} - \delta\cos\left(\frac{2\pi t}{365}\right), \quad t > 0,$$

where δ is the amplitude of the extrinsic incubation period over a year, and $1/\epsilon_V(0)$ is the mean extrinsic incubation period over the year in days. As previously stated, time is initialised to the middle of the hot (and wet) season where the extrinsic incubation period is shortest (Figure 2.4).

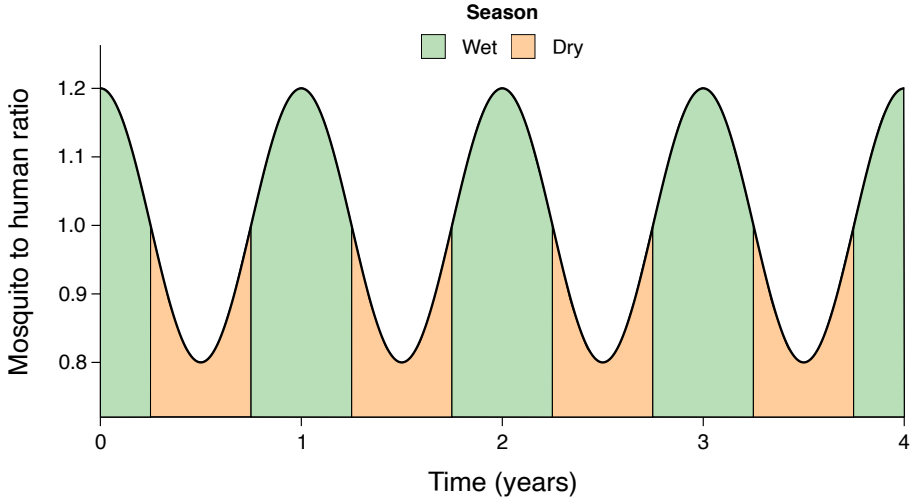

Figure 2.3. Seasonality of mosquito density. Mosquito population density is dependent on annual oscillations in rainfall. The figure shows variation in the mosquito-to-human ratio size over time with $(m, M) = (0.8, 1.2)$. The green time segments highlight the wet season, and the orange segments represent the dry season.

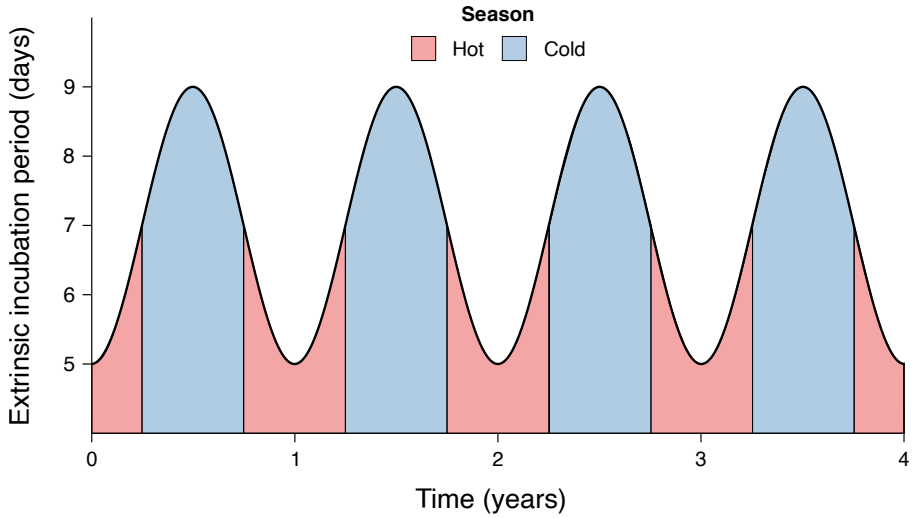

Figure 2.4. Seasonality of the extrinsic incubation period. Extrinsic incubation period of the virus is dependent on annual oscillations in temperature. The figure shows variation in extrinsic incubation period over time with $\delta = 2$ and mean incubation period of 7 days. The red time segments highlight the hot (and humid) season, and the purple segments represent the colder dry season.

Spatial structure

We incorporated spatial structure by dividing the host and vector populations into sets of communities, which were then organised into a non-wrapping lattice (Figure 2.5). Humans and mosquitoes were homogeneously distributed throughout the lattice, and individuals were assumed to mix homogeneously within each community. We then allowed infection events from each community to disperse:

(i) locally, where infectious individuals can infect those in surrounding communities according to some local disease dispersal kernel, and

(ii) across long distances, where infectious individuals can infect anyone within the lattice with probability ω.

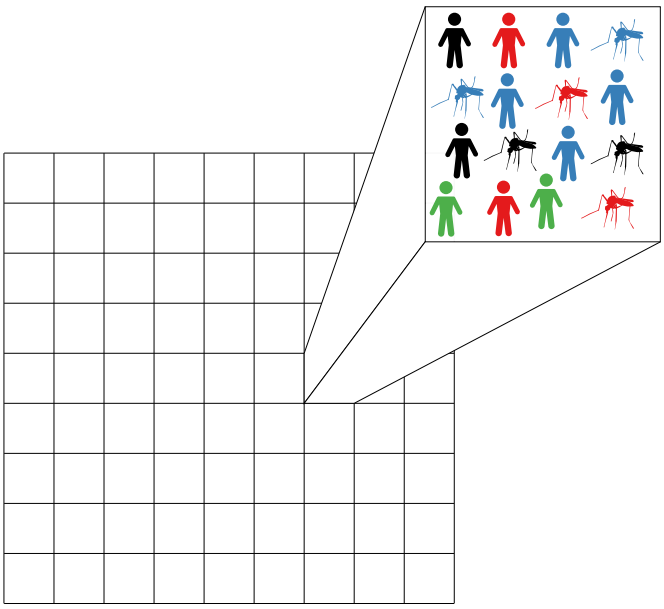

Figure 2.5. Spatial structure of the individual based model. Humans and mosquitoes were grouped into communities which were then arranged into a non-wrapping lattice. All individuals were assumed to mix homogeneously within each community. Each community was assumed to have the same number of individuals.

Community-level force of infection

Let C be the set of all communities in the meta-population and define $\vec{c} = [r, c] \in C$ where $[r, c]$ is the community's row and column number in the lattice. In a model with no disease dispersal, communities are isolated from one another, so the force of infection terms of each community $\vec{c} \in C$ are

$$\lambda_V^{(s)}(t, \vec{c}) = p_V \beta \frac{I_H^{(s)}(t, \vec{c})}{N_H(\vec{c})},$$
$$\lambda_H^{(s)}(t, \vec{c}) = p_H \beta \frac{I_V^{(s)}(t, \vec{c})}{N_H(\vec{c})},$$

where $I_H^{(s)}(t, \vec{c})$ and $I_V^{(s)}(t, \vec{c})$ are the number of infected hosts and vectors with serotype s in community $\vec{c}$ at time t, respectively, and $N_H(\vec{c})$ is the number of humans in community $\vec{c}$.

Local mobility

Infection between communities was considered by allowing vectors to also bite in surrounding communities and humans to temporarily visit local neighbourhoods. The total number of transmissions in community $\vec{c} \in C$ at time t is the force of infection multiplied by the population size. Inter-community infections were included by setting the total number of transmissions of each community to be a weighted sum of the total number of transmissions for each sub-population in the isolated community model:

$$N_V(t, \vec{c})\, \Lambda_V^{(s)}(t, \vec{c}) = \sum_{\vec{\zeta} \in C} \Phi_{\vec{\zeta}}(\vec{c})\, \lambda_V^{(s)}\left(t, \vec{\zeta}\right) N_V\left(t, \vec{\zeta}\right),$$
$$N_H(\vec{c})\, \Lambda_H^{(s)}(t, \vec{c}) = \sum_{\vec{\zeta} \in C} \Phi_{\vec{\zeta}}(\vec{c})\, \lambda_H^{(s)}\left(t, \vec{\zeta}\right) N_H\left(\vec{\zeta}\right),$$

where $\Phi_{\vec{\zeta}}(\vec{c})$ are the normalised weights, $N_V(t, \vec{c})$ and $N_H(t, \vec{c})$ are the total number of vectors and hosts respectively in community $\vec{c}$ at time t, and $\Lambda_V^{(s)}(t, \vec{c})$ and $\Lambda_H^{(s)}(t, \vec{c})$ are the force of infection terms on vectors and hosts respectively in community $\vec{c}$ at time t for serotype s with inter-community transmission.

Local disease dispersal kernel

The normalised-weight function $\Phi_{\vec{\zeta}}$ is called the local disease dispersal kernel (Figure 2.6), and was dependent on the distance between the source community $\vec{\zeta}$ and each destination community $\vec{c} \in C$. It was assumed that the distance dependence follows a normal distribution centred at mean zero with variance σ^2. The absolute weight $\phi_{\vec{\zeta}}(\vec{c})$ was computed as the volume of the two dimensional normal distribution centred around $\vec{\zeta}$ with covariance matrix $\mathbf{\Sigma}$ evaluated at the community $\vec{c}$. Since $|C| < \infty$, the absolute weights were normalised over all $\vec{\xi} \in C$.

$$\vec{X}_{\vec{\zeta}} \sim \mathcal{N}_2\left(\vec{\mu}_{\vec{\zeta}}, \mathbf{\Sigma}\right), \qquad \vec{\mu}_{\vec{\zeta}} = \vec{\zeta} = [r, c], \qquad \mathbf{\Sigma} = \begin{bmatrix} \sigma^2 & 0 \\ 0 & \sigma^2 \end{bmatrix}.$$

$$\begin{aligned} \phi_{\vec{\zeta}}(\vec{c}) = {} & \mathbb{P}\left(\vec{X}_{\vec{\zeta}} \leq \vec{c} + [0.5, 0.5]\right) + \mathbb{P}\left(\vec{X}_{\vec{\zeta}} \leq \vec{c} - [0.5, 0.5]\right) \\ & - \mathbb{P}\left(\vec{X}_{\vec{\zeta}} \leq \vec{c} + [-0.5, 0.5]\right) - \mathbb{P}\left(\vec{X}_{\vec{\zeta}} \leq \vec{c} + [0.5, -0.5]\right), \\ \Phi_{\vec{\zeta}}(\vec{c}) = {} & \frac{\phi_{\vec{\zeta}}(\vec{c})}{\sum_{\vec{\xi} \in C} \phi_{\vec{\zeta}}\left(\vec{\xi}\right)}. \end{aligned}$$

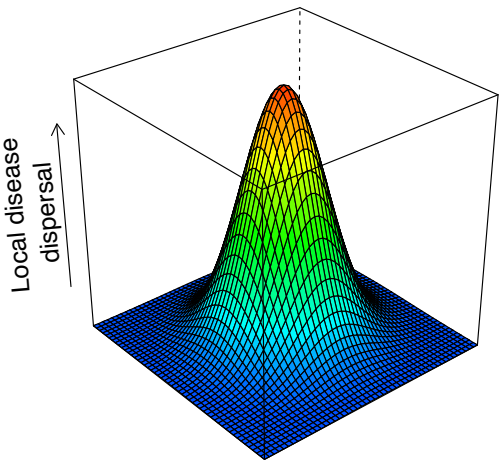

Figure 2.6. Local disease dispersal kernel. Force of infection of the homogeneous mixing model of each community $\vec{\zeta}$ contributes to the spatially dependent force of infection of every community $\vec{c} \in C$ over a two dimensional normal distribution centred at $\vec{\zeta}$. The figure depicts the local disease dispersal kernel, or weights $\Phi_{\vec{\zeta}}$ with smoothing between communities, with $\vec{\zeta}$ positioned in the centre of the lattice.

Long-distance mobility

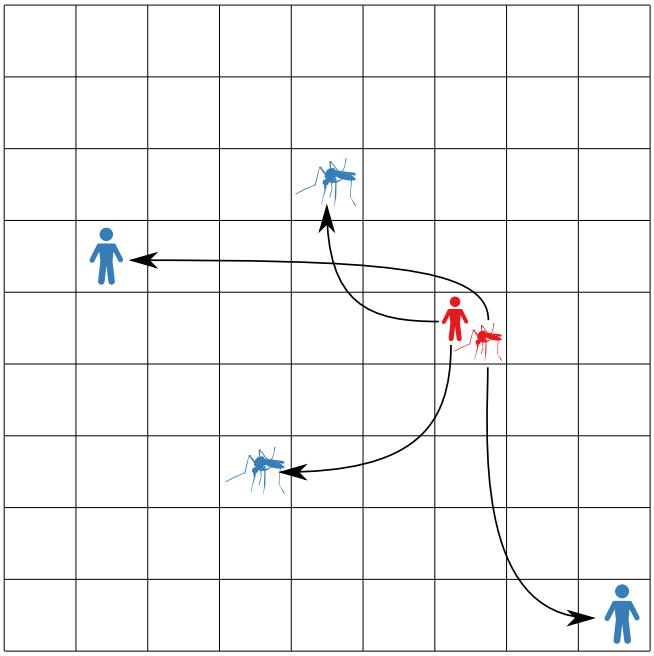

Figure 2.7. Long-distance mobility model. Human individuals were permitted to temporarily visit any community within the lattice with probability ω. This gave rise for the potential for susceptible individuals (blue) of distant communities to become infected if they had come into contact with an infectious individual (red).

Long-distance human mobility was incorporated by allowing human individuals to temporarily visit any sub-population in the lattice with probability ω. Combining this with the local disease dispersal kernel,

$$N_V\left(t,\vec{c}\right)\Lambda_V\left(t,\vec{c}\right)=\sum_{\vec{\zeta}\in C}\left[\left(1-\omega\right)\Phi_{\vec{c}}\left(\vec{\zeta}\right)+\frac{\omega}{|C|}\right]\lambda_V\left(t,\vec{\zeta}\right)N_V\left(t,\vec{\zeta}\right),$$

$$N_H\left(\vec{c}\right)\Lambda_H\left(t,\vec{c}\right)=\sum_{\vec{\zeta}\in C}\left[\left(1-\omega\right)\Phi_{\vec{c}}\left(\vec{\zeta}\right)+\frac{\omega}{|C|}\right]\lambda_H\left(t,\vec{\zeta}\right)N_H\left(\vec{\zeta}\right),$$

where, as before, $N_V\left(t,\vec{c}\right)$ and $N_H\left(\vec{c}\right)$ are the number of humans and mosquito individuals in community $\vec{c}$ at time t, respectively, and $|C|$ denotes the number of communities in the lattice.

2.2.2 Computational implementation

Due to the complex nature of an individual based model, here we outline in detail the implementation of the model and describe the typical computational pathway that is followed (Figure 2.8).

Tracked properties of individuals

Within the framework, a human individual was defined by their current age, life expectancy, the community in which they reside, their infection history and their infection status (susceptible, exposed, infected or recovered). Infected human individuals were also defined by the serotype with which they were infected, the age at which they became infectious and the age at which they would recover. Similarly, an individual mosquito was defined by their age, life expectancy and home community. Infected mosquitoes also had information on their infecting serotype and age at which they became infectious. Since mosquitoes were assumed to be infected for life, there was no need to define recovery times or infection history for mosquitoes.

Demographic and epidemiological initialisation

An individual's age was initialised from the survival function, here defined as the probability that an individual survives beyond a given age (Figure 2.9). For an age-dependent mortality rate $\mu(t)$, such as Equation 2.1 and Equation 2.2, the survival function $S(t)$ was given by

$$S(t) = \int_t^\infty \mu(t).$$

Life expectancy was then randomly assigned according to the survival function. Individuals were then assigned to communities uniformly, and a small proportion (approximately 0.01%) of individuals were infected with a random serotype.

Demographic update

For the demographic update, the cumulative probability function of mortality risk was evaluated at the individuals current age and compared with their life expectancy. If their age exceeded their life expectancy, the individual died, otherwise the individual aged by a

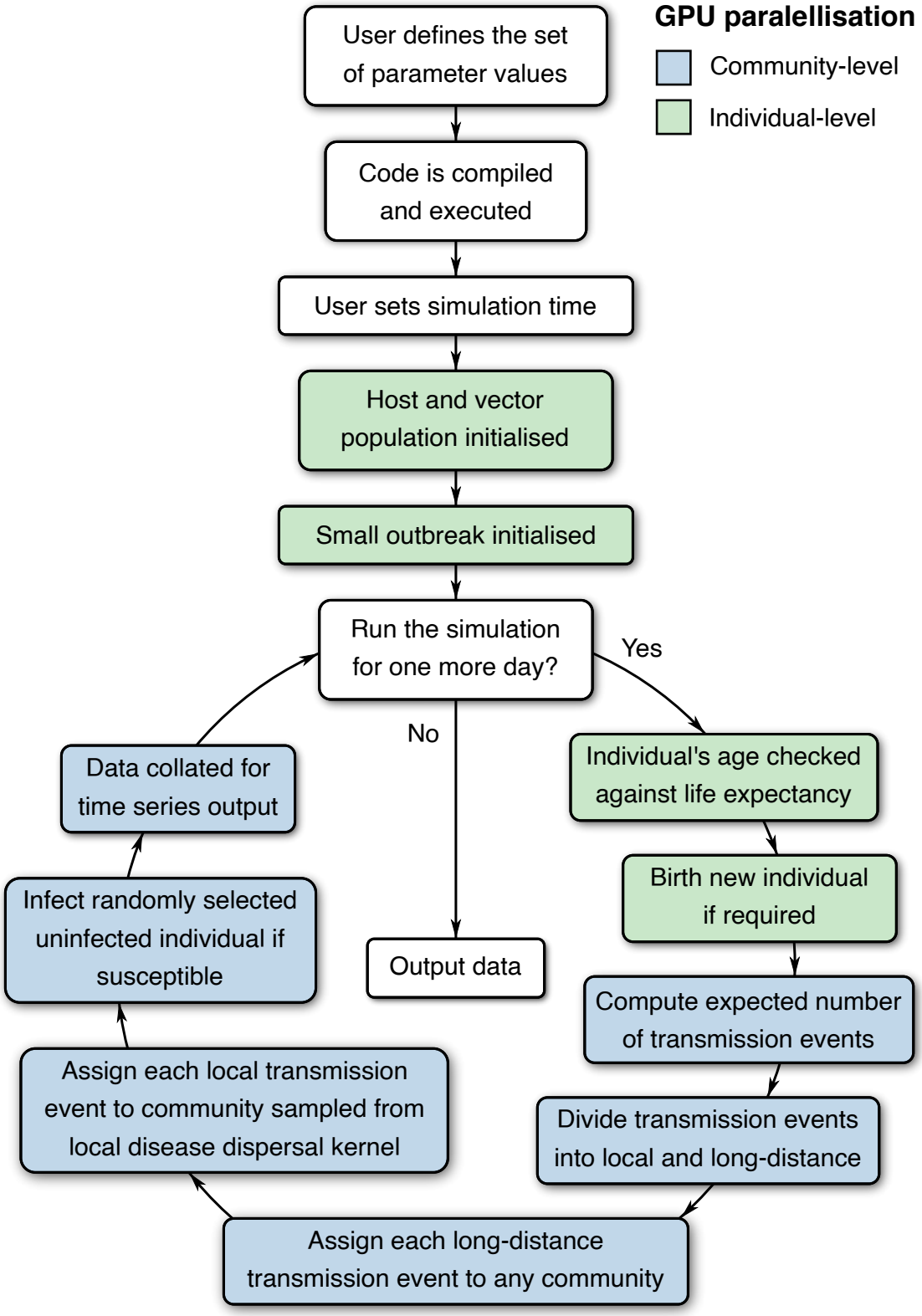

Figure 2.8. Computational walk-through. A flow diagram illustrating the computational footpath of the GPU-accelerated spatial multi-strain dengue transmission model. Colours indicate levels of parallelism at each stage of the computational design: white indicates no parallelism, blue is partial parallelism (e.g. community-level) and green represents full (individual-level) parallelism.

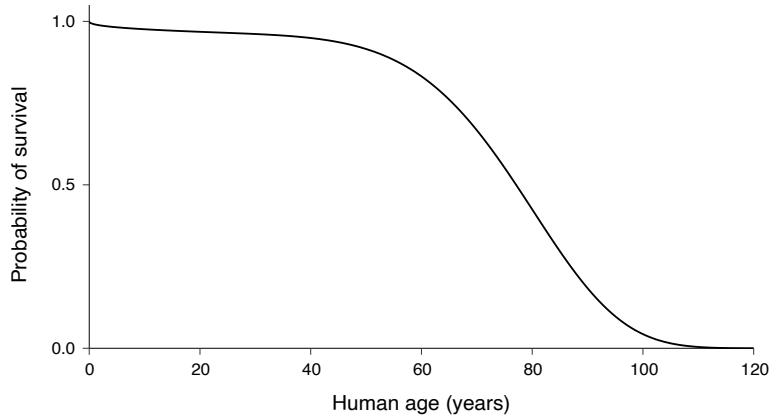

Figure 2.9. Human survival function. The survival function of the five-parameter bi-Weibull human mortality distribution with $(L, a_H, b_H, c_H, d_H) = \left(8 \cdot 365, 0.4, 3.65 \times 10^{-7}, 6, 75 \cdot 365\right)$, or the probability that an individual survives beyond each age.

single day. A new human always replaced a deceased human, whereas a dead mosquito is replaced conditional on the expected mosquito population size at time t, $N_V(t)$. In the demographic update, exposed and infectious individuals were checked if they should become infectious or if humans should recover from the disease respectively. Within the demographic update, the total number of infected individuals per community per serotype was tracked.

Epidemiological update

The expected number of transmission events of each sub-population and serotype was then computed for both host ($\lambda_H N_H$) and vector populations ($\lambda_V N_V$), which were then divided into long-distance and local transmission events. Each long-distance transmission event was assigned a community in the lattice at random. Every local transmission event was assigned to a community based on the local disease dispersal kernel $\Phi_{\vec{\zeta}}$. After all transmission events from each sub-population and serotype were assigned, each community now had a new total number of serotype-specific transmission events to pass to its individuals ($\Lambda_{\{H,V\}} N_{\{H,V\}}$). For every transmission event within each community, an individual from that community was selected at random. If the individual was not already infected and did not have immunity to the infecting serotype, the individual was infected. A newly infected mosquito was assigned the age at which they would become infectious equal to the (temperature-dependent) extrinsic incubation period at the time of infection. Similarly, a human is was assigned the age at which they would become infectious and recover.

GPU-acceleration

Individual based models are notoriously computationally expensive as at each time step, every individual is passed through some demographic process that involves birthing new individuals, terminating old individuals, and ageing individuals. In order to alleviate this high computational cost, the individual based model described above was implemented computationally in a low level programming language, C/C++, with NVIDIA CUDA GPU-acceleration.

In the CUDA environment, a process can be split across multiple threads (of execution). Often, these threads can be executed in parallel to one another (i.e. simultaneously). In CUDA, these threads are (conceptually) grouped into 3D blocks, which are (conceptually) organised into a 3D grid. The maximum dimensions of each block and the grid depends on the GPU architecture of the machine running the code. For the purposes of this model, the grid and blocks are only one dimensional (i.e. an array of threads). Provided serialised code is already written, the general approach for implementing GPU-acceleration on each process is as below.

I. Identify the level at which the process can be parallelised. For the demographical process, this is at the individual level. For the epidemiological process, this is at the sub-population/strain level.

II. Convert existing functions to GPU-accelerated code:

   (i) append `__global__` to the function so that the compiler knows this is a function to operate on the GPU, to be initiated by the central processing unit (CPU),

   (ii) uniquely identify the thread (i.e. individual or sub-population/strain) at the start of the function using `block.Idx`, `blockDim.x` and `thread.Idx` and

   (iii) re-write function to operate on a single thread (i.e. individual or sub-population/strain).

III. Append calls to GPU-accelerated functions with `<<< gridDim, blockDim >>>` before the function name, where `gridDim` and `blockDim` specify the dimensions of the grid (in blocks) and each block (in threads) respectively.

IV. Optimise functions to ensure that a maximal number of threads are operating concurrently.

Within the GPU-acceleration framework, the demographics of many individuals can be updated simultaneously. To that end, the demographic update for both mosquitoes and humans was parallelised across individuals, with individuals assigned to each scheduled thread of the GPU. The epidemiological update was also parallelised on the GPU. Furthermore, during calculation and dispersal of the total number of transmission events per community per serotype, each scheduled thread of the GPU was uniquely assigned to each community and strain. However, in order to prevent the simultaneous infection of the same individual with different serotypes, the infection of individuals was only carried out with each scheduled thread assigned to each community only. The complete documentation and commented source code for the model presented in Chapter 2 are provided in Appendix A and B.

Optimisation

GPU-accelerated code is straight-forward to implement (provided there is existing serialised code). However, GPU-accelerated code generally only executes faster if several optimisations have been implemented. Optimisation ensures that the number of threads executing in parallel at a given time is maximised and are running as fast as possible. This number is almost always less than (otherwise equal to) the maximum number of threads that can execute concurrently, which is determined by the GPU architecture of the machine running the code. To maximise this number, it is crucial to identify the reasons behind why some threads are inactive. In general, thread inactivity is due to waiting for other threads to complete tasks before the inactive threads can continue (or even start). The exact reasons for waiting though are usually code-specific. For this model, the following optimisations were implemented:

I. *Minimise warp divergence*

GPU-acceleration with CUDA works based on the single-instruction-multiple-threads (SIMT) principle. This means that a single instruction is passed to multiple threads at once. In the case of CUDA, a single instruction is passed to 32 threads at a time,

known as a warp. If any path of the 32 threads in the warp diverge from one another, through conditional branching statements (e.g `if`), then single instructions will be passed to each branch one at a time. Nested conditional statements can therefore result in the severe warp divergence with single instructions being passed to only a handful of threads at a time, while others are left waiting. It is therefore important to minimise warp divergence.

In the case of this model, it is challenging to avoid conditional statements. For example, it is necessary to check whether an individual has reached their life expectancy at each time step. However, it is useful to avoid branching over the same conditional statement multiple times throughout the code. This is to ensure that simulation run-time is not penalised multiple times throughout the code because of the same (or similar) branching process. This was originally applicable with the death and birth processes for humans. First an individual would be checked if it was due to die and only after all individuals were checked, later on in the code, a new individual was birthed in place of those who had died. This was naturally very inefficient. Instead, death and birth processes are implemented simultaneously, where a new individual is birthed in place of one who had died immediately.

II. *Eliminate strided memory access*

In CUDA, memory is accessed simultaneously by each thread in a warp. The number of transactions required to access the memory depends upon the relationship between the thread index of the warp and the address of memory being accessed. If thread addresses have a clear 1-to-1 mapping to memory addresses (i.e. the k^{th} thread accesses the k^{th} entry in the data), then memory access is said to be coalesced. Coalesced memory access is optimal as it generally only requires a single transaction to access all information required by the warp. However, if the mapping from thread index to memory address is random or strided (i.e. the k^{th} thread accesses the $s*k^{th}$ entry in the data, where s is the stride), then the number of transactions required is greater than one. Random or strided memory access should therefore be substituted with coalesced memory access. This can be done through careful and consistent memory management and thread assignment. In this model, this is applied when

accessing the total number of infected individuals with strain `s` in a subpopulation `subPop`. As this data is ordered according to strain and then sub-population, i.e. `idx = s*h_subPopTotal + subPop`, then thread identity should correspond directly to this index and similar variables should also have this ordering.

III. *Reduce global memory access*

In CUDA, there are six different memory types which reside on the GPU. The most important is the global memory space. All threads can access global memory, however the memory bandwidth is low. It is therefore optimal to ensure that frequently used data reside in lower memory spaces, such as registers and shared memory, and/or that it is cached in L1 or L2 caches, which have much higher bandwidths. However, the total space in registers and shared memory is fairly small. The compiler will generally place frequently used variables in memory of higher bandwidth after the first read from global memory. Read-only variables can be forced to be stored in low-level caches using the `__ldg()` function.

Reducing the total number of reads and writes to global memory is by far the best way of reducing latency. It is therefore recommended that each thread only reads and writes at most once to a variable within each function. This is not always possible, particularly if data needs to be shared between threads. In the model, this is crucial in the demographic process for mosquitoes, where it is essential the know how many individuals died before birthing new ones to ensure the correct total population size is achieved at each time step (recall that mosquito population size is seasonal). This means that all threads need to communicate with one another through the low bandwidth global memory. One solution is to use shared memory. Shared memory is low latency memory that each thread on a block can access. Instead, the expected total population size of mosquitoes per block is calculated at each time step and the total number of mosquitoes that died in that time step is communicated between the threads in the block. This way, only threads in a block need to communicate with one another about how many mosquitoes died. Although this introduces stochasticity into the overall total population size of mosquitoes, this design is where the largest speed up in simulation runs was found.

IV. *Maximise warp occupancy*

In CUDA, blocks of threads are distributed amongst several streaming multiprocessors on the GPU, which are then split into the warps of 32 threads. If work is not equally distributed across all warps (due to branching processes), then some streaming multiprocessors may finish early while others are still occupied by several warps. The game is to maximise the total number of warps executing at each possible point in time, known as warp occupancy. One way maximise warp occupancy is to adjust the number of threads on a block (and consequently the number of warps on a block). The optimal number of threads per block can be worked out experimentally by comparing simulation run times with different block sizes. It should be noted however that smaller block sizes decreases the total amount of low latency memory per block. In this model, setting 128 threads per block ensures that the total number of streaming multiprocessors doing work throughout the simulation is maximised.

V. *Reduce copying between host and device*

In order for data to be used by the GPU, data needs to be copied from the host memory to the GPU's global memory. Likewise, after computation, for data to be written to file, data needs to be copied back to host memory. This process is computationally expensive. Therefore, it is generally recommended that data only be copied once from host to device and once from device to host. This may not always be applicable if there is a mixture of parallelisable and serial code. In this case, the costs associated with copying data from the device to host, executing serial code on the CPU and copying it back to the device for parallelised functions may outweigh the costs of executing the same code serially on the GPU. This dilemma applies to the model, where individual mosquitoes and humans are infected sequentially. This part of the epidemiological process would likely run faster on the CPU. However, as the infection process is at least parallelised across sub-populations, the cost of running this section of code on the GPU outweighs the high costs associated with copying individual-level data between the host and device. If the meta-population consists of only one community, then an argument could be made where copying is worthwhile.

VI. *Reduce data output*

This optimisation strategy need not only apply to GPU-accelerated models, but to most individual based models, where the computational cost of writing data to file can exceed the cost of the simulation itself. This problem is more apparent in a GPU-accelerated model after simulation times have been reduced. For example, in the case of this model, if the user wants to record the total number of infections per subpopulation per serotype over time for a large number of subpopulations, writing this data to file can take longer than the simulation itself. If this data is required, then this problem is unavoidable, however if only the total number of infections per serotype across the entire meta-population is required as output, it is much faster to sum across the subpopulations in the simulation rather than outputting the files and summing elsewhere.

Model speed-up

The model was simulated for 100 years on the CPU parallelised and GPU-accelerated model with three and a half million human individuals and seven million mosquitoes, then simulation run times were compared. Without GPU-acceleration the model was shown to take 40 times longer to run (Table 2.1). The speed-up permitted extensive analyses into the quantification of ecological and environmental effects on dengue emergence, spread and persistence.

Table 2.1. Simulation run-times on the CPU versus GPU. The simulation using both the CPU and GPU-accelerated implementations of the model using the parameters listed in Table 2.2 for 100 simulation years, with 3.5 million humans and 7 million mosquitoes ($M = 2$). The CPU implementation of the model was parallelised on 8 threads and executed on an Intel Core i7-4790K CPU @ 4.00Ghz processor. The GPU implementation of the model was both executed using an NVIDIA GeForce GTX 970 2GB graphics card and an NVIDIA GeForce GTX 1080 Ti 11GB graphics card.

Parallelisation	Runtime	Speed-up
CPU	40 minutes	-
GPU with GTX 970	1 minute	40
GPU with GTX 1080Ti	25 seconds	96

2.2.3 Epidemiological metrics

To quantify the emergence, spread and persistence of the dengue virus, the output of each simulation was quantified by:

(i) mean prevalence,
(ii) annual variability,
(iii) seroprevalence at age nine based on a cross-sectional survey at the end of each simulation,
(iv) age of each heterotypic infection,
(v) inter-epidemic period,
(vi) epidemic period,
(vii) serotype period,
(viii) serotype extinction risk and
(ix) serotype co-circulation,

which are defined in detail below. Each epidemiological metric, except seroprevalence at age nine, was measured over a 75-year simulation period.

Mean prevalence

A standard measure in epidemiology indicating the level of transmission intensity. Mean prevalence was defined as the mean number of infected humans per 100,000 (human) individuals over the analysed time series.

Annual variability

Annual variability indicated if disease incidence was stable (Figure 2.10CA) or unstable (Figure 2.10CB) over time. Annual variability was calculated as the standard deviation between the annual epidemic peaks over the study period.

Seroprevalence at age nine

Seroprevalence at age nine (SP9) was the percentage of nine year old humans who have been exposed to the virus at least once. SP9 is a previously used indicator of transmission severity (Flasche et al., 2016).

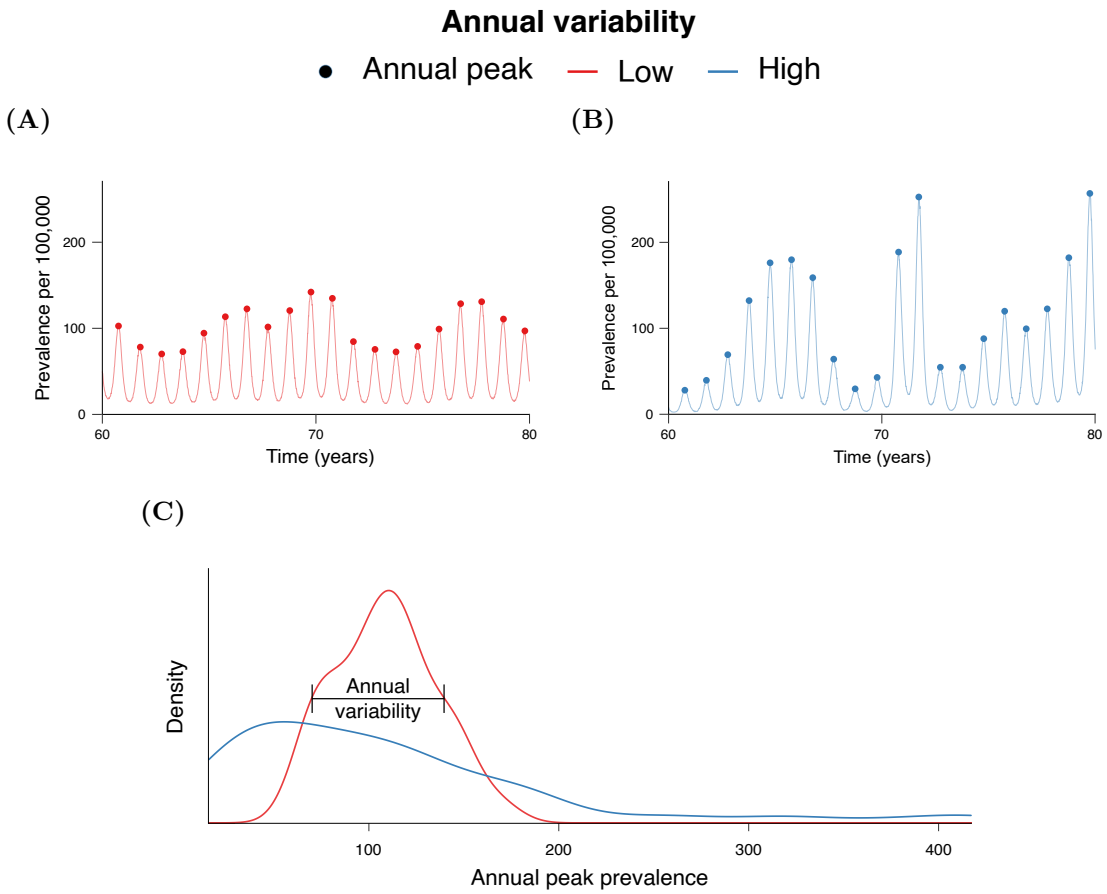

Figure 2.10. Measuring the annual variability of dengue outbreaks. Comparison of time series with low annual variability and high annual variability. An area where dengue is an epidemic has much greater annual variation, whereas a region where dengue is endemic has reduced annual variation. **(A)** High annual variation. **(B)** Low annual variation. **(C)** Distribution of the peaks of annual outbreaks. The width of the distribution visually indicates the level of annual variation.

Age of infection

Age of infection was defined as the median age at which individuals acquire each heterotypic (first, second, third and fourth) infection.

Inter-epidemic period

Inter-epidemic period was defined as the time between outbreaks of relatively high incidence (Figure 2.12A). That is, the mean consecutive time that the low-frequency oscillations are below mean prevalence, where the low frequency oscillations were obtained by passing the time series of total prevalence through the Butterworth filter. The Butterworth filter is a low-pass filter, which reduce frequencies of a signal above a given cut-off frequency, defined

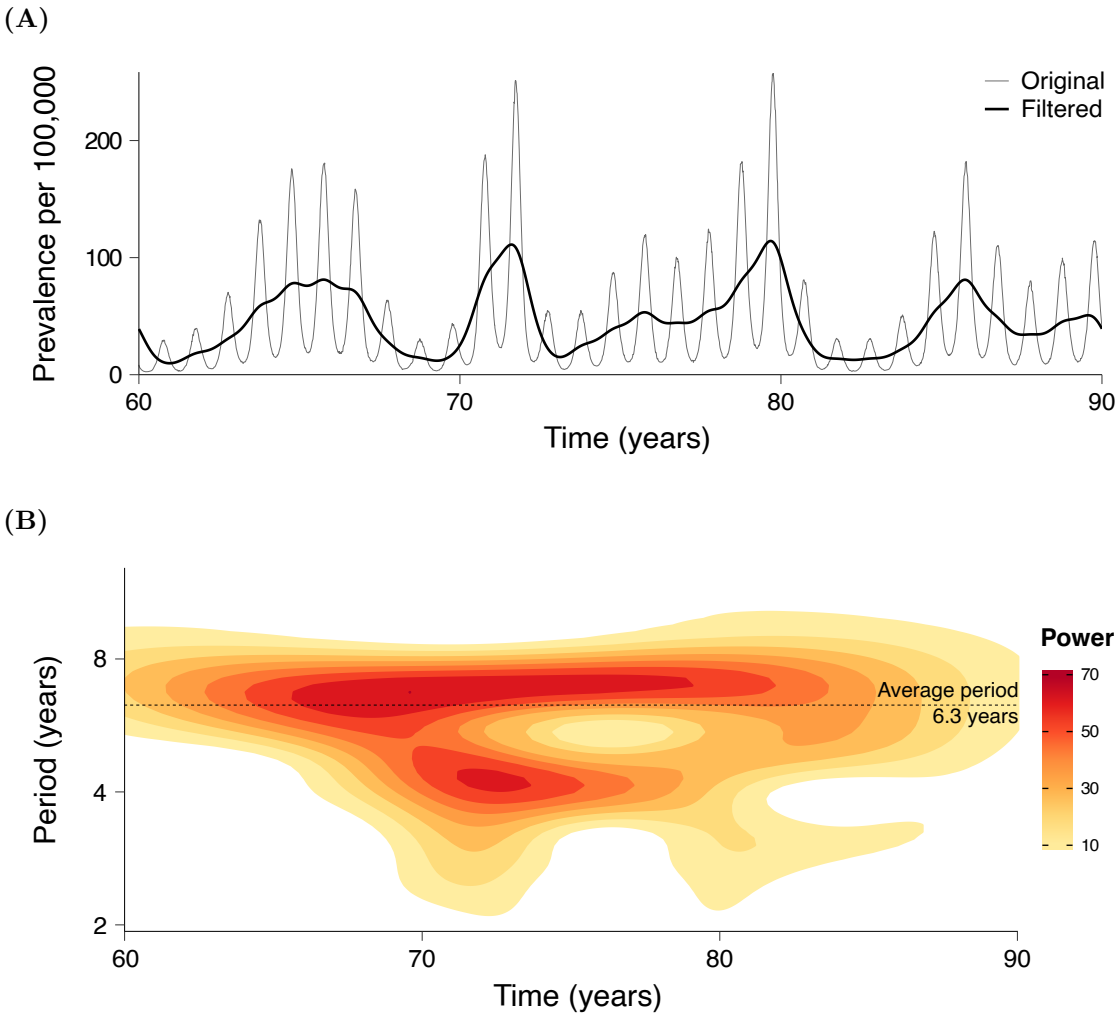

Figure 2.11. Time series filtering and wavelet analysis. The time series of total prevalence per 100,000 individuals over 30 years is filtered and periodicity calculated. **(A)** The original time series (dashed) is passed through a Butterworth filter of order 5 with cut-off frequency 0.5 years^{-1} (solid). The chosen cut-off frequency filters out the annual oscillations to obtain the long-term behaviour of the time series. **(B)** The Wavelet Power Spectrum is computed for the filtered time series. The average period of the time series is 6.3 years.

as

$$G(f) = \frac{1}{\sqrt{1 + \left(\frac{f}{f_c}\right)^{2n}}},$$

where f is the signal frequency, f_c is the cut-off frequency, n is the order of the filter, and $G(f)$ is known as the gain of the frequency (Proakis and Manolakis, 1996). The gain is equivalent to the fraction of each frequency that is preserved when passed through the filter. Each time series was transformed to the frequency domain using the Discrete Fourier Transform (DFT), then each frequency was multiplied by the corresponding gain and

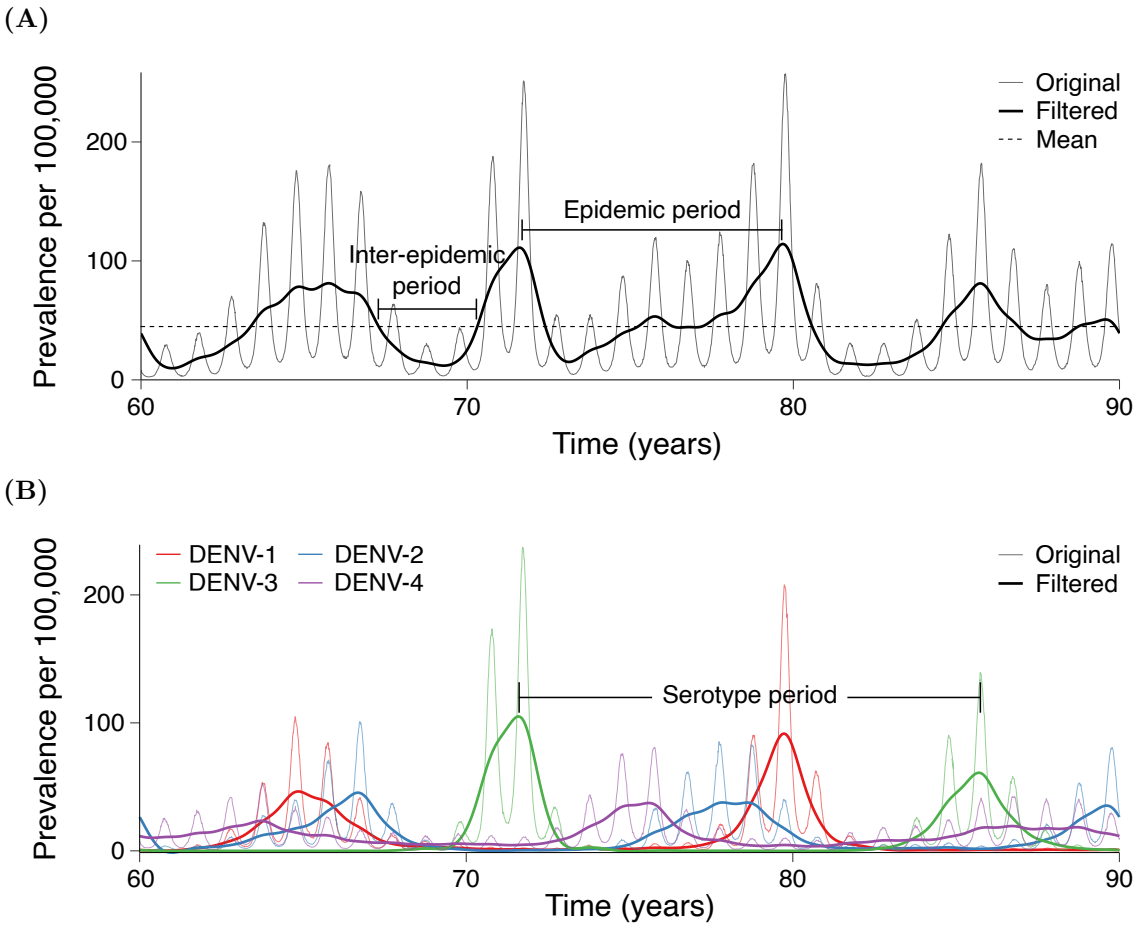

Figure 2.12. Measuring the periodic behaviour of dengue. A visualization of the periodicity metrics. The time series for each serotype and total prevalence is filtered using the Butterworth filter. The serotype and epidemic period are the times for the filtered time series to complete one cycle for serotype and total prevalence respectively. The inter-epidemic period is the time between epidemic outbreaks. **(A)** Serotype period. **(B)** Epidemic and inter-epidemic period.

finally transformed back into the time domain using the Inverse Discrete Fourier Transform (IDFT). Here, the cut-off frequency, f_c, and the order of the filter, n, were set to 2 years and 5, respectively, to ensure attenuation of annual frequencies.

Epidemic period

Epidemic period was defined as the mean time for the low frequency oscillations of total prevalence to repeat (Figure 2.12A), measured using Wavelet analysis (Percival and Walden, 2000). We used the Mortlet Wavelet, defined as the product of a sinusoid and a normal distribution. Convolving Mortlet Wavelets of different frequencies and shifts with the dengue time series generated wavelet coefficients in the time domain, which were then squared and corrected (Liu et al., 2007) to get the Wavelet Power Spectrum (WPS). The WPS was used to indicate the strength of each period at every time point in the time

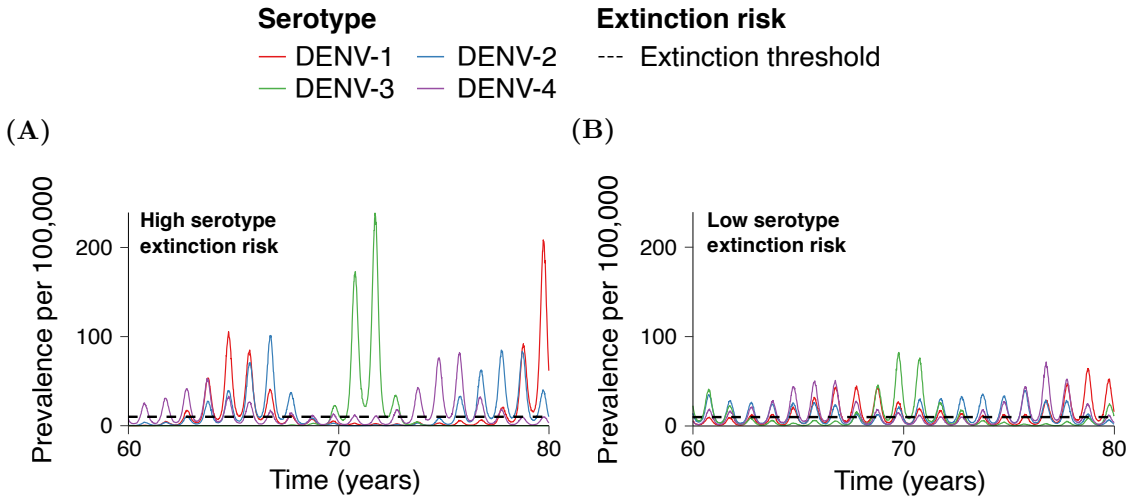

Figure 2.13. Measuring the extinction risk of dengue serotypes. The serotype extinction risk is the percentage of time serotype prevalence is above the extinction risk threshold (dashed). Stable annual oscillations permit serotypes to persist through each off-season, whereas large epidemic outbreaks create a high likelihood that a serotype will become temporarily extinct. **(A)** High serotype extinction risk. **(B)** Low serotype extinction risk.

series (Figure 2.11B). The overall periodicity for a dengue time series was defined as the weighted average of periods and the mean power of each period across time.

Serotype period

Similarly, serotype period was defined as the average time across each serotype for the low frequency oscillations of sero-specific prevalence to repeat (Figure 2.12B), measured from the WPS.

Serotype extinction risk

Serotype extinction risk was defined as the likelihood that a serotype will become extinct, thus relying on external introduction to be reintroduced into circulation. This was measured as the mean percentage of time at which a serotype is present in less than 10 individuals (Figure 2.13).

Serotype co-circulation

Serotype co-circulation indicated a measure of the mixing of serotypes among communities within the lattice. Co-circulation of each serotype was defined as the percentage of time of which there were concurrent infections of more than one serotype in a randomly chosen community.

2.2.4 Model parameters

Table 2.2 provides an overview of the parameters and parameter values used throughout this work (unless stated otherwise). The values were chosen to reflect the epidemiological dynamics of dengue in an endemic urban setting. Human demographic parameters were selected to best represent the demography of Brazil as reported in (United Nations, 2015). An age-dependent vector mortality rate was chosen motivated lab and field studies by (Harrington et al., 2008) and (Styer et al., 2007). Parameters for vector mortality were chosen to give a mean life expectancy of approximately 3 weeks (Mohamed et al., 2013; Yang et al., 2009). The vector to human ratio and its variability throughout the year are challenging to empirically estimate so parameters were selected based on previous modelling work (Liu-Helmersson et al., 2016; Lourenço and Recker, 2013). Incubation periods were chosen based on a lab based study conducted by (Chan and Johansson, 2012). The parameter for amplitude of oscillations in the extrinsic incubation period was selected to represent variability due to temperature (Chan and Johansson, 2012). The mean daily biting rate of mosquitoes was chosen based on the length of the gonotrophic (reproductive) cycle, 3–7 days (Goindin et al., 2015), and number of blood meals required to complete the gonotrophic cycle, 1–3 (Scott et al., 1993b). Given the wide range of infectivity and transmissibility probabilities for dengue (Mordecai et al., 2017), the probability of successful viral transmission from hosts to vectors and vectors to hosts was chosen arbitrarily at 0.5 for each. Sensitivity analyses were performed on the local and long-distance dispersal of disease in order to better understand their epidemiological effects. The external introduction rate was sufficiently high to ensure reasonable persistence of all four serotypes, as in an endemic setting, however low enough that it would not dominate the epidemiological dynamics. Human population and community size were fixed at five million and 320 respectively.

Table 2.2. The default set of parameter values used in the simulation of the spatially-explicit individual based model.

Parameter	Description	Value
L	Host mortality location parameter	8×365
a_H	Infant host mortality shape parameter	0.4
b_H	Infant host mortality scale parameter	3.65×10^{-7}
c_H	Adult host mortality shape parameter	6
d_H	Adult host mortality scale parameter	75×365
c_V	Adult vector mortality shape parameter	4
d_V	Adult vector mortality scale parameter	23
M	Maximum vector to human ratio	1.2
m	Minimum vector to human ratio	0.8
$1/\gamma$	Mean recovery time	4 days
$1/\epsilon_H$	Mean intrinsic incubation period	6 days
$1/\epsilon_V$	Mean extrinsic incubation period	7 days
δ	Amplitude of extrinsic incubation period oscillations	2 days
β	Per day biting rate	0.6
p_H	Probability of successful viral transmission to hosts	0.5
p_V	Probability of successful viral transmission to vectors	0.5
σ	Local disease dispersal kernel standard deviation	4
ω	Long distance transmission probability	1×10^{-4}
ι	External introduction rate per day per strain	0.01
N_H	Human population size	5000000
$N_H/\lvert C\rvert$	Community size	320

2.3 General model behaviour

2.3.1 Spatio-temporal dynamics

In order to assess the model's potential for modelling dengue, and other arboviral diseases, we first qualitatively compared the simulation output with notified dengue case data. Figure 2.14 shows the typical time-series under default parameter settings, exhibiting the characteristic oscillations of dengue incidence and serotype prevalence. Simulations also exhibited large variations in dengue incidence across space, creating heterogeneous distributions in susceptibility. These in turn promoted the uneven spread of individual serotypes into regions of relatively low immunity, inducing strong spatio-temporal heterogeneity in dengue seroprevalence (Figure 2.15).

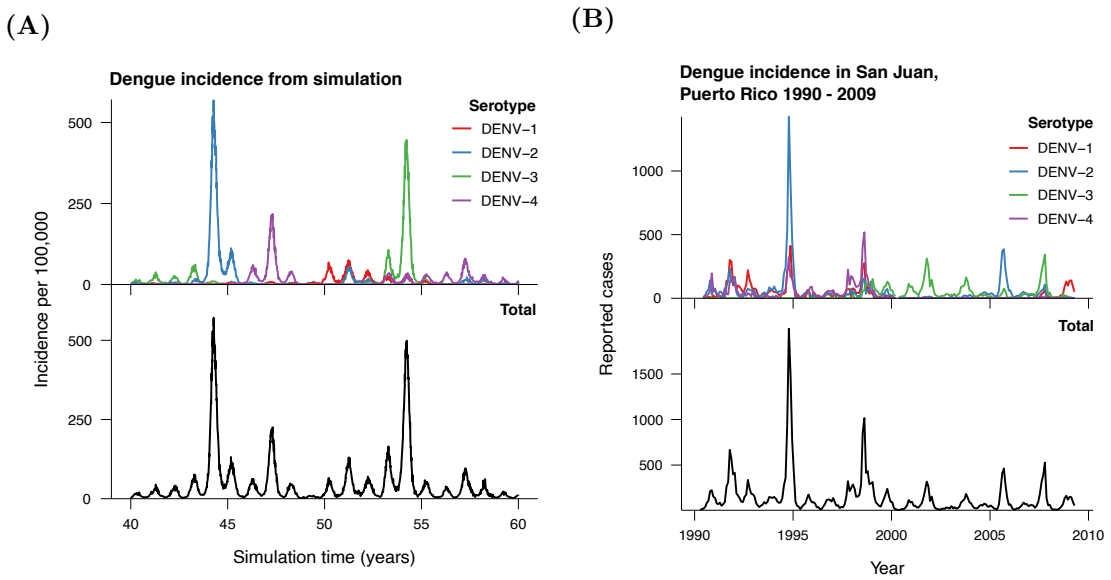

Figure 2.14. Simulated temporal dynamics. Simulated dynamics behaved similarly to empirical data: there are irregular epidemic outbreaks with sequential dominance of dengue's four serotypes. **(A)** Simulated dengue incidence per 100,000 individuals. **(B)** Reported cases of dengue in San Juan, Puerto Rico between 1990 and 2009. Furthermore, simulations exhibited strong spatio-temporal heterogeneity in susceptibility to each dengue serotype, where gradients in immunity across space facilitate virus spread into regions of relatively high susceptibility.

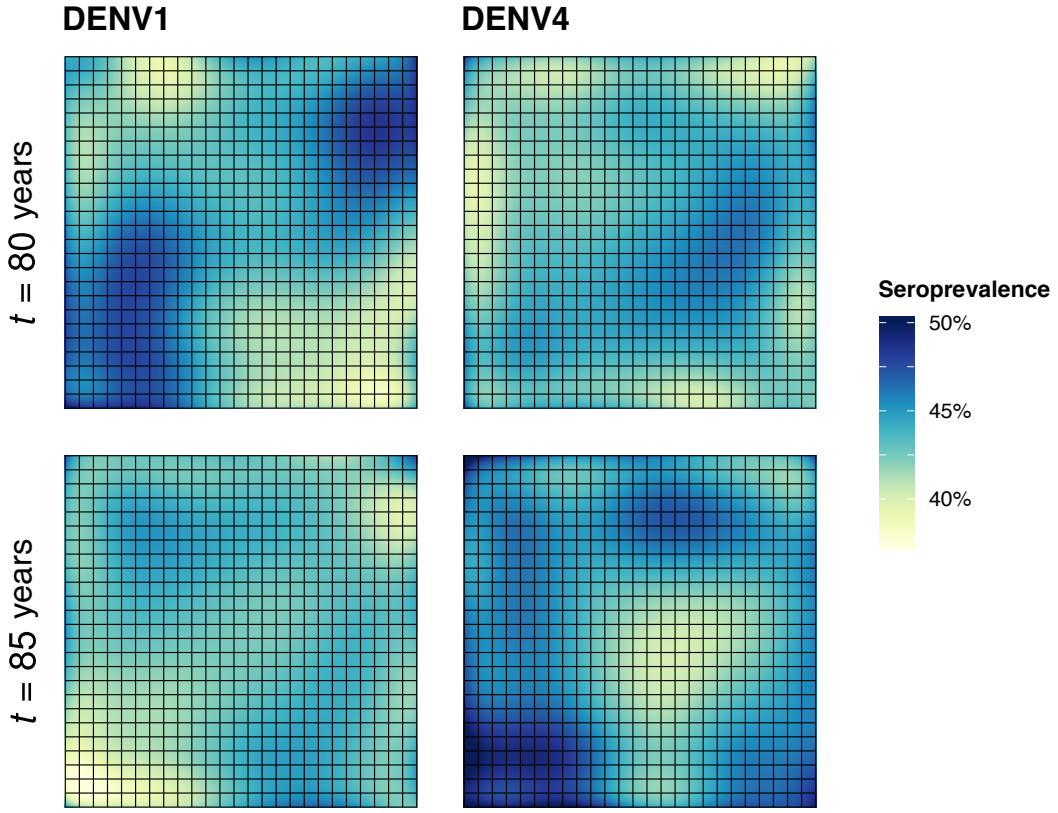

Figure 2.15. Spatio-temporal heterogeneity of dengue serotypes. The spatial landscape of seroprevalence to each serotype within each community of the lattice changes over time. Each grid square represents a group of 25 communities.

2.3.2 Epidemiological metrics under default settings

Each epidemiological metric was measured from simulations under default settings. Sensitivity analyses were then performed on virus transmissibility, $p_{\{H,V\}}$, community size, $N_H/|C|$, local mobility, σ, long-distance transmission, ω, and external introduction rates, ι, to understand the model's behaviour under changes to these critical parameters.

Based on 50 simulations run over 75 years, the mean daily prevalence of dengue was approximately 50 per 100,000 individuals. However, in each time series, we observed a large variability in incidence over time, highlighting the stochastic nature of the individual based model (Figure 2.16A).

The mean age of first, second, third and fourth dengue infection was 9, 21, 36 and 52 years, respectively (Figure 2.16B). Human immunity to dengue, or seroprevalence, increased alongside the age, with almost all individuals that were 50 years or older being immune to at least one dengue serotype (Figure 2.17A). Furthermore, the rate at which seroprevalence increased with age expectedly decreased with the number of novel exposures.

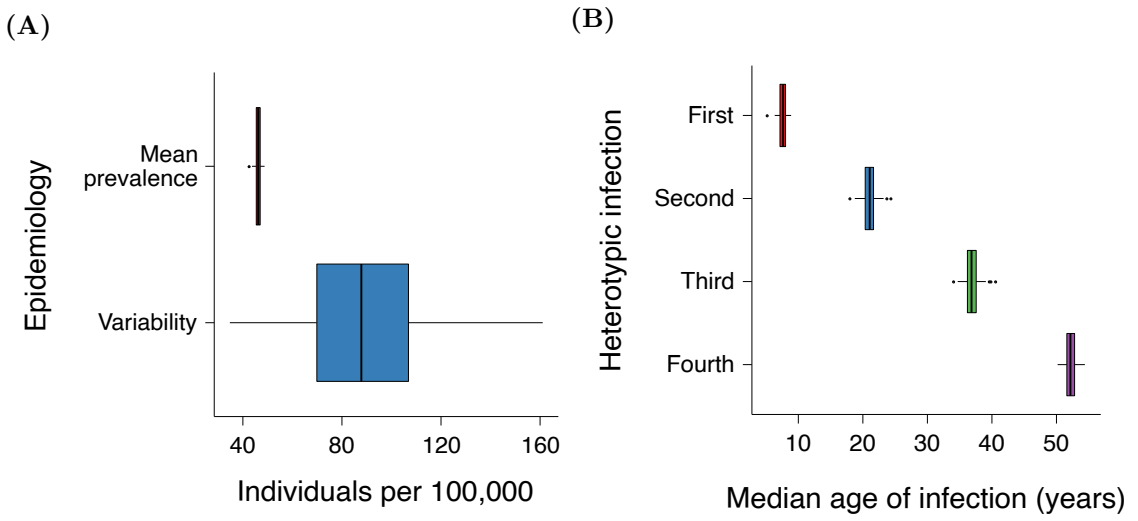

Figure 2.16. Epidemiological dynamics of the individual based model. The individual based model was executed 50 times for 75 simulation years, and the following epidemiological metrics were used to quantify epidemiological behaviour: **(A)** mean daily prevalence and variability in the magnitude of annual outbreaks, and **(B)** the median age of each novel dengue infection.

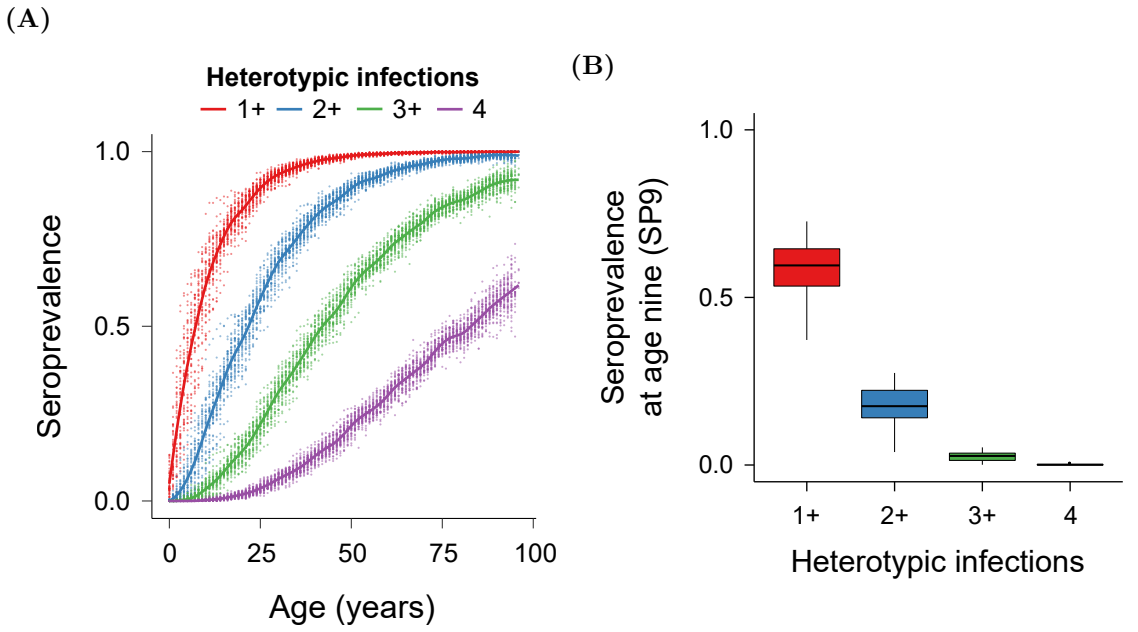

Figure 2.17. Population-wide seroprevalence. **(A)** The probability of individuals at every age year having been exposed to a first, second, third or fourth novel serotypes. Curves fitted using local regression (LOESS). **(B)** Seroprevalence at nine years old (SP9) for their first through fourth exposure event. Data was generated from 50 stochastic simulations.

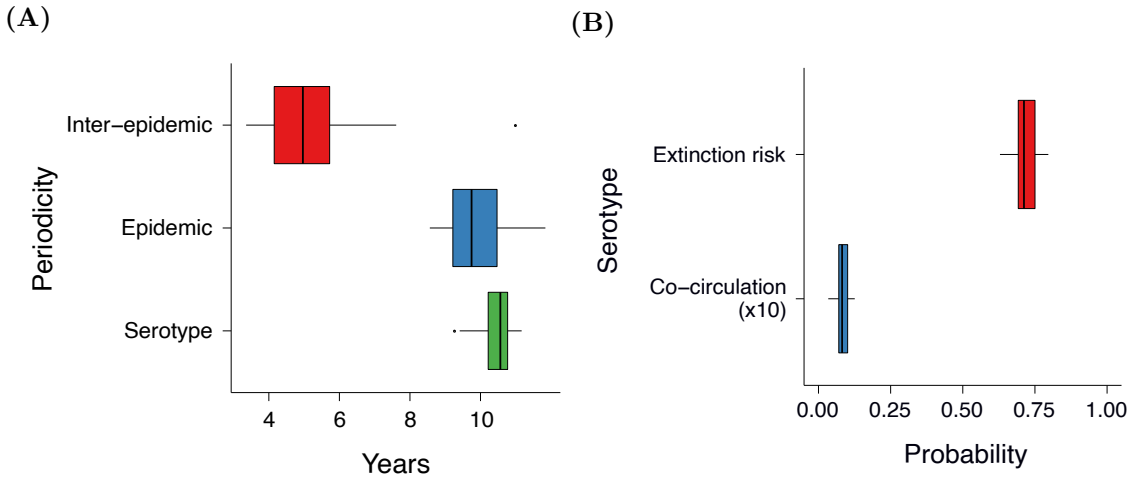

Figure 2.18. Epidemiological dynamics of the individual based model. The individual based model was executed 50 times for 75 simulation years, and the following epidemiological metrics were used to quantify epidemiological behaviour: **(A)** mean time between epidemics and long-term periodicity of (total and sero-specific) dengue prevalence, and **(B)** probability of serotype extinction and co-circulation of serotypes within a random community.

Mean seroprevalence levels by the age of 9 was just over 50%, and the variance in event-specific seroprevalence between the 50 stochastic simulations decreased with each heterotypic infection (Figure 2.17B). Note that immunity was only assessed at the end of the simulation. This induced variability in seroprevalence across simulations, dependent on where the simulation was stopped.

With the selected parameters in Table 2.2, epidemic and serotype periodicty was found to be between 9 to 11 years, with roughly an inter-epidemic period of only 5 years (Figure 2.18A). Dengue incidence was also very low in the off-season when mosquito density was low and the extrinsic incubation period was high. As a result, the mean extinction risk of each serotype was high, around 75%. Mirroring the high extinction risk, the likelihood of two or more serotypes co-circulating within a randomly selected community was extremely small, less than 1% (Figure 2.18B).

2.3.3 Effects of dengue transmissibility

To assess the suitability of our model further, we performed a sensitivity analysis on the probability of infection given a contact event between an infected and susceptible individual, $p_{\{H,V\}}$. Here, we assumed equal probabilities of human and mosquito infection. The output of each simulation was then quantified by means of the epidemiological metrics defined above.

Epidemiological dynamics

Higher transmission probabilities increased the number of transmission events caused by a single infected individual, which increased the mean prevalence. However, the increase in mean prevalence eventually reached a plateau, as immunity within the human population became saturated (Figure 2.19A). Additionally, annual variability initially rose as the transmissive capability of the virus increased, resulting in smaller, more stable, annual outbreaks. Alongside the increase in mean prevalence, the serotype prevalence and the median age of infection also increased (Figure 2.19B).

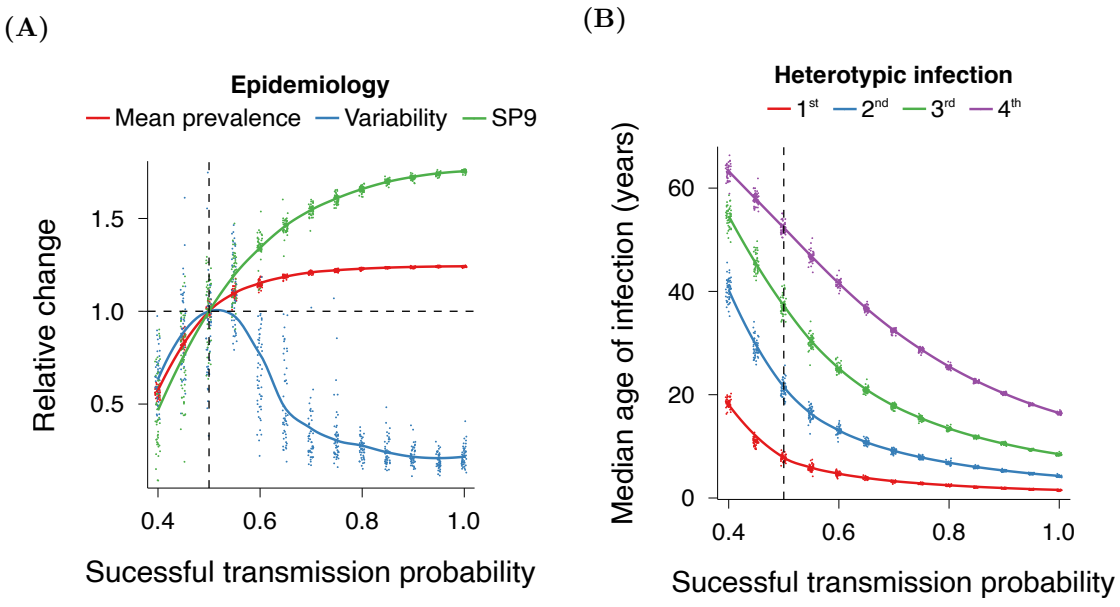

Figure 2.19. Effects of the probability of transmission success, $\mathbf{p_{\{H,V\}}}$, on dengue epidemiology. (A) The mean daily prevalence and seroprevalence at age nine increased with transmission probability, however epidemic variability decreased as annual outbreaks stabilised. **(B)** The median age of novel serotype exposure decreased with higher virus transmissibility. Each point indicates the metrics of one simulation, with 50 stochastic simulations per 13 tested transmission probability values. Local polynomial regression (LOESS) curves were fitted through the points, and vertical dashed lines indicate the baseline parameter value.

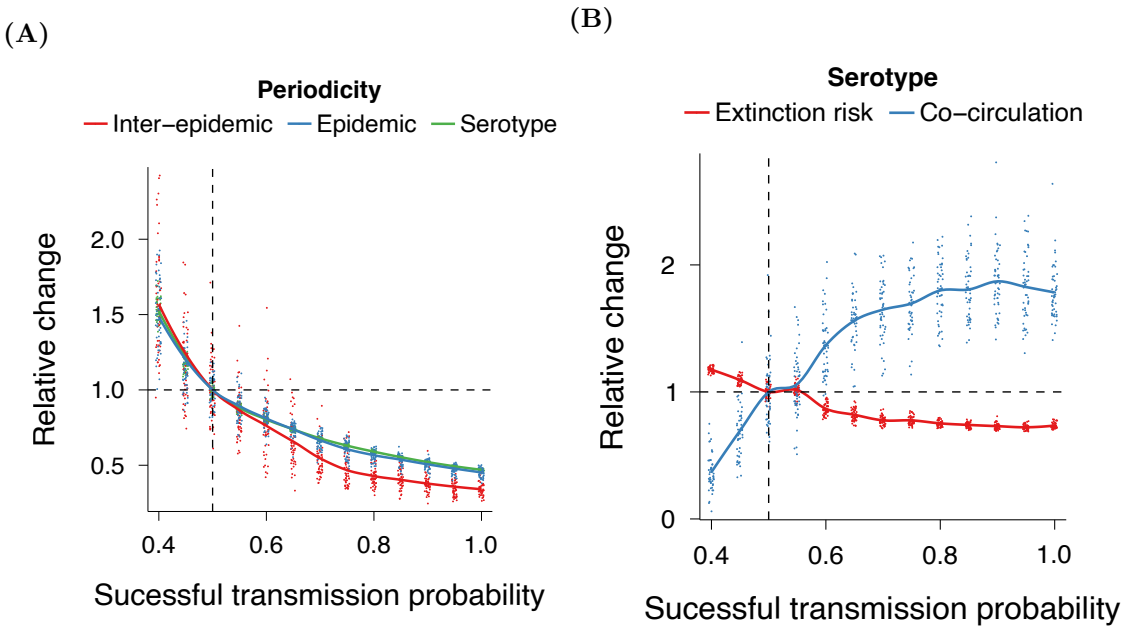

Figure 2.20. Effects of the probability of transmission success, $\mathbf{p}_{\{\mathbf{H,V}\}}$, on dengue periodicity and persistence. **(A)** The period of serotypes decreased with higher transmission probabilities, alongside shorter inter-epidemic periods. **(B)** The extinction risk and co-circulation of serotypes increased with higher dengue pathogeneicity. Each point indicates the metrics of one simulation, with 50 stochastic simulations per 13 tested transmission probability values. Local polynomial regression (LOESS) curves were fitted through the points, and vertical dashed lines indicate the baseline parameter value.

Inter-epidemic and serotype periodicity

All periodic behaviours decreased with higher transmission probability as it permits faster virus dissemination and re-invasion into populations of low susceptibility (Figure 2.20A). Alongside the dramatic decrease in annual outbreak variability, the period of time between epidemics approached one with higher transmission, suggesting that without seasonal forcing, incidence would be at an endemic equilibrium. Furthermore, the variation in the inter-epidemic period between simulations decreased with higher virus transmissibility for the inter-epidemic period as annual oscillations in prevalence became more stable.

Serotype extinction risk and co-circulation

Serotype extinction risk also decreased with higher transmission probabilities (Figure 2.20B). Reflecting this behaviour, the probability of two or more serotypes co-circulating within the same community increased and stabilised at higher levels of transmissibility.

2.3.4 Effects of community size

The effects of community size, $N_H/|C|$, on dengue epidemiology was explored by considering 16 different community sizes, including a model with only a single community. The output of each simulation was then quantified with selected epidemiological metrics (see Methods and materials).

Inter-epidemic and serotype periodicity

As community size was decreased, the inter-epidemic period marginally increased, and the variation between simulations greatly increased (Figure 2.21A). Furthermore, serotype and epidemic period greatly increased because of the greater restriction on pathogen access to susceptible areas of the community structure which resulted in slower spatial spread.

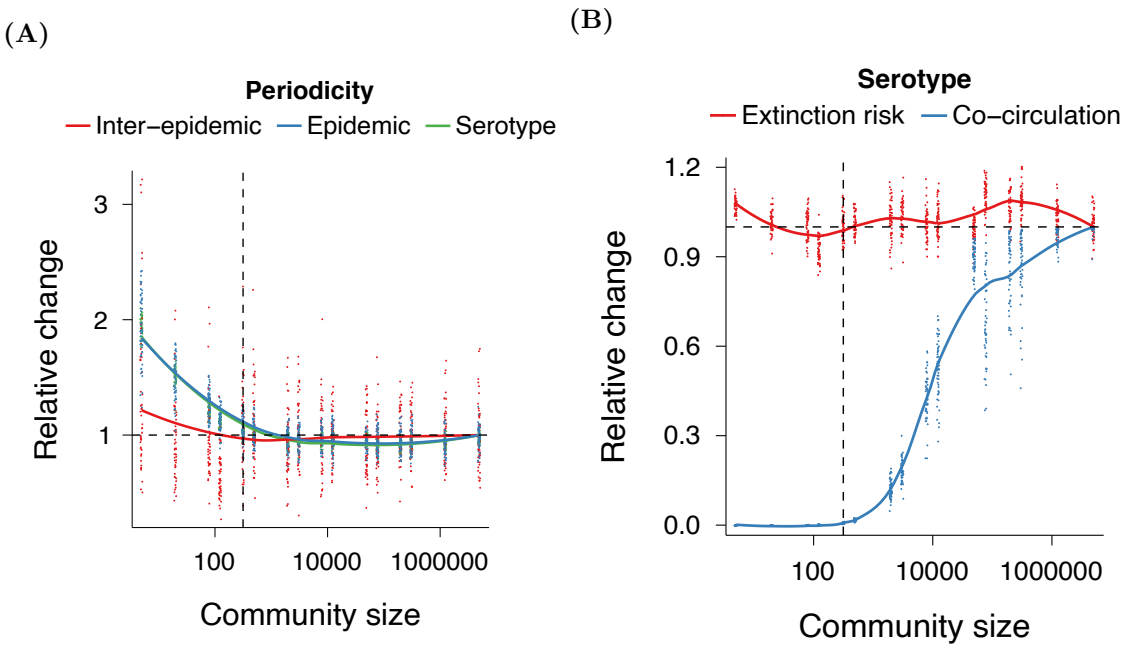

Figure 2.21. Effects of community size, $N_H/|C|$, on dengue periodicity and persistence. (A) The period of serotypes increased with reduced community size due to increased spatial constraints. Furthermore, **(B)** the co-circulation of serotypes greatly decreased with increasing lattice sizes, whereas serotype extinction risk was mostly unchanged. Each point indicates the metrics of one simulation, with 50 stochastic simulations per 16 tested community sizes. Local polynomial regression (LOESS) curves were fitted through the points, and vertical dashed lines indicate the default simulation parameter value. Here, the reference value for relative change was taken from the non-spatial (maximum community size) model.

Serotype extinction risk and co-circulation

Serotype extinction risk roughly mirrored the dynamics of annual outbreak variability. Higher annual variability coincides with larger epidemic outbreaks, which in turn cause more pronounced rises in population-wide seroprevalence and serotype extinction risk. However, with small community sizes, the chance for stochastic extinction was high (Figure 2.21B). Additionally, virus co-circulation dramatically decreased with community size because serotypes became spatially segregated from one another.

2.3.5 Effects of local mobility

To elucidate the role of local mobility on dengue epidemiology, a sensitivity analysis on the standard deviation of the local disease dispersal kernel, σ, was performed.

Epidemiological dynamics

Without local mobility, the virus could not persist, as virus propagation relied solely on random long distance transmission and small external introduction rates.

As soon as the virus was allowed to infect neighbouring communities, it was able to persist. Increased local mobility permitted the virus to spread further, however we found that the mean prevalence did not change (Figure 2.22A). This was because the transmissibility of the virus is unaffected by changes in mobility. In contrast, at reduced local mobility, seroprevalence at age nine was very low. This was due to the virus spreading more slowly across the lattice, requiring a longer period of time for the entire population

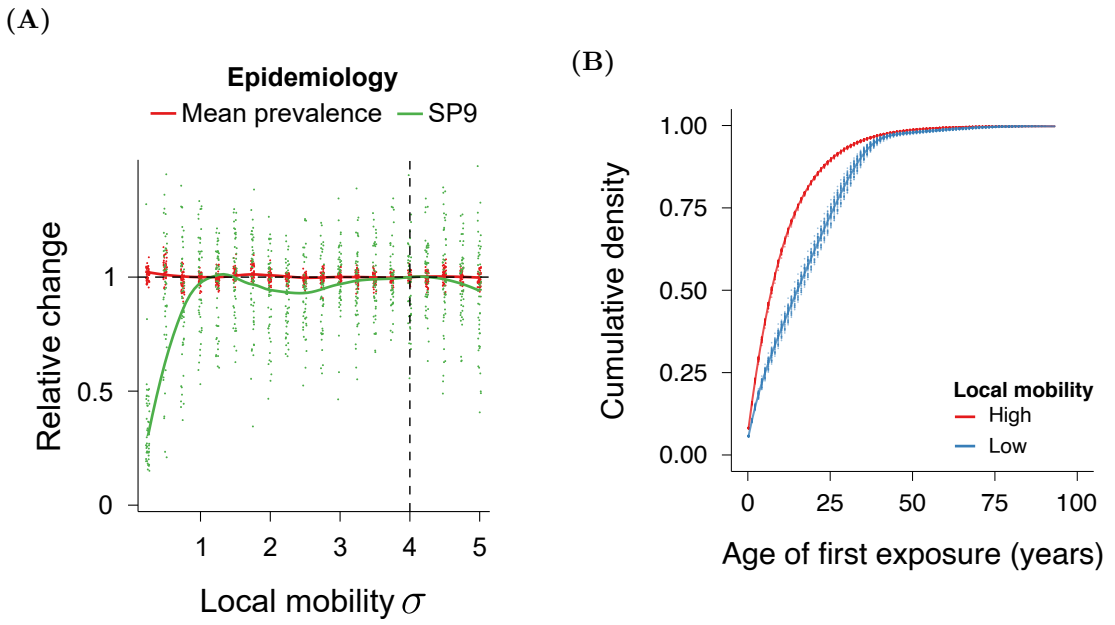

Figure 2.22. Effects of local mobility, σ, on dengue epidemiology. (A) Mean daily prevalence did not change as local mobility was introduced into the system, however at very low levels of mobility, the seroprevalence at age nine decreased, as the pathogen was unable to spread across the meta-population as rapidly. **(B)** Compared to high mobility ($\sigma = 4$), low local mobility ($\sigma = 0.25$) causes the pathogen to spread less rapidly, introducing a lag in the age at which individuals are first exposed to a serotype. Each point indicates the metrics of one simulation, with 50 stochastic simulations per 20 tested local mobility parameter values. Local polynomial regression (LOESS) curves were fitted through the points, and vertical dashed lines indicate the default simulation parameter value.

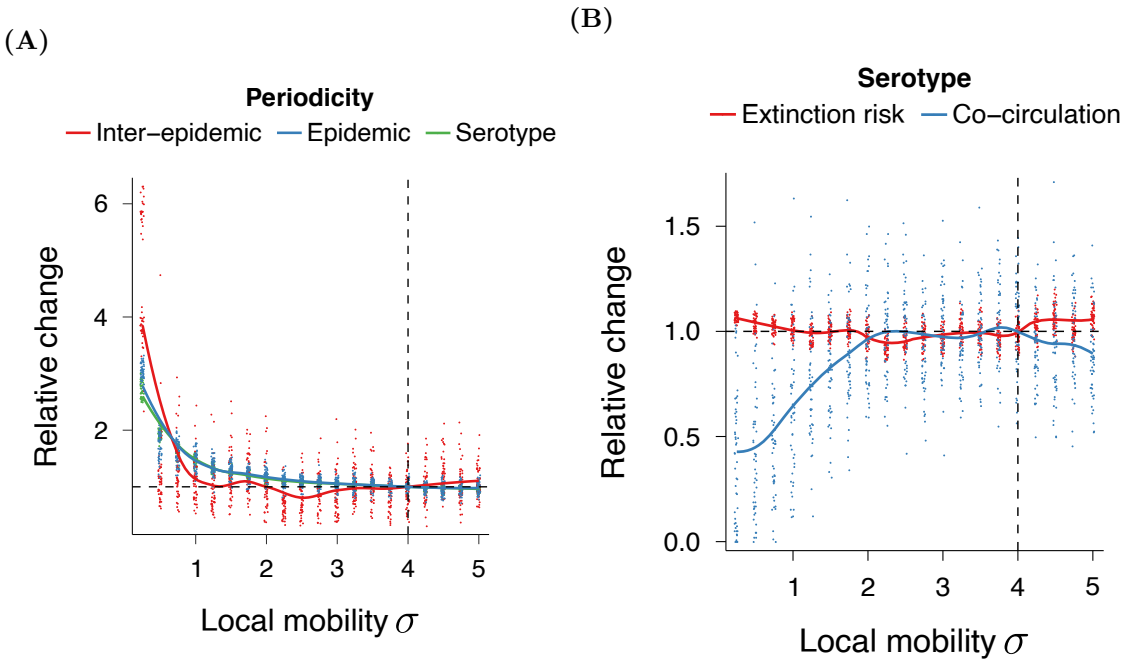

Figure 2.23. Effects of local mobility, σ, on dengue periodicity and persistence. **(A)** The periodic behaviours of dengue decreased with increased local mobility as greater spatial spread facilitated pathogen re-introduction. Furthermore, **(B)** the co-circulation of serotypes increased with higher mobility, whereas serotype extinction risk remained reasonably stable. Each point indicates the metrics of one simulation, with 50 stochastic simulations per 20 tested local mobility parameter values. Local polynomial regression (LOESS) curves were fitted through the points, and vertical dashed lines indicate the default simulation parameter value.

to become exposed. Therefore, the rate at which individuals were exposed was lower (Figure 2.22B), and so seroprevalence among nine year olds was smaller. However, upon consideration the entire population, we found that (poulation-wide) seroprevalence only marginally decreased at reduced local mobility levels.

Inter-epidemic and serotype periodicity

Inter-epidemic, epidemic and serotype period decreased with an increase in local mobility (Figure 2.23A). As the pathogen spread more easily, it would leave focal regions faster. In these regions, sufficient levels of susceptibility then accumulated earlier, and thus serotype re-introduction could occur sooner. Interestingly, at very low local mobility ($\sigma = 0.25$), the inter-epidemic period exhibited very large variance between four and six times the baseline value, or 12 to 18 years respectively. This is because epidemic periodicity increased to around 30 years, and thus within the 75 year simulation, often only a single inter-epidemic period could be calculated.

Serotype extinction risk and co-circulation

Extinction risk was barely influenced by changes in local mobility (Figure 2.23B). However, the co-circulation of serotypes was facilitated by an increase in local mobility, as wider dispersion of serotypes raised the likelihood of one community becoming infected with two or more serotypes.

2.3.6 Effects of long-distance transmission

In order to better understand the role of long-distance transmissions on the epidemiological behaviour of the individual based model, a sensitivity analysis on the probability that each transmission event is randomly dispersed throughout the lattice, ω, was performed.

Epidemiological dynamics

As long distance mobility is introduced, mean prevalence, observed seroprevalence (Figure 2.24A) and the median age of infection did not change, as transmissibility of the virus was unaltered. However, the variability between annual epidemic outbreaks increased with higher probability of long distance dispersion of transmission events, as the potential for the introduction of the pathogen into highly susceptible regions of the lattice rises, consequentially producing large epidemic outbreaks.

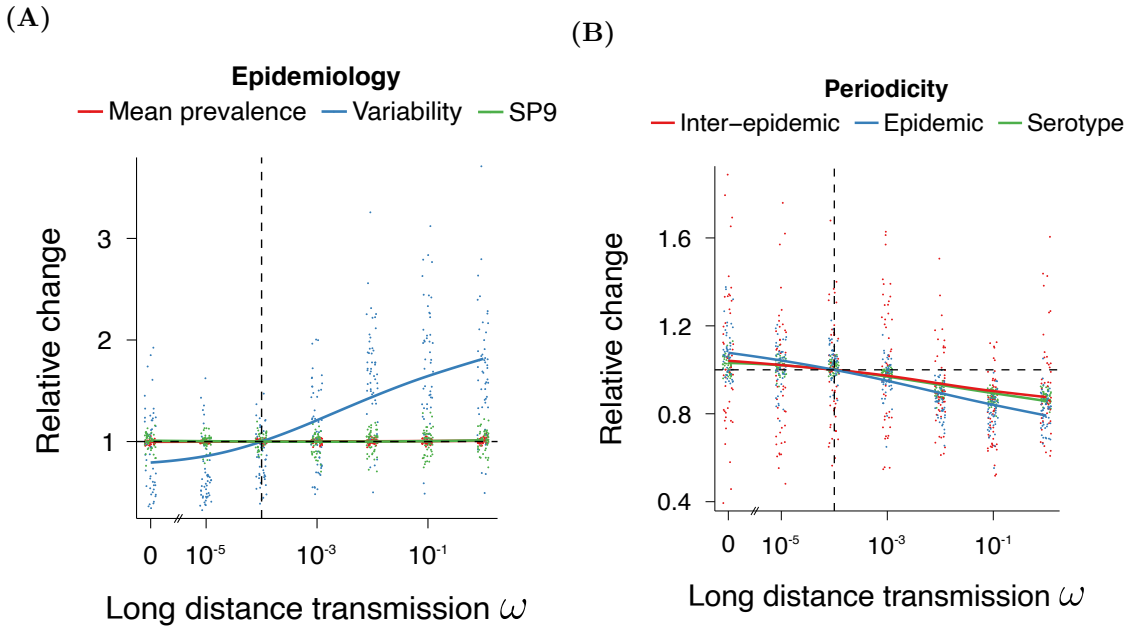

Figure 2.24. Effects of long-distance transmission, ω, on dengue epidemiology. **(A)** Mean daily prevalence and seroprevalence at age nine did not change as long-distance mobility was introduced, however higher long-distance mobility enabled pathogen introduction into highly susceptible regions of the lattice, increasing epidemic variability. **(B)** The periodic behaviours of dengue decreased with increased long-distance mobility as greater spatial spread facilitated pathogen re-introduction. Each point indicates the metrics of one simulation, with 50 stochastic simulations per 7 tested long-distance mobility parameter values. Local polynomial regression (LOESS) curves were fitted through the points, and vertical dashed lines indicate the default simulation parameter value.

Inter-epidemic and serotype periodicity

In alignment with the effects of local mobility on dengue periodicity, the inter-epidemic, epidemic and serotype period decreased with an increase in long-distance transmission (Figure 2.24B). As the pathogen is able to spread more readily, re-invasion into previously infected communities where immunity had waned sufficiently could occur faster, thus allowing serotype re-invasion to occur earlier.

2.4 Discussion

To describe the effects of ecological drivers on dengue epidemiology in detail, we extended a spatially-explicit individual based model previously proposed by Lourenço and Recker (2013). The computational complexity of the model was alleviated by implementing the model within a GPU-accelerated framework, reducing simulation run-times by just under a factor of 100. This then allowed us to assess the suitability of framework in capturing the epidemiology dynamics of dengue.

Under default parameter settings, the model captured the temporal qualitative behaviour of endemic dengue, with irregular epidemic outbreaks (Cuong et al., 2011; Johansson et al., 2009), 8–12 year cycles of serotype-specific prevalence (Nisalak et al., 2003), and spatio-temporal heterogeneity in dengue seroprevalence (Stewart-Ibarra et al., 2014). Furthermore, the simulated age of first and second infection, as well as seroprevalence levels in children, were in line with previous empirical studies in (highly) endemic regions (Garg et al., 2017; Lee et al., 2018; Rodríguez-Barraquer et al., 2014; Thai et al., 2011). There was also good agreement with previous theoretical work by Lourenço and Recker (2013) on the effects of dengue transmissibility and community size on simulated dynamics.

Human and mosquito movement have been implicated as important epidemiological drivers for the global emergence and local spread of dengue worldwide (Gubler, 2011), so we assessed model suitability by performing a detailed exploration into the effects of local and long-distance mobility on dengue epidemiology. We found that beyond small ranges, increasing the distance of local disease spread did not affect the overall epidemiological dynamics of dengue. These findings were in alignment with previous work on exploring the effects of different local disease dispersal kernels on dengue epidemiology (Barmak et al., 2011, 2016; Chao et al., 2013). We also found that higher rates of long-distance human movement increased the annual variability of outbreaks, as the virus was able to be introduced into regions of high susceptibility more frequently. This re-emphasised the importance of long-distance mobility in seeding large epidemic outbreaks in dengue-naïve regions (Tian et al., 2017). Overall, both of these analyses helped us to better understand the sensitivity of our model to movement patterns across different scales.

Other spatially detailed transmission models have provided insight into social human movement effects at small spatial scales (Chao et al., 2012; Karl et al., 2014; Perkins et al., 2014). However, in this framework, only a simple model for local movement was considered. We explored the epidemiological effects of mobility in a highly connected community structure, and it remains unclear how structured movement influences dengue epidemiological dynamics across larger regions. Mosquito and human density was also assumed to be homogeneous across space, however heterogeneity in population size has been demonstrated to correlate highly with dengue incidence patterns (Sirisena et al., 2017). Therefore, the presented spatial model was modified to include different community structures and heterogeneity in human and mosquito population density (see Chapter 4). Finally, only a simple model for the influence of climate on dengue was considered here. The framework could easily be extended to more accurately capture the seasonal oscillations influencing vector suitability and virus transmissibility (Caminade et al., 2017; Johansson et al., 2009; Li et al., 2019) (see Chapter 5).

As demonstrated here the model is suitable in capturing the epidemiological dynamics of endemic dengue. The effects of different local and large-scale movement patterns was further re-emphasised within this framework, which allows a more detailed investigation into the effects of spatio-temporal heterogeneities on dengue epidemiology, such as the influence of community structure and ecological heterogeneities on the spread and persistence of the virus (see Chapter 4). Furthermore, the reduced simulation run-times of framework allow the inherently expensive process of model fitting to epidemiological data within a fully Bayesian framework to be considered for the first time (see Chapter 5), which in turn can then be used to better assess the impacts of intervention strategies such as vaccine deployment programs (Flasche et al., 2016; Lourenço and Recker, 2016).

Chapter 3

Robustness of the reproduction number estimates in vector-borne disease systems

3.1 Introduction

The dependence on insects for transmission between vertebrate hosts has a number of important implications. First, they are frequently subject to strong spatial and temporal fluctuations due to environmental and climatic variations, such as seasonality in rainfall or temperature. Second, these pathogens should be amenable to vector control. That is, disease transmission can, at least in theory, be interrupted simply by removing the insect vector (e.g. use of insecticides) or by preventing contact between the vector and the host (e.g. use of bednets). Furthermore, it has been suggested that only a fraction of insects need to be removed or vector-host contacts to be prevented for the disease to die out. This concept is largely based on mathematical theory that can be traced back to the first formal description and mathematical treatment of the malaria life-cycle by Ross (Ross, 1911). Unfortunately, translating theoretical predictions to practical applications, especially with regards to disease elimination through vector control, has only resulted in partial success.

The epidemiological reasoning behind the theory relies on a particular threshold condition involving the so-called basic reproduction number, R_0, which denotes the expected number of secondary cases arising from a single infection in a totally susceptible population (Heesterbeek and Dietz, 1996). To date, R_0 is frequently used either to predict the extent

of an epidemic outbreak or to derive the necessary conditions to prevent this outbreak from happening, e.g. by means of vaccination. The crux of the problem is how to robustly derive or estimate this number in the first place. Compartmentalised systems of ordinary differential equations (ODEs) have been in use for decades to understand infectious diseases at the population level and provide the backbone for most formulas for R_0 (Reiner et al., 2013). These allow the reproduction number to be computed either exclusively using empirically informed parameter estimates or from the initial growth rate of an outbreak (Dietz, 1993). Although the latter is the more common approximation method for directly-transmitted disease (Breban et al., 2007; Cairns, 1991; Chowell et al., 2004; Lipsitch et al., 2003), it has equally been applied to vector-borne pathogens (Chowell et al., 2007; Massad et al., 1996; Towers et al., 2016).

An important consideration for R_0 estimates of vector-borne pathogens is that these can vary substantially across space and time. For example, reported R_0 estimates for the complete transmission cycle of *Plasmodium falciparum* in Africa range from 1 to more than 3,000 (Bousema et al., 2012; Smith et al., 2007). Based on nine epidemics in Brazil between 1996 and 2003, the reproduction number for dengue has been estimated to be somewhere between 2 to 103 (Favier et al., 2006), and median estimates for Zika range between 2.6–4.8 in French Polynesia (Kucharski et al., 2016) and 4–9 in Rio de Janiero (Villela et al., 2017). The reasons for such wide variations are manifold. As mentioned earlier, the dependence on insect vectors for transmission can naturally introduce large spatio-temporal heterogeneities. That is, a disease introduced during the dry season will behave very differently to the same disease being introduced during the rainy seasons. Equally, an outbreak in a densely populated urban area will likely take a different course than an outbreak in a sparsely populated rural area. Here we argue that in addition to these natural variations and potential differences in data collection and analyses, the actual methodologies used to derive R_0 estimates can also introduce substantial discrepancies.

A crucial component of the reproduction number for a vector-borne disease is the mean time that an infected vector is able to transmit to a host, or the infectious vector-to-host transmission period (VHTP) (Mendes Luz et al., 2003). As infectious vectors are assumed to continue to transmit the disease until death, the VHTP is determined both by the

life expectancy of the vector and the extrinsic incubation period of the pathogen. For mathematical simplicity, most epidemiological models of vector-borne diseases assume that vectors have a constant (daily) mortality rate. However, this assumption is in stark contrast to findings from lab-based and field mark-and-recapture studies. For example, survival probabilities of the dengue mosquito vector *Aedes aegypti* and the principal malaria vectors *Anopheles stephensi* and *An. gambia* have been shown to be strongly age-dependent (Dawes et al., 2009; Harrington et al., 2008; Hugo et al., 2014; Styer et al., 2007). Although it should be clear that current lab and field-based studies of vector survivorship come with their own set of limitations and uncertainties, constant, i.e. age-independent mortality rates are biologically less likely than assuming a general decrease in the survival probability with age.

Previous work has looked into the effects of logistic mortality rates on the vectorial capacity, the mosquito-related components of R_0 (Bellan, 2010). However, the effects of assuming constant vector mortality on R_0 in a system where death rates are strongly age-dependent have not yet been explored. In this chapter, we compared a commonly used R_0 formula based on continuous-time differential equation model using constant mortality rates to an R_0 estimate derived from first principles under relaxed assumptions about vector mortality. Using the stochastic, individual-based simulation model (IBM) introduced in Chapter 2, which permits the direct measurement of the average number of secondary cases, we demonstrate how the underlying assumptions of vector survivorship can significantly inflate R_0 estimates. We further show how estimates based on endemic equilibria are generally more robust and derive a correction factor to ameliorate R_0-inflations in estimation methods based on epidemic growth curves.

3.2 Methods

3.2.1 Modelling frameworks

We derived R_0 estimates from two different epidemiological frameworks: (i) a simple, single-strain vector-borne disease model based on ordinary differential equations (ODE), where vector mortality is assumed to be constant, leading to an exponentially distributed vector survivorship, and (ii) a stochastic individual-based model (IBM), which permits more explicit control over the demographic processes regulating birth and death rates (see Chapter 2).

ODE model

The classical ODE approach to model infectious diseases is obtained by dividing the population into those that are susceptible (S), exposed but not yet infectious (E), infectious (I) and recovered (R). The same principle is then applied to extend these models to vector-transmitted diseases, except for the fact that vectors usually do not recover from infection but are instead assumed to remain infectious until death. This model can be realised by the following set of differential equations

$$
\begin{aligned}
\frac{\mathrm{d}S_H}{\mathrm{d}t} &= \mu_H N_H - p_H \beta I_V \frac{S_H}{N_H} - \mu_H S_H \\
\frac{\mathrm{d}E_H}{\mathrm{d}t} &= p_H \beta I_V \frac{S_H}{N_H} - \epsilon_H E_H - \mu_H E_H \\
\frac{\mathrm{d}I_H}{\mathrm{d}t} &= \epsilon_H E_H - \gamma I_H - \mu_H I_H \\
\frac{\mathrm{d}R_H}{\mathrm{d}t} &= \gamma I_H - \mu_H R_H \\
\\
\frac{\mathrm{d}S_V}{\mathrm{d}t} &= \mu_V N_V - p_V \beta S_V \frac{I_H}{N_H} - \mu_V S_V \\
\frac{\mathrm{d}E_V}{\mathrm{d}t} &= p_V \beta S_V \frac{I_H}{N_H} - \epsilon_V E_V - \mu_V E_V \\
\frac{\mathrm{d}I_V}{\mathrm{d}t} &= \epsilon_V E_V - \mu_V I_V
\end{aligned}
$$

Here, $1/\mu_{H,V}$ are the mean host and vector life expectancies; β is the daily biting rate; $p_{H,V}$ are the per-bite transmission probabilities from vector to human and human to vector,

respectively; $1/\epsilon_{H,V}$ are the incubation periods in the host and vector, respectively; and $1/\gamma$ is the mean infectious period in the host. This model is illustrated by means of a flow diagram in Figure 3.1A.

Stochastic IBM

The individual based model used here was that described in Chapter 2 with seasonal oscillations in extrinsic incubation period and mosquito density removed. For simplicity, only one virus strain was considered as the simultaneous invasion of multiple serotypes is unlikely. Here, we also neglected infant mortality.

3.2.2 Estimation of R_0

ODE-based reproduction number

We derived the R_0 estimates from the ODE model by applying the next generation approach (Diekmann et al., 2010), which relates the number of newly infected individuals in the compartments in consecutive generations to one another, yielding

$$R_0 = \frac{Mp_H p_V \beta^2}{(\gamma + \mu_H)} \frac{1}{\mu_V} \frac{\epsilon_H}{\epsilon_H + \mu_H} \frac{\epsilon_V}{\epsilon_V + \mu_V}$$

where M is the vector:host ratio ($N_V : N_H$) and $\frac{\epsilon_H}{\epsilon_H+\mu_H}$ and $\frac{\epsilon_V}{\epsilon_V+\mu_V}$ are the probabilities of hosts and vectors surviving the intrinsic and extrinsic incubation period of the pathogen respectively.

For many vector-borne disease systems, such as malaria and dengue, both the human recovery rate, $1/\gamma$, and the intrinsic incubation period $1/\epsilon_H$, are much shorter than the mean human life expectancy, $1/\mu_H$. We can therefore make the following approximation of the above formula, which, here, we define as R_0^{ODE}.

$$R_0^{\mathrm{ODE}} = \frac{Mp_H p_V \beta^2}{\gamma} \frac{1}{\mu_V} \frac{\epsilon_V}{\epsilon_V + \mu_V}$$

IBM-based reproduction number

The basic reproduction number for the individual-based model, R_0^{IBM}, was derived from first principles using the transmission cycle of the pathogen, similar to (Smith et al., 2007)

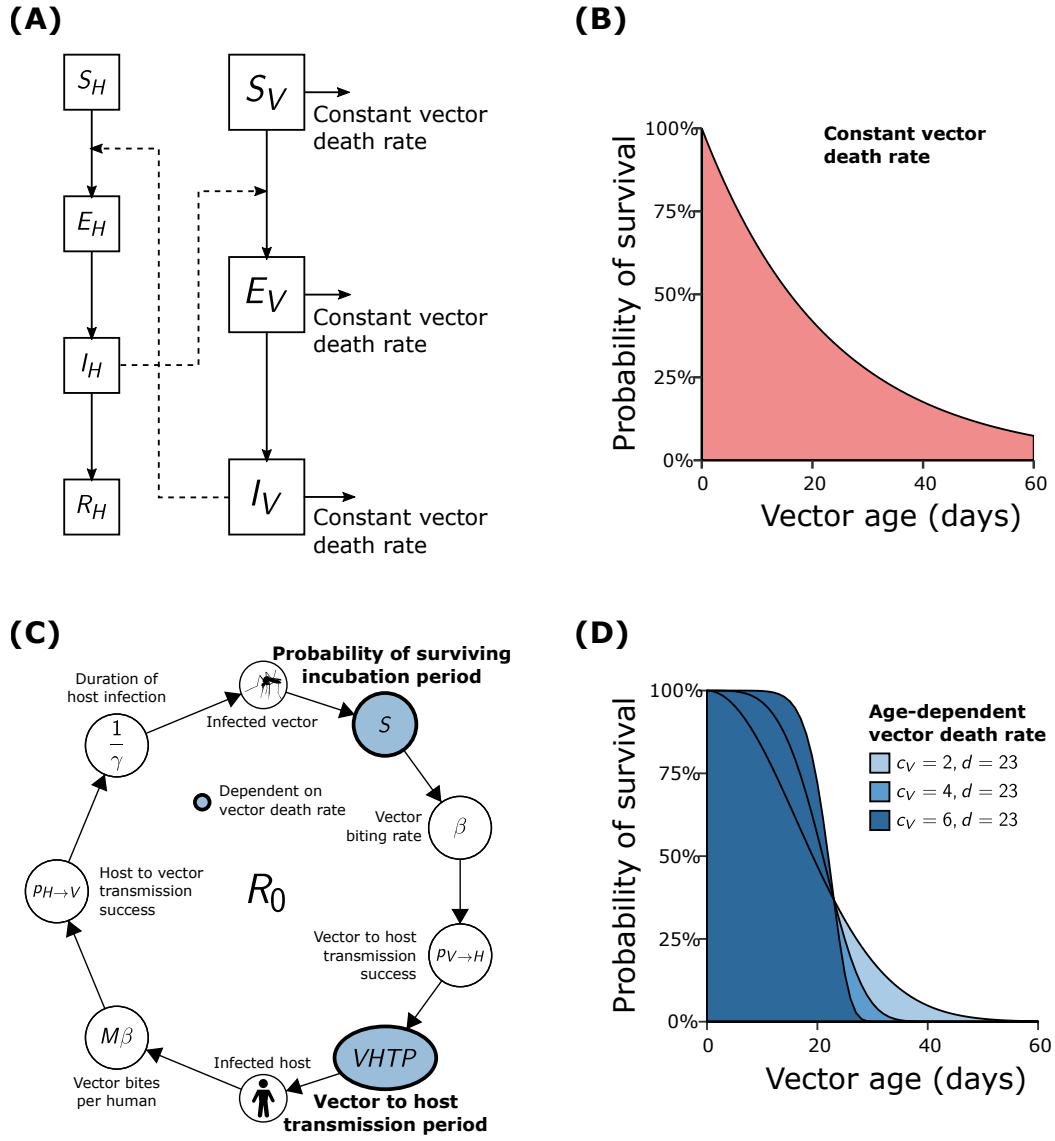

Figure 3.1. Comparison of flow diagrams and vector death rates between the ODE and IBM frameworks. (A) The compartmentalised system of differential equations for a vector-borne pathogen assumes constant vector mortality rates from each state of infection. **(B)** Constant vector mortality rates result in exponential age distribution of vectors (with $1/mu_V = 21.3$), with a high proportion of individuals living far beyond their life-expectancy. **(C)** The transmission cycle of a vector-borne pathogen used in the individual-based model highlighting the dependency of the infectious period and the probability of surviving the extrinsic incubation period on the mortality rate of vectors. **(D)** The age distribution of vectors under three different Weibull distributed mortality risks with an increasing dependency on vector age (light blue to dark blue with $c_V = 2, 4, 6$ and $d_V = 23$).

(Figure 3.1C). Starting with an infected host in an entirely susceptible population, this individual will infect on average $Mp_V\beta$ vectors per day and will remain infected for $1/\gamma$ days. Therefore, a single infected host is expected to infect a total of $Mp_V\beta/\gamma$ vectors.

A single infected vector will infect on average $p_H\beta$ hosts per day (in a totally susceptible population). As vectors remain infectious for the rest of their lives, the infectious period is defined as the difference between the mean life expectancy, $1/\mu_{I_V}$, and the mean age at which a vector becomes infectious, $1/\delta_{I_V}$, meaning that an infectious vector will have $(1/\mu_{I_V} - 1/\delta_{I_V})$ days to infect hosts.

Furthermore, the proportion of infected vectors that survive the extrinsic incubation period, denoted by $\rho_{E_V \to I_V}$, also depends on the vector mortality risk. Combining all these terms, the basic reproduction number of the individual based model can be derived as

$$R_0^{\text{IBM}} = \frac{Mp_H p_V \beta^2}{\gamma} \left(\frac{1}{\mu_{I_V}} - \frac{1}{\delta_{I_V}} \right) \rho_{E_V \to I_V}$$

Note, the first term is identical to the first term of R_0^{ODE}. However, the second term, which denotes the infectious period of the vector, and the third term, which denotes the probability of vectors surviving the incubation period, differ between R_0^{ODE} and R_0^{IBM}. This is because the formula for the reproduction number derived from the transmission cycle in the individual based model takes into account alternative (Weibull distributed) vector mortality risks, whereas the ODE system assumes a constant mortality rate (an exponential distributed mortality risk).

Timeseries-based reproduction number

In addition to the direct R_0 formulas derived above, we also considered two common methods to estimate the reproduction number from timeseries data: one approach based on the initial growth rate of an epidemic outbreak (Figure 3.2A), and one based on the dynamic equilibrium of an endemic scenario (3.2B). Both epidemic and endemic cases were simulated using the individual based model.

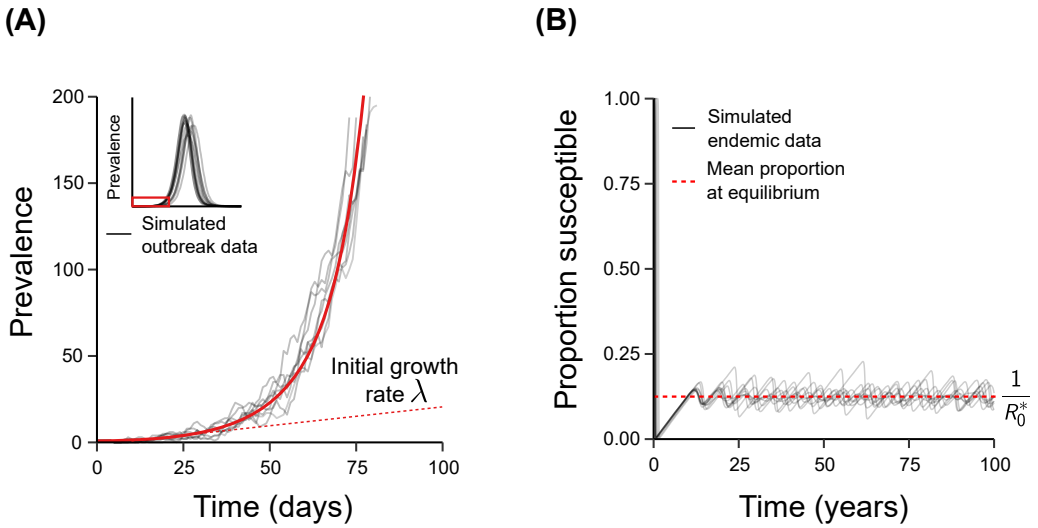

Figure 3.2. Estimating R_0 from empirical data. **(A)** The reproduction number can be estimated from epidemic outbreak data assuming an initially exponential growth rate, λ. **(B)** The dynamic equilibrium of susceptible individuals in a population can also be used to estimate R_0.

Epidemic growth rate

The epidemic outbreak method for estimating the reproduction number requires timeseries data for the introduction of the disease into a completely susceptible population (Ribeiro et al., 2010). The initial (exponential) growth rate, λ, was obtained by fitting a Poisson generalised linear model to the initial outbreak data (Figure 3.2A). Using the classical SEIR-SEI system of ordinary differential equations for a vector-borne disease, the formula for the basic reproduction number, R_0^λ, can be derived as

$$R_0^\lambda = \left(1 + \frac{\lambda}{\gamma}\right)\left(1 + \frac{\lambda}{\mu_V}\right)\exp\left(\lambda\left(\frac{1}{\epsilon_V} + \frac{1}{\epsilon_H}\right)\right)$$

where $1/\mu_V$ is the mean life expectancy of the vector.

Endemic equilibrium approach

The asymptotically stable steady state of susceptible individuals in an ODE-based SEIR system for a directly transmitted disease can be used to estimate the basic reproduction number (Dietz, 1993) as

$$R_0^* = \frac{N_H}{S^*}$$

where S^* is the number of susceptible individuals at equilibrium. The directly transmitted disease R_0 estimate was then used as an approximation for the basic reproduction number of an endemic vector-borne disease. For the stochastic IBM this required the system to reach a dynamic equilibrium, where the proportion of susceptibles oscillates around the deterministic equilibrium (as the inherent stochasticities prevent the system from reaching an equilibrium state). We calculated R_0^* using susceptibility levels at a single time point, as well as the mean proportion of susceptible individuals over the final five years of the timeseries (Figure 3.2B).

Table 3.1. The default set of parameter values used in the simulation of the individual-based model describing the spread of a vector-borne disease.

Parameter	Description	Value [range†]
$\|C\|$	Number of communities in lattice	400 $[1, 16384]$
N_H	Host population size	100000
M	Maximum vector to host ratio	1.2
m	Minimum vector to host ratio	1.2
c_H	Host mortality shape parameter	6
d_H	Host mortality scale parameter	75×365 days
c_V	Vector mortality shape parameter	4 $[1, 4]$
d_V	Vector mortality scale parameter	23 days $[10, 40]$
$1/\mu_H$	Mean human life expectancy in ODE	70 years
$1/\mu_V$	Mean vector life expectancy in ODE	21.3 days
$1/\gamma$	Host recovery time	4 days
$1/\epsilon_H$	Intrinsic incubation period	6 days
$1/\epsilon_V$	Extrinsic incubation period	5 days
δ	Amplitude in extrinsic incubation period oscillations	0 days
β	Per day contact rate	0.6 days^{-1}
p_H	Pathogen transmission success to host	0.5
p_V	Pathogen transmission success to vector	0.5
σ	Local disease dispersal kernel standard deviation	2
ω	Long distance transmission probability	10^{-4}
ι	External infection rate per 100,000 hosts per day	10^{-2} $\left[10^{-5}, 1\right]$

† range considered for sensitivity analyses.

3.2.3 Parameter values

Table 3.1 provides an overview of the parameters and parameter values used throughout this work (unless stated otherwise). The values were chosen to reflect the epidemiological dynamics of an arboviral disease, such as dengue or Zika. However, the results presented here are qualitatively independent of the particular choice of parameters; Figure 3.9 show the results of model sensitivity analyses with respect to the dependency of R_0 estimates on particular parameter values.

3.3 Results

A multitude of the mathematical models put forward to study the dynamics of vector-borne diseases are based on compartmental models described by systems of ordinary differential equations (ODE). Crucial to these types of models is the assumption of constant death rates. As vectors are assumed to remain infectious for life, such assumptions can crucially influence not only the resulting dynamics but also the estimates of the disease's basic reproduction number R_0 (and relatedly the (time-varying) effective reproduction number $R_e(t)$). Here we aimed to quantify the effects of relaxing the assumption of constant vector death rate on R_0 estimates within the same theoretical setting. This was done by comparing the R_0 values derived from an SEIR-SEI system of ODEs with a formula derived from first principles using the transmission cycle of a generic vector-borne disease using different assumptions about vector mortality rates (see Methods). We then verified these estimates by means of a stochastic individual-based model, which allowed us to directly *measure* R_0 from running repeat simulations of introducing an infected individuals into a fully susceptible population. The same model was also used to derive R_0 estimates from simulated timeseries data.

3.3.1 Constant vector mortality rates over-estimate R_0

Assuming constant vector mortality rates leads to exponentially distributed age profiles (Figure 3.1B), which permit some vectors to live in excess of four times their mean life expectancy and potentially to transmit the pathogen for an unusually long period of time. Even more concerning is that the vector life expectancy in each compartment of the infection process is essentially the same (Figure 3.1A). That is, independent of when in its life a vector becomes infected and infectious, its remaining life expectancy remains exponentially distributed around the mean life expectancy. As a consequence, all infectious vectors have a vector-to-human transmission period (VHTP) equal to the mean life expectancy of all vectors, $1/\mu_V$, with obvious consequences for R_0 estimates.

In contrast to ODE models, individual-based models (IBM) permit much greater control over vector mortality rates. Here we used (Weibull distributed) age-dependent vector death rates (see Methods), which yield a range of sigmoid age profiles (Figure 3.1D) but which all

Table 3.2. Comparison of the reproduction numbers derived from a system of ordinary differential equations and the individual-based model under the assumption of constant versus age-dependent mortality rates.

Model	Vector death rate	Life expectancy	Infectious period	R_0
ODE	Constant	21 days	21 days	7.5
IBM	Constant	21 days	21 days	7.2
IBM	Age-dependent	21 days	11 days	3.2

prevent vectors from living severely extended lives. More importantly, an individual vector's remaining life expectancy remains unchanged when transitioning between susceptible and infected state or between infected and infectious state, resulting in shorter and more realistic infectious periods.

We demonstrate the effect of assuming different vector mortality rates by comparing the R_0 estimates derived from the ODE model to the individual-based model (see Methods). As expected, using parameters as listed in Table 3.1 we find that the reproduction numbers from the ODE and IBM systems are similar under the assumption of constant vector mortality rates (Table 3.2). The small discrepancy between the two models is due to the ODE model's assumption of an exponentially distributed extrinsic incubation period, whereas the IBM assumes this to be a fixed length of time. Using the IBM approach to track individual mosquitoes and infection events we also find that under this assumption the mean age at which vectors become infected is 20 days and infectious at an age of 25 days, i.e. days beyond their average life-expectancy. Furthermore, those vectors that have become infectious live for an average of 46 days, which means that their infectious period is 21 days (equal to the life expectancy of all vectors). This clearly highlights the discrepancy between model outputs based on constant mortality rates and biological reality. In contrast, assuming age-dependent mortality rates (Weibull shape parameter, $c_V = 4$) results in biologically more reasonable infectious periods of 11 days (Table 3.2) and R_0 estimates that are less than half of those based on a model with constant mortality.

To further demonstrate the dependency of R_0 on different distributions of mosquito survivorship, we changed the Weibull distribution of vector mortality to transition smoothly between an exponential ($c_V = 1$) and a sigmoid ($c_V > 1$) age profiles and by keeping the average life expectancy constant. As illustrated in Figure 3.3, relaxing the assumption of

constant mortality and resultant exponential age profile shortens the average infectious period and lowers the reproduction number as derived from the transmission cycle of the pathogen, i.e. R_0^{IBM}. This clearly demonstrate that as well as the vector life expectancy, the actual shape of the survival curve strongly determines the estimated values of a pathogen's reproduction number.

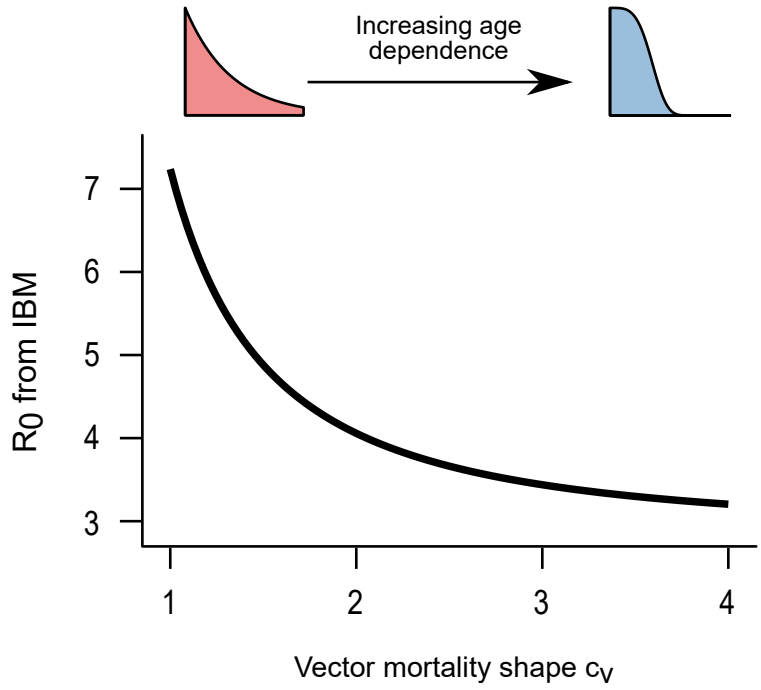

Figure 3.3. The effects of age-dependent mortality rates on the reproduction number. Starting by assuming a constant death rate ($c_V = 1$), the reproduction number derived from the transmission cycle of the individual based model, R_0^{IBM}, rapidly decreases as vector mortality becomes increasingly more age-dependent ($c_V > 1$) under constant average life spans.

3.3.2 Comparing R_0 estimates through direct measurement

The scenario defined by the reproduction number, whereby a single infectious case enters an entirely susceptible population, is arguably unrealistic for most diseases. Furthermore, disease transmission is an inherently stochastic process, such that each realisation of a disease introduction event is likely to take a different course. We should therefore expect that R_0 estimates derived from such introductory events should come with a certain degree of variation. In order to better understand the variability of the expected number of secondary cases and then to directly compare the above formula-based R_0 estimates, we simulated disease introduction events into a completely susceptible population using our

IBM framework and kept track of all secondary host infections resulting from the index case.

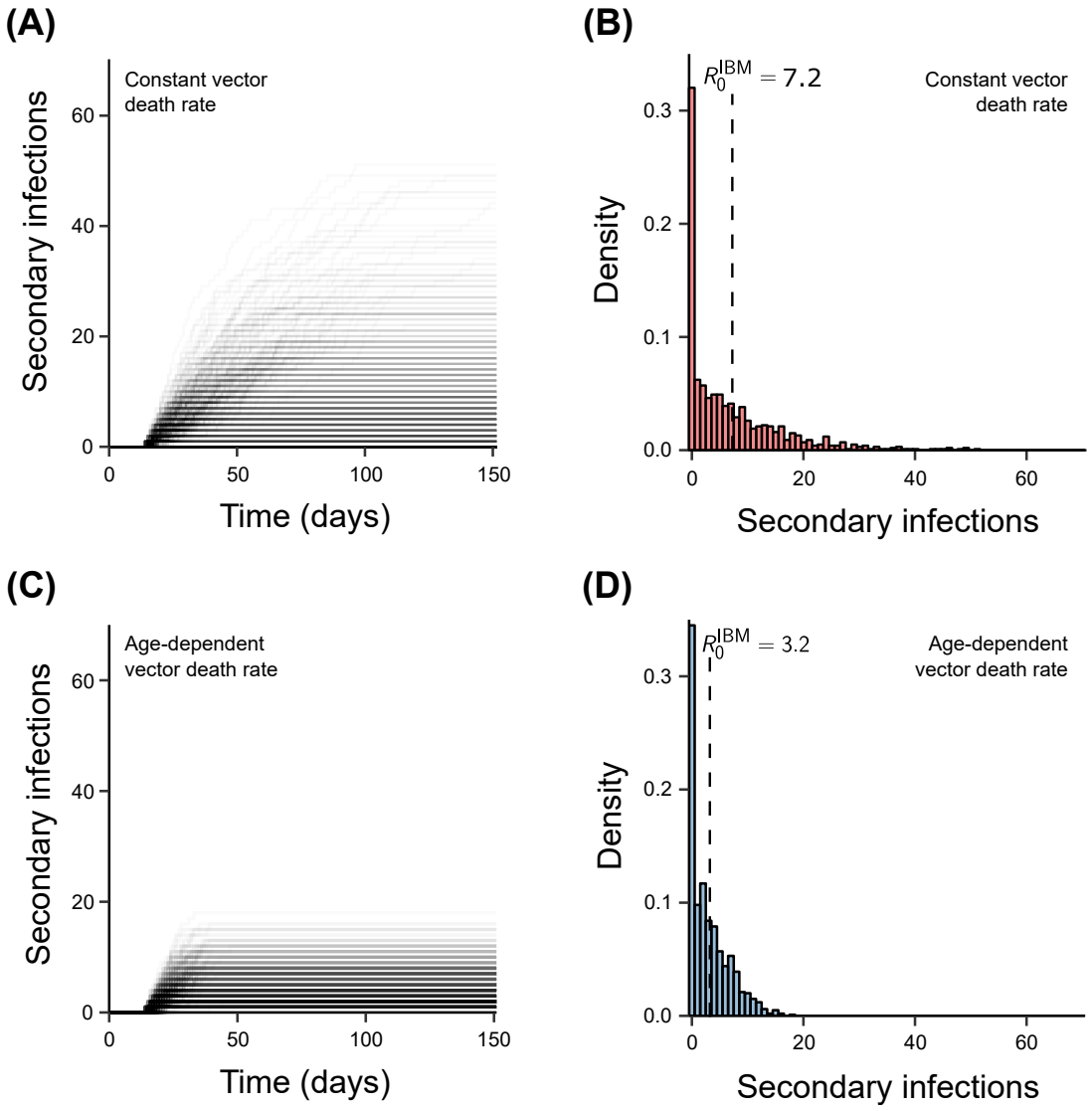

Figure 3.4. Reproductive number measured from the individual-based model. **(A,C)** Keeping track of the number of secondary infections in an entirely susceptible population starting with a single human case over time illustrates how in the model with constant vector death rates, secondary cases can still occur more than 100 days after the disease is introduced. This is in stark contrast to the model with age-dependent mortality, where most secondary infections occur within the average life-expectancy of the mosquito. Each solid line represents the accumulation pathway of secondary infections over time with darkness indicating the percentage of simulations that follow each pathway. The individual-based model was executed 500 times yielding a distribution of total number of secondary infections, or R_0, assuming either constant **(B)** or age-dependent vector mortality rates **(D)**. The two dotted lines are the reproduction numbers calculated from the theoretical IBM calculation. The mean number of secondary infections was comparable to the reproduction number derived from the transmission cycle of a vector-borne pathogen. Results are based on 100 model runs. Parameter values as in Table 1, except $c_v = 1, d_v = 20.8$ (constant mortality) and $c_v = 4, d_v = 23$ (age-dependent mortality).

As before we compared the two different assumptions regarding vector life expectancy: constant vs. age-dependent mortality rates. As shown in Figure 3.4, there is a wide distribution in the number of secondary infections, particularly when we assumed constant vector death rates (Figure 3.4A, B). In that case it was not unusual to observe 40-60 secondary infections, due to the aforementioned unrealistically high life-expectancies for some of the vectors, permitting the accumulation of secondary cases well after the primary human case has recovered (Figure 3.4A). The mean number of secondary infection (i.e. R_0) across 500 model simulations was around 7, more than twice that of the model which assumed age-dependent mortalities. In the latter case we observed secondary infections in the range of 0 to 18 (due to the model's stochastic nature where some vectors may be infected for their entire life) and with a mean of around 3.2 (Figure 3.4C, D), in line with theoretical expectations. Please refer to Figure 3.5 for sensitivity on model parameters on the direct measurement of mean secondary infections from the IBM.

An interesting observation is that under both assumptions of vector mortality, over 30% of our simulations resulted in zero secondary infections, as either the single primary case did not infect any vectors, the infected vectors failed to survive the extrinsic incubation period, or the infectious vectors failed to transmit the pathogen. Shorter infectious periods for both the host and the vector, a longer extrinsic incubation period, and lower transmissibility naturally decrease the overall likelihood of transmission from primary to secondary cases. Therefore, the proportion of failed outbreaks crucially depends on all these factors (Figure 3.6).

3.3.3 Initial growth rate methods can lead to over-estimation of R_0

In most cases, only successful disease introductions that lead to epidemic outbreaks are observed. These outbreaks can then be used to estimate the reproduction number based on the initial epidemic growth rate λ (see Methods). Formulas to calculate R_0 from λ are usually based on ODE modelling frameworks assuming constant vector death rates. To investigate the effect of this assumption on estimating a disease's basic reproduction number from epidemic growth rates, R_0^{λ}, we used our IBM framework to generate 50 epidemic outbreaks (discounting failed introductory events) under identical initial conditions for

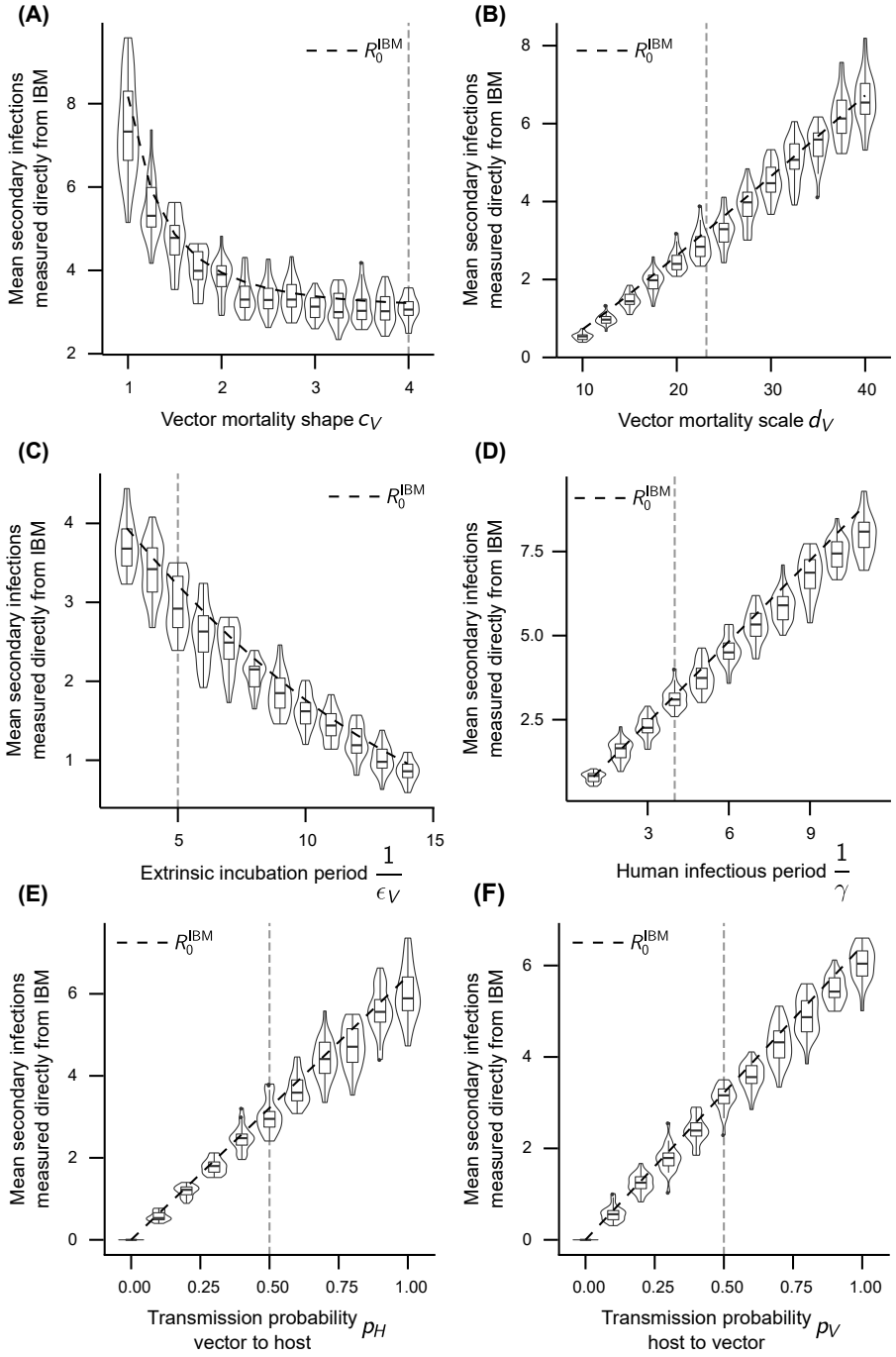

Figure 3.5. Sensitivity of R_0 direct measurement on model parameters The direct measurement of the mean number of secondary infections from a single introduction in the individual based model reliably estimates the theoretical calculation R_0^{IBM}. Longer vector life expectancies **(A)**, decreased age-dependence of vector mortality rates **(B)**, longer human infectious periods **(C)**, shorter extrinsic incubation periods **(D)**, and higher transmissibility **E**–**F**, increase R_0^{IBM} and the mean number of secondary infections simulated in the individual based model. Each parameter was tested 2500 times, where the mean number of secondary infections was calculated from groups of 100 simulations. The dashed vertical lines represent the baseline values selected.

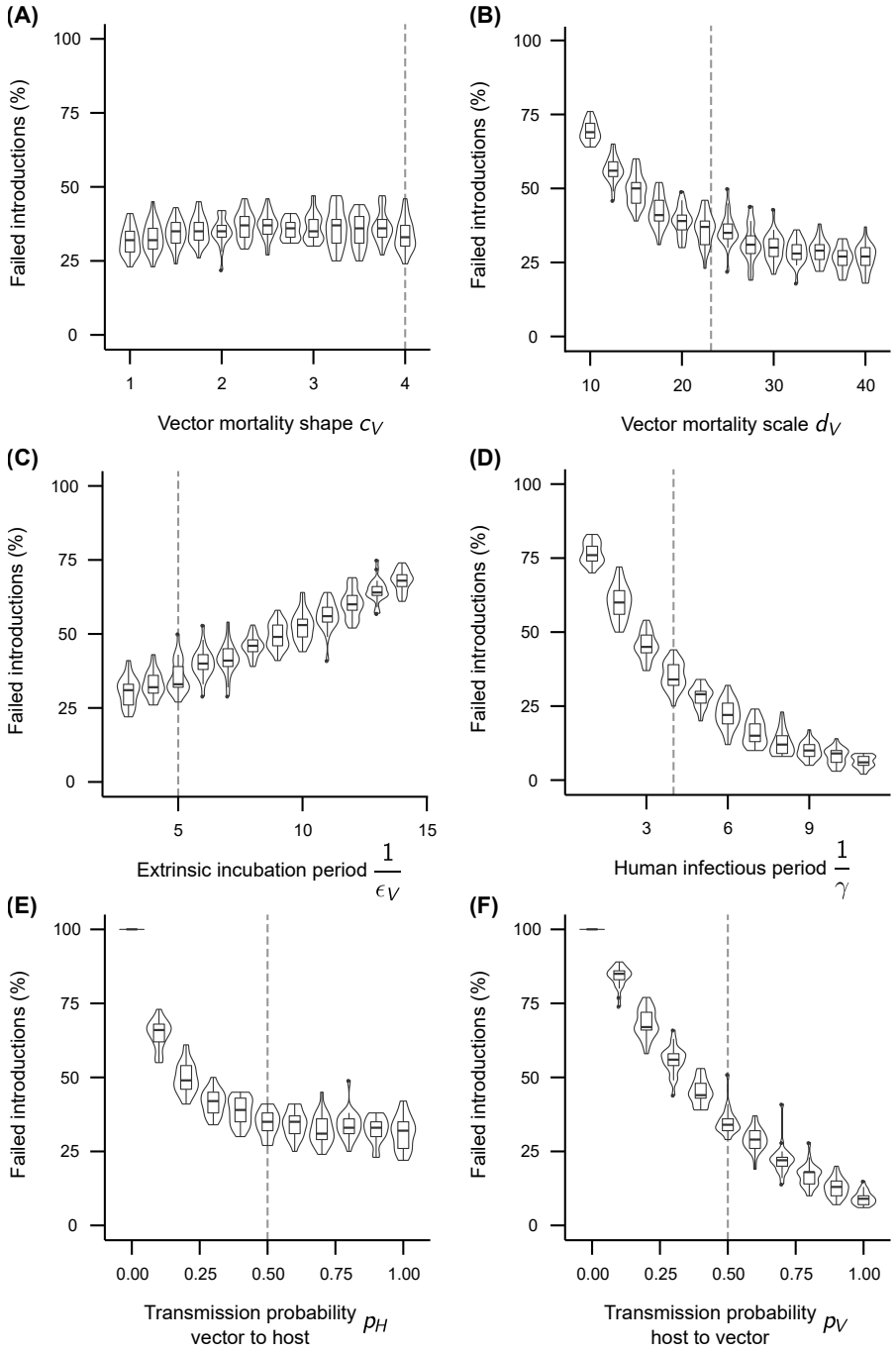

Figure 3.6. Sensitivity of stochastic fadeout on model parameters. The proportion of failed disease introductions in the individual based model depends upon a variety of factors. Longer vector life expectancies decrease the number of failed outbreaks **(A)**, whereas the age-dependence of vector mortality has little effect **(B)**. Longer human infectious periods **(C)**, shorter extrinsic incubation periods **(D)**, and higher transmissibility **(E–F)** naturally increase the overall likelihood of transmission from primary to secondary cases. Each parameter was simulated 2500 times, where the proportion of failed introductions was calculated from groups of 100 simulations. The dashed vertical lines represent the baseline values selected.

both constant and age-dependent mortality rates.

As illustrated in Figure 3.7, estimating the reproduction number from initial outbreak data is fairly reliable as long as the empirical age profiles of the mosquitoes match the one assumed in the model. That is, if mosquito mortality was indeed independent of age, leading to exponentially distributed age profiles, then R_0^λ can provide good estimates of the real reproduction number. However, if the risk of dying does increase with age, then R_0^λ, as derived form the ODE framework, is once again significantly over-inflated. Likewise, assuming age-dependent death rates when mortality is in fact constant, this could lead to an underestimation of the true reproduction number; note, however, that the latter scenario is arguably less relevant in biological terms.

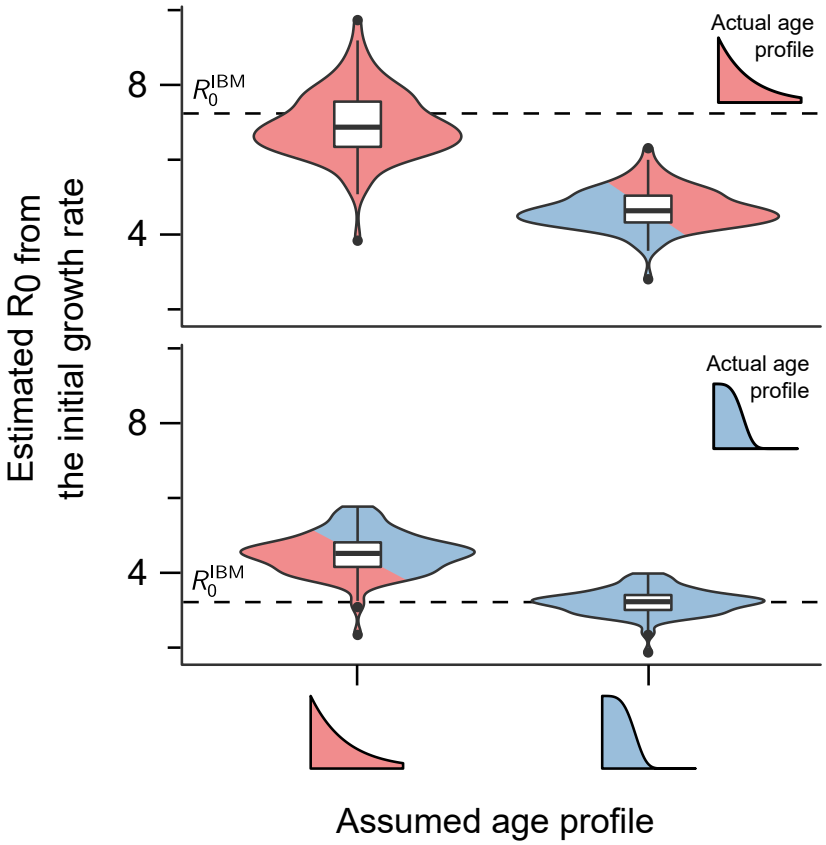

Figure 3.7. R_0 estimates based on the initial growth can over-estimate R_0. Violin plots showing the density distribution of R_0 estimates based on the initial epidemic growth rates. When the model assumptions regarding vector mortality correctly reflect the true mortality rates (solid colour) we find that the method based on epidemic growth generates fairly robust estimates of the true R_0. When the assumed mortality rate does not coincide with the real one (mixed colour), estimates can be off by a wide margin. Results are based on 100 model runs for each scenario; the inserted boxplots indicate the median and interquartile ranges.

Noticeable in all situations is the considerable variance in R_0^λ. This is due to the stochastic nature of our spatial IBM framework, which to a certain extend should also reflect the natural stochasticities underlying real vector-borne disease systems. Changing the model's spatial and demographic set-up will obviously affect the variance reported here; however, the results, related to the mean values, are to be understood as independent of the model's underlying structure.

3.3.4 Correction for initial growth rate methods

As shown in Figure 3.7, using the initial epidemic growth rate is only appropriate when empirical vector mortality is indeed age-independent, whereas it can lead to significant over-estimations otherwise. In order to compensate for this and include age-dependent vector mortality rates into the ODE-derived formula for R_0^λ, we replaced this critical term by the vector to host transmission period (VHTP), denoted by ν_{I_V}, calculated directly from an assumed vector age profile (see Methods), which yields the corrected estimate

$$\hat{R}_0^\lambda = (1 + \lambda\nu_{I_V}) \left(\frac{\mu_V}{\mu_V + \lambda} \right) R_0^\lambda \tag{3.1}$$

where μ_V is the constant vector mortality rate in the classical system of ordinary differential equations. Crucially, a vector age profile has to be assumed explicitly to calculate the VHTP. And as before, if the assumed profile in $\hat{R}_0^\lambda$ matches the simulations' profile, we find that the derived reproduction numbers are good estimates of the actual ones, with the same variance as before (Figure 3.7).

3.3.5 Endemic equilibrium can provide robust estimates of R_0

Finally, we sought to estimate R_0 from the dynamic equilibrium distribution of susceptibles in the human population (see Methods). Crucially, this approach does not require any *a priori* knowledge of mosquito survivorship and should therefore provide more robust estimates regardless of the underlying assumptions regarding vector mortality rates. Indeed, and as demonstrated in Figure 3.8, using the endemic state can provide reasonable estimates of a disease's true (i.e. theoretical) R_0 value, even though the formula itself was derived from a directly transmittable disease, which might explain why R_0^* slightly underestimates

R_0.

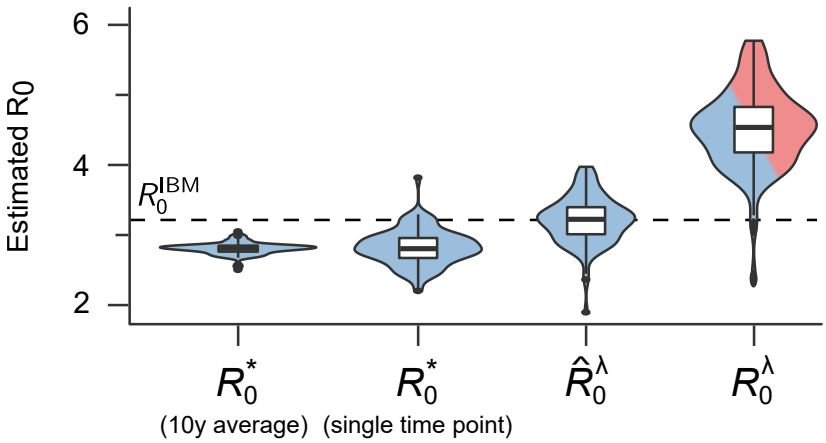

Figure 3.8. R_0 estimates from the endemic equilibrium are more robust. Estimating R_0 from the endemic equilibrium distribution of susceptibles (R_0^*) requires no assumptions about underlying vector survival rates and proves more robust than estimates based on the initial growth rate (R_0^λ), especially when using longitudinal data (compare single time point with 10 year average). Note that corrected values of R_0^λ, $\hat{R}_0^\lambda$, can yield good estimates but are still subject to significant variations around the mean. Results are based on 100 model runs for each scenario; the inserted boxplots indicate the median and interquartile ranges, and the dashed line denotes the theoretical R_0 value.

As before we find a significant degree of variation around the mean estimates, due to the stochastic nature of disease transmission. This can somewhat be reduced by taken longer term averages (compare single time point estimates with 10 year average in Figure 3.8), which in reality will be limited due to data availability. Equally, the model and population structure itself, including as population size, importation rates, spatial structuring and mixing, all affect the stability of the dynamic equilibrium and with it the variance and hence robustness of R_0^* (Figure 3.9). Although this method is only applicable for diseases that have reached at least a semi-endemic state, its parameter and assumption-free approach means that it should be considered as one of the most robust ways to estimate a disease's reproduction number.

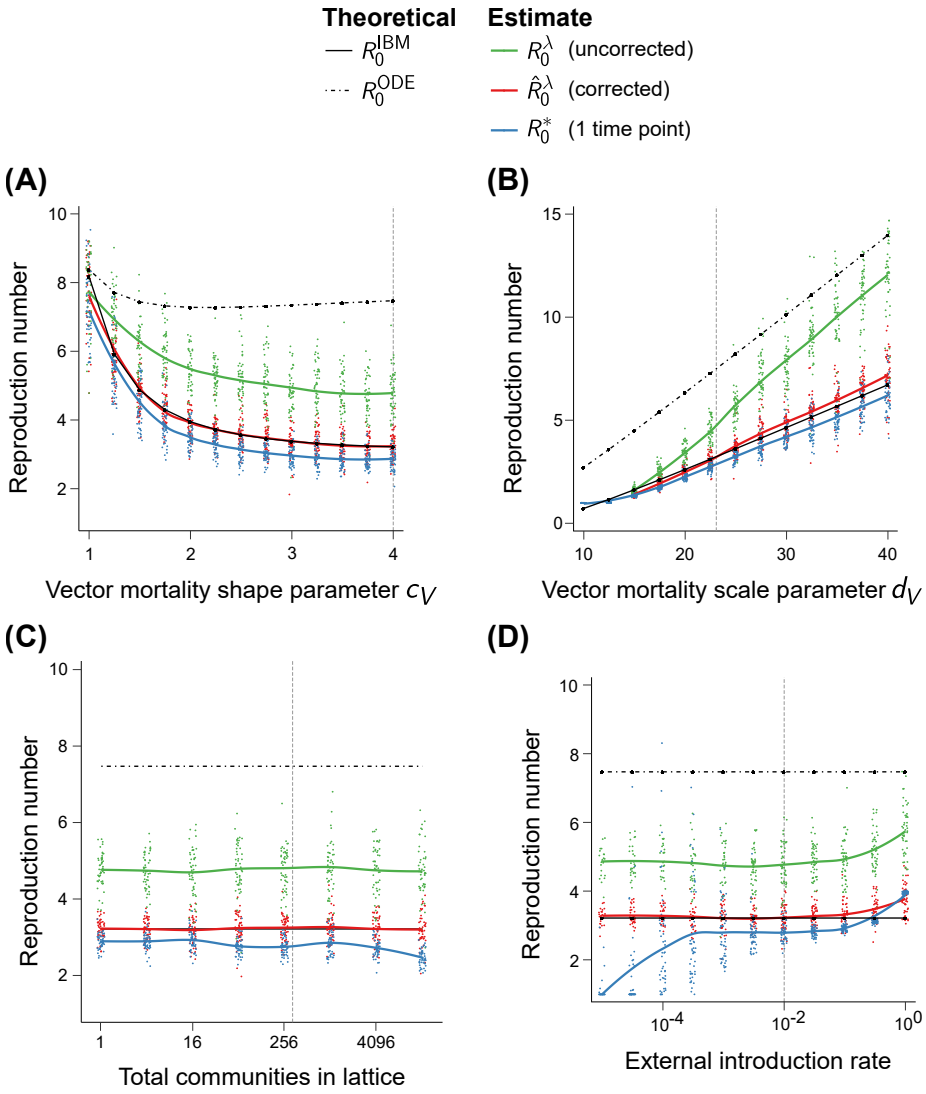

Figure 3.9. Sensitivity of R_0 estimates on model parameters. **(A)** Vector mortality shape parameter c_V. Increasing the shape of the vector mortality distribution from constant to age-dependent survival rates shows that traditional theoretical approaches to R_0 significantly overestimate the reproduction number R_0^{IBM}. Both estimates from the endemic equilibrium and post-correction initial growth rate continue to be robust over this range of shape parameters. **(B)** Vector mortality scale parameter d_V. Both theoretical calculations scale linearly with the vector mortality scaling parameter, as this directly influences vector life expectancy and thus the vector-to-human transmission period (VHTP). Across all tested parameters, both estimates from the endemic equilibrium and post-correction initial growth rate continue to be reliable for $R_0 > 1$. **(C)** Number of communities in the lattice $|C|$. The theoretical calculations of R_0 presented do not explicitly contain any spatial dynamics. Increasing the number of communities (starting with a homogeneous mixing model) does not affect the robustness of R_0 estimates. **(D)** External infection rate ι. The theoretical calculations of R_0 presented do not explicitly contain the external infection rate. Increasing the external infection rate, does not influence the robustness of R_0 estimates from the initial growth rate unless external infection rates are high enough to start driving the epidemiological dynamics. Furthermore, R_0 estimates from the endemic equilibrium continue to be reliable until re-introduction of the disease into the system is too low for disease persistence. For each parameter value tested, 50 stochastic simulations were executed and R_0 estimated for each simulation. The dashed vertical lines represents the baseline values selected.

3.4 Discussion

Mathematical models describing the population dynamics of an infectious disease provide the necessary frameworks by which we can calculate an infectious disease's reproduction number, R_0, based on specific parameters related to infection and transmission probabilities. One of the most important factors influencing R_0 is the length at which an individual remains infectious. For vector-transmitted diseases this places huge significance on vector mortality rates as vectors usually do not clear an infection and instead remain infectious for life. Many formulas to estimate R_0 are based on systems of ordinary differential equations (ODEs), which commonly assume that vector mortality is constant, i.e. independent of age. As we have demonstrated here, the resulting exponential distribution and the effective *resetting* of life expectancies as individuals transition through the infection stages permit some vectors to live for an extraordinary length of time. As a result, vectors are potentially able to transmit the disease multiple times that of what should biologically be possible, leading to significantly inflated R_0 estimates.

In comparison to ODE models, individual-based models (IBMs) provide much greater control over the dynamics that govern both demography and disease transmission. Here we used an individual-based modelling approach to elucidate the influence of vector mortality on R_0 estimates and to highlight the discrepancy between model predictions based on constant vs. age-dependent mortality. Because individual infection events can easily be tracked within an IBM, the basic reproduction number can essentially be measured simply by counting the number of secondary infections arising from a single index case. This in turn not only allowed us to compare different formulas for estimating R_0 but also provided us with a better understanding of the degree of uncertainty surrounding these estimates.

As demonstrated here, the assumption of constant vector death rates can lead to significant over-estimation of R_0. Importantly, it is not so much that the formulas commonly used to estimate R_0 are inherently wrong but rather that the underlying assumption of the models from which they are derived are not necessarily aligned with biological reality. We found that one of the most robust methods to estimate a pathogen's R_0 is based on the proportion of susceptible individuals at endemic equilibrium, as this is entirely parameter

free and does not require any assumption about vector death rates. Unfortunately, this only works for diseases that are well established in a population, and its reliability is strongly dependent on the stochasticity of the underlying endemic equilibrium, i.e. the (multi-annual) variations around the mean. For emerging diseases this is obviously not practical and estimation methods in those cases usually make use of epidemic growth curves instead. However, these also implicitly assume exponential vector age profiles and are therefore subject to inflation. In order to account for this we have here derived a correction factor that can be applied to classical R_0 estimation formulas and which adjusts for most of the discrepancy between the vector-to-human transmission period (VHTP) of the biological system and the assumed system with constant vector mortality.

In this work we made use of an individual-based modelling framework to test the effect of non-exponential vector age distributions on R_0. Alternative methods that allow for the (partial) relaxation of the assumption regarding constant mortality or vector senescence have also been proposed, including lumped-age class models (Hancock and Godfray, 2007) or systems of partial differential equations (Rock et al., 2015). However, these methods can still suffer from the same issues as simpler ODE models, where transition rates between life and infection stages are usually exponentially distributed and where information about individual ages is lost at every transition stage. The ease at which different distributions that govern host and vector mortality, infection recovery and other epidemiological factors can be incorporated, make IBM frameworks the natural choice to examine the influence of vector mortality or other such factors on R_0 estimations. Here we only concentrated on the effect of vector mortality, whereas similar arguments are equally valid for the distribution underlying the extrinsic incubation period (Brand et al., 2016), for example. Nevertheless, our work strongly suggests that vector mortality rates, or rather our assumptions about the age-dependency of survivorship, are the predominant factors, as our correction term for R_0 estimates based on epidemic growth essentially recovers the true value.

Another important observation from this study was that when simulating the spread of a disease from a single infected individual and then calculating R_0 based on the number of secondary infections, the stochastic nature of such events resulted in very wide distributions in R_0. Although the assumption of age-dependent mortality rates generally prevented

extremely high values of secondary cases, and therefore R_0, the variance was still in the region of twice the mean and included a significant proportion of zero cases. That is, in around a third of the simulations we observed no secondary case at all despite starting off with the same initial conditions. This then begs the question whether these events should be counted towards the estimated R_0 or not, as in reality we never observe such failed introductions. Comparing the expected with the observed R_0 value would suggest that zero cases should be counted, which on the other hand implies that even high values of the reproduction number are by no means a guarantee that an outbreak should ensue if a disease gets introduced in a fully susceptible population (sufficient conditions to prevent stochastic fade-out at the start of an epidemic have been previously discussed (Hartfield and Alizon, 2013)). The high variation also suggests that control strategies based on R_0 estimates generated from initial growth rates should be treated with caution and that estimations based on one particular setting might not be adequate to generalize and predict pathogen behaviour across all other spatial contexts (Smith et al., 2014).

We here concentrated solely on the basic reproduction number, which describes an arguably unusual and often artificial situation. However, it should be clear that the same arguments also hold for the effective reproduction number, R_e, which is essentially R_0 multiplied by the fraction of the population that is susceptible to a disease, as well as their time-dependent counterparts $R(t)$ and $R_e(t)$. Furthermore, the serial and generation intervals, which can be understood as temporal analogues of the reproduction number, also rely on the vector to host transmission period and are usually assumed to be exponentially distributed (Huber et al., 2016; Siraj et al., 2017). This implies that these intervals, and alternative R_0 estimation methods that depend upon them (Cori et al., 2013; Perkins et al., 2018; Reiner et al., 2015), may equally be over-estimated.

Our work thus reiterates the importance of obtaining empirical vector mortality rates in the field. The original Ross-MacDonald model for the spread of *Plasmodium falciparum* and *P. vivax* malaria assumed constant vector mortality as laboratory and field studies seemed to suggest that death rates were age independent (MacDonald, 1952). However, re-analysis of laboratory data showed that mosquito mortality is in fact age-dependent for several *Anopholes* species (Clements and Paterson, 1981). More recent studies also

confirmed that mosquito mortality is dependent on age for *Anopheles* mosquitoes (Dawes et al., 2009) and *Aedes aegypti* (Harrington et al., 2008; Styer et al., 2007). What is clear is that more work needs to be done to fully elucidate realistic, i.e. field-relevant vector mortality rates, perhaps with more accurate spectroscopic methods (Lambert et al., 2018), as well as their environmental drivers. That is, seasonal variations in temperature and rainfall have been shown to affect the birth and death rate of vectors (Alto and Juliano, 2001; Brady et al., 2013; Valdez et al., 2017; Yang et al., 2009), the vectorial competence (Lambrechts et al., 2011) as well as the extrinsic incubation period (Beck-Johnson et al., 2013; Brady et al., 2014; Chan and Johansson, 2012; Focks et al., 1995). It has also been emphasized that other spatio-temporal heterogeneities, such as community structures and host and vector movement, should be considered when assessing R_0 (Lloyd et al., 2017; Perkins et al., 2013). All this needs to be factored in if we are to develop better models to understand the epidemiological and ecological determinants of vector-borne diseases, guide outbreak prevention strategies or monitor ongoing intervention measures.

Chapter 4

Effects of community structures and vector ecology on dengue spread and persistence

4.1 Introduction

As discussed in Chapter 1 and demonstrated in Chapter 2, human movement is implicated as an important driver of dengue epidemiology across space and time (Stoddard et al., 2009). Local, regional and global mobility naturally give rise to networks of communities, that play an essential role in the epidemiology of directly transmitted disease (Aparicio and Pascual, 2007; Cauchemez et al., 2011; Meyers et al., 2005; Newman, 2002; Riley and Ferguson, 2006; Salathé and Jones, 2010; Wang et al., 2015), and in capturing the qualitative patterns of endemic dengue at fine spatial scales (Adams and Kapan, 2009; Favier et al., 2005; Perkins et al., 2016; Reiner et al., 2014; Stoddard et al., 2013). At wider spatial scales, Massad et al. (2008) demonstrated that a scale-free transmission network is able to explain the 2005 dengue epidemic in Singapore and emphasised the impact of understanding social contact structures on dengue control measures, where intervention strategies could be focused within super-spreading communities. Furthermore, Gardner and Sarkar (2013) constructed a global dengue importation risk model informed by the air transport network, again suggesting priority locations for targeted dengue surveillance in order to inhibit emergence within dengue-naïve regions. However, little work has been done to elucidate the influence of community structure on the spread and persistence of

dengue across larger geographical regions.

Heterogeneity in human population density between these communities has also been shown to be associated with dengue epidemiology, although the nature of which is unclear. Empirical-based findings are in conflict about whether there is a positive, negative, or no correlation between human population density and dengue incidence rates (Díaz-Quijano and Waldman, 2012; Kong et al., 2018; Qi et al., 2015; Schmidt et al., 2011; Siqueira et al., 2004; Sirisena et al., 2017). In fact, a pair of studies carried out within the same city at different times discovered that higher host population sizes were both associated and dissociated with dengue incidence Ko (1989); Lin and Wen (2011), suggesting that some time-based factor may be at play. Existing modelling frameworks have also exhibited contrasting results: one study stating that mosquito-borne disease prevalence scales with human density (Smith et al., 2004), and another implying there to be an attack abatement effect of host aggregation (Cummins et al., 2012). These conflicting findings may be down to the habitat preference of the vector.

As the primary vector of dengue, *Aedes aegypti*, is anthropophilic, it naturally prefers well-populated urban environments (Ponlawat and Harrington, 2005; Scott et al., 1993a, 2000a), giving rise spatio-temporal heterogeneity in vector abundance. These patterns have been shown to be important in characterising the transmission of mosquito-borne disease (Manore et al., 2014; Perkins et al., 2013; Romeo-Aznar et al., 2018), with increased heterogeneity in vectors facilitating dengue persistence (Acevedo et al., 2015). On the other hand, Cromwell et al. (2017) found there to be no correlation between DENV seroconversion and *Aedes aegypti* abundance. These results suggest the ecological factors dictating dengue transmission may vary by spatial setting, but overall, it is unclear how the interplay of heterogeneity in both host and vector population density influences dengue epidemiology.

In order to better understand the complex relationships between spatial ecology and dengue's epidemiological dynamics, in this chapter, we quantify the effects of community structure and ecological heterogeneities on dengue epidemiology within the spatially explicit individual based modelling framework presented in Chapter 2. We explore the effects of three different community structures describing local mobility on dengue incidence in an endemic setting, in addition identifying the critical community connectivity necessary

for the spread and persistence of the virus. Additionally, we investigate the influence of spatio-temporal heterogeneity in mosquito-to-human ratios on the epidemiological dynamics of dengue globally and at the community-level.

4.2 Methods

We employed the spatially-explicit individual based model developed in Chapter 2 and adjusted the spatial aspects of the model as follows:

4.2.1 Community structure

A community structure, or network, was described as a graph, G, of a set of communities (or vertices), C, and connections (or edges), E. We considered the following arrangement of communities were considered: a lattice, random network, and scale-free network (the distribution of the number of connections of each community follows a power-law).

Lattice

Here, the non-wrapping lattice community structure described in Chapter 2 was realised as a network (Figure 4.1A), such that communities are arranged into a grid and community $\zeta \in C$ is connected to community $c \in C$, or equivalently $\{\zeta, c\} \in E$, if ζ and c are adjacent within the grid.

Random network

The random network was generated using the Erdős–Rényi algorithm (Erdős and Rényi, 1959), where an edge $\{\zeta, c\} \in E$ exists with probability p_E (Figure 4.1B).

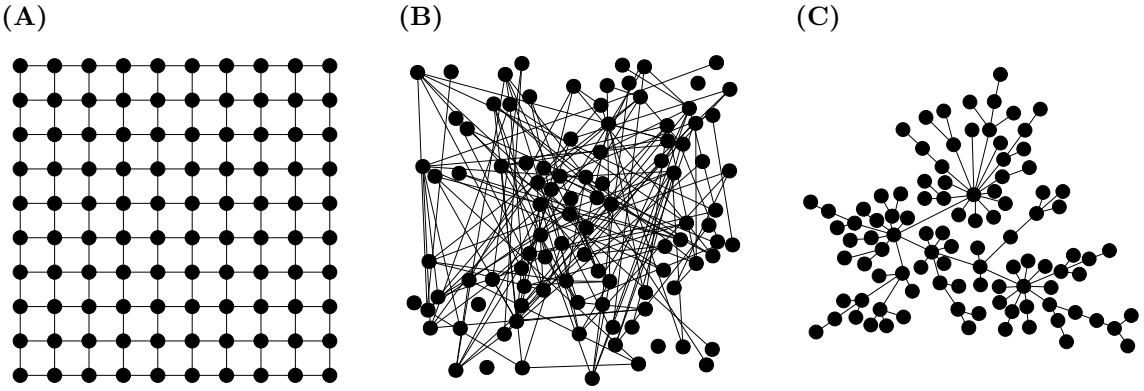

Figure 4.1. Different community structures. Communities of individuals (circles) were represented as networks, where the existence of a connection between two communities represented local disease transmission potential. Communities were either arranged as a **(A)** lattice, **(B)** random network ($p_E = 0.05$), or **(C)** Barabási-Albert network.

Scale-free network

The scale-free network used was a Barabási-Albert network, generated using the igraph software package (Csardi and Nepusz, 2006). The network was generated by adding communities one at a time, where each newly introduced community $\zeta \in C$ is connected to n other communities. The probability that $\{\zeta, c\} \in E$ is proportional to

$$\deg(c)^k + z,$$

where $\deg(c)$ is the number of connections, or degree, of community c, k denotes the constant power of preferential attachment (see Figure 4.2), and z represents the constant appeal of nodes with no connections.

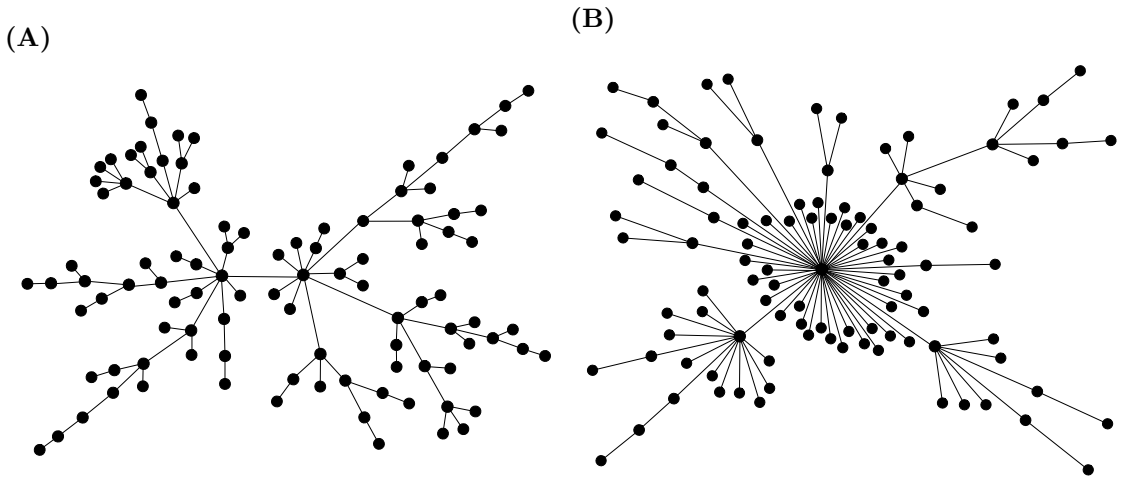

Figure 4.2. Effect of preferential attachment parameter k on Barabási-Albert network generation. **(A)** Barabási-Albert networks with low power of preferential attachment ($k = 0.5$) have more communities with more than one connection, whereas **(B)** Barabási-Albert networks with high power of preferential attachment ($k = 1.5$) have very few communities with more than one neighbour.

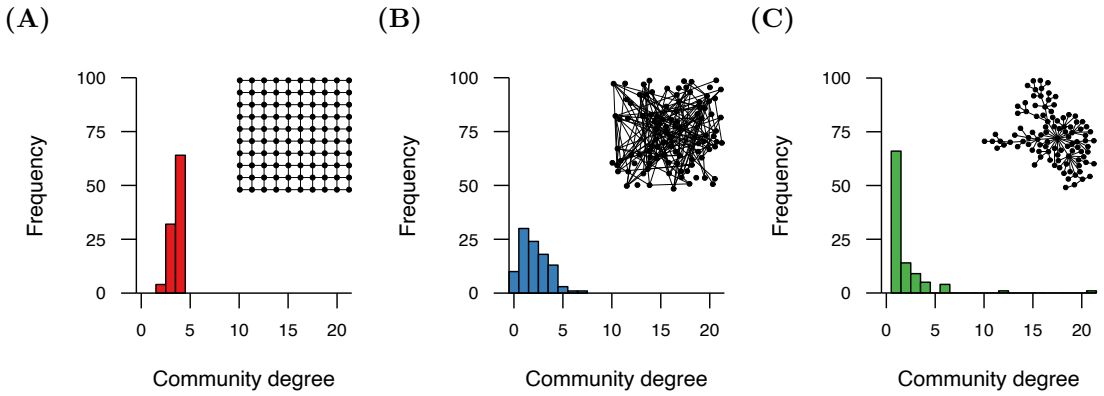

Figure 4.3. Connectivity of the lattice, random, and Barabási network. Each community in the **(A)** lattice community structure had similar connectivity, in contrast to the **(B)** random or **(C)** Barabási network, which exhibited a wide variation in connectivity between communities. Here, the degree of a community is the number communities to which it is connected.

Community degree

The degree of a community is defined as the number of communities that a given community $c \in C$ is connected to in the network. The degree of each community in a lattice is very similar (Figure 4.3A), whereas the degree distribution of a random network follows a binomial distribution with the number of trials and probability of success given by the number of communities in the network, $|C|$, and the probability of the existence of an edge between two communities in the network, p_E, respectively (Figure 4.3B). The degree distribution of communities in the Barabási network follows a power law, dependent on the power of preferential attachment k, with an inflated number of communities with degree equal to one (Figure 4.3C).

4.2.2 Local disease dispersal kernel

As described in Chapter 2, the local disease dispersal kernel, Φ_ζ, describes the local spread of transmission events from a source community, ζ, to each destination community, $c \in C$, where C is the set of all communities. Although the two dimensional normal distribution is an appropriate choice for the regular lattice arrangement of communities where the location of each community is described in Cartesian space, it makes little sense in a network of irregular connectivity where, the spatial location of each community is assumed to be unimportant. Therefore, transmission events were dispersed from the source community, ζ, to only those communities with an immediate connection with ζ. This is given in the following way:

$$\Phi_\zeta(c) = \begin{cases} p_\sigma & \zeta = c, \\ \frac{1-p_\sigma}{\deg(\zeta)} & \{\zeta, c\} \in E, \\ 0 & \text{otherwise}, \end{cases}$$

where p_σ is the probability of transmission events remaining in community ζ, $\deg(\zeta)$ is the degree of ζ in the network, and $\{\zeta, c\} \in E$ denotes the edge between the source community, ζ, and destination community, c.

4.2.3 Population density heterogeneity

In addition to investigating the effects of community structure on the epidemiology of dengue, the influences of heterogeneities of human and mosquito population densities were explored (Figure 4.4). Here, it is assumed that the maximum number of individuals in a community follows a power law and is proportional to the degree of a community. Furthermore, it was assumed that mosquito density was seasonal. Therefore,

$$N_H^c \propto N_H \deg(c)^{\eta_H},$$

$$\max_{t\geq 0}\{N_V^c(t)\} \propto \max_{t\geq 0}\{N_V(t)\} \deg(c)^{\eta_V},$$

where N_H^c and $N_V^c(t)$ are the number of humans and mosquitoes at time t, respectively, in community $c \in C$, $\deg(c)$ is the degree of community c, and η_H and η_V are scalars for the strength of the power-law relationships between community degree and population size for humans and mosquitoes, respectively.

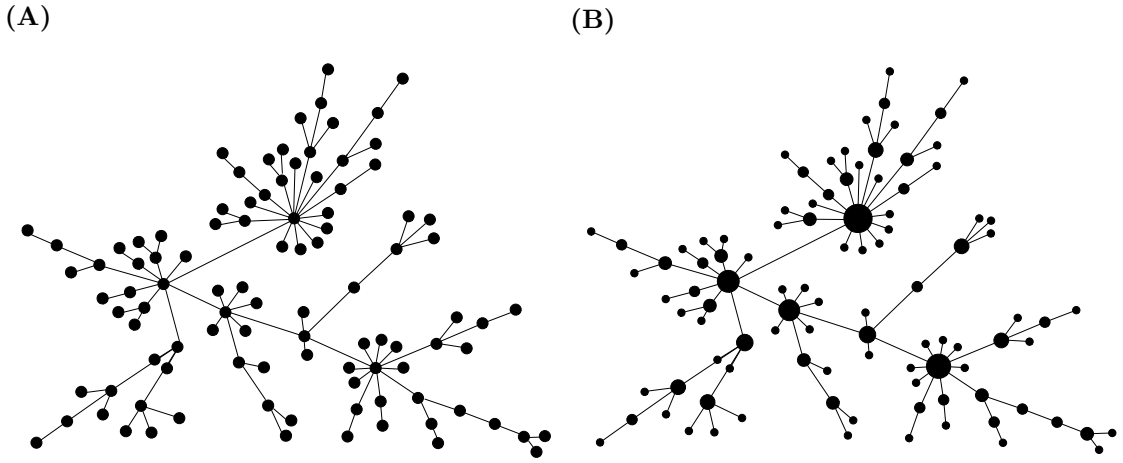

Figure 4.4. Community size heterogeneity. The population size in a community is determined by a power-law relationship and the degree of each community in the network. **(A)** Barabási-Albert network with no heterogeneity in community size ($\eta_H = \eta_V = 0$). **(B)** Barabási-Albert network with high heterogeneity in community size ($\eta_H = \eta_V = 0.5$).

4.2.4 Epidemiological properties

We quantified the effects of community structure and heterogeneity in community size by using the epidemiological metrics as described in Chapter 2, including mean prevalence per 100,000 individuals, annual variability, and basic reproduction number estimates, as presented in Chapter 3. All epidemiological metrics were calculated from simulations over a 75 year period. Each epidemiological metric was also calculated for (i) the entire meta-population, (ii) an urban community, defined by the community with the most number of connections, and (iii) a rural community, defined by the community with the fewest number of connections and furthest from the urban community (Figure 4.5). If there was more than one community satisfying conditions in (ii) or (iii), the community selected was randomly selected.

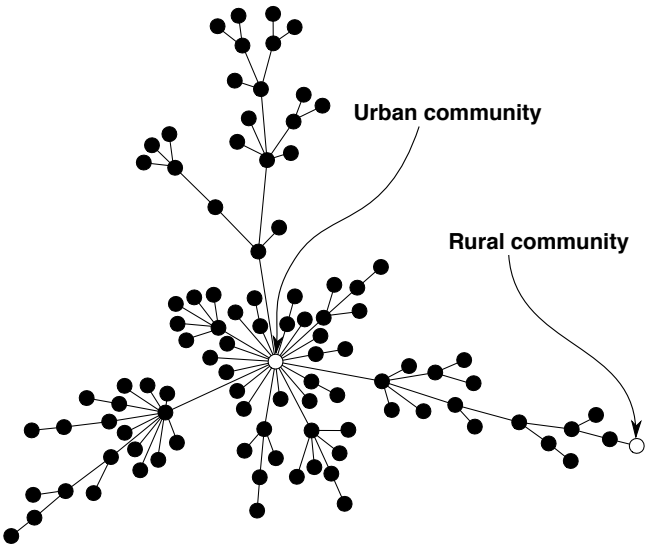

Figure 4.5. Urban and rural communities of a complex network. Epidemiological metrics were calculated for both an urban and rural community. The urban community was defined as the community with the highest degree. The rural community was defined as the community with the lowest degree, furthest from the urban community.

d

Table 4.1. The default set of parameter values used in the comparing the effects of different community structures on the emergence, spread and persistence of the dengue virus.

Parameter	Description	Value
N_H	Human population size	5000000
$\|C\|$	Number of communities	100
p_E	Probability of connection in random network	0.02
k	Power of preferential attachment of Barabási algorithm	1
n	Connections generated at each step of Barabási algorithm	1
z	Appeal of communities with zero connections in Barabási algorithm	1
p_σ	Probability of local transmission event not being dispersed	0.8
η_H	Strength of community size heterogeneity in hosts	0
η_V	Strength of community size heterogeneity in vectors	0

4.2.5 Model parameters

The default set of parameters, unless stated otherwise, selected for simulations are shown in Table 4.1. Other parameters were the same as in Table 2.2. Parameters were selected to best represent the epidemiology of dengue. The default connectivity of the random network was selected to be the same of a scale-free network generated through the Barabási algorithm with $(k, n) = (1, 1)$. This allowed the epidemiological effects of organised local mobility (scale-free network) to be assessed against random local mobility (random network). In the scale-free network, the appeal of communities with zero connections, z, was fixed at 1 to ensure every community was connected. Only one connection was generated at each step of the Barabási algorithm to generate a complete (fully connected) network with minimal connectivity. This gave a network containing no loops in contrast to both the lattice community structure and random network. With these parameter values, three qualitatively different community structures, in terms of type of local connectivity, could be compared. Sensitivity analysis of results were also performed on the probability of connection in a random network, p_E, power of preferential attachment of the Barabási algorithm, k, and strength of community size heterogeneity in hosts, η_H, and vectors, η_V.

4.3 Results

4.3.1 Effects of community structure

To discern the influence of community structure and connectivity on the epidemiological dynamics of dengue, the model was simulated for 3 different network structures: the lattice, the scale-free network, and a random network with a similar number of connections to the scale-free network. All simulations on each network produced multi-annual irregular epidemic outbreaks and sequential dominance of the four serotypes (Figure 4.6). However, the variability in magnitude of annual outbreaks was much lower when simulating over the lattice (Figure 4.6A) and random network (Figure 4.6B) than the scale-free network (Figure 4.6C).

Higher global annual variability was due to the increased discrepancy in connectivity between communities in the organised scale-free network versus the other networks (see Figure 4.3). That is, the abundance of highly connected communities in the lattice and random network allowed the virus to be more easily dispersed, resulting in very little spatio-temporal heterogeneity in dengue incidence (Figure 4.7A). On the other hand, the reduced number of highly connected communities in the scale-free network acted as a bottle neck for transmission. This resulted in the much slower dissemination of the virus throughout the network, inducing increased spatio-temporal heterogeneity in dengue incidence but with synchronised outbreaks between different regions of the network (Figure 4.7B).

Global, urban and rural prevalence and extinction risk

Comparing the effects of each community structure on the epidemiology of dengue within different communities showed that there was little difference in the global, urban or rural behaviour in the lattice network. This was again attributed to the small differences between the connectivity of communities within the lattice. In contrast, the urban versus rural communities exhibited very different epidemiological dynamics in the random and scale-free networks.

Mean prevalence decreased globally when the community structure was switched from the uniform lattice to the random and scale-free networks due to the overall reduction in

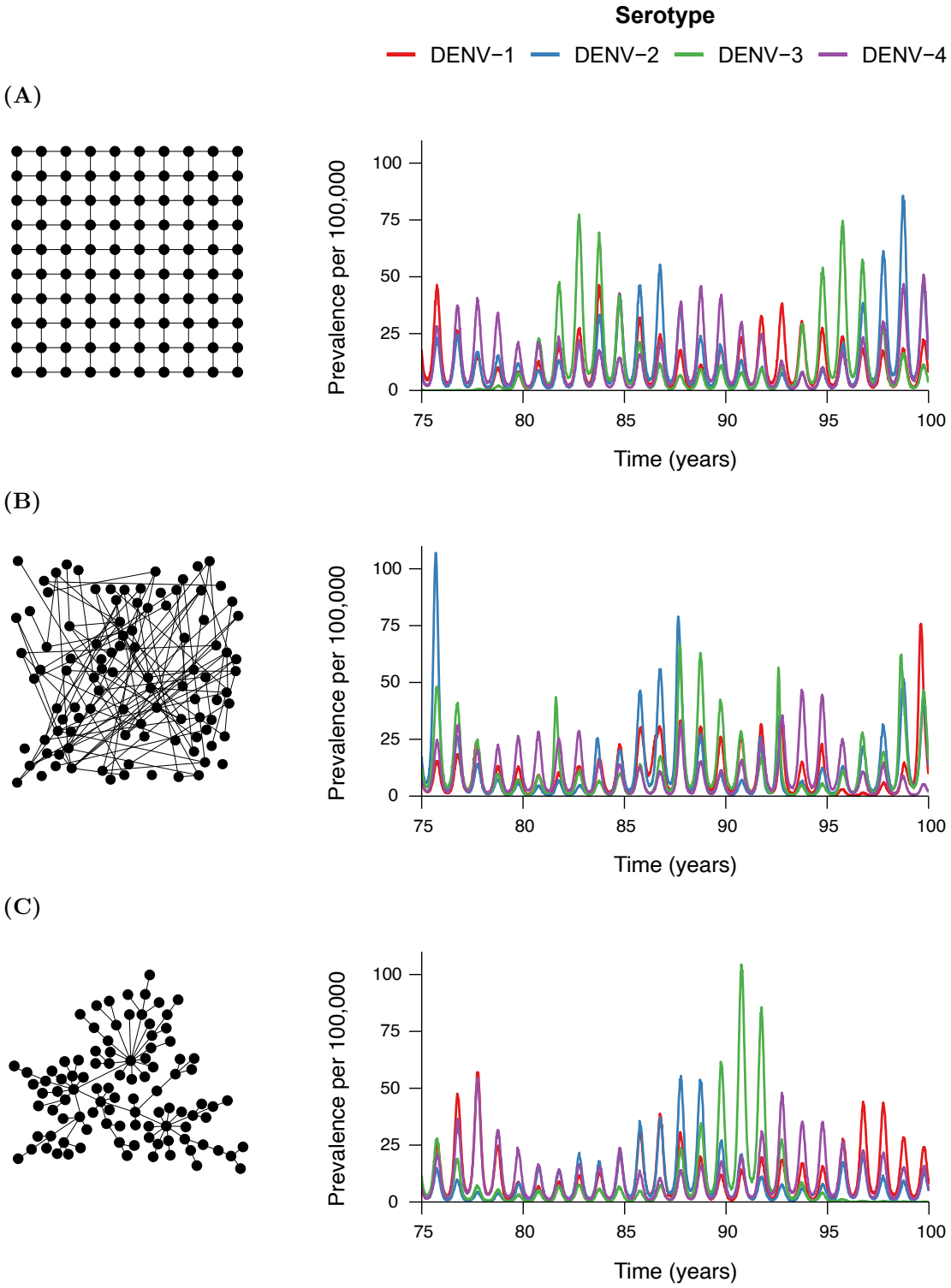

Figure 4.6. Comparing prevalence over time in the lattice, random, and scale-free network. Across all community structures, simulations produced annual outbreaks of varying magnitude characteristic of dengue, in addition to sequential dominance of the four serotypes. Simulating on the **(A)** lattice or **(B)** random network gave lower annual variability in outbreaks than the **(C)** scale-free network, because of the greater heterogeneity in connectivity of communities in the scale-free network versus the lattice or random networks.

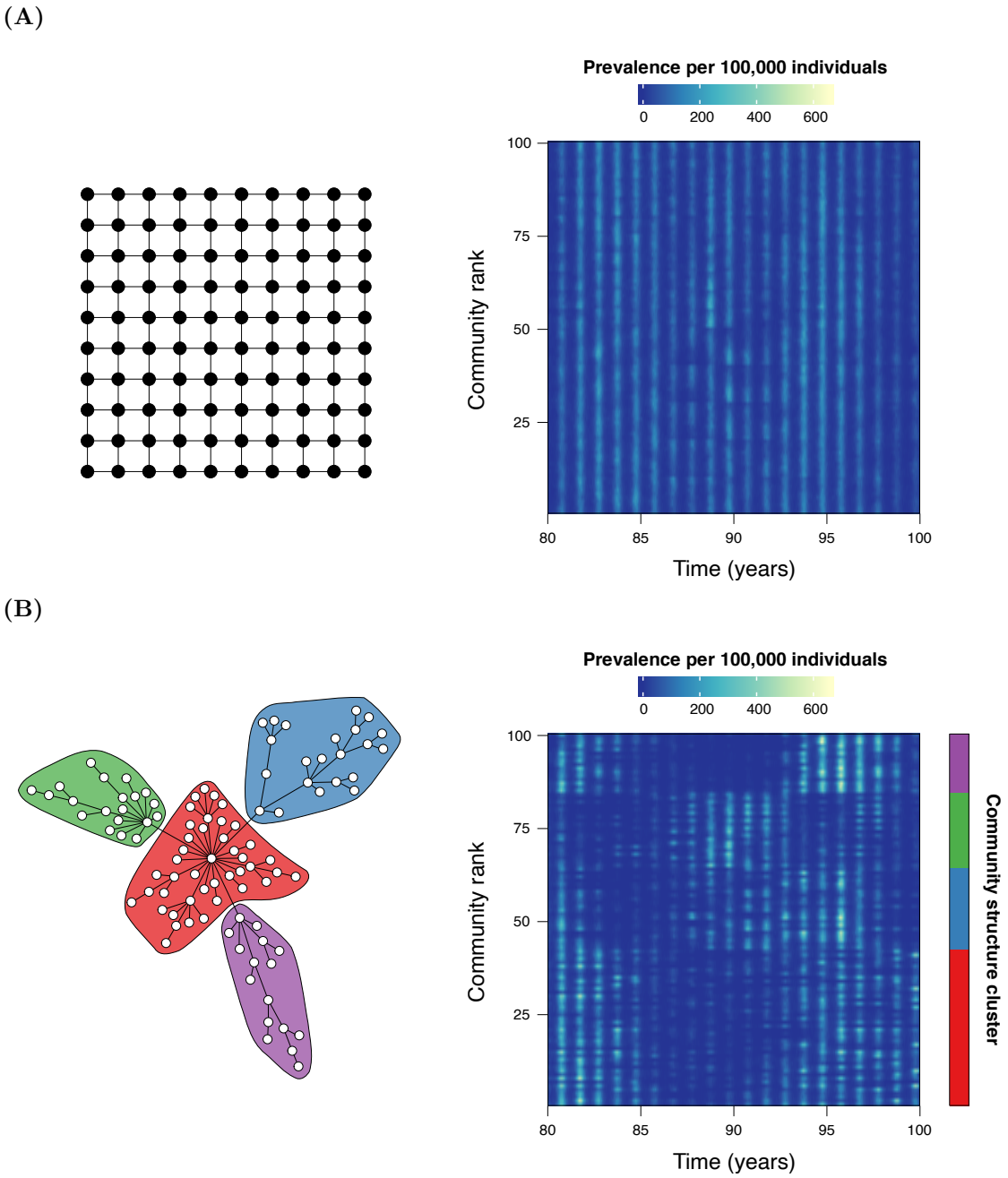

Figure 4.7. Spatio-temporal heterogeneity of dengue prevalence in the lattice and scale-free network. (A) The high connectivity of the lattice permitted the pathogen to disperse more readily, in turn resulting in little spatio-temporal heterogeneity in dengue prevalence, with disease outbreaks synchronised across the network. **(B)** In contrast, the reduced connectivity of the scale-free network restricts dengue dispersion, resulting in high spatio-temporal heterogeneity of prevalence with clear synchronisation across clustered regions of the network. The scale-free community structure was clustered using the Newman-Girvan algorithm (Newman and Girvan, 2004).

connectivity hindering transmission (Figure 4.8A). However, mean prevalence in the urban communities was much greater than rural areas in both the random and scale-free networks as the increased connectivity of the urban communities resulted in greater chance of exposure to the virus through local mobility. Interestingly, the rural communities of the random

network exhibited a very large variance in mean prevalence between simulations. This was mainly due to some rural communities having no local connections. Their infection potential was therefore solely determined by long distance transmission or external introduction, often resulting in extremely large, yet infrequent, epidemic outbreaks of a single serotype.

We also observed a global increase in the risk of serotype extinction from the lattice, to the random, and then to the scale-free network (Figure 4.8B). First, the reduced overall connectivity of the random and scale-free networks resulted in decreased opportunity for the escape of serotypes to communities of sufficiently low seroprevalence. This effect was worsened through the increased organisation of the scale-free network because of the greater number of communities with only a single neighbour. However, the risk of serotype extinction in the urban communities was much lower than global measurements since the greater connectivity of the urban communities naturally facilitated maintenance of serotypes. In alignment with this, the extinction risk of rural communities was very high due to their reduced connectivity. Crucially, the isolated communities in the random network

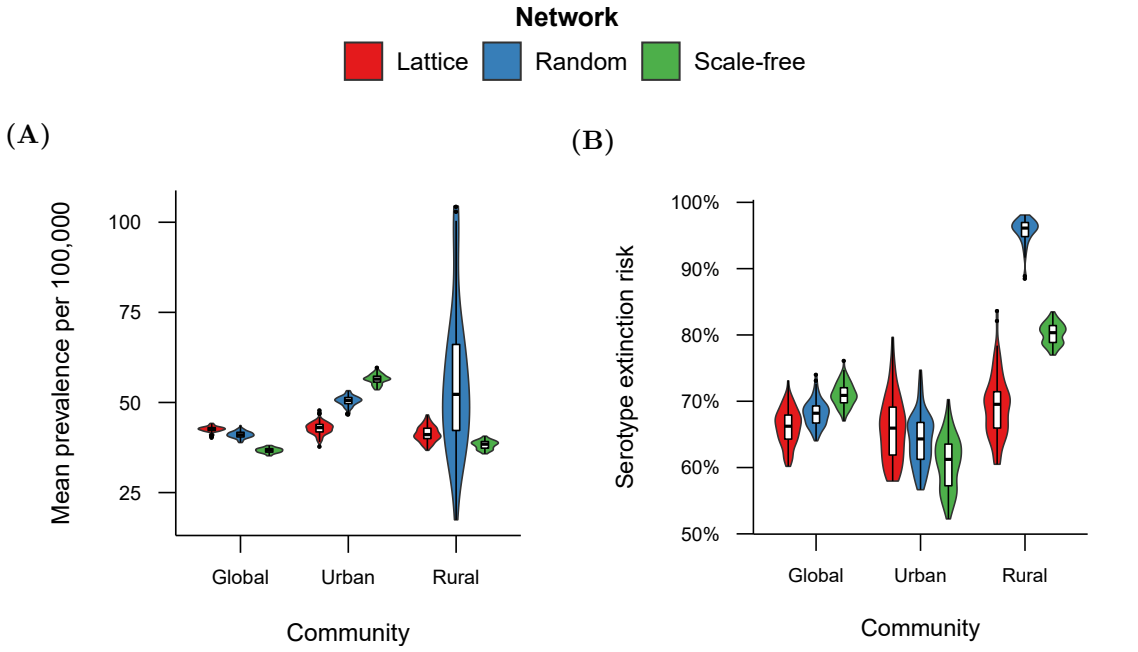

Figure 4.8. Influence of different community structures on global dengue epidemiology and dynamics in urban and rural communities. Endemic dengue simulated in a lattice (red), random (blue) and scale-free (green) network exhibit different quantitative behaviour, in addition to strong heterogeneity across different communities. The high connectivity of urban communities naturally gave rise to higher **(A)** mean prevalence, lower **(B)** local extinction risk of serotypes. All epidemiological metrics were calculated from 100 stochastic simulations of each network.

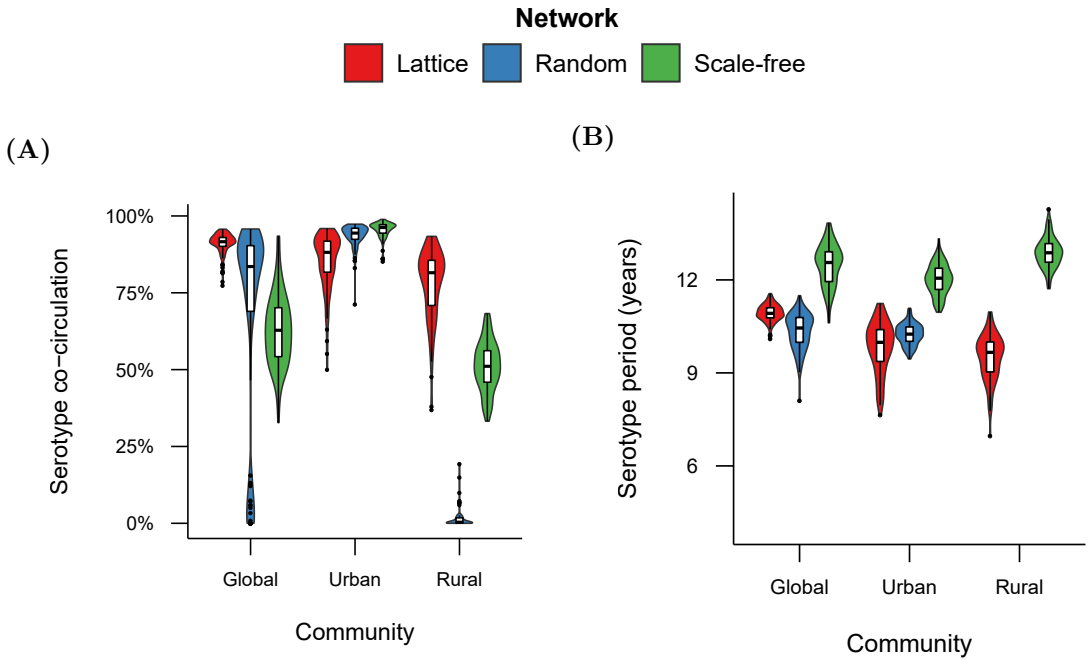

Figure 4.9. Influence of different community structures on global, urban and rural serotype dynamics. Endemic dengue simulated in a lattice (red), random (blue) and scale-free (green) network exhibit different quantitative behaviour, in addition to strong heterogeneity across different communities. **(A)** The high connectivity of urban communities naturally gave rise to higher co-circulation of serotypes, in contrast to the much reduced local connectivity of rural communities in each network. **(B)** However, a high degree of homogeneity across space was observed for the periodicity of serotypes. All epidemiological metrics were calculated from 100 stochastic simulations of each network.

were again exclusively reliant global transmission, and thus exhibited an exceptionally high serotype extinction risk.

Global, urban and rural serotype co-circulation and periodicity

The behaviour of serotype extinction risk was mirrored by the probability that two or more serotypes are co-circulating at a particular point in time in a randomly selected community (Figure 4.9A). Serotype co-circulation was consistently high in the two organised networks, although lower in the scale-free network due to reduced overall connectivity. Intriguingly, serotype co-circulation showed a bimodal response in the random network due to communities being selected at random. Higher co-circulation was observed in connected parts of the network, whereas very low values co-circulation came from the isolated communities. These again struggled to maintain the virus, let alone multiple co-circulating serotypes. This phenomenon became clearer after breaking down the global observation into measurements from specific communities, where the rural communities (or

isolated communities) were indeed shown to have very low serotype co-circulation, and the well-connected urban sub-populations had high co-circulation of serotypes (Figure 4.9A).

The periodicity of serotypes was found to be higher in the scale-free network than in the lattice or random network (Figure 4.9B). The highly connected communities of the scale-free network acted as a bottle-neck for transmission, and so the global resurgence of serotypes was only able to occur once immunity levels had waned sufficiently in these communities. In contrast, the higher, more uniform connectivity of the lattice, and haphazard connectivity of the random network, allowed serotypes to more easily disperse around regions of high seroprevalence. However, the serotype period appeared invariant to the community in which it was measured, with the exception of the isolated communities in the random network, where the high infrequency of outbreaks rendered serotype periodicity incalculable.

4.3.2 Effects of local connectivity in random and scale-free networks

Different levels and forms of local connectivity in the random and scale-free networks were further explored in order ascertain the importance of local connectivity on dengue epidemiology.

Random network connectivity p_E

First, the influence of adjusting the probability of a connection between two randomly selected communities, p_E, was investigated in the random network. Above a connectivity level of approximately 3.5%, $p_E = 0.035$, roughly equivalent to the connectivity of the lattice, the observed epidemiological dynamics remained unchanged. However, below this threshold, seroprevalence at age nine decreased, and variability between annual outbreaks increased globally (Figure 4.10A) as lower values of connectivity resulted in a greater number of isolated communities that depended on long distance and external transmission. Surprisingly, mean prevalence was unaffected by levels of local connectivity in the random

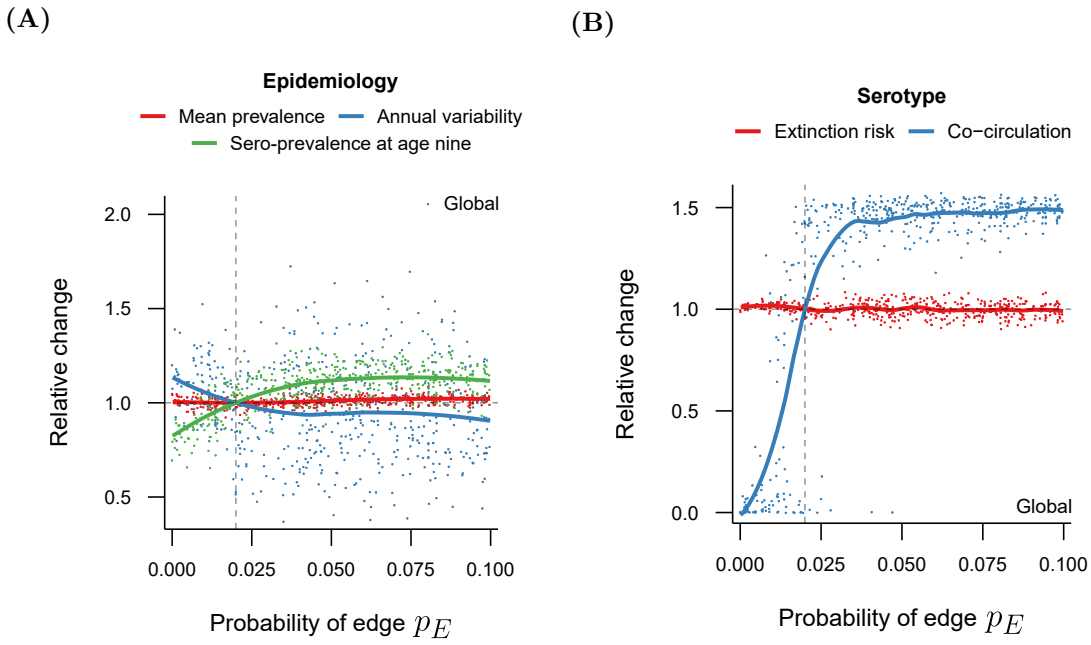

Figure 4.10. Effect of probability of edge creation p_E in random network on epidemiological metrics. **(A)** The frequency of epidemic outbreaks decreased globally while connectivity between communities in the random network was reduced below $p_E = 0.35$. **(B)** More local connections between communities facilitated the co-circulation of serotypes across the entire network. Epidemiological metrics were calculated from 500 stochastic simulations of randomly sampled levels probability of edge between two communities p_E, and LOESS curves were fitted to show the change in general behaviour of each metric over different probabilities of a connection between two randomly selected communities in the random network.

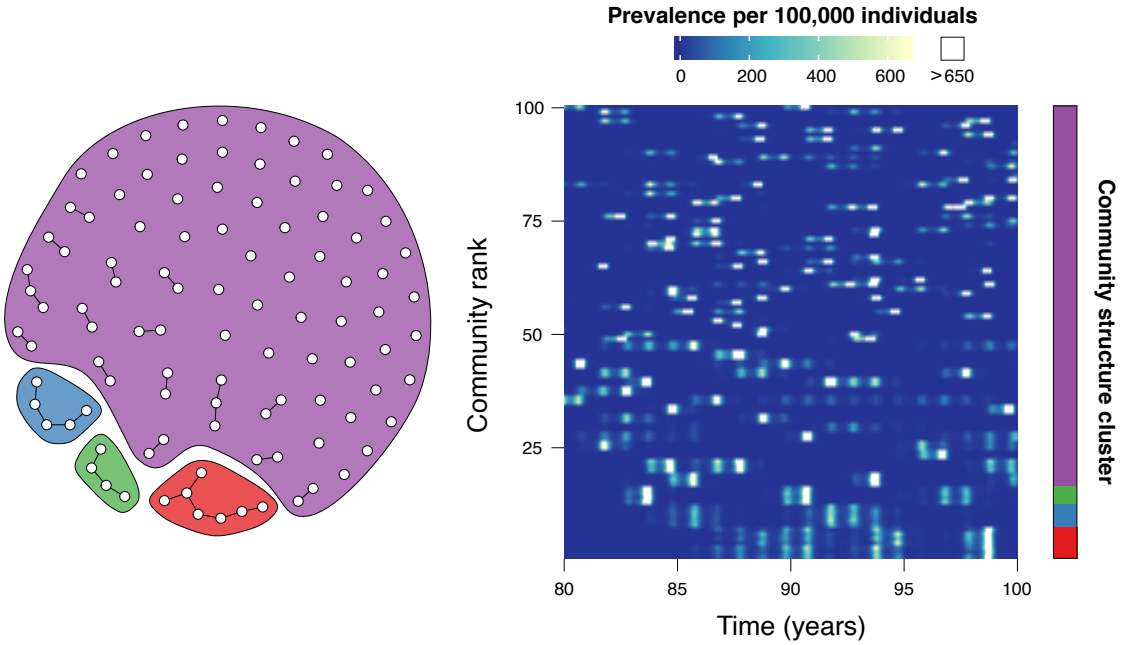

Figure 4.11. Small local connectivity induced isolated epidemic outbreaks in the random network. In random networks with very low local connectivity ($p_E < 0.1$), many communities are isolated from another, resulting in infrequent dengue exposure. Therefore, isolation induces infrequent epidemic outbreaks, driven by long-distance transmissions and external importations. The random network was clustered using the Newman-Girvan algorithm (Newman and Girvan, 2004).

network, because overall exposure was mostly driven by key transmission parameters, such as virus transmissibility and mosquito density. Similar to the result in Chapter 2, reduced local mobility decreased the rate at which seroprevalence accumulated. However, accounting for the age distribution among humans, population-wide seroprevalence levels were unaffected.

As local connectivity was reduced, co-circulation of serotypes dramatically decreased globally and was decoupled from the extinction risk of serotypes (Figure 4.10B). This was because lower connectivity caused infrequent, yet severe, epidemics of only a single serotype within isolated communities (Figure 4.11). However, global dengue prevalence, and thus extinction risk of serotypes, was unaffected.

Scale-free network connectivity n

In a similar manner, we investigated the effects of increasing local connectivity in the scale-free network by performing a sensitivity analysis on the number of connections generated at each step in the Barabási algorithm n. Increasing local connectivity in this way produced similar results to the random network: epidemic variability steadily decreased (Figure 4.12A)

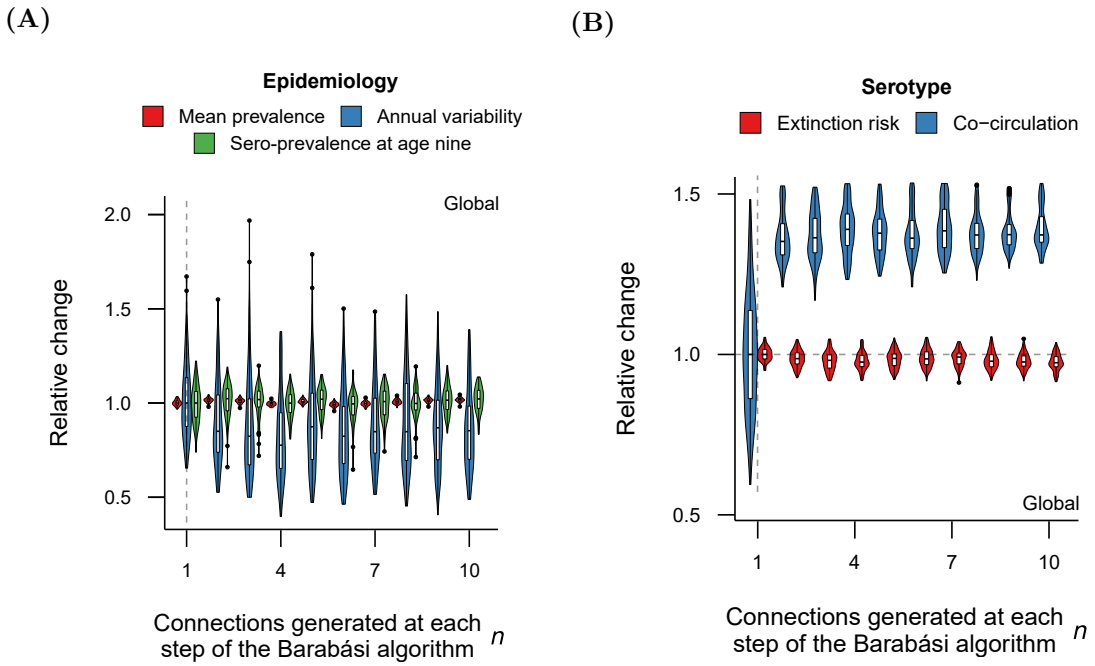

Figure 4.12. Global effects of increased local connectivity n in the scale-free network. (A) Similar to increasing local connectivity in the random network, increasing the number of connections generated at each time step of the Barabási algorithm n, and thereby increasing the local connectivity of the network, decreased annual variability. **(B)** More local connections between communities in the scale-free network removed spatial bottlenecks in transmission, increasing co-circulation of serotypes throughout the network. Epidemiological metrics were calculated from 100 stochastic simulations for each parameter value tested.

and the co-circulation of serotypes increased and then plateaued (Figure 4.12B). These findings were due to the removal of spatial bottlenecks for transmission through a scale-free network with $n = 1$. For $n > 1$, alternative transmission routes permitted previous bottlenecks to be bypassed, facilitating mixing of serotypes throughout the network and increasing stability of the dynamic endemic equilibrium.

Local connectivity of urban centres in the scale-free network

In order to further elucidate the effects of local connectivity on the spatio-temporal heterogeneity of dengue dynamics, the importance of local connectivity to the well-connected urban communities was explored in the scale-free network by varying the power of preferential attachment k. Increasing the power of preferential attachment above a given threshold resulted in a general decrease in mean prevalence, seroprevalence levels and annual outbreak variability (Figure 4.13A). This was because high preferential attachment increased the number of direct connections to a single urban centre, resulting in many communities having only a single connection. In turn, the increased connectivity of the urban community

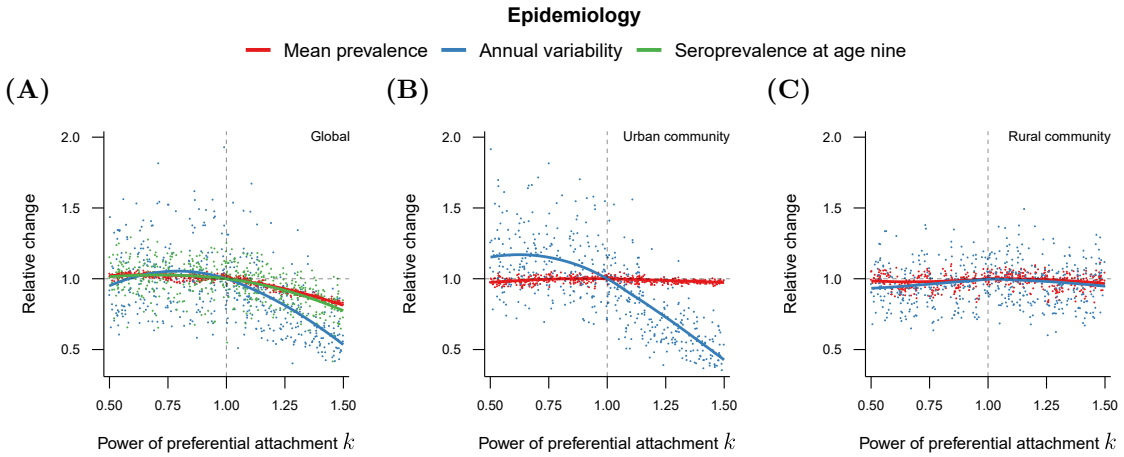

Figure 4.13. Effect of changing the power of preferential attachment k in the scale-free network on epidemiological dynamics. Increases in the power of preferential attachment in the scale-free network increased the focus of connectivity in the network towards urban centres. Above $k = 1$, **(A)** overall viral exposure and annual outbreak variability decreased due to **(B)** increased stability of epidemiological dynamics within urban communities. **(C)** The epidemiology within communities on the periphery of the network remained unchanged. Epidemiological metrics were calculated from 500 stochastic simulations of randomly sampled powers of preferential attachment k in the scale-free network, and LOESS curves were fitted to show the change in general behaviour of mean prevalence, annual variability and seroprevalence levels by age nine over different powers of preferential attachment.

facilitated virus maintenance, giving rise to increased endemic stability (Figure 4.13B). In contrast, lower preferential attachment resulted in epidemiological dynamics that were widely unchanged from baseline parameter values. This was due to the broader distribution of connectivity amongst communities which could sustain sufficiently high spatial heterogeneity in seroprevalence. Interestingly, the mean prevalence and outbreak variability of rural communities also remained mostly unchanged (Figure 4.13C) for all powers of preferential attachment, as the communities on the edge of the network already had low infection potentials. Furthermore, as mean prevalence was unaffected in both the rural and urban communities, the decrease in mean prevalence globally was due to the decrease in overall viral exposure in some intermediate sub-populations.

4.3.3 Effects of heterogeneity in population density

From here onwards, only the scale-free network was considered due to its irregular connectivity and high spatio-temporal heterogeneity in dengue incidence. In order to explore the effects of heterogeneous community size, it was first assumed that the mosquito-to-human ratio was fixed across all communities. That is, any changes to the heterogeneity of humans, denoted by η_H, directly changed the level of mosquito heterogeneity, η_V, to the same level, i.e. $\eta_H = \eta_V$. For simplicity, we then denoted heterogeneity in community size by η such that $\eta = \eta_H = \eta_V$.

From 500 stochastic simulations of randomly sampled values of community size heterogeneity, mean prevalence and seroprevalence levels by age nine marginally increased as heterogeneity increased (see Figure 4.4A). This is because the increased concentration of individuals in well-connected communities resulted in greater exposure of the population overall. Variability between annual outbreaks initially decreased and then increased as more individuals were concentrated into more highly connected communities. However, the variance in the epidemic variability between different simulations of similar heterogeneity levels was extremely large. Larger well-connected communities create the potential for much greater outbreaks when immunity levels are sufficiently low, thus creating a larger contrast between annual outbreaks that involve communities of high population densities versus those that are not.

Epidemiological dynamics

Concentrating more individuals into communities of higher degree, resembling densely populated urban centres, had little influence on the duration between large epidemic outbreaks (Figure 4.14A), but exhibited high variance between simulations. Estimates for the basic reproduction number from initial growth rates were consistent across all levels of heterogeneity (Figure 4.14B). This was because the infection risk of each community across the entire network was the same at the initialisation of an outbreak. However, the estimate depended upon the community in which the outbreak, as introductions to more central, well-connected communities resulted in a higher number of secondary infections than when the virus was introduced into less connected communities on the

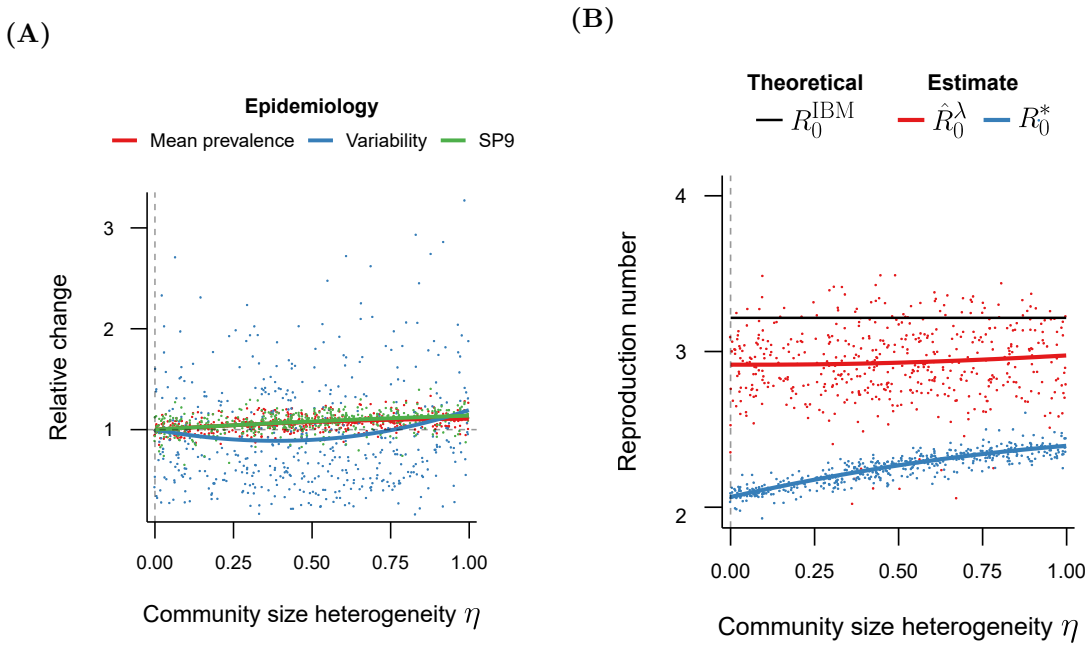

Figure 4.14. Effect of heterogeneity in community size on spread and persistence of dengue. Concentrating individuals into more well connected communities resulted in **(A)** a global increase in the life-time of exposure of individuals to the virus. **(B)** The rate at which the disease emerged was shown to remain unchanged across different levels of community size heterogeneity, although the overall increased exposure of individuals resulted in higher estimates of R_0 from the endemic equilibrium, R_0^*. Epidemiological metrics were calculated from 500 stochastic simulations of randomly sampled levels of heterogeneity $\eta = \eta_H = \eta_V$, and LOESS curves were fitted to show the change in general behaviour of each metric over different heterogeneities in community size.

periphery of the network. Reproduction number estimates from the endemic equilibrium slightly increased with heterogeneity, as concentrating individuals into the well-connected communities increased virus exposure. Furthermore, this R_0 estimate was lower than theoretical expectations because the estimate averaged over the seasonal oscillations in mosquito density and extrinsic incubation period, whereas theoretical expectations were based upon the maximum transmission season.

Inter-epidemic and serotype periodicity

The inter-epidemic period was unaffected under changes in population density heterogeneity because it does not alter the rate at which immunity wanes, and so the potential for, and thus duration between, epidemics remained the same (Figure 4.15A). In contrast, the periodicity of serotype-specific and overall incidence decreased. More heterogeneous population distributions resulted in shortened periods of high incidence. This was because herd-immunity acquired faster in the network globally due to increased population densities

in well-connected communities.

Serotype extinction risk and co-circulation

The local extinction risk of each serotype decreased overall as heterogeneity increased, with a slight increase as the difference in population density between well-connected and poorly-connected communities grew larger (Figure 4.15B). The overall decrease was due to the tendency of larger communities to contain a sufficient number of susceptible individuals and infected mosquitoes to sustain transmission out of season. As heterogeneity was taken to the extreme ($\eta = 1$), the ability for the virus to transmit throughout the network, and thus persist during the off-season, decreased due to the much smaller population densities of the non-central communities. Mirroring the decreasing behaviour of serotype extinction risk, the co-circulation of serotypes greatly increased with heterogeneity. Only at very high levels of population density heterogeneity did we observe a decrease due to higher extinction risks.

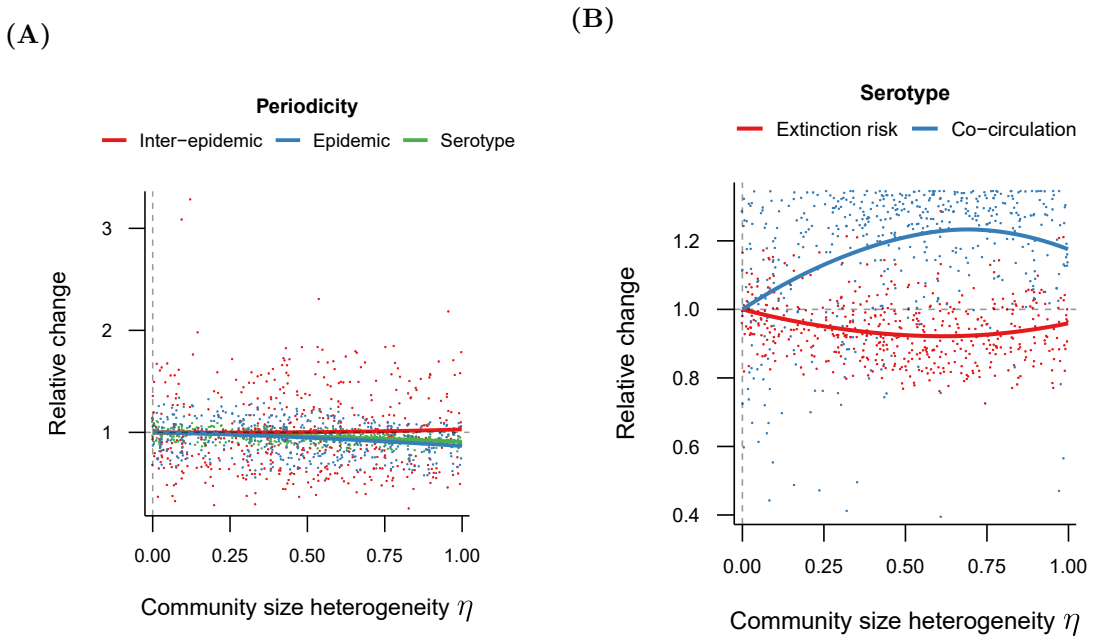

Figure 4.15. Effect of heterogeneity in community size on spread and persistence of dengue. Increased heterogeneity facilitated **(A)** the co-circulation multiple serotypes but reduced the probability of serotype extinction and **(B)** shortened periodic behaviour. The rate at which the disease emerged was shown to remain unchanged across different levels of community size heterogeneity, although the overall increased exposure of individuals resulted in higher estimates of R_0 from the endemic equilibrium. Epidemiological metrics were calculated from 500 stochastic simulations of randomly sampled levels of heterogeneity $\eta = \eta_H = \eta_V$, and LOESS curves were fitted to show the change in general behaviour of each metric over different heterogeneities in community size.

4.3.4 Effects of ecological heterogeneity

Local ecological heterogeneities naturally give rise to differences in vector suitability between communities. To investigate the effects of these ecological heterogeneities on dengue epidemiology, the ratio of mosquitoes to humans was varied by changing the community size heterogeneity for humans and mosquitoes independently.

Global dynamics

Decoupling the relationship between the heterogeneity of humans and mosquitoes gave rise to a landscape of high epidemiological diversity. Generally, higher mosquito-to-human ratios in urban areas than rural communities ($\eta_H < \eta_V$) gave rise to decreased epidemic variability (Figure 4.16A) due to the maintenance of sufficiently high mosquito densities throughout the off season in the well-connected communities. The increased global stability coincided with a reduction in the epidemic period (Figure 4.16B) and an increase in the co-circulation of serotypes (Figure 4.17A). In contrast to the increased maintenance of

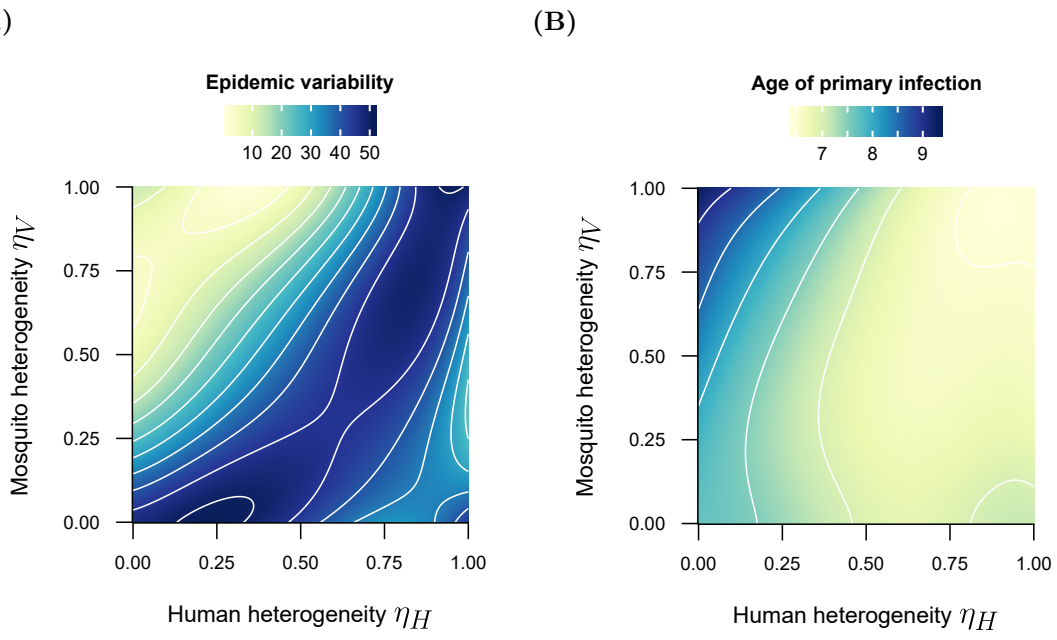

Figure 4.16. Effects of spatial heterogeneity in the mosquito-to-human ratio on global epidemiological dynamics. High mosquito-to-human ratios in better connected communities ($\eta_H < \eta_V$) exhibited **(A)** increased annual stability, **(B)** decreased epidemic periodicity, **(C)** increased co-circulation of serotypes and **(D)** primary age of infection. Epidemiological metrics were calculated from 2500 stochastic simulations of randomly sampled community size heterogeneity η_H and η_V. Quintic polynomials were then assessed and fit across each epidemiological metric to show the general behaviour of each metric on different heterogeneity parameters.

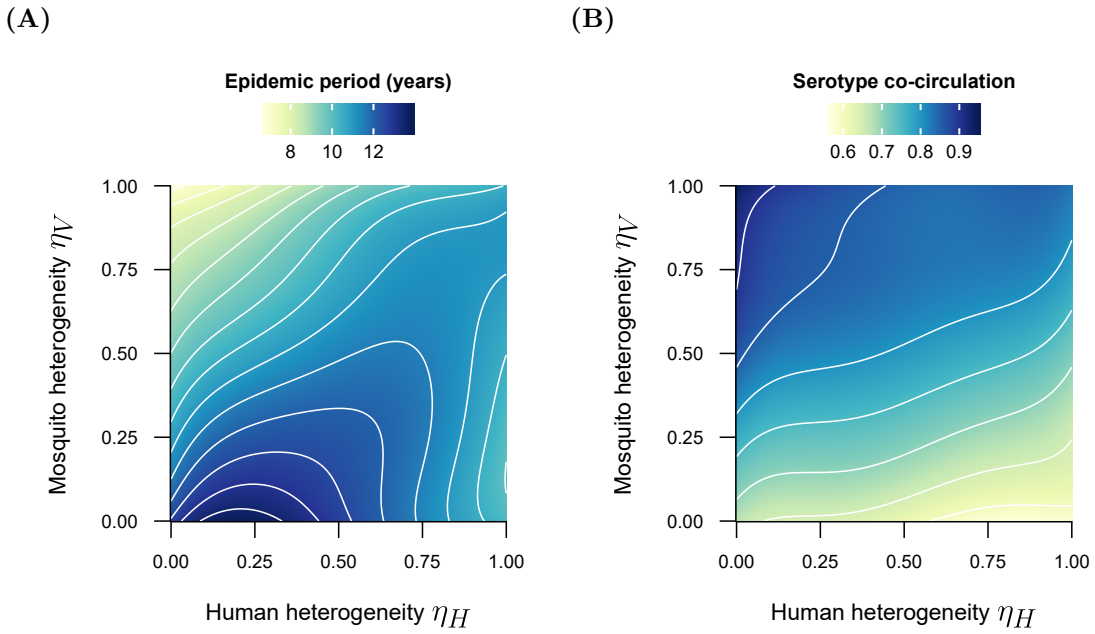

Figure 4.17. Effects of spatial heterogeneity in the mosquito-to-human ratio on global epidemiological dynamics. High mosquito-to-human ratios in better connected communities ($\eta_H < \eta_V$) exhibited **(A)** increased annual stability, **(B)** decreased epidemic periodicity, **(C)** increased co-circulation of serotypes and **(D)** primary age of infection. Epidemiological metrics were calculated from 2500 stochastic simulations of randomly sampled community size heterogeneity η_H and η_V. Quintic polynomials were then assessed and fit across each epidemiological metric to show the general behaviour of each metric on different heterogeneity parameters.

dengue, the mean age of first infection increased globally (Figure 4.17B). This was because although well-connected communities were able to better sustain serotypes, the virus struggled to propagate through the rest of the network, in which some communities had very low numbers of mosquitoes.

The reverse scenario exhibited contrasting results. Higher mosquito-to-human ratios in rural areas than urban communities ($\eta_H > \eta_V$) resulted a consistently high global variability between annual outbreaks. This was because the poor local connectivity of the rural communities resulted in very high virus extinction rates, which was not compensated for by higher mosquito densities. To that end, the co-circulation of serotypes, the epidemic period and the mean age of primary infection remained at similar levels to a network with no heterogeneity in community size.

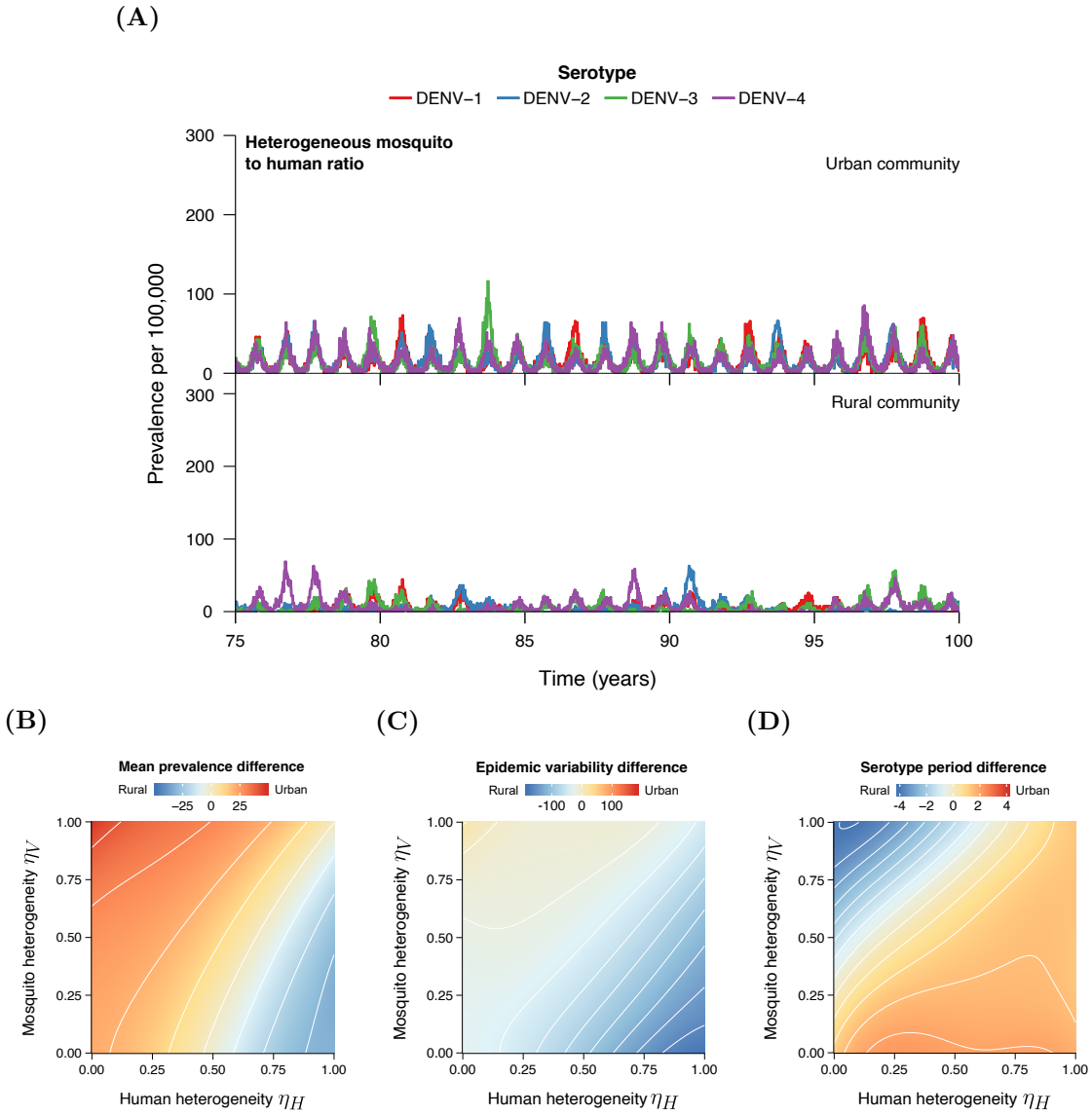

Figure 4.18. Effects of spatial heterogeneity in the mosquito-to-human ratio on urban and rural epidemiology. **(A)** High mosquito-to-human ratios in well-connected communities ($\eta_H < \eta_V$) homogenised the epidemiological dynamics in urban and rural communities. Although **(B)** mean prevalence was higher in urban areas, **(C)** epidemic variability stabilised throughout the network, and **(D)** the periodicity of serotype re-emergence was longer in rural communities, with high concentrations of mosquitoes in well-connected communities than that of poorly connected subpopulations. Epidemiological metrics were calculated from 2500 stochastic simulations of randomly sampled community size heterogeneity η_H and η_V. Quintic polynomials were then assessed and fit across each epidemiological metric to show the general behaviour of each metric on different heterogeneity parameters.

Urban versus rural dynamics

The epidemiological dynamics were further split down into those for urban and rural communities. In contrast to the high spatio-temporal heterogeneity of dengue incidence across the scale-free network under a spatially homogeneous mosquito-to-human ratio (Figure 4.7B), the behaviour of both urban and rural communities became more similar under higher mosquito-to-human ratios in the well-connected communities (Figure 4.18A). Mean prevalence was much higher in urban communities than rural communities when the mosquito-to-human ratios were higher within better connected communities, because the combined poor connectivity and small mosquito densities within rural communities hindered transmission (Figure 4.18B). In alignment, concentrating mosquitoes into communities on the edge of the network resulted in higher transmission in rural than in urban communities. There was little difference in epidemic variability across the network with higher mosquito-to-human ratios in urban centres than rural areas (Figure 4.18C) due to a combination of reduced transmission in the rural communities and overall increased stability within urban areas. This was reflected by the much higher serotype periodicity in rural communities than in urban communities for high ratios in the urban community versus the rural community (Figure 4.18D). In turn, high mosquito densities in poorly connected communities, facilitated faster reintroduction within those areas.

4.4 Discussion

Community structure and human mobility have been implicated as key drivers for the observed heterogeneity in dengue incidence across space and time (Reiner et al., 2014; Velázquez-Castro et al., 2018). However, quantification of the effects of human mobility networks, and spatio-temporal heterogeneity in human demography on dengue epidemiological dynamics remains lacking. Furthermore, the anthropophilic nature of dengue's primary vector, *Aedes aegypti*, induces an urban habitat preference (Ponlawat and Harrington, 2005; Scott et al., 1993a, 2000a; Weeraratne et al., 2013), but it is unclear how spatially heterogeneous mosquito density may influence the spread and persistence of dengue. Here, we extended the individual based model introduced in Chapter 2 to include different community structures and heterogeneity in population density of humans and mosquitoes in order to assess the impact of these factors on dengue epidemiology.

Organised community structure, the scale-free Barabási-Albert network, induced significant spatio-temporal heterogeneity in the epidemiological dynamics of dengue compared to a lattice structure due to the irregularity of community connectivity. Well-connected urban communities in the organised network facilitated co-circulation of dengue's four serotypes. Poorly-connected rural populations, on the other hand, exhibited high risks for local virus extinction, resulting in long time periods of low incidence and increased the risk of large outbreaks in subsequent years. These findings are in agreement with the observed epidemiological dynamics of dengue in Vietnam, which shows enormous spatial heterogeneity in incidence, with regular oscillations in cases nationally and within Ho Chi Minh City, the capital of Vietnam, in addition to irregular rural outbreaks of increased severity (Cuong et al., 2013).

We then found that reducing local connectivity in combination with a random community structure hindered transmission, and halted the co-circulation of serotypes within communities. However, if community structure was organised as in the scale-free network, the virus was able to persist even with minimal local connectivity. These findings imply that the structure of local human movement may be more important driving dengue epidemiology compared with the total level of local human movement in a region. We also

demonstrated that preference of local human movement towards urban centres increased overall endemic stability. At the same time, the epidemiological dynamics of rural communities were unaffected by changes to the preference of local human movement, suggesting long distance human mobility or external importations are more important at driving dengue epidemiology in poorly connected regions. In conjunction with results from Chapter 2, the role of national mobility and international travel is crucial for arboviral emergence and spread (Massad et al., 2013; Polwiang, 2016; Tian et al., 2017; Wesolowski et al., 2015; Wilder-Smith and Gubler, 2008).

There were only minimal global effects in concentrating individuals into well-connected communities - overall incidence slightly increased and the periodicity of serotypes shortened - inferring that heterogeneity in population density across space does not play a large role in overall dengue dynamics, at least while the number of mosquitoes per human is spatially homogeneous. These findings agree with a study by Pavía-Ruz et al. (2018) on dengue seroprevalence in three communities within Yucatan, Mexico, which showed that dengue immunity was highest in the community with lowest population density. On the other hand, after normalising for external introductions, Padmanabha et al. (2012) showed that the number of transmission events per household grew with an increasing numbers of residents. Together with our results, this suggesting that the role of human population density may differ across spatial scales. Our results further demonstrate that greater heterogeneity in community sizes aided dengue persistence. However, this may have been confounded by the increased local connectivity of well-connected, well-populated communities. Desynchronizing the relationship between human and mosquito population density also shifted the epidemiological dynamics of dengue, highlighting the significance of mosquito ecology on arboviral epidemiology and emphasising the continuing need for detailed maps of vector abundance.

In this chapter, we have not explored the influence of human mobility networks on the emergence and establishment of indigenous dengue, as has for example been implicated in China, where annual importation is necessary for regional outbreaks (Lai et al., 2015). Our findings have also implied that the effects of heterogeneity in human population density may vary across different spatial scales, and so lattice community structures could be integrated

within each node of the global network, as in Silva et al. (2007), in order to more deeply explore the interaction between fine-scale and regional epidemiological dynamics. The effects of heterogeneity in mosquito and human population density were explored within a scale-free network, however Broido and Clauset (2019) recently showed that social networks are at best weakly scale-free. Therefore, it may be beneficial to integrate social network data within the model to more realistically infer the relationships between population density and dengue epidemiology. Although it is known that climate influences mosquito demography and susceptibility to dengue (Carrington et al., 2013; Harris et al., 2018; Liu et al., 2017; Messina et al., 2016), it is not also clear from our results how spatio-temporal heterogeneity in climate factors might influence the spread persistence of dengue.

Here, we have shown that community structure plays a crucial role in the epidemiological dynamics of dengue, with sufficient local connectivity required for viral persistence. Furthermore, we have demonstrated that heterogeneity in human population density only minimally affects dengue spread and persistence. Our results highlight that the consideration of mosquito ecology is vital in capturing the epidemiological dynamics of dengue. These findings emphasise the importance of considering community structure and vector ecology to understand dengue epidemiology, which in turn will impact our approaches toward dengue control through community led mosquito management or vaccine deployment.

Chapter 5

Inferring the ecological drivers of arboviral outbreaks

5.1 Introduction

Climate has been shown to play a key role in arboviral transmission dynamics as annual oscillations in temperature, precipitation and humidity induce seasonal fluctuations in vector suitability and virus transmissibility (Caminade et al., 2017; Johansson et al., 2009; Li et al., 2019). Increased temperatures have been associated with faster virus replication rates and shorter extrinsic incubation periods for Zika (Tesla et al., 2018), Chikungunya (Mbaika et al., 2016), dengue (Mordecai et al., 2017; Xiao et al., 2014), and yellow fever (Johansson et al., 2010). Higher temperatures also increase survivorship of mosquitoes, including *Aedes* (Alto and Bettinardi, 2013; Alto and Juliano, 2001), *Anopholes* (Lyons et al., 2013) and *Culex* (Ciota et al., 2014) species, the three most common genera of mosquito-borne disease vectors (Erlanger et al., 2009; Gubler, 2009; Jupp et al., 2002; Sinka et al., 2012). Elevated precipitation and humidity levels have also been shown to correlate with arboviral outbreaks by creating additional mosquito breeding sites and decreasing mosquito mortality, respectively (Harris et al., 2018; Hu et al., 2006; Li et al., 1985; Messina et al., 2016; Scott et al., 2000b). However the exact relationships between these climate factors and vector suitability are often disputed and have yet to be rigorously established within in the field (Alto and Juliano, 2001; Canyon et al., 2013; Da Cruz Ferreira et al., 2017; Descloux et al., 2012; Naish et al., 2014).

Fitting epidemiological models to empirical data provides a way of quantifying these

relationships. This has been done to successfully determine the local epidemiological drivers of Zika (Kucharski et al., 2016; Lourenço et al., 2017). Multiple facets of serological, spatial, phylogenetic or surveillance data can be linked to better elucidate the underlying mechanisms of disease, e.g. Kucharski et al. (2018); Wesolowski et al. (2015). Systems of ordinary differential equations have already been extensively fit to empirical data (Chowell et al., 2007; Lourenço et al., 2017; O'Reilly et al., 2018; Tuncer et al., 2018). However these deterministic frameworks fail to capture the inherent stochasticity and spatio-temporal heterogeneities of arboviral disease, and place strong implicit assumptions on vector ecology (see Chapter 3 page 103). Although discrete-time statistical transmission models (Li et al., 2018), such as Bayesian hierarchical dynamic Poisson models (Martínez-Bello et al., 2017), spatio-temporal risk models (Lowe et al., 2014; Martínez-Bello et al., 2018), and mixed models (Lowe et al., 2017), encapsulate the stochastic dynamics of arboviruses, they fail to capture the associations between epidemiological determinants and essential transmission drivers. Individual based models are arguably better suited to capture the spatio-temporal dynamics of arboviral disease whilst allowing for an unrestricted relationship between extrinsic and intrinsic factors.

Due to computational inefficiency, agent based frameworks have only previously been fit to surveillance data using a maximum likelihood approach, reporting only point estimates of each parameter of interest (Soda et al., 2018). In this chapter, we demonstrate that the computational speed-up from GPU-acceleration (see Chapter 2) allows us to fit a spatially-explicit, climate-driven individual based model to arboviral disease incidence data within a Bayesian framework. For this, we first fit our model to simulated data before considering empirical data from the first Zika outbreak in Brazil. This data was chosen because it has two clean disease outbreaks and as the primary vector for Zika and dengue is the same, dengue's ecological drivers can be reasonably inferred. We also explore the effects of relaxing classical ordinary differential equation model assumptions of mosquito mortality rates and spatial dynamics on inference about mosquito ecology, and emphasise the importance of inter and intra-urban human mobility on vector-borne disease outbreaks. Finally, we show the forecasting potential of this framework, highlighting its potential usage as a real-time analysis tool for epidemiological outbreaks.

5.2 Methods and materials

5.2.1 Individual based model

Here, the individual based model as described in Chapter 2 was used in conjunction with the lattice community structure and fixed local mobility. We modelled the influence of temperature, humidity and rainfall on transmission parameters similarly to Lourenço et al. (2017).

Temperature-dependent parameters

For the relationship between the extrinsic incubation period, $1/\epsilon_V$, and temperature, T, (in Celsius) we applied the formulation by Focks et al. (1993) (see also Focks et al. (1995) and (Otero et al., 2006)), motivated by the enzyme kinetic model by Sharpe and DeMichele (1977), which assumed that replication is determined by a single rate-controlling enzyme. In order to match the daily time steps of the individual based model, the formula was multiplied by 24, as the replication rate given by Focks et al. (1995) is defined per hour:

$$r_V\left(T_k\right) = \frac{8.0616 \times 10^{-2}\left(T_k/298\right)\exp\left[\left(15000/1.986\right)\left(\left(1/298\right)-\left(1/T_k\right)\right)\right]}{1+\exp\left[\left(6.203\times 10^{21}/1.986\right)\left(-2.176\times 10^{-30}-\left(1/T_k\right)\right)\right]},$$

where $r_V\left(T_k\right)$ denotes the developmental rate of the virus in the mosquito at an environmental temperature T_k in degrees Kelvin. Under all environmentally realistic temperatures, the denominator can be approximated by one, thus we reduced the equation to:

$$r_V\left(T_k\right) = 8.0616\times 10^{-2}\frac{T_k}{298}\exp\left[\frac{15000}{1.986}\left(\frac{1}{298}-\frac{1}{T_k}\right)\right],$$

As the parameters in the temperature-dependent equation were estimated from laboratory experiments, where mosquitoes were infected with a fixed titre of virus, we scaled the replication rate linearly to account for the difference between infected blood meal virus titre in the laboratory versus in the field:

$$\epsilon_V\left(T\right) = \alpha r_V\left(T+273.15\right),$$

for some scalar $\alpha > 0$ and $\epsilon_V(T)$ represents the developmental rate of the virus in the mosquito at an environmental temperature T in degrees Celsius.

The dengue virus vector-to-human transmission probability, p_V, generally increases with temperature but then sharply declines at very high temperatures (Lambrechts et al., 2011). This was described using a non-monotonic function given in the work by Mordecai et al. (2017) as:

$$p_H(T) = \begin{cases} 8.49 \times 10^{-4}\, T\,(T - 17.05)\sqrt{35.83 - T}, & 17.05 \leq T \leq 35.83, \\ 0, & \text{otherwise.} \end{cases}$$

Similarly, the human-to-vector transmission probability was influenced by temperature (Mordecai et al., 2017) and given as:

$$p_V(T) = \begin{cases} 4.91 \times 10^{-4}\, T\,(T - 12.22)\sqrt{37.46 - T}, & 12.22 \leq T \leq 37.46, \\ 0, & \text{otherwise.} \end{cases}$$

The relationship between temperature and mean life expectancy of a mosquito ($1/\mu_V$) was obtained using a fourth degree polynomial fit to data from a study by Yang et al. (2009):

$$\mu_V(T) = \frac{8.692}{10} - \frac{1.59}{10}T + \frac{1.116}{10^2}T^2 - \frac{3.408}{10^4}T^3 + \frac{3.809}{10^6}T^4.$$

where $\mu_V(T)$ denotes the mortality rate of vectors at a temperature T in Celsius.

Humidity-dependent parameters

Humidity, $\hat{H}$, has a complex relationship with rainfall and temperature and is known to affect the birth and death rate of vectors. Humidity effects wee therefore modelled explicitly. Humidity was normalized between $[0, 1]$ and then standardised about the mean, $\bar{H}$, as follows (Tran et al., 2013):

$$H = \frac{\hat{H} - \bar{H}}{\sqrt{1 + \left(\hat{H} - \bar{H}\right)^2}}.$$

Therefore $H \in [-1, 1]$. The death rate of vectors was assumed to have a negative relationship with humidity (Alto and Juliano, 2001):

$$\mu_V(H) \propto (1 - H)^{\rho_H},$$

with some power $\rho_H > 0$ and $\mu_V(H)$ denotes the effect of humidity on the mosquito mortality rate. Combining this with the temperature-dependent effects on mosquito mortality gave,

$$\mu_V(T, H) = \eta \mu_V(T)(1 - H)^{\rho_H},$$

for some linear scalar $\eta > 0$.

Rainfall-dependent parameters

Increased rainfall ensures additional breeding sites resulting in increased rates of mosquito oviposition (Scott et al., 2000b) and mosquito density. Rainfall, R, was smoothed using a moving average and then normalised, such that $R \in [0, 1]$.

The expected change in the number of mosquitoes between time t and $t + 1$ (in days), denoted by $\Delta N(t + 1)$ was assumed to follow a logistic growth model with carrying capacity dictated by rainfall.

$$\Delta N(t + 1) = rN(t)\left(1 - \frac{N(t)}{\kappa(R(t) + 1)^{\rho_R}}\right),$$

where κ is the minimum carrying capacity of the environment, r is the maximum growth rate and $\rho_R > 0$ scales the carrying capacity with rainfall.

Model parameters

Unless stated otherwise, the parameters changed from Table 2.2 during individual based model fitting are shown in Table 5.1. We fixed these parameters values to best represent the epidemiology of arboviral disease within an urban setting as informed by the literature. The per day biting rate was set at 0.25 given findings by Mordecai et al. (2017). This change also allowed a more direct comparison to be made between inference from fitting

Table 5.1. The default set of fixed parameter values used in fitting the individual based model to weekly Zika incidence from 2015–2017 in Feira de Santana, Brazil.

Parameter	Description	Value
β	Per day biting rate	0.25
ω	Long distance transmission probability	0.01
ι	External introduction rate per 100,000 individuals per day	5
N_H	Total number of humans	100,000
$N_H/\left\|C\right\|$	Size of each community	100
$1-p_\sigma$	Probability of transmission event dispersing locally	0.25
η_H	Strength of community size heterogeneity in hosts	0
η_V	Strength of community size heterogeneity in vectors	0

our individual based model with parameter inference presented by Lourenço et al. (2017). Long-distance transmission was set at 0.01 to account for the small probability of long-distance human movement in an urban environment (Perkins et al., 2014). Multiple model fits were carried out on different external introduction rates, $\iota \in \{0.5, 1, 5\}$, and local transmission probabilities, $p_\sigma \in \{0.25, 0.5, 0.75\}$, in order to assess the impact of human movement on parameter inference. The parameters that were inferred during model fitting are shown in Table 5.2. We sought to estimate these unobserved parameters as they are not currently well established.

5.2.2 Bayesian specification

We specified the model in a Bayesian framework to determine the posterior distribution, $\pi(\theta|y)$, the set of unobserved parameters, θ, (given in Table 5.2) given outbreak data y, such that,

$$\pi(\theta|y) = \frac{\pi(y|\theta)\,\pi(\theta)}{\pi(y)},$$

where $\pi(.)$ represents a probability distribution, $\pi(y|\theta)$ denotes the likelihood distribution of the data y given the parameters θ, $\pi(\theta)$ is the prior distribution of θ, and $\pi(y)$ is the marginal likelihood of the data y, specified by,

$$\pi(y) = \int_\Theta \pi(y|\theta)\,\pi(\theta)\,d\theta,$$

where Θ denotes the multi-dimensional space of all possible values of θ.

Table 5.2. Unobserved parameters to be estimated by the Metropolis-Hastings MCMC algorithm in conjunction with the individual based model and given climate and incidence data.

Unobserved parameter	Description	Range
α	Scalar for extrinsic incubation period	$(0, \infty)$
η	Scalar for mean adult vector life expectancy	$(0, \infty)$
ρ_R	Non-linear scalar for rainfall seasonality	$(0, \infty)$
ρ_H	Non-linear scalar for humidity seasonality	$(0, \infty)$
κ	Minimum mosquito carrying capacity	$(0, \infty)$
ϕ	Reporting dispersion	$(0, 1)$
p_{obs}	Observation rate	$(0, 1)$

5.2.3 Markov chain Monte Carlo model fitting

For a complex model, the posterior distribution of parameters θ cannot be written in a closed form as the integral $\pi(y)$ is impractical to evaluate analytically. Instead, we fit the model using Markov chain Monte Carlo (MCMC) methods. A Markov chain is a sequence of stochastic events where the probability of each event occurring depends only upon the state of the previous event. Under certain conditions, a Markov chain can be set up such that it converges to a stationary distribution, and thus once converged, the chain produces random samples from the stationary distribution. This allowed us to take a large number of samples from the posterior distribution for the parameters through construction of a Markov chain that converges to the desired posterior distribution. From this, we could estimate the posterior mean and credible intervals of the parameters of interest. The classical Metropolis-Hastings MCMC algorithm that was employed here is defined in Algorithm 1 (Hastings, 1970; Metropolis et al., 1953).

Likelihood

Let y denote the time series of the number of weekly infected cases in the observed data. The likelihood of the data y given parameters θ is often dependent on some unobserved, or hidden, variables, which we denote z and corresponds to the time series of the actual number of weekly infected cases, where y is the number of those actual cases which are

Algorithm 1: Random-walk Metropolis-Hastings MCMC

Let θ_k denote the set of parameters at position $k \in \mathbb{N}_0$ in the Markov chain, then the next set of unobserved parameters in the chain, denoted θ_{k+1}, is generated as follows:

1. Sample a candidate parameter set θ' from some proposal distribution with probability density function $q\left(.|\theta_k\right)$.
2. Compute the acceptance ratio of the proposed parameter set θ' given the previous parameter set θ_k:

$$\alpha_{\text{MCMC}} = \frac{\pi\left(y|\theta'\right)\pi\left(\theta'\right)q\left(\theta_k|\theta'\right)}{\pi\left(y|\theta_k\right)\pi\left(\theta_k\right)q\left(\theta'|\theta_k\right)}.$$

3. Generate $u \sim \text{Unif}\left(0,1\right)$.
4. If $u < \alpha_{\text{MCMC}}$, accept the candidate parameters and set $\theta_{k+1} = \theta'$, else reject the candidate parameter set and use the previously accepted parameter set, thereby setting $\theta_{k+1} = \theta_k$.

Here $\pi\left(y|\theta\right)$ is the likelihood of the case data y given a parameter set θ, $\pi\left(\theta\right)$ is the joint-prior distribution of all unobserved parameters, and $q\left(.\right)$ is the probability density function of the proposal distribution of θ.

observed. Therefore, the likelihood can be written as,

$$\pi\left(y|\theta\right) = \int_Z \pi\left(y|z,\theta\right)\pi\left(z|\theta\right)dz,$$

where Z is the multidimensional space containing all possible values of z. Here, $\pi\left(y|z,\theta\right)$ represents an observation process from the time series of the actual number of weekly infected cases, z, to the observed data y, and $\pi\left(z|\theta\right)$ denotes the probability density of the unobserved states z given parameters θ. As this integral is intractable in practice, we approximate the integral using importance sampling, such that for a given number of simulated data sets, N, we can write the following unbiased estimator for the integral (Andrieu and Roberts, 2009; Beaumont, 2003):

$$\pi\left(y|\theta\right) = \frac{1}{N}\sum_{i=1}^{N}\pi\left(y|z^{(i)},\theta\right)\frac{\pi\left(z^{(i)}|\theta\right)}{q\left(z^{(i)}|\theta\right)},$$

where $z^{(i)}$ denotes the i-th replicate from the model and $q\left(z^{(i)}|\theta\right)$ denotes the probability density of the importance sampling distribution. By simulating from the underlying individual based model, we have $\pi\left(z^{(i)}|\theta\right) = q\left(z^{(i)}|\theta\right)$, and so the estimated likelihood

became:

$$\pi\left(y|\theta\right) = \frac{1}{N}\sum_{i=1}^{N}\pi\left(y|z^{(i)},\theta\right).$$

Each data set was a weekly time series, and the observation process was assumed independent given the hidden states z, therefore,

$$\pi\left(y|\theta\right) = \frac{1}{N}\sum_{i=1}^{N}\prod_{t=0}^{T-1}\pi\left(y_t|z_t^{(i)},\theta\right),$$

where T is the maximum number of weeks in the data, y_t is the incidence at week t in the observed data, and $z_t^{(i)}$ is the incidence in week t of the i-th replicate simulated from the model. Finally, we assumed that the observation process is negatively binomially distributed with some observation rate p_{obs} and dispersion parameter ϕ to account for under-reporting of the true incidence:

$$\begin{aligned}\pi\left(y|\theta\right) &= \frac{1}{N}\sum_{i=1}^{N}\prod_{t=0}^{T-1}\pi\left(y_t|z_t^{(i)},\theta\right),\\ y_t|z_t^{(i)} &\sim \text{NegBinom}\left(\phi, \frac{\phi p_{obs} z_t^{(i)}}{1+\phi p_{obs} z_t^{(i)}}\right),\end{aligned}$$

where the negative binomial distribution was parametrised with mean $\mathbb{E}\left[y_t\right] = p_{obs}z_t^{(i)}$ and variance $\mathbb{E}\left[y_t\right] + \phi\mathbb{E}\left[y_t\right]^2$.

It has been shown that substitution of an unbiased estimate of $\pi\left(y|\theta\right)$, such as above, into the classical Metropolis-Hastings MCMC algorithm, Algorithm 1, produces samples from the true posterior distribution of y given θ in probability (Andrieu and Roberts, 2009).

Prior distributions

Weakly informative priors were inferred from field and laboratory experiments (for Zika and dengue), the parametrization of our simulation model, and previous model fits from the scientific literature. These are listed in Table 5.3. The prior distribution for the scalar of the extrinsic incubation period, α, was chosen to give an extrinsic incubation period at $30°C$ between 3–14 days (Krow-Lucal et al., 2017). Similarly, the prior for the scalar of

Table 5.3. The prior of each parameter used during fitting the stochastic individual based model to outbreak incidence data, where TruncNorm (μ, σ, a, b) denotes a truncated normal distribution with mean μ, standard deviation σ, lower bound a, and upper bound b.

Parameter	Prior
α	TruncNorm $(2.5, 0.5, 0, \infty)$
η	TruncNorm $(3, 0.5, 0, \infty)$
ρ_R	TruncNorm $(1, 0.4, 0, \infty)$
ρ_H	TruncNorm $(1, 0.4, 0, \infty)$
κ	TruncNorm $(3, 1, 0, \infty)$
ϕ	Beta $(20, 80)$
p_{obs}	Beta $(10, 90)$

mosquito longevity, η, was chosen sufficiently wide to give mean mosquito life expectancy at $30°C$ between 5–30 days Maciel-de Freitas et al. (2007); Marinho et al. (2016); Muir and Kay (1998). Weak priors were chosen for the effect of rainfall on mosquito density, ρ_R, effect of humidity on mosquito longevity, ρ_H, and minimum mosquito to human ratio κ given the lack of estimates from the literature. The prior for the over-dispersion of observed cases, ϕ, was chosen to account for uncertainty in the reporting rate over time due to possible misclassification of different arboviral diseases and changes to surveillance systems over the study period. The prior for the reporting rate, p_{obs}, was selected to reflect observation rates estimated in other model-based studies Kucharski et al. (2016); Lourenço et al. (2017); Shutt et al. (2017).

Proposal distribution

Candidate parameters θ' were sampled from a multivariate normal distribution with probability density function $q(.)$ using a random-walk adaptive routine as detailed in Roberts and Rosenthal (2009). The routine uses the existing covariance structure in the Markov chain of θ to adapt toward an optimal proposal distribution that promotes well-mixing chains. Using the optimal scaling parameter defined in Sherlock et al. (2015), we define the proposal distribution here as,

$$q\left(\theta'\right) = (1-\beta)N\left(\theta_k, (2.562)^2\,\Sigma_k/d\right) + \beta N\left(\theta_k, (0.1)^2 I_d/d\right),$$

where θ_k is the set of parameters at the k-th position in the chain, Σ_k is the current estimate of the covariance structure of the posterior distribution, defined as the empirical covariance matrix from the current samples for θ at iteration k, d is the dimension of Σ_k, β is a small positive constant (here, we take $\beta = 0.05$), and I_d is the identity matrix of dimension d.

Convergence

In order to allow the chains to converge to the posterior distribution, unless stated otherwise, the first 20,000 iterations of each model fit were discarded as burn-in. Furthermore, in order to assess the independence of parameter initialisation on the convergence of the model, each unobserved parameter in the Markov chain was initialised from multiple randomly sampled points of each prior. Convergence to the same posterior distribution was then assessed through visual inspection of the Markov chains. Thereafter, each parameter in the Markov chain was initialised from the mean of each prior.

Optimisation

The likelihood of the data y given parameters θ depended upon the number of simulated data sets, N, generated per proposed parameter set. At the cost of computational runtime, a higher number of simulations per step increases decreased the variance in the estimated likelihood, $\pi(y|\theta)$, and so the acceptance rate of the algorithm increased and in turn, the mixing chains improved. To balance computational cost with how well-mixed each of the parameters were, N was selected such that the variance of the log of the estimate of the marginal likelihood was approximately equal to one (Doucet et al., 2015; Pitt et al., 2012; Sherlock et al., 2015). As a result, unless stated otherwise, the number of simulated time-series per step of the MCMC, N, was set to be equal to thirty.

5.2.4 Incidence and climate data

The presented climate and weekly notified case data of Zika in Feira de Santana, Brazil from 1st February 2015 to 31st December 2016 were taken from Lourenço et al. (2017).

5.3 Results

As a proof of concept, we first fit the model to simulated data from the model itself, before fitting to weekly notified case data of Zika in Feira de Santana. Because there was only one circulating serotype during the first Zika outbreak in Brazil, both cases only considered a single serotype and a fully susceptible host population.

5.3.1 Model fit to simulated incidence data

Table 5.4. Values of the unobserved parameters used to generate (simulated) weekly case data.

Unobserved parameter	Description	Value
α	Scalar for extrinsic incubation period	3
η	Scalar for mean adult vector life expectancy	2
ρ_R	Non-linear scalar for rainfall seasonality	1.5
ρ_H	Non-linear scalar for humidity seasonality	0.5
κ	Minimum mosquito carrying capacity	2.3
ϕ	Reporting dispersion	0.05
p_{obs}	Observation rate	0.2

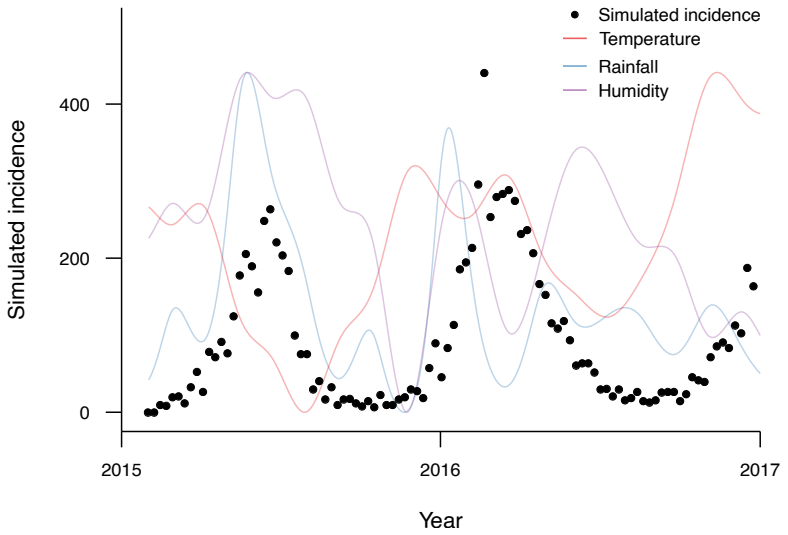

Figure 5.1. Observed (simulated) incidence and climate dynamics. Together with climate data from Feira de Santana, the individual based model was executed on pre-selected unobserved parameter values shown in Table 5.4, resulting in two disease outbreaks over two years.

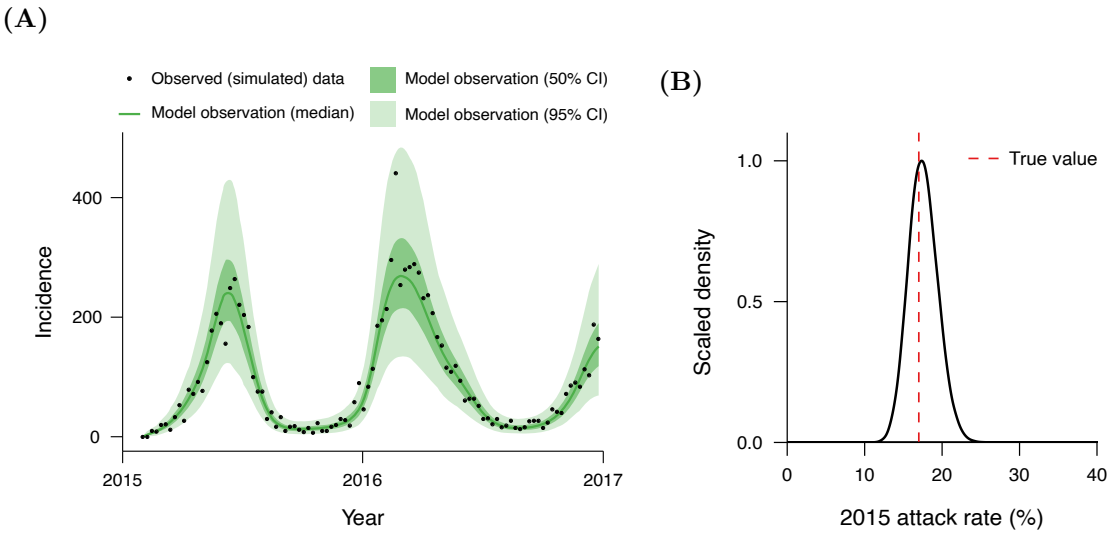

Figure 5.2. Posterior predictive distribution and attack rate of simulated data. **(A)** The temporal dynamics of the distribution of simulated observed cases exhibited similar underlying dynamics to the incidence data generated from the individual based model. **(B)** The inferred percentage of the total population infected during the first year contained the true attack rate of the simulated data. The posterior predictive distribution was calculated from randomly sampling 1,000 sets of parameter values from the posterior distribution and simulating these within the individual based model. Weekly observed cases are then randomly sampled from the negative binomial distribution, representing the observation process from total to notified cases, using the dispersion parameter, ϕ, mean probability of observing a single case, p_{obs}, and simulated total weekly cases from the individual based model.

In order to demonstrate that the unobserved parameters can be inferred from relatively sparse data, the model was first fit within a Bayesian framework to incidence data simulated from the individual based model with the preselected unobserved parameter values listed in Table 5.4. The incidence data exhibited two disease outbreaks, one in the middle of the first year, peaking with high rainfall and humidity levels, and the other at the start of the second year (Figure 5.1).

Model generated observations matched the temporal signature of the original incidence data (Figure 5.2A). There was large uncertainty around the simulated dynamics, however this was not due to the stochasticity of the individual based model itself, but due to the underlying observation process. That is, we selected an overly-dispersed observation process, which created high variance between credible outcomes in observed incidence during both epidemic peaks. In contrast, the variance in the estimated percentage of the population that were affected during the first year of the outbreak, or 2015 attack rate, was small, with the true attack rate of the original simulated data contained within the 95% credible interval of the attack rate (Figure 5.2B).

These findings demonstrated that the individual based model could accurately fit to incidence data. However, fitting to this data came as no surprise, as a model with so many parameters is likely to give a good fit. What was important here, were the inferred unobserved parameter values themselves.

Parameter recovery

The 95% credible intervals of the posterior distribution of each unobserved parameter also included the true values used to generate the incidence data to which the individual based model was fit (Figure 5.3). We also tested parameter recovery for six other parameter sets

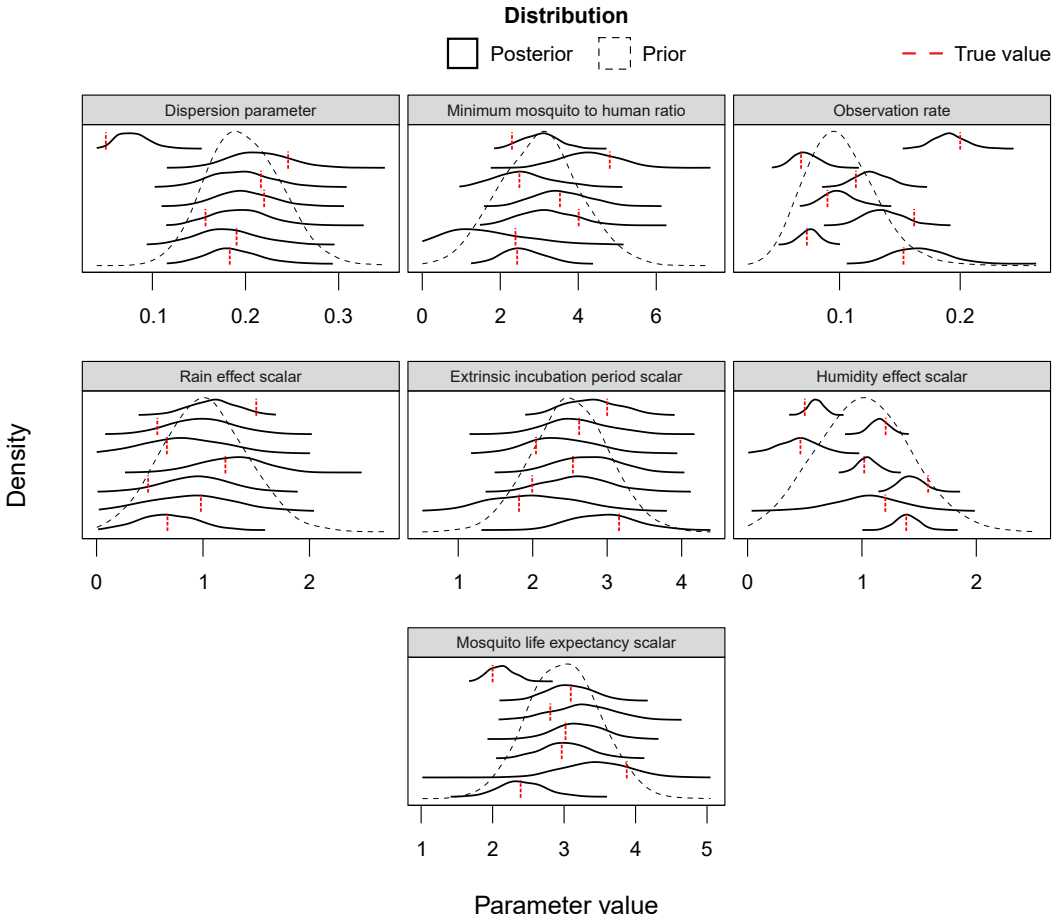

Figure 5.3. Posterior distributions of unobserved parameters. All estimated posterior distributions from the fitting process were unimodal and often improved upon the weakly informative prior distributions. The inferred unobserved parameters were the dispersion parameter for the observation process that mapped total weekly cases to notified cases, ϕ, the minimum mosquito-to-human ratio, κ, the probability of a case being notified, p_{obs}, the effect of rainfall on mosquito density, ρ_R, the scalar that influences the extrinsic incubation period, α, the effect of humidity on mosquito longevity, ρ_H, and the scalar that controls mosquito mortality rates, η. Posterior distributions were calculated from chains of 125,000 iterations with a burn in period 20,000 iterations. Here, seven posterior distributions are shown from model fits to data generated from seven different parameter sets (including those in Table 5.4).

randomly sampled from the prior, with results consistent with the parameters presented in Table 5.4. We found that the true parameter values for all parameters were reasonably well-inferred, however true values for the rainfall effect scalar, ρ_R, and the minimum mosquito to human ratio, κ, were less likely to be inferred than the other parameters across all seven simulated parameter sets. The high uncertainty in the these parameters was likely a result of strong correlations between the inferred values for each. There was also large uncertainty over the inferred over-dispersion parameter because precisely estimating a reasonably large overdispersion parameter from only 100 samples (the length of the presented time series) is not possible.

Correlations of unobserved parameters

There was a strong negative correlation (Pearson's $r = -0.68$) between the minimum mosquito-to-human ratio and the effect of rainfall on the mosquito-to-human ratio (Figure 5.4A). In order to achieve the same transmission potential during outbreaks, smaller baseline values of mosquito density required an increasing effect of rainfall so that the mosquito density during outbreaks is maintained. Similarly, there was a strong positive correlation ($r = 0.62$) between the minimum mosquito-to-human ratio and the effect of humidity on mosquito longevity (Figure 5.4B). This was because lower mosquito densities throughout periods of low transmission required increased mosquito longevity in order sustain sufficient transmission potential.

Key transmission parameters were well-inferred

To demonstrate the relationships between key epidemiological parameters, the posterior distributions of these parameters were translated into distributions of the mosquito-to-human ratio and mosquito life expectancy at the peak of the first epidemic. The inferred estimates of the human to mosquito ratio and mosquito life expectancy during the first outbreak were well-inferred, with the true values of each contained within the 95% credible intervals of each distribution (Figure 5.5).

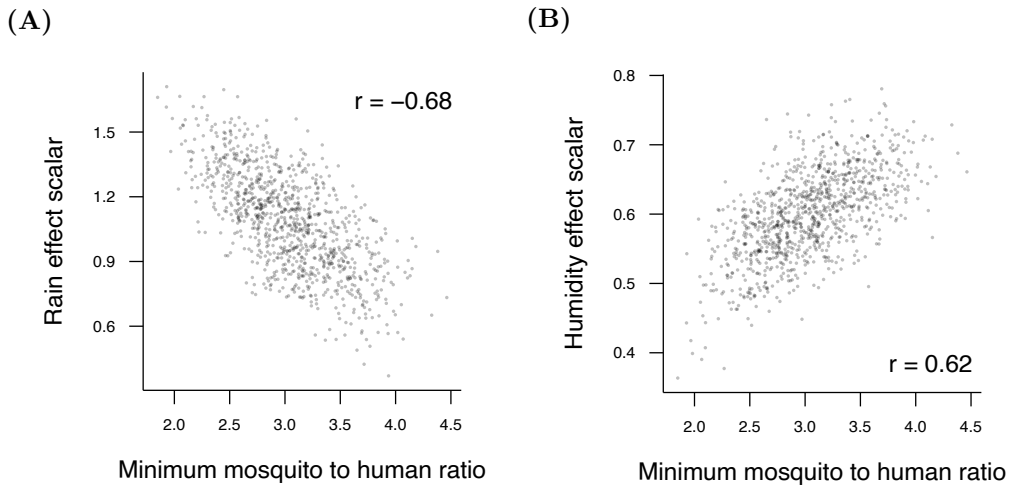

Figure 5.4. Correlation between rainfall, humidity and the minimum mosquito-to-human ratio. (A) There was a strong negative correlation between the effect of rainfall on mosquito density and the minimum mosquito-to-human ratio. **(B)** Similarly, there was a strong positive correlation between the minimum mosquito-to-human ratio and the effect of humidity on mosquito longevity. Pearson's correlation coefficients, r, shown were calculated from 2000 samples from the posterior distribution.

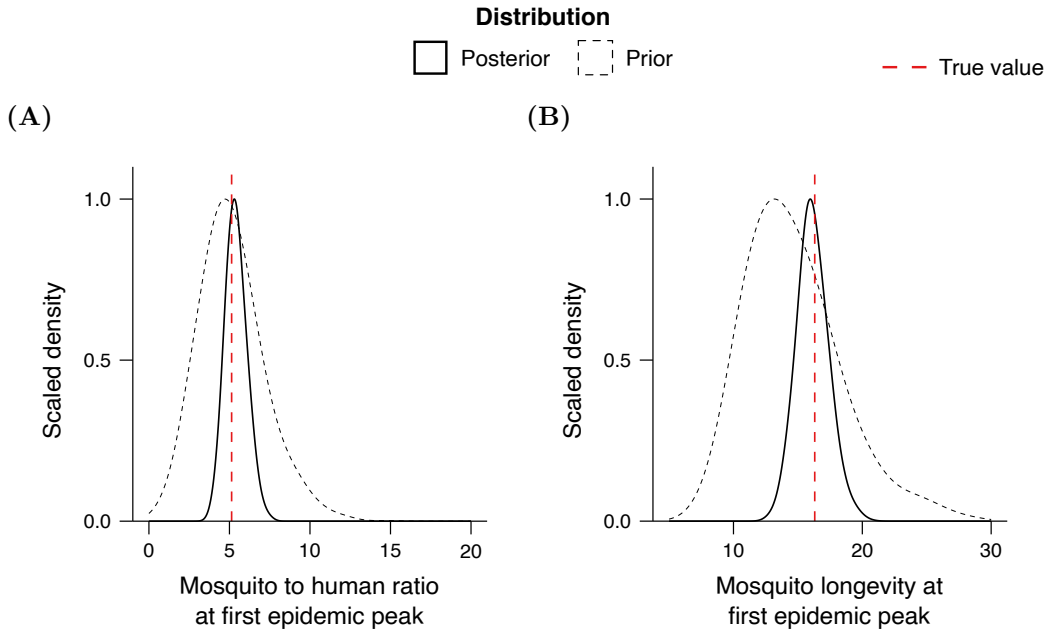

Figure 5.5. Key transmission parameters can be well-inferred from sparse data. By fitting the modelling framework to incidence data that was generated from the individual based model itself, the posterior distributions for **(A)** the mosquito-to-human ratio and **(B)** mean mosquito life expectancy contained the true parameter values used to generate the incidence data.

5.3.2 Model fit to empirical incidence data

From February 2015 to December 2016, Feira de Santana experienced two outbreaks of Zika: a large epidemic from April 2015 to August 2015 when rainfall and humidity levels were at their maximum over the two year period, and a small outbreak at the start of 2016

when temperature, humidity, and rainfall were consistently high (Figure 5.6). Now that we have shown that the model can capture the temporal dynamics of simulated incidence data, the individual based model was fit to weekly notified cases of Zika in Feira de Santana, Brazil.

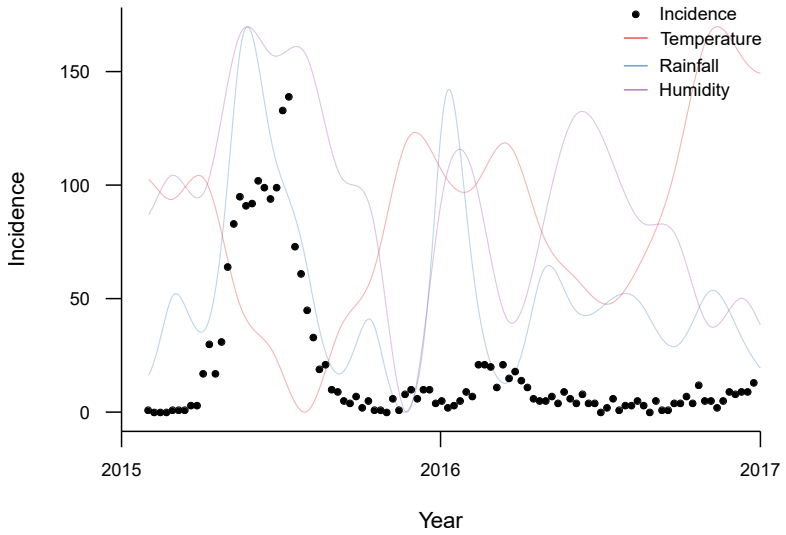

Figure 5.6. Observed incidence and climate dynamics. The individual based model was fit to weekly notified cases of Zika in Feira de Santana, Brazil between February 2015 and December 2016. There was one large epidemic during 2015 and a smaller outbreak at the start of 2016, both which correlate with elevated levels of rainfall and humidity, and the latter with high temperatures.

Fitting the model yields simulated dynamics that behave similarly to the dynamics of the empirical data, with a large epidemic from April 2015 to August 2015, and a much smaller outbreak at the start of 2016 (Figure 5.7A). There were high levels of uncertainty in the distribution of model generated observed cases during the 2015 outbreak because of the large variance in the underlying observation process. Furthermore, the percentage of the total human population infected, or attack rate, of around 50% during 2015 (Figure 5.7B) with very large uncertainty, between 20% and 70%.

The chains of accepted unobserved parameter values were well-mixed (Figure 5.8) and converged in probability to the posterior distributions of each unobserved parameter. All posterior distributions were unimodal, and the majority of posteriors narrowed from the weakly informative prior distributions that were set (Figure 5.9). The dispersion parameter, ϕ, for the negative binomial observation process from total weekly cases to notified weekly cases was slightly lower than expected, but exhibited a high degree of uncertainty. The posterior for the minimum mosquito-to-human ratio, κ, constricted to values between

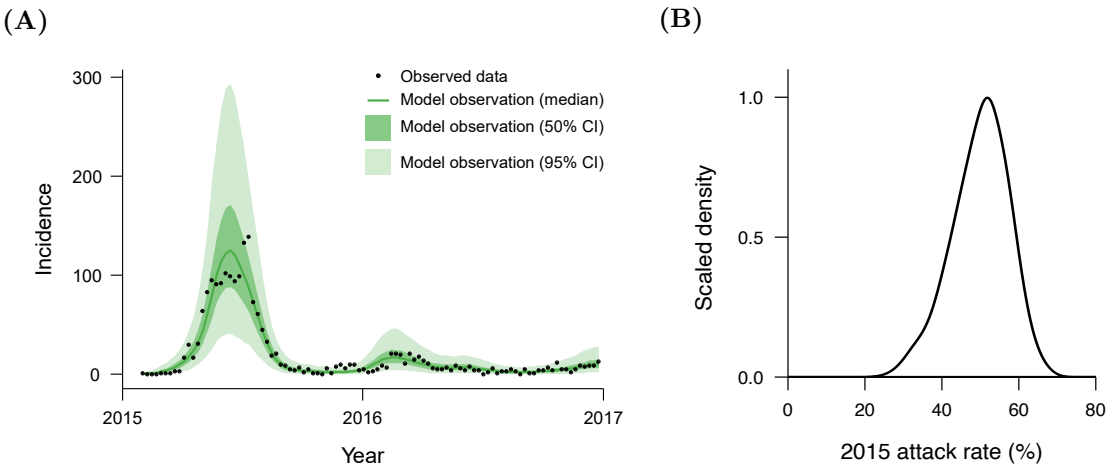

Figure 5.7. Posterior predictive distribution and attack rate. (A) The temporal dynamics of the distribution of simulated observed cases exhibited similar underlying dynamics to the weekly notified cases for Zika in Feira de Santana, Brazil, between 2015 and 2017. **(B)** The percentage of the total population infected during 2015 was inferred to be around 50%. The posterior predictive distribution is calculated from randomly sampling 1,000 sets of parameter values from the posterior distribution and simulating these within the individual based model. Weekly observed cases are then randomly sampled from the negative binomial distribution, representing the observation process from total to notified cases, using the dispersion parameter, ϕ, mean probability of observing a single case, p_{obs}, and simulated total weekly cases from the individual based model.

2.0 and 6.0, as a sufficiently high baseline level of mosquito density was required for out of season transmission. Interestingly, the posterior distribution of the observation rate, p_{obs}, greatly shifted and narrowed from the prior to values of less than 5%. The posterior distributions for the scalar that influences mosquito mortality rates, η, also exhibited a strong shift towards lower values from the prior distribution, and thus in the direction of longer mean mosquito life expectancy. In contrast, the scalar that controls the replication rate of the virus, α, moved towards higher values from the prior distribution, resulting in shorter extrinsic incubation periods.

Fitting the model to the empirical data suggests that the effect of rainfall on mosquito population density, ρ_R, was weak (less than a linear response). This means that rainfall only caused small amplitude oscillations in mosquito density. However due to the association of high rainfall levels with large numbers of reported cases, the maximum number of mosquitoes per human was also found to be highest at peak incidence (Figure 5.10A). In contrast, the strong effect of humidity on mosquito longevity yielded pronounced oscillations in mean mosquito life expectancy that also correlated with incidence (Figure 5.10B). However, the relatively low temperatures during the 2015 outbreak modulated mosquito longevity from

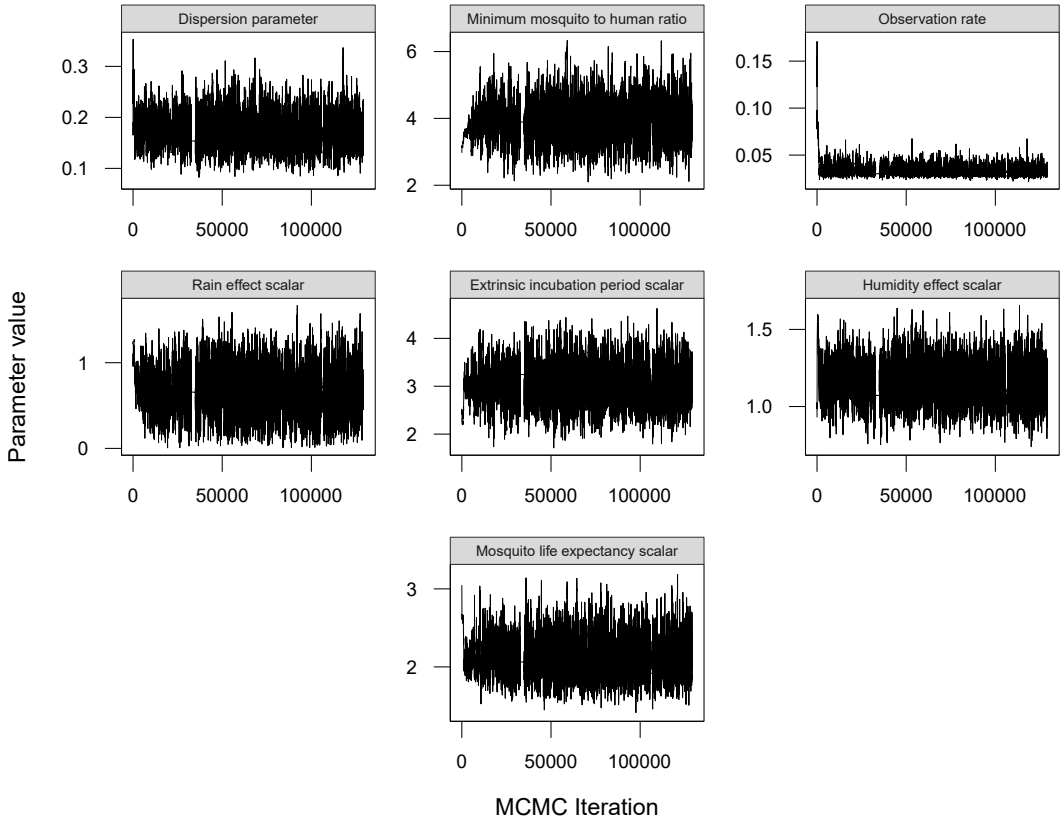

Figure 5.8. Markov chain Monte Carlo (MCMC) chains. Fitting the individual based model with the pseudo-marginal method with $N = 30$ particles per iteration to the empirical data yielded chains of accepted unobserved parameter values that were well-mixed.

extremely high values. The extrinsic incubation period was also found to be consistently low throughout the study period, maximising during the 2015 epidemic due to relatively low temperatures (Figure 5.10C).

Similarly to fitting to the simulated data, we also found correlations between some of the inferred unobserved parameters (Figure 5.11). The minimum mosquito-to-human ratio, κ, was found to be moderately negatively correlated with the effect of rainfall on mosquito population density, ρ_R, with a Pearson's correlation coefficient of $r = -0.54$. Lower mosquito densities required an increased influence of rainfall on mosquito density in order to maintain sufficiently high maximum mosquito-to-human ratios, and thus transmission potential, during periods of high transmission. Furthermore, the minimum mosquito-to-human ratio was positively correlated ($r = 0.5$) with the mosquito life expectancy scalar, η. This was because in order to sustain transmission potential, lower mosquito densities need to be counterbalanced by higher mosquito life expectancies, gained through lower scalars for mosquito longevity. The effect of rainfall on mosquito density was negatively correlated

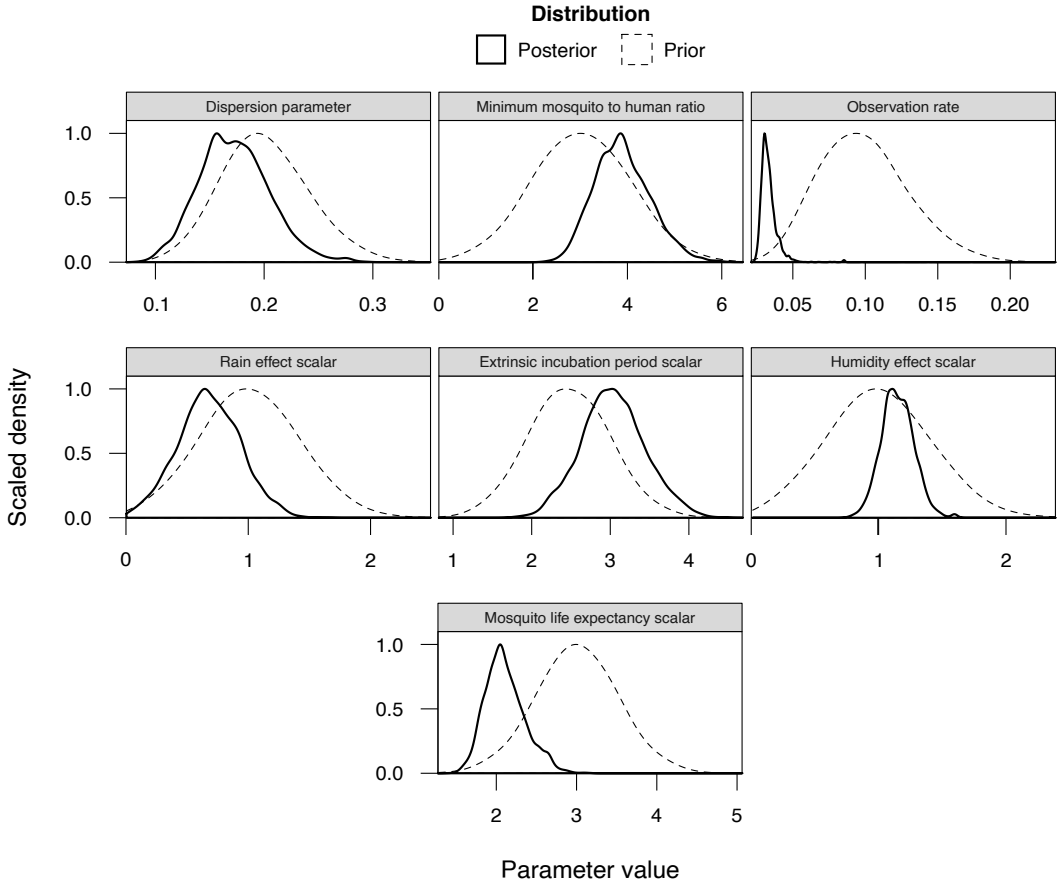

Figure 5.9. Posterior distributions of unobserved parameters. All estimated posterior distributions from the fitting process were unimodal and improved upon the weakly informative prior distributions. The inferred unobserved parameters were the dispersion parameter for the observation process that mapped total weekly cases to notified cases, ϕ, the minimum mosquito-to-human ratio, κ, the probability of a case being notified, p_{obs}, the effect of rainfall on mosquito density, ρ_R, the scalar that influences the extrinsic incubation period, α, the effect of humidity on mosquito longevity, ρ_H, and the scalar that controls mosquito mortality rates, η. Here, posterior distributions were calculated from chains of 125,000 iterations with a burn in period 20,000 iterations.

($r = -0.5$) with the effect of humidity on mosquito longevity, ρ_H, due to the strong positive correlation between the temporal dynamics of rainfall and humidity ($r = 0.73$). In line with the previous results, there was a moderate positive correlation ($r = 0.46$) between the effects of humidity on mosquito longevity and the scalar for mosquito mortality rates. Amplified oscillations in mosquito longevity need to be offset by a reduction in mean mosquito longevity, and thus an increase in the corresponding scalar, to maintain high transmission during the 2015 outbreak. Expectedly, the scalar influencing mosquito mortality rates was positively correlated with the scalar controlling the extrinsic incubation period ($r = 0.34$) as shorter mosquito life expectancy compensates for shorter extrinsic incubation periods, keeping the mean duration of infectivity in mosquitoes consistent.

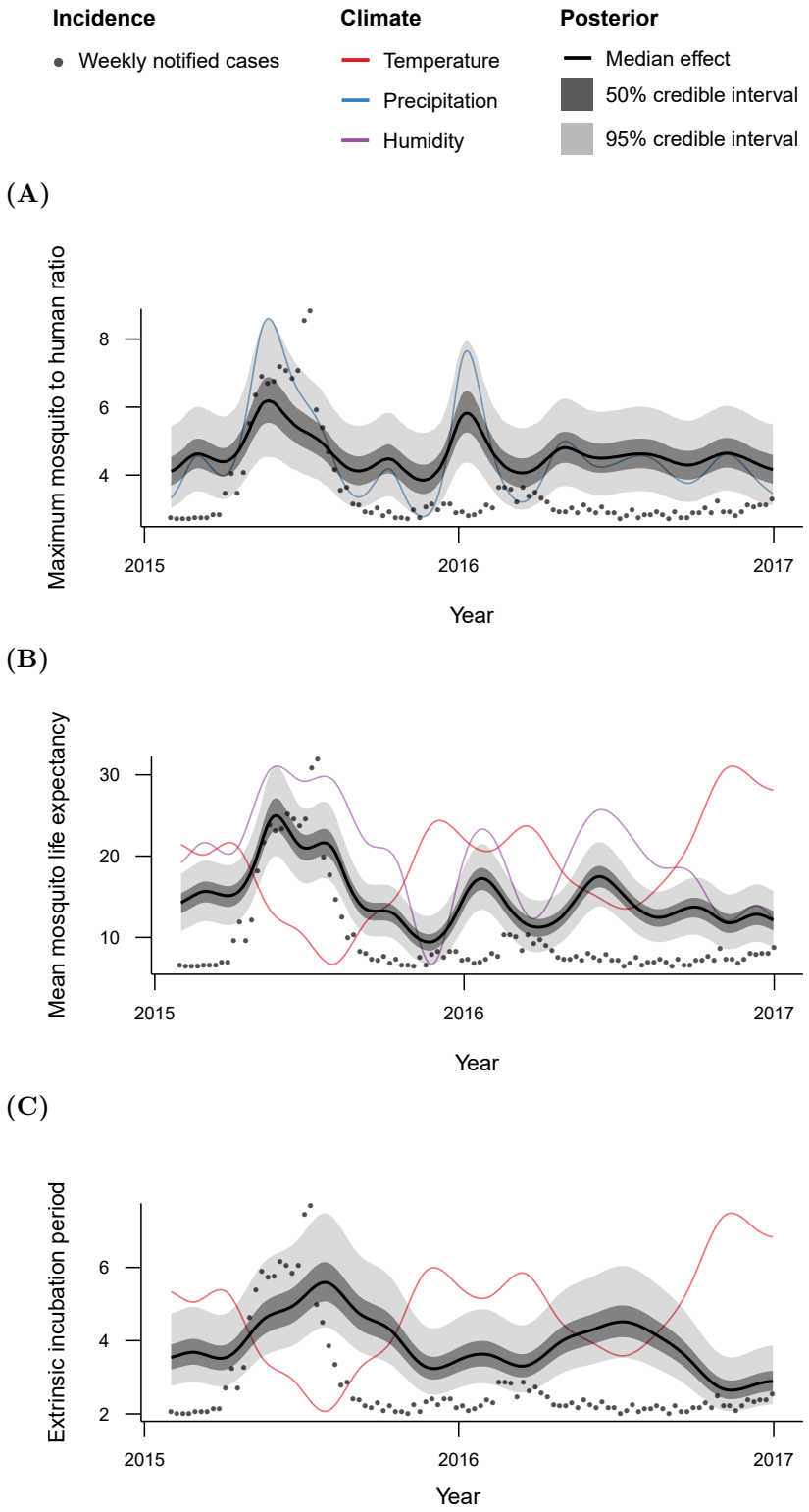

Figure 5.10. Inferred temporal dynamics of transmission parameters. Combining the inferred posterior distributions of unobserved parameters with the relationships defined between climate parameters and key transmission parameters, yield posterior distributions of each transmission parameter over time. During the 2015 outbreak, there was high **(A)** mosquito longevity, **(B)** mosquito density and **(C)** extrinsic incubation periods. During the periods of low transmission, extrinsic incubation periods, mosquito longevity and mosquito density were inferred to be consistently low.

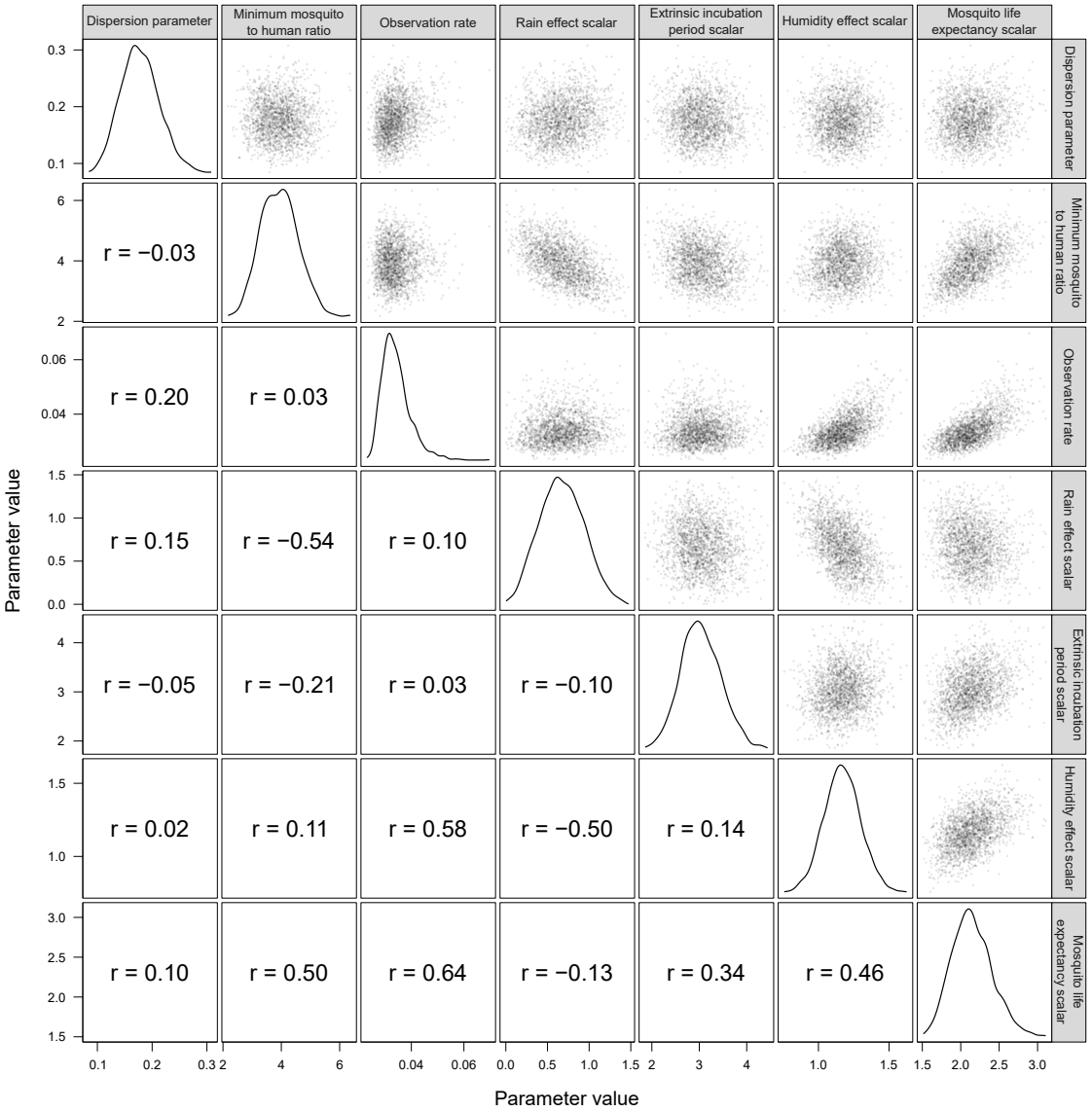

Figure 5.11. Correlations between accepted unobserved parameters. Moderate correlations were found between several of the inferred parameters in order to maintain the transmission potential, R_0, of the Zika virus throughout the study period. A shorter inferred extrinsic incubation period was associated with reduced mosquito longevity, amplified oscillations in mosquito longevity were correlated with attenuated oscillations in mosquito density and a reduction in the minimum mosquito-to-human ratio. Pearson's correlation coefficients, r, were calculated from 2,000 random samples from the posterior distribution.

Finally, the mean probability of observing an infected human case, p_{obs}, was strongly positively correlated with the effect of humidity on mosquito longevity ($r = 0.58$) and the scalar controlling mosquito longevity ($r = 0.64$). Increased humidity effects and scalers for mosquito longevity decreased the mean mosquito life expectancy throughout the study period. Therefore, the overall reduction in transmission potential was compensated for by increased observation rates.

5.3.3 Model forecasting

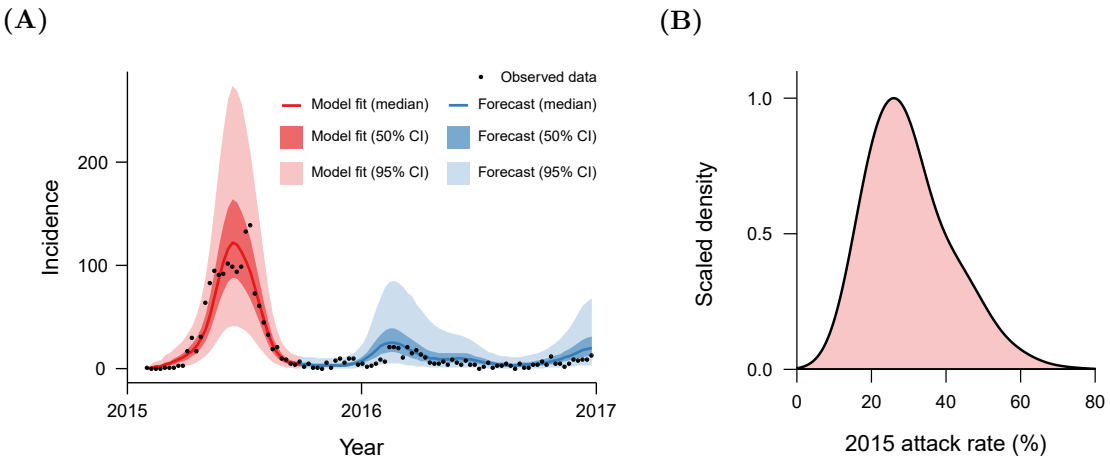

Figure 5.12. Forecasting from the 2015 epidemic. Fitting the model to the 2015 epidemic only and simulating forward in time yielded **(A)** forecasts which accurately predicted the temporal signature of the empirical data. **(B)** In turn, estimates of the percentage of the population infected during 2015 were lower than the inferred attack rate from fitting to entire empirical data set. Distributions shown were calculated from 2,000 randomly sampled values from the posterior distributions of the model fit.

In order to highlight the model's capacity to forecast outbreaks, the model was fit to empirical data from the 2015 epidemic only. Then, the model was simulated forward in time until the end of 2016 using parameter values inferred from the 2015 model fit and the 2016 climate data. The temporal dynamics of observed incidence from the forecast closely tracked the behaviour of the weekly notified case data (Figure 5.12A). However, due to the large uncertainty in forecasted incidence, there was a tendency for the number of cases to be overestimated during the 2016 outbreak. In alignment, the inferred attack rate during 2015 was on average 20% lower than estimates gained from fitting to both outbreaks (Figure 5.12B), indicating that in order to capture the small magnitude of the 2016 outbreak, sufficiently high levels of herd-immunity prior to 2016 was required to inhibit transmission.

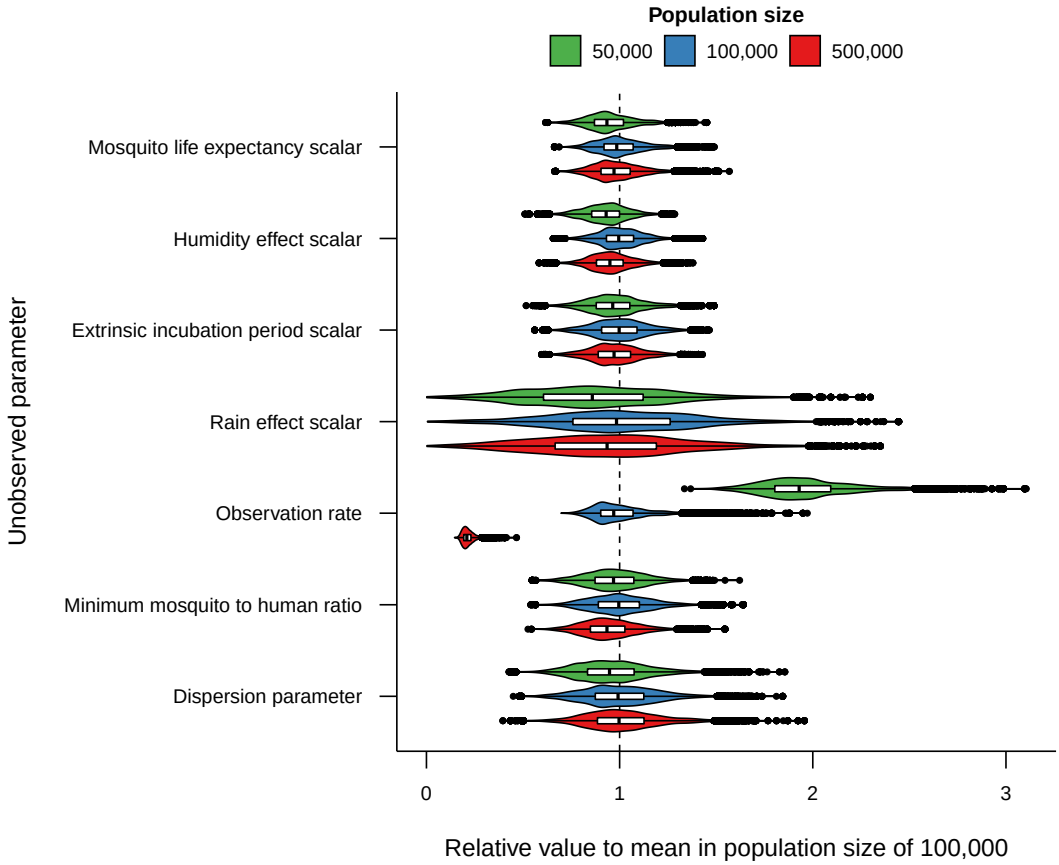

Figure 5.13. Invariance of population size on posterior distributions. The model was fit to the empirical data with three different population sizes, 50,000, 100,000 and 500,000 individuals, with fixed community size, and posterior distributions compared. The majority of posterior distributions for the three model fits were analogous with the exception of the observation rate, p_{obs}, which had mean and variance that scaled by the number of individuals in each model fit.

5.3.4 Effects of population size on model inference

The presented model fit was executed on a human population of 100,000 individuals, whereas the full population of Feira de Santana has over 500,000 individuals (United Nations, 2015). In order to assess the influence of population size on the inference of unobserved parameters, the model was also fit with 50,000 and 500,000 individuals, keeping community size constant, and the resulting posterior distributions were compared. An increase in the number of individuals reduced the stochasticity of the simulation, and thus the variance in the estimate of the marginal distribution was reduced. Therefore, in order to optimise the pseudo-marginal method the number of particles selected at each step in the MCMC algorithm was adjusted to be $N = 50$ for 50,000 individuals and $N = 10$ for 500,000 individuals.

The majority of posterior distributions of unobserved parameters were extremely comparable between the fits for the three different population sizes (Figure 5.13). However, the mean and variance of the posterior distribution for the observation rate, p_{obs}, was found to scale linearly with population size. This was because total infections increased with population size, thus observation rates needed to be reduced such that model generated observations matched the incidence data.

5.3.5 Effects of mosquito mortality rates on model inference

In order to determine the effects of different assumptions about mosquito mortality rates on inferred parameters, motivated by the significant difference between R_0 estimates under different assumptions of vector mortality rates (see Chapter 3), the individual based model was fit to the empirical data under the assumption of constant vector mortality ($c_v = 1$) and then compared to the previous model fit assuming age-dependent vector mortality ($c_v = 4$).

We found that there was little difference between most posterior distributions of constant and age-dependent vector mortality. However, the inferred linear scalar that controls mean mosquito mortality rates, η, was higher on average under the assumption of constant daily vector mortality rates, resulting in lower mean mosquito life expectancy (Figure 5.14A). In order to maintain the same transmission potential, or R_0, the mean infectious period of mosquitoes is required to be the same under both assumptions, which, as suggested in Chapter 3, is achieved with lower mosquito life expectancies under the assumption of constant vector mortality.

Interestingly, the uncertainty in the percentage of the population infected during 2015, or the 2015 attack rate, decreased under the constant vector mortality rate assumption (Figure 5.14B). This was due to the decrease in the variance of mean mosquito longevity, resulting in decreased variance of R_0 between simulations.

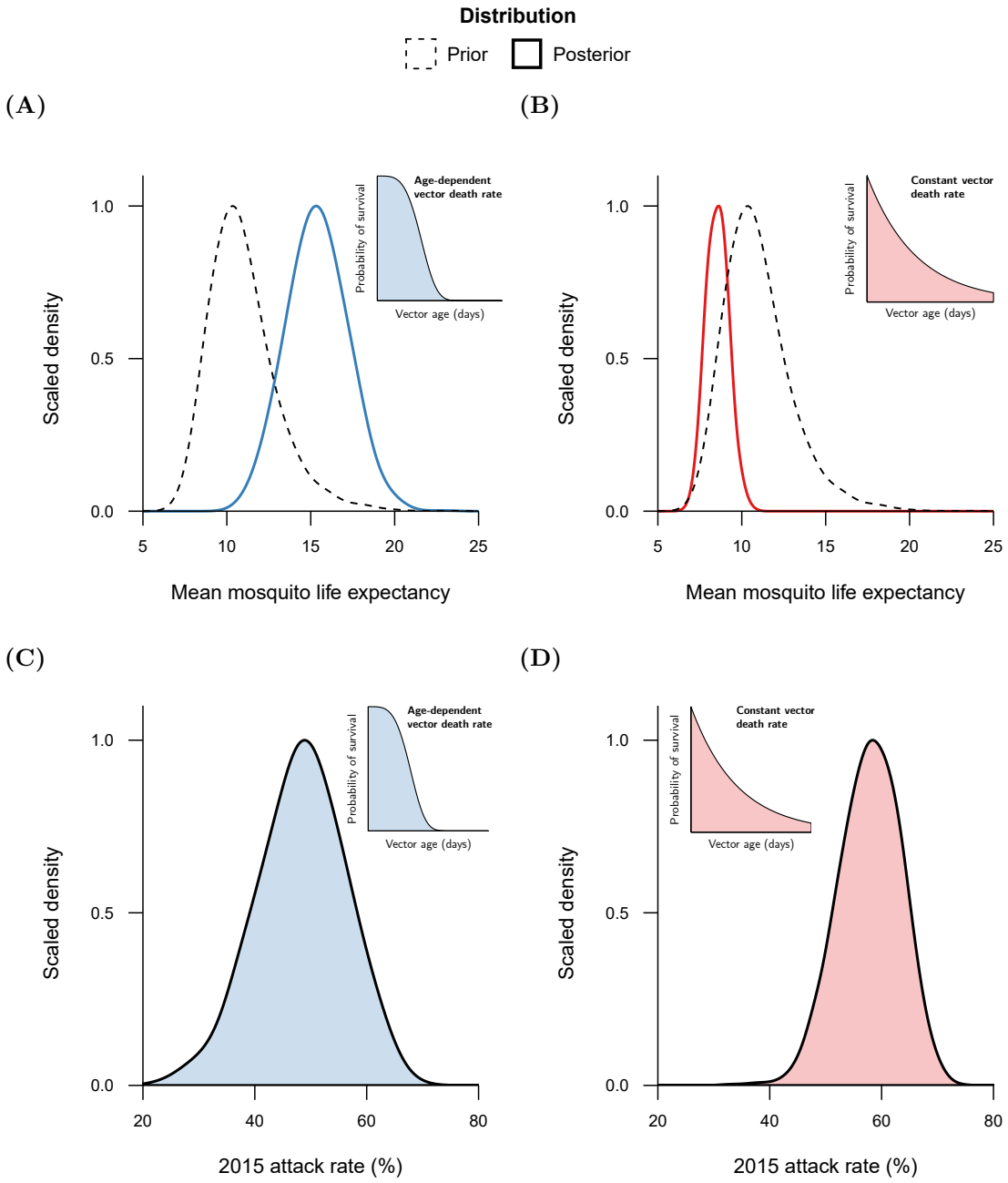

Figure 5.14. Effect of constant and age-dependent vector mortality rates on model inference. The model was fit under the assumption of constant vector mortality ($c_v = 1$) and age-dependent vector mortality ($c_v = 4$). In order to maintain transmission potential, or R_0, estimates of mean mosquito longevity were reduced from the model fit with **(A)** age-dependent mortality versus the model fit with the **(B)** constant vector mortality rate assumption. Smaller variance in the posterior distribution of mean mosquito life expectancy produced decreased uncertainty in estimates of the proportion of the total population that were infected during 2015, or 2015 attack rate, from the **(C)** age-dependent fit to the model fit under the assumption of **(D)** constant vector mortality.

5.3.6 Effects of spatial structure on model inference

To investigate the impact of space structure on model inference, the model was fit under the assumption of three lattice configurations of increasing number of communities ($|C| = 100; 1,000; 10,000$) with fixed local mobility and total human population size. Increasing the number of communities in the lattice, or decreasing number of individuals within each

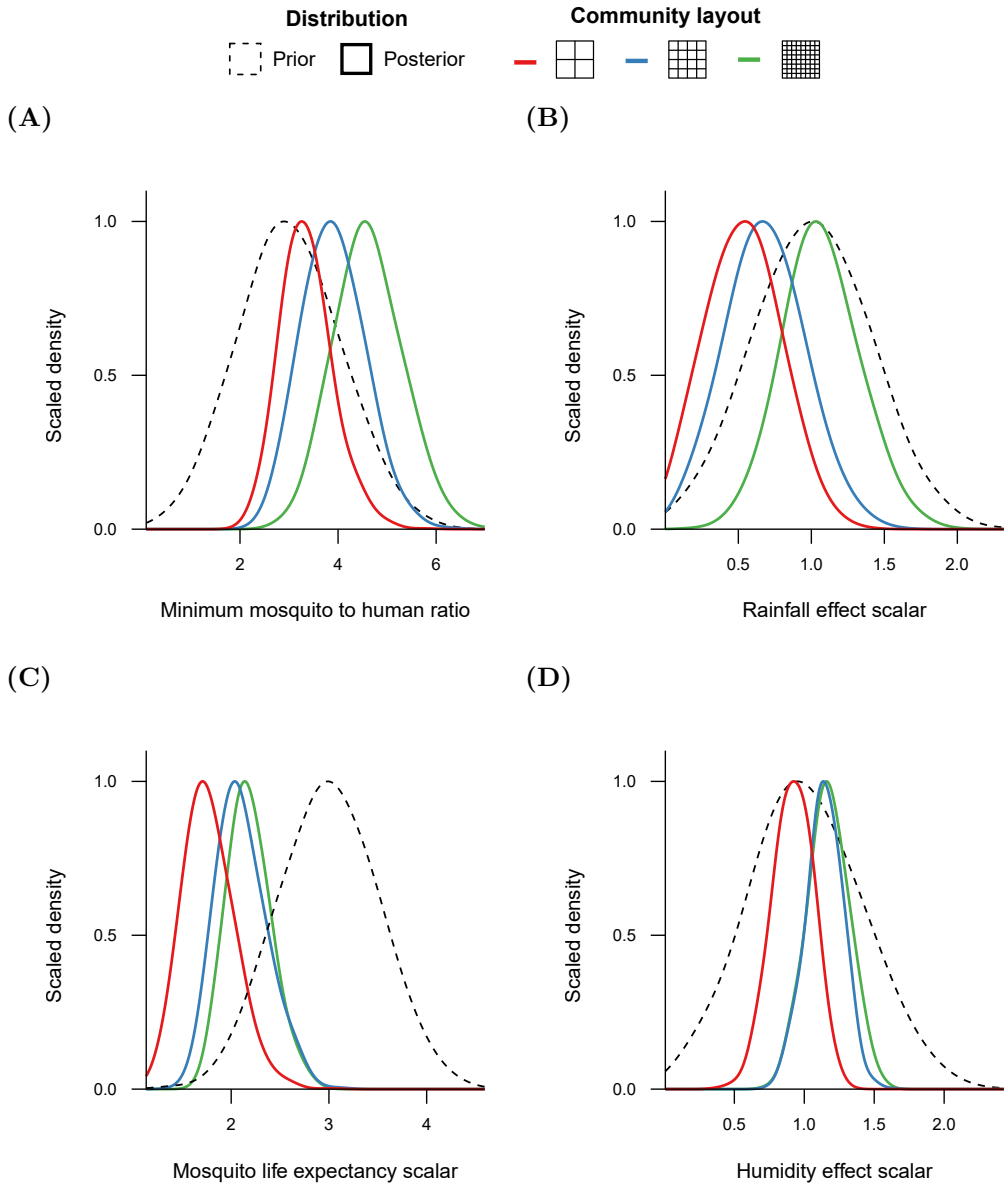

Figure 5.15. Effect of spatial refinement on model inference. The individual based model was fit to the empirical data with three different community layouts: a small lattice ($|C| = 100$), the default lattice ($|C| = 1,000$) and a large lattice ($|C| = 10,000$). As the size of the lattice increased, posterior distributions for **(A)** minimum mosquito density, κ, and **(B)** effect of rainfall on mosquito density, ρ_R, shifted toward larger values, inferring a higher mean mosquito population density. In contrast, the scalar for **(C)** mean mortality rates, η, and the **(D)** effect of humidity on mortality rates, ρ_H increased with greater spatial refinement.

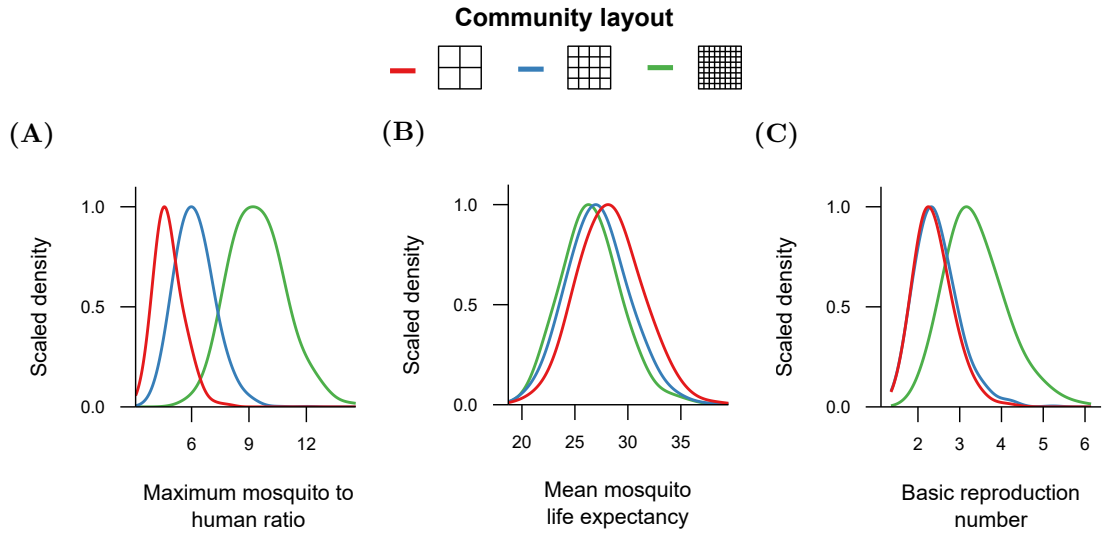

Figure 5.16. Drivers of transmission during the 2015 epidemic. Posterior distributions for estimates of key transmission parameters at peak transmission during the first epidemic, defined here as 1st June 2015, were calculated from model fits under three different community layouts: a small lattice ($|C| = 100$), the default lattice ($|C| = 1,000$) and a large lattice ($|C| = 10,000$). With increased spatial refinement, increased **(A)** mosquito density and consistently high **(B)** mosquito longevity yielded increased values of **(C)** the basic reproduction number, R_0.

community, increased the mean minimum mosquito-to-human ratio (Figure 5.15A) and the effect of rainfall on mosquito density (Figure 5.15B). In stark contrast, the scalar controlling mean mosquito mortality rates increased as spatial resolution was refined (Figure 5.15C), resulting in a decrease of mean mosquito longevity throughout the study period, counterbalancing the aforementioned rise in mean mosquito density. Furthermore, there was a slight increase in the effect of humidity on mosquito life expectancy as the number of communities was increased (Figure 5.15D), amplifying the seasonal oscillations in mosquito longevity.

To further discern the influence of space on key drivers of transmission, the accepted unobserved parameter values were transformed into the maximum mosquito-to-human ratio, expected mosquito life expectancy, and basic reproduction number, R_0, at the peak of the first epidemic, here defined as the first day of June in 2015. On average, the maximum mosquito-to-human ratio during the first epidemic increased as the number of communities in the lattice was increased (Figure 5.16A). At high humidity levels, the amplification of the oscillations of mosquito mortality rates from increased humidity effects counterbalanced the overall decrease in mosquito longevity as spatial dimension was refined, yielding a similar mosquito life expectancy during peak transmission (Figure 5.16B). Therefore, the

basic reproduction number increased alongside the number of communities in the lattice (Figure 5.16C) because higher levels of transmission, via the increase in peak mosquito density, facilitated the virus diffusion throughout lattices of increased dimension.

5.3.7 Effects of importation and mobility on model inference

The spatially-explicit framework further permits us to determine the influence of national and local mobility on disease outbreaks, so the model was fit to the empirical data with three fixed values for external introduction rates ($\iota = 0.5, 1, 5$) and three levels of local mobility ($1 - p_\sigma = 0.25, 0.5, 0.75$).

All model fits exhibited shifts in the posterior distribution of each unobserved parameter. Critically, there was a bistable response in inferred seroprevalence levels at the end of the two year study period. High external introduction rates, in combination with any level of local mobility, consistently produced immunity levels of over 50% (Table 5.5). This was in stark contrast to inferred immunity levels of less than 10% for lower introduction rates and local human mobility. However, high local mobility also consistently inferred human seroprevalence of over 50%, as the virus could disseminate throughout the community structure more easily.

Table 5.5. Bi-stability of inferred seroprevalence. the individual based model was fit under three external introduction rates, $\iota = 0.5, 1, 5$, and three levels of local mobility, $1 - p_\sigma = 0.25, 0.5, 0.75$. All model fits with high external infection rates or high local mobility were consistently attracted towards posterior distributions of high immunity levels of greater than 50%. However, fits lower introduction rates and local mobility hindered transmission and inferred very low attack rates of less than 10% from February 2015 to December 2016. Results shown are the median and 95% credible intervals of human seroprevalence at the end of 2016.

		Local mobility		
		Low	Medium	High
External infection rate	High	58.8% [41.4%, 69.7%]	59.2% [44.9%, 68.5%]	61.1% [52.4%, 68.7%]
	Medium	6.0% [4.4%, 8.1%]	6.2% [4.5%, 8.4%]	59.6% [51.6%, 67.4%]
	Low	3.1% [2.2%, 4.2%]	3.0% [2.1%, 4.3%]	58.8% [50.1%, 65.4%]

Model fits inferring high seroprevalence produced high values of R_0 prior to the 2015 outbreak before falling below one in July 2015 (Figure 5.17A). In contrast, inferred R_0 estimates were much smaller in model fits that estimated low seroprevalence (Figure 5.17B). There was little difference between R_0 estimates at peak levels of transmission across all tested combinations of external introduction rate and local mobility, however, which implies that lower seroprevalence estimates were induced simply by shorter durations of high transmission potential.

Spatially, high external introduction rates ($\iota = 5$) and low local mobility ($1 - p_\sigma = 0.25$) generated hundreds of outbreaks which slowly spread locally over the first epidemic in 2015. A large number of small clusters of high susceptibility provided ideal conditions for virus invasion during the second smaller outbreak, yielding a heterogeneous immunity landscape (Figure 5.18A). As the external introduction rate was lowered ($\iota = 0.5$), the number of initial

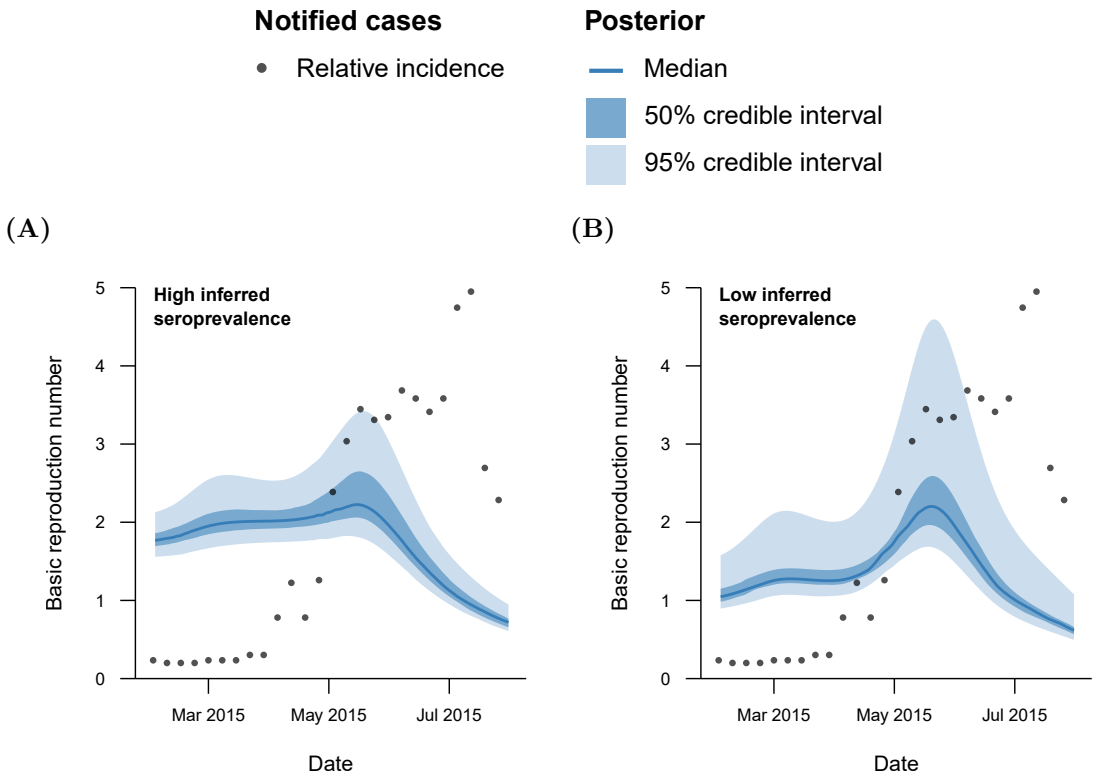

Figure 5.17. Effect of inferred seroprevalence on the basic reproduction number R_0. **(A)** High external introduction rates or local mobility (here, $\iota = 1$ and $1 - p_\sigma = 0.75$) consistently inferred high seroprevalence due to sustained levels of R_0 from February 2015 until July 2015. **(B)** In contrast, model fits inferring low immunity levels (here, $\iota = 1$ and $1 - p_\sigma = 0.5$) exhibited R_0 estimates which were almost singular prior to May 2015, hindering transmission. Median and credible intervals shown were calculated from 1,000 randomly sampled values from the posterior distribution of each model fit.

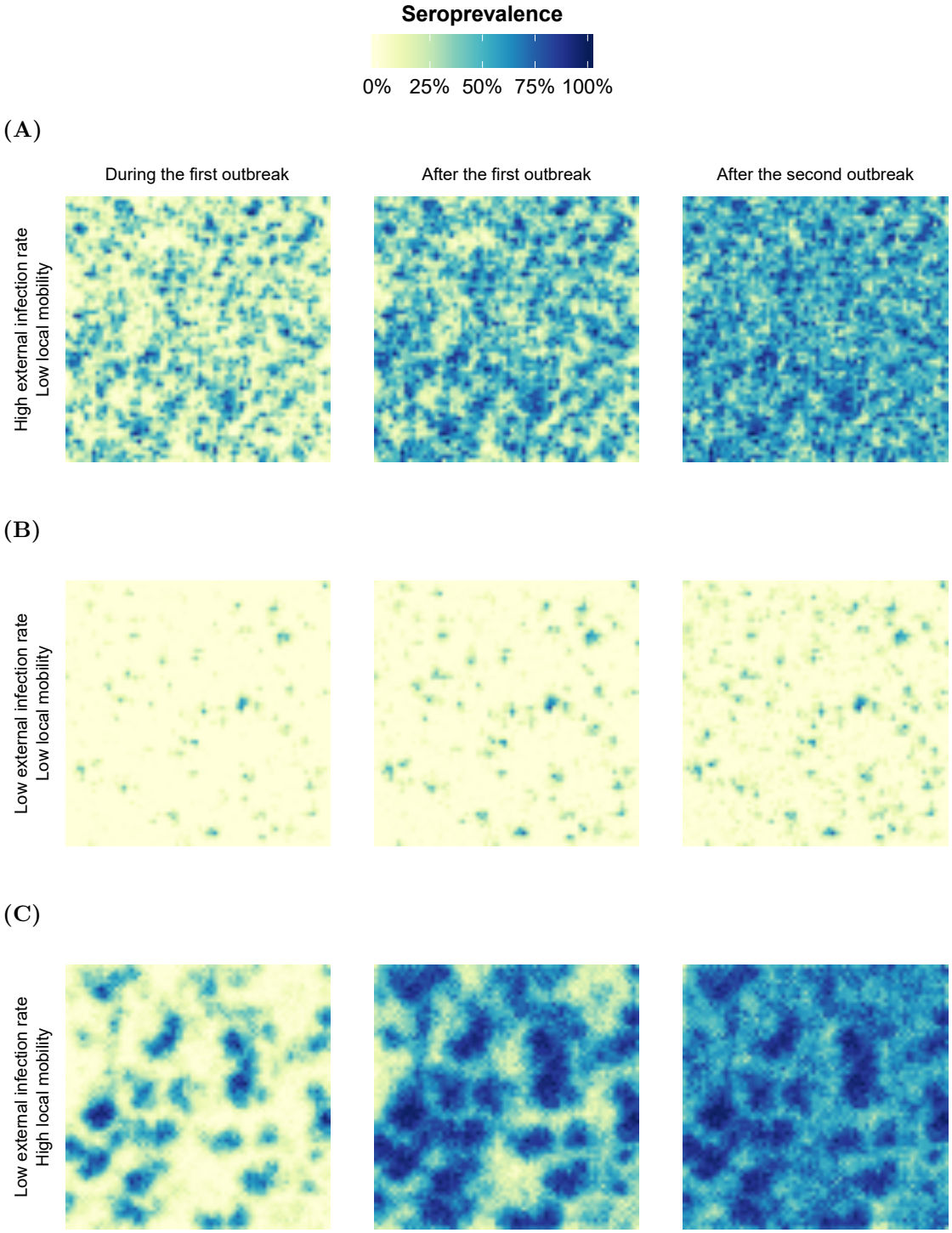

Figure 5.18. Effect of mobility and external introduction rates on Zika spatial dynamics. The individual based model was fit under different assumptions of local and global mobility and accumulation of immunity across the lattice was observed. **(A)** High external introduction rates ($\iota = 5$) and poor human mobility ($1 - p_\sigma = 0.25$) gave rise to many initial outbreaks culminating in a highly heterogeneous landscape of immunity. **(B)** Low local mobility in combination with poor introduction rates ($\iota = 0.5$), greatly hindered spatial spread of the virus, yet **(C)** increasing human mobility ($1 - p_\sigma = 0.75$) enabled rapid diffusion of the virus throughout the lattice. Seroprevalence was calculated from the percentage of individuals within each community with acquired immunity to the virus.

cases in 2015 greatly decreased. Hindered by low local mobility, very high susceptibility levels were maintained throughout the lattice that failed to be penetrated during the 2016 outbreak due to poor external introduction and local mobility (Figure 5.18B). However, increased local mobility ($1 - p_\sigma = 0.75$) enabled the small number of initial cases to rapidly spread throughout the lattice, creating large spatial clusters of high immunity and susceptibility. During the second peak, neighbourhoods of complete susceptibility permitted further local expansion of the virus, creating an overall landscape of moderate immunity with collections of entirely immunised communities (Figure 5.18C).

5.4 Discussion

Climate has previously been shown to play a crucial role in the emergence and spread of arboviral disease (Huber et al., 2018; Lowe et al., 2013, 2016; Tesla et al., 2018). However, the exact relationships between climate and epidemiological drivers are not yet well established in the field. Therefore, in this chapter, we sought to quantify the influence of climate on vector suitability and virus transmissibility by fitting a spatially-explicit, climate-dependent individual based model to incidence data. The computational speed-up (see Chapter 2) allowed us to perform individual based model fitting within a fully (as opposed to approximate) Bayesian framework for the first time.

The model was first fit to simulated data from the individual based model itself. Fitting to the simulated data showed that several parameters of interest could be recovered within our framework, but other parameters, such as the effect of rainfall on mosquito density, could not be recovered as reliably. However, we then demonstrated that epidemiological transmission parameters, such as mosquito density during an epidemic, could be reliably inferred given that all other parameters were well-informed. We also found correlations between several inferred parameters, such as the effects of rainfall and humidity on mosquito density and longevity. This was confounded by the strong relationship between rainfall and humidity within the climate data, suggesting potential for model simplification in this spatial context. Understanding the behaviour of our model by fitting to simulated data meant that we could fit to empirical data with the confidence that posterior distributions for each parameter of interest could capture the hidden empirical value.

By fitting to Zika incidence data from Feira de Santana, Brazil, the framework was able to identify posterior distributions of parameters, defining relationships between climate and mosquito demography. These results generally indicated a strong influence of humidity on mosquito longevity and a relatively weaker effect of rainfall on mosquito density. However, we found that there was little impact of annual oscillations in temperature on the Zika outbreak within this region. This was likely due to the fact that temperatures were high throughout the study period, thereby enabling short extrinsic incubation periods of the virus.

Our findings aligned with a climate-driven ordinary differential equation model fit by Lourenço et al. (2017), including inferred reporting rates of less than 1%. These reporting rates were in stark contrast to estimated Zika reporting rates of 16% and 18% in El Salvador and Suriname, respectively (Shutt et al., 2017), but in agreement with observation rates of 2.7% in Cabo Verde Islands, West Africa (Lourenço et al., 2018a). High seroprevalence levels of over 50% were also agreed with previous cross-sectional serological studies in Salvador, Brazil (Netto et al., 2017), Yap Island (Duffy et al., 2009), French Polynesia (Cauchemez et al., 2016), and Nicaragua (Zambrana et al., 2018). This combination of low observation and high attack rates suggest great potential for asymptomatic infected individuals to transmit the virus. Indeed, Zika, as has dengue, has been shown to have a high proportion of asymptomatic cases (Haby et al., 2018; Ladhani et al., 2016), although it is not yet clear whether asymptomatic and symptomatic individuals have the same transmission potential (Duong et al., 2015; Moghadas et al., 2017; ten Bosch et al., 2018).

Traditional modelling frameworks, such as the one by Lourenço et al. (2017), implicitly assume constant vector mortality rates, which have been shown to impact the basic reproduction number, R_0, of arboviral diseases (see Chapter 3). Due to the flexibility of the individual based model, the model was fit to the empirical data under constant and age-dependent mosquito mortality rates. Under both assumptions, mean mosquito life expectancies were within previously found bounds of one to two weeks (Maciel-de Freitas et al., 2007; Marinho et al., 2016; Muir and Kay, 1998). However, in order to maintain the same basic reproduction number, the inferred mosquito life expectancy greatly shortened under the assumption of constant death rates.

To further demonstrate the importance of model assumptions for inferring parameter values, we fit the model to three community structures with different spatial resolutions. Finer spatial resolutions required increased effects of rainfall on mosquito density to capture the explosive dynamics of the 2015 outbreak. A similar behaviour was also found when fitting the model under different local human mobility and importation rates. Dependent on these rates, inferred seroprevalence levels at the end of 2016 exhibited a bistable behaviour. For high seroprevalence, external introduction rates were required to be high. This implies that the introduction of infected individuals into multiple locations within Feira de Santana

was necessary to achieve the high seroprevalence levels subsequently observed within the city. From a public health perspective, identifying common socio-ecological features of these locations could therefore be useful in determining infection risk factors and informing outbreak prevention strategies.

To that end, we further investigated the potential of the model to be used in disease forecasting by fitting exclusively to the 2015 epidemic and simulating forward in time. Our predictions matched the temporal signature of the 2016 empirical data, and demonstrated that reliable climate data would be required in order to accurately predict disease outcomes. That is, although climate factors exhibit general annual trends, these alone are likely insufficient to forecast incidence accurately. Due to the high variance of inferred attack rates, the framework would also benefit from cross-sectional serological data in order to more precisely quantify the relationships between climate and vector suitability. This may not be enough in cases where climate drivers are strongly correlated, such as here. In these cases, more robust estimates of some model parameters, such as mosquito longevity, could go a long way in more accurately quantifying ecological features that are challenging to measure empirically, such as the mosquito carrying capacity.

With that in mind, many epidemiological and demographical parameters were fixed within our framework. Inference on the parameters of interest was therefore dependent upon the choices of these fixed parameters. For example, we demonstrated the impact of different fixed human mobilities on the spatio-temporal dynamics of disease incidence. This suggests the strong benefits that including spatio-temporal incidence or social data sets could have on assessing the importance of human movement on arboviral disease outbreaks. It should also be noted that the model is still relatively computationally expensive (each model fit takes approximately one week) in comparison to a spatially homogeneous deterministic system. Model fitting run-times could further be reduced by instead using Particle marginal Metropolis Hastings methods to estimate the posterior distribution (Andrieu et al., 2010). However, with the alleviation of computational costs and invariance of inferred climate drivers to the number of individuals within the model, it may already be practical to use an individual based model as a real-time disease control management tool, at least when analyses only needed to be carried out on longer time

scales (e.g. monthly).

Here, we have demonstrated that not only is it possible to fit an individual based model to relatively sparse data within a fully Bayesian framework, but we can quantify relationships between climate and epidemiological features, such as mosquito longevity or the extrinsic incubation period. Unlike previous modelling approaches, the spatially-explicit nature of our framework showed that the virus was likely introduced into multiple spatial foci in order to create the observed temporal dynamics and rapid accumulation of Zika immunity in Feira de Santana. These findings emphasise the added benefits that cross-sectional serological and spatio-temporal incidence data sets could bring in more precisely inferring the ecological drivers of arboviral epidemiology. Overall, our results indicate the significant impacts that spatio-temporal ecological heterogeneities have on mosquito-borne disease inference, and should thus be explicitly considered when informing control efforts through mosquito elimination and vaccine deployment programs.

Chapter 6

Summary and conclusion

The main aim of this thesis was to gain a better understanding of the ecological drivers of dengue epidemiology. Previous approaches have concentrated on capturing dengue's epidemiological dynamics under strong modelling assumptions, and thus, an additional goal of this thesis was to assess the impacts of model simplification. Given the increased availability of epidemiological, environmental and social data, focus was placed upon the spatio-temporal heterogeneous patterns found in these data, by integrating epidemiological and ecological factors into a flexible and detailed mathematical framework. The results presented in Chapters 3–5 advance the current literature by quantifying the effects of human movement, vector ecology and climate on dengue epidemiology, and by stressing the limitations of simplifying model assumptions on elucidating the epidemiological determinants of vector-borne disease.

In order to capture the complex socio-ecological interactions that drive dengue epidemiology, spatially-explicit transmission models need to be developed. One such example is an individual based model, which is a flexible framework in which these spatio-temporal factors can be integrated. In Chapter 2, I presented a spatially-explicit individual based model to describe dengue transmission. Within our framework, I explicitly represented individual humans and mosquitoes which underwent daily demographical and epidemiological processes. Humans and mosquitoes were grouped into communities, which were then arranged into a grid to represent spatial structure. Transmission events from mosquitoes to humans, and humans to mosquitoes alike, were dispersed between these communities across different spatial scales. This model could then be used to extensively assess the epidemiological effects of different ecological scenarios.

However, it is well established that individual based models can be extremely computationally expensive compared to traditional modelling approaches, because at every single time step, each individual needs to be passed through some demographic process which involves ageing, birthing and death, plus processes related to infection. Recently, some individual based models have been implemented on the graphics processing unit (GPU) instead. The GPU is fantastic at the on-mass, simultaneous processing of tedious arithmetic tasks, such as the demographic update of an IBM. GPU-accelerated implementations of individual based modelling frameworks, outside of epidemiology, have had remarkable success at dramatically reducing computational run times (Hidayat et al., 2016; Li et al., 2009). In Chapter 2, I presented the first example of a GPU-accelerated spatially-explicit epidemiological model which alleviates the aforementioned high computational costs.

Remarkably, GPU-accelerated simulations of the individual based model were just under 100 times faster compared to simulations executed on the CPU. This was over five times faster than one previous approach of a spatially-homogeneous, epidemiological individual based model (Galvão Filho et al., 2016). However, Galvão Filho et al. (2016) concentrated on speed-up factors between simulations of relatively small numbers of individuals, and may have discovered larger gains at higher population sizes where the increased overhead of GPU setup becomes more worthwhile. Other agent based model approaches have exhibited speed-ups of up to 617 times in a bird flocking model (Hidayat et al., 2016) and up to 240 times in a fish schooling model (Li et al., 2009) compared to serialised CPU implementations. In contrast, individual based models simulating the dynamics of molecules demonstrated more modest speed-up factors of between 2 and 11 (Le Grand et al., 2013; Yang et al., 2007).

In fact, the range in speed-up factors between GPU-accelerated and CPU-exclusive model implementations is exceptionally wide throughout the literature: 115-fold speed-up for a Runge-Kutta solver (Murray, 2012), 10 to 26-fold speed-up in weather models (Michalakes and Vachharajani, 2008; Shimokawabe et al., 2010), from 30 to 200 in medical simulations (Dean-Ben et al., 2013; Kutter et al., 2009; Mosegaard and Sorensen, 2006; Wu and Heng, 2004), between 1.2 and 50 for image processing algorithms (Farrugia et al., 2006; Zhang et al., 2010), from 14 to 500 in fluid flow models (Hernández Pérez et al., 2018;

Sunarso et al., 2010; Sweet et al., 2018), a 223-fold speed-up in a tsunami propagation model (Satria et al., 2012), 360 in a biochemical network model (Zhou et al., 2011), and up to 115-fold speed-up in finding numerical solutions to a particular set of Navier-Stokes equations (Griebel and Zaspel, 2010). This huge variance highlights that expected speed-ups depend heavily upon the model being GPU-accelerated, the quality of the baseline and GPU-accelerated code, in addition to the hardware used to execute the CPU and GPU implementations.

Regardless, with the speed-up of the individual based model, I could more deeply explore the influence of socio-ecological drivers on dengue epidemiology. In Chapter 2, the suitability of the framework for capturing the observed epidemiological dynamics for dengue was assessed. I executed the modelling framework with empirically estimated parameters, finding that it could capture the irregular epidemic outbreaks and asynchronous oscillations in dengue's four serotypes, in addition to the typical spatio-temporal heterogeneous patterns in dengue incidence. Model sensitivity to dengue transmissibility and spatial structure was then assessed, with results largely agreeing with previous theoretical approaches.

Importantly, I then performed sensitivity analyses on the dispersion of infection events across different spatial scales in order to better understand model behaviour. First, the influence of increasing the range at which transmission events could locally disperse was investigated. I found that beyond a certain small threshold, the epidemiological dynamics of dengue were unaffected by increased local mobility, often becoming limited by other epidemiological and ecological effects, such as seasonal forcing or mosquito biting rate. These results were not only in agreement with previous empirical and theoretical studies, but suggests that the added benefit of increased dengue transmissibility would be lost given that dengue's primary vector, *Aedes aegypti*, only travel short distances. Therefore, it seems likely that DENV has evolved to maximise its transmissibility due to the limited flight range of the primary vector.

Plateauing epidemiological effects of increased local human movement once again implicated long distance mobility in driving dengue transmission, and so, I looked at the effects of increasing the rate at which transmission events were dispersed to anywhere within the spatial structure. I found that higher long-distance mobility increased epidemic-like

behaviour, by allowing the virus to be introduced into clusters of entirely susceptible populations more frequently. However, this hindered the virus persistence, as increasingly explosive outbreaks resulted in catastrophic crashes in dengue incidence during the off-season. This finding was also in full agreement with previous modelling approaches by Barmak et al. (2016) and Lourenço and Recker (2013). As demonstrated by these results, the model was more than suitable in capturing dengue epidemiology, and re-emphasised the importance of understanding human mobility patterns across different spatial scales.

The flexible nature of this framework also permits the addition of ecological features, such as more detailed descriptions of human movement, or the influence of climate. In addition, its prescriptive nature permitted for the relaxation of assumptions around key epidemiological parameters, such as mosquito mortality rates. Traditional modelling approaches, such as ordinary differential equations, frequently assume constant vector mortality rates, which naturally influences the vector-to-human transmission period (VHTP). The VHTP is crucial in calculating R_0 estimates for an infectious disease, which can then be used to inform disease control through mosquito elimination or vaccination. However, several empirical studies have instead found that the mortality rate of *Aedes aegypti* is dependent upon the age of the mosquito (Harrington et al., 2008; Styer et al., 2007). Therefore, in Chapter 3, I was interested in the effect of relaxing assumptions of mosquito survivorship on R_0 estimates using my modelling framework.

Initially, I compared a formula for R_0 based on systems of differential equations to one derived from first principals under different assumptions about mosquito mortality rates. I found that assuming constant vector morality rates grossly inflates R_0 estimates derived from empirically-estimated parameters and initial growth rates of epidemics. For the latter, I provided a correction factor, which takes into account the true mortality rate of mosquitoes, demonstrating a clear need for more realistic (in-field) mosquito longevity studies in order to accurately inform R_0 estimates.

These formulae were ratified by directly measuring the total number of secondary infections from a single human infection in a fully susceptible population within the individual based model itself. Doing this multiple times under the same initial conditions gave rise to a distribution of secondary infections with considerable variance, highlighting

that R_0 is simply a mean estimate of this distribution. Furthermore, approximately a third of disease introductions failed to cause any secondary infections, as either no mosquitoes became infected before the single primary human recovered, or infected mosquitoes died before going on to infect any humans. This suggested that a large number of disease introductions may be going empirically unobserved even in cases where R_0 estimates are reasonably high.

Additional variance in the reproduction number may be brought in through the socio-ecological context in which the disease is circulating. For example, as dengue is climate-dependent, disease introduced at the start of the transmission season will take a very different course to one introduced later on. Alternatively, introduction of the virus into a well-populated urban area will behave very differently to one introduced in a sparsely populated rural region. This massively highlights the importance of understanding the ecological drivers of dengue in order to more robustly estimate R_0, and emphasises the importance of not generalising R_0 estimates for a particular pathogen across all spatial contexts. However, I found that R_0 estimates based solely on serological surveys are generally much more robust. The longitudinal-nature of these surveys can absorb the stochastic variation of socio-ecological features, and thus provide more consistent estimates of R_0. Therefore, I would recommend that R_0-based control strategies are motivated by R_0 estimates from such surveys, rather than the initial growth rate of disease outbreaks.

In order to eradicate a disease, the theory of the basic reproduction number states that R_0 needs to be reduced to less than one. In contrast to this, my findings have shown that even if R_0 estimates were less than one, there is still the possibility of an outbreak. This has similarly been demonstrated in other stochastic modelling frameworks (Britton, 2010; Clancy and O'Neill, 2008). This perhaps offers an explanation behind why transmission can still occur amongst active mosquito elimination efforts. Alternatively, there may be inconsistency in control efforts across space and time, whereby outbreaks can occur through dengue importation from regions where control is less stringent. Therefore, in addition to the establishment of realistic vector mortality rates, socio-ecological features need to be considered in order to better understand the epidemiological and ecological determinants of vector-borne disease, and ultimately advise prevention strategies.

Recently, there have been several empirical and theoretical approaches to further elucidate these ecological drivers, e.g. Stoddard et al. (2013), Reiner et al. (2014), and Perkins et al. (2014). These studies found that socially-structured human movement drives the empirically observed spatio-temporal heterogeneity of dengue across fine-spatial scales. However, the effects of community structure on dengue epidemiology over wider geographical regions had yet to be explored in detail. To that end, in Chapter 4, I first sought to improve understanding on how structured human movement influences the epidemiological dynamics of dengue across wider spatial scales, by extending the individual based model presented in Chapter 2 to include different community structures.

Within this chapter, the epidemiological effects of three community structures were explored: a lattice, a random network and a scale-free network. Outbreaks were found be synchronised across different regions of these networks, meaning that if the underlying movement structure can be understood, then perhaps accurate spatio-temporal transmission risk maps could be built. Given the increased availability of social data sets, such as Salathé et al. (2012), such approaches could be achievable at present. I also found that introducing discrepancies into community connectivity induced further spatio-temporal heterogeneity in epidemiological dynamics. Namely, the co-circulation of multiple serotypes, and therefore the likelihood of more severe disease outcomes, was higher in well-connected communities than in poorly connected communities in both the random and scale-free networks. Further disrupting the connectivity completely hindered the persistence of dengue in the random network. However, in the more organised scale-free network, the pathogen could still persist and multiple serotypes could continue to co-circulate.

These diverse findings re-emphasise the importance of understanding the general nature of socially-driven human movement across different spatial scales and geographic locations. That is, models for national human movement may be inappropriate at capturing intra-urban mobility patterns, e.g. Yan et al. (2014), or human movement across Southeast Asia may have different cultural drivers to those found in South America, e.g. Kang et al. (2012). Gathering rich social data sets and developing appropriate modelling approaches will be essential to informing the socio-ecological drivers of human movement, and in turn accurately quantifying their impact on the spread and persistence of dengue.

Common to the results above was the assumption that population density of humans and mosquitoes was the same across all communities. However, it is well established that the habitat preference of *Aedes aegypti* gives rise to a heterogeneous distribution of mosquito density, but there are conflicting findings on how this, in combination with heterogeneity in human population density, influences dengue epidemiology (Cromwell et al., 2017; Kong et al., 2018; Morrison et al., 1998; Sang et al., 2014; Sirisena et al., 2017). In Chapter 4, I therefore investigated the influence of heterogeneity in human and mosquito population density on dengue epidemiology. If the mosquito-to-human ratio, and thereby the transmission rate, was fixed across all communities, increasing heterogeneity in population density across the network only marginally influenced epidemiological dynamics. However, destabilising the relationship between heterogeneity in mosquito and human density introduced a wide landscape of possible epidemiological behaviour.

To summarise here, annual outbreaks of multiple serotypes were stabilised throughout the network when mosquitoes were more highly concentrated into well-connected (urban) communities. In contrast, higher mosquito density in poorly connected (rural) communities induced increasingly severe, yet infrequent, epidemic outbreaks, demonstrating potential for dengue's secondary (rural-dwelling) vector, *Aedes albopictus*, to seed large epidemic outbreaks. From these results, it is clear that building realistic maps of vector abundance is fundamental in assessing dengue transmission risk. This could then inform more accurate disease prevention strategies, and perhaps the reasons behind why some disease control measures are unsuccessful could be justified. Overall, I need to develop better techniques to construct fine-grain maps across large spatial scales in order to gain a better understanding of vector ecology. Alongside this, it will continue to be important to understand how vector abundance across these regions is affected by changes in climate.

Climate is a well-documented driver of dengue epidemiology, whereby oscillations in temperature, rainfall and humidity induce spatio-temporal heterogeneity in both vector suitability and dengue transmissibility. Although there have been many laboratory-based studies to quantify these relationships, they're yet to be rigorously established in the field. Furthermore, the relative importance of each climate factor in driving dengue transmission has been shown to vary across environmentally different regions (Lauer et al., 2018; Xu

et al., 2019). To bridge this gap, mathematical transmission models are often employed and fit to empirical data, which is typically very sparse. However, individual based models were previously too computationally expensive to be fit with the Bayesian paradigm in mind, but due to the remarkable speed-up of the model, as presented in Chapter 2, this was now possible. Therefore, in Chapter 5, I were interested in what could be inferred about climate drivers by fitting the individual based model to disease surveillance data within a Bayesian framework.

First, I extended the model to include relationships between climate and mosquito longevity, population density, virus transmissibility and its extrinsic incubation period. In order to assess the ability of my framework to reliably infer these relationships, the model was then fit to incidence data that came from the IBM itself. As expected, the individual based model could capture the temporal dynamics of the simulated data, as can many systems of ordinary differential equations, but what was important here was that the framework correctly inferred key epidemiological parameters, such as mosquito longevity and the extrinsic incubation period. However, due to the strong correlations between these parameters, the climate-dependent relationships could not be inferred to a high degree of accuracy. Despite the fact that the exact relationships between climate and its epidemiological drivers could only be modestly established at best, the models capacity to capture key features of an outbreak was verified.

To that end, the framework was then fit to empirical data. As the primary vector of dengue and Zika is the same, and the epidemiological properties of the viruses themselves have been shown to be similar, selecting Zika data to infer the ecological drivers of dengue, and other arboviral disease, is plausible. Under this choice, only a single serotype would need to be considered, and the absence of any pre-existing immunity within the population prior to 2015 could be safely assumed. Therefore, I fit to Zika incidence data from Feira de Santana, Brazil, between 2015 and 2017. The modelling framework presented here could capture the temporal dynamics of this data, and inferred 40–60% seroprevalence after the first outbreak, which were in line with empirical findings. Importantly, humidity was identified as a key driver of arboviral outbreaks within this region, although it remains unclear whether this is the case across the rest of Brazil.

This data set had previously been fit to by Lourenço et al. (2017) with a system of differential equations, and so, motivated by my results in Chapter 3, I decided to investigate the effects of different modelling assumptions on model inference. By assuming homogeneity in space and a constant vector mortality rate, the main results of the previous study could be reproduced. However, assuming age-dependent vector mortality rates greatly influenced the relationships of temperature and humidity with mosquito longevity, a result which was consistent with the findings in Chapter 3. That is, in order to maintain the same transmission potential, mosquito longevity needed to be higher under age-dependent than constant mortality rates. This re-emphasised the importance of obtaining realistic estimates of mosquito survivorship in order to robustly inform relationships between climate and vector suitability. Similarly, by spatially segregating individuals, more mosquitoes were required to disseminate the virus. Combining this with the findings from Chapter 4, I wanted to know how human movement across different spatial scales influenced the outbreak in Feira de Santana.

The model was then fit under three different ranges for local human movement, and three different external introduction rates. This revealed bimodal behaviour in inferred attack rates: either less than 10% or greater than 40%. As the attack rate was in fact high (based on empirical evidence), my results implied the virus needed to be introduced into multiple distinct regions of Feira de Santana in order for the virus to spread across the city and affect enough individuals during the first transmission season. In that sense, it emphasised the benefit that either serological data, or spatio-temporal incidence data, could bring to model fitting exercises.

By fitting a complex epidemiological model to empirical data, I have demonstrated the capacity for my framework to be used in quantifying the relationships between ecological and epidemiological drivers. I have again highlighted that model assumptions about ecological factors can easily influence the inference made by these modelling frameworks. Therefore, obtaining reliable data sets on these ecological features is key in improving the understanding of dengue epidemiology, and thus more robustly informing control measures. However, building increasingly rich data sets into epidemiological models can harm the generalisability of findings across different spatial contexts. To that end, a balance needs

to be struck between informing these models with data sets that are both easily accessible, and simultaneously do not harm model interpretation, and thus will largely depend on the spatio-temporal scale upon which the research question is set.

Dengue has been at the forefront of mosquito-borne epidemiological modelling research for decades now and yet, effective control has still to be realised. Although the dengue modelling literature feels saturated, many approaches do not take ecological drivers of dengue epidemiology into consideration. However, in this thesis, I have clearly demonstrated that these spatio-temporal factors shape the epidemiological behaviour of dengue and thus, their absence within modelling frameworks throws up boundaries in the face of effective disease prevention. To that end, they should be explicitly included within transmission models, at least as far as informing control is concerned. I've also made a case for the benefits of obtaining more robust empirical data sets to reliable inform such modelling frameworks. In turn, this could refine our predictions and help us to assess the effectiveness of different vaccine strategies in the long-term. Alternatively, perhaps the data would inform us that maybe something is still; something that current models are incapable of capturing. That is, existing modelling approaches should not be taken for granted.

Whilst my framework has relaxed several assumptions of classical modelling approaches, the increased complexity of an individual based model naturally raises questions about model sensitivity and thus, the generalisability of my findings. I attempted to alleviate this concern by better understanding the effect of model parameters on its output throughout this thesis, and certainly do not claim that interpretations discussed here are generalisable to all transmission settings. In fact, my approach has reaffirmed the substantial effect that ecological drivers can have on dengue epidemiology, and has improved the understanding of how they can induce the marked spatio-temporal heterogeneity in dengue's epidemiological dynamics. I have also provided a mathematical framework in which any number of epidemiological questions can be answered due to the alleviation of computational costs by means of GPU acceleration. To that end, here, I make several suggestions for future work.

I. In this thesis, I have consistently demonstrated that findings should only be interpreted within the spatial location on which the theoretical model is based. This raises questions as to if the relationships between climate, vector suitability and dengue transmissibility are generalisable to other spatial contexts. Therefore, the natural next steps from the model fitting work presented in Chapter 5 are three-fold.

   (i) My results suggested that temperature was sufficiently high throughout 2015–2017 to sustain transmission, however this may not always be the case. For example, the south of Brazil is a more temperate region, and thus outbreaks here may be dictated by temperature rather than humidity. Therefore, the framework presented should be fit to incidence data from environmentally distinct regions to allow more generalised conclusions to be made about the influence of climate on arboviral epidemiology. However, it would be advisable that the regions compared be socio-economically similar, whereby the effects of underlying human movement patterns and other ecological features on inference are minimised.

   (ii) The framework should be fit to dengue incidence data from Feira de Santana over the same time period in order to confirm that the relationships between ecological drivers and arboviruses are generally comparable. More importantly, fitting to serotype specific incidence data could further elucidate the influence of immunological interactions on the spread of dengue.

   (iii) Throughout this thesis, I have proposed that more detailed data sets could further elucidate dengue's epidemiological drivers, and so the model should fit within a setting where very rich data sets are available in order to assess which data is most beneficial. From this, surveillance strategies could be refined, and in turn, modelling frameworks would be better informed to test the impact of different control strategies.

II. With that in mind, the findings presented in this thesis have important implications in assessing the efficacy and long-term impacts of disease control. That is, to make the prediction of disease outcomes as robust as possible, more data is required to inform data-driven modelling approaches. Similar epidemiological models have been

used to recently assess the impact of introducing Dengvaxia into populations of different pre-exposure levels (Flasche et al., 2016). However, many of these assumed a homogeneous distribution of humans and mosquitoes, or homogeneous mixing of individuals across space. I have demonstrated that these factors also matter when characterising the spread and persistence of dengue, and so the consequences of vaccine introduction could be evaluated under different community structures or heterogeneity in population density. Then, if the underlying network of human movement is better understood, then perhaps regions of high transmission risk could be targetted in order to maximise immunising effects. Although continued vaccine development is highly beneficial to dengue eradication, community-based control efforts will continue to sit at the heart of mosquito-borne disease control, and thus the effects of different mosquito control strategies could also be assessed within the framework. This could determine some of the factors behind why some vector control strategies have had little impact on dengue transmission.

III. The framework presented here made several simplifying assumptions over human mobility, namely that the movement of individual humans and mosquitoes was not explicitly represented. Instead, a model describing how infections are passed between communities was presented, with humans and mosquitoes assuming to mix homogeneously within each community. These assumptions could be relaxed such that human and mosquito mobility is accurately described. Previous modelling approaches have focused upon individual human movement at very fine spatial scales, and in this thesis, I addressed general movement patterns across larger spatial scales. However, to capture fine-grain dynamics over these large geographical regions without harming computational efficiency, other approaches may need to be considered. An empirical or theoretical approach could be taken, whereby social networks from regions of interest are assimilated into the framework *a priori*, or increasingly complex models for human movement are integrated, such as a gravity or radiation model. Either approach has its merits and pitfalls, but they would allow a much more thorough exploration into the effects of human movement across multiple spatial scales on dengue epidemiology.

IV. There are several other proposed ecological drivers of dengue that have not been discussed here. One example is that, in this thesis, I have focused upon the effects of vector ecology on dengue epidemiology, however little work has been done on how shifts in human demography have influenced dengue transmission. Cummings et al. (2009) showed that demographic shifts can explain changes in the age distribution of DHF incidence in Thailand from 1985 to 2005, but many other dengue-endemic countries have socio-economically developed alongside the global spread of dengue, and the role that this has played on dengue emergence is unknown. Secondly, the epidemiological effects of dengue's secondary vector, *Aedes albopictus*, are rarely explored within transmission models. However, from my findings in Chapter 4, the rural habitat preference of *Ae. albopictus* may give rise to more serious epidemic outbreaks in urban regions. Combining this with the wider global distribution of *Ae. albocpitus*, and its demonstrated ability to transmit DENV, studying its influence on dengue emergence and spread is worthwhile. As *Ae. albopictus* also circulates the sylvatic strain of dengue, it has been hypothesised that it could act as bridge vector between the endemic and sylvatic transmission cycles, thereby forming reservoirs of DENV in rural regions from which dengue may persist.

V. It could also be important to understand the influence of ecological factors on evolution of the dengue virus. The evolutionary history of dengue is largely attributed to the different immunological environments of host and vector, in addition the competition between serotypes. However, the effects of ecological heterogeneities on the observed spatio-temporal patterns of DENV phylogenetics is not prominent within the literature (Lourenço et al., 2018b). Within my model, a phylodynamic modelling framework could be integrated to test the impacts of spatio-temporal heterogeneity in host immunity, for example, on dengue evolution. Therefore, with the increased availability of (spatial) genetic data, the phylogenetic history of dengue could be used to further elucidate ecological drivers of dengue epidemiology.

In this thesis, I have demonstrated the importance of ecological drivers in the transmission of dengue across space and time within a robust mathematical framework. With that in mind, I have clearly shown that the simplifying assumptions of ordinary differential equations severely limit the interpretation of theoretical results in an ecological context. From this, I have stressed the need to gather realistic data sets on mosquito survivorship, in addition to fine-scale surveillance data, in order to inform detailed epidemiological models, and in turn, more robustly inform the influence of human movement, vector ecology and climate on not just dengue, but the epidemiology of other vector-borne pathogens. To that end, it is clear that findings interpreted from theoretical approaches based on a single geographical setting should not be generalised across larger spatial contexts, as the ecological drivers of dengue themselves dictate its emergence, spread and persistence.

Appendix A

Model documentation

The spatially-explicit individual based model (outlined in Chapter 2) was implemented initially in C/C++, where individuals are characterised in vectors of various census data. Populations of human and mosquito individuals are first initialised and then at each time step, individuals are passed through demographic and epidemiological processes. Throughout the simulation, numerous counters are kept track of to save outputting the entire census at every time step. These are outlined in the variables section below. Please refer to source files for additional details; all source files are thoroughly commented throughout. The following is a brief overview of how each process in the model was implemented.

Initialisation

At the start of every simulation, the human and mosquito populations are created in the following way:

I. The ages of humans and mosquitoes were initialised according to the survival function of human and mosquito demography (i.e. demography is at equilibrium).

II. All mosquitoes were initialised to be alive at the start of the simulation.

III. Humans and mosquitoes were uniformly distributed across all communities in the meta-population.

IV. All humans and mosquitoes were first initialised to be susceptible, then a small proportion of humans and mosquitoes are infected.

Demographic process

For every individual:

I. Check if individual is alive. This is applicable to mosquitoes only, as their population size can vary.

II. Check the individual's current age against their life expectancy:

   (i) If the individual has exceeded their life expectancy, then they are removed. If it is a human, then all census data for that individual are reset and new life expectancy generated. If it is a mosquito, then they are marked as being dead.

   (ii) If the individual has exceeded their life expectancy then their age is increased by one.

   For mosquitoes, the total number of alive individuals is then compared to the expected number of individuals given the pre-defined seasonality function of mosquito density for the model. If the number is lower than expected, then new mosquitoes are created in order to match the expectation.

III. Check if the individual is infected:

   (i) If the individual's age has exceeded the age at which they were due to become infectious, make them infectious.

   (ii) If the individual's age has exceeded the age at which they were due to recover (humans only), remove their infection.

Epidemiological process

For every sub-population / strain combination:

I. Calculate the total number of expected transmission events from humans to mosquitoes and from mosquitoes to humans according to the force of infection term multiplied by the total number of individuals alive in that sub-population.

II. Split the expected number of transmission events into long-distance and local transmission events.

III. For each long-distance transmission event, randomly assign it to any community within the meta-population (uniform).

IV. For each local disease transmission event, randomly assign it to any community defined by the local disease dispersal kernel.

Then, for every sub-population:

I. Go through each expected transmission event for each serotype (one-by-one), and select a random individual belonging to that sub-population.

II. If that individual is alive, not immune to the infecting serotype and not currently infected, infect the individual. Record the age of infection, and assign an age at which the individual becomes infectious and recovers (if human), record a successful transmission event.

III. If that individual is alive, but immune, or currently infected, record an unsuccessful transmission event.

IV. If that individual is not alive, try again.

GPU-acceleration

The model was implemented in NVIDIA's GPU acceleration environment: CUDA. Due to the novelty of using GPU-acceleration in epidemiological models, it is worth briefly explaining the motivation behind using GPU-acceleration. Details on the implementation and optimisation of the GPU-accelerated code can be found in the main text in Chapter 2.

Motivation

Individual based models are highly computationally expensive because at each time step, every individual (possibly millions) needs to be passed through some demographic and/or epidemiology process. This results in model run-times being very long (on the order of several minutes for large numbers of individuals). This limits their usefulness in real-time responses to epidemiological outbreaks when often many simulations are computed. However, the graphics processing unit (GPU) is very good at processing a large number of arithmetic tasks simultaneously. Therefore, this modelling framework was implemented

using GPU-acceleration because of the parallelisable nature of the demographic and epidemiological processes outlined above, whereby each individual or sub-population can be processed simultaneously.

Compilation

The model was compiled in Microsoft Visual Studio 2019 on Windows 10 64-bit using the compiler `nvcc` that is included in the NVIDIA CUDA Toolkit 10.1. The C/C++ only code was compiled through `nvcc` using `cl` distributed in Microsoft Visual Studio 2019. The necessary flags to pass to the compiler were as follows:

```
-O2 -Xcompiler="/std:c++14 /MD /O2" --gpu-architecture=sm_61 --machine 64
-cudart static -use_fast_math
```

The option for `--gpu-architecture` should be changed in accordance with the machine executing the code. A Microsoft Visual Studio 2019 solution file is provided also here.

Parameters

Parameters for the spatially-explicit individual based model for a multi-strain vector-borne pathogen. Simulation parameters can be adjusted within `parameter.cpp`. Default parameter values are given Table 2.2 in Chapter 2. Descriptions of each parameter are outlined below, along with the corresponding mathematical notation.

Parameter in code	Description
`nSize`	The number of human individuals, N_H, in the simulation.
`metaPopRows`	The number of rows of communities in the lattice meta-population.
`metaPopCols`	The number of columns of communities in the lattice meta-population.
`maxMosToHuman`	The maximum mosquito to human ratio, M.
`minMosToHuman`	The minimum mosquito to human ratio, m. The mosquito to human ratio follows a sinusoidal function which is maximised at time points $t = 365n$ and minimised at time points $t = 365n + 365/2$ with $n \in \{0, 1, 2, ...\}$.
`nShapeInfantMortality`	The burn-in shape parameter, a_H, for the bi-Weibull distribution of human mortality risk.
`nScaleInfantMortality`	The burn-in scale parameter, b_H, for the bi-Weibull distribution of human mortality risk.
`nShapeLifeExpectancy`	The fade-out shape parameter, c_H, for the bi-Weibull distribution of human mortality risk. Loosely, this defines the shape of the age distribution of individuals which survive infant mortality.

`nScaleLifeExpectancy`	The fade-out scale parameter, d_H, for the bi-Weibull distribution of human mortality risk. Loosely, this defines the mean life expectancy of an individual which survives infant mortality risk.
`nLocWeibull`	The age at which the burn-in phase transititions to the fade-out phase of the bi-Weibull distribution for human mortality risk, L.
`mShapeLifeExpectancy`	The shape parameter, c_V, for the Weibull distribution of mosquito mortality risk. Defines the shape of the age disitribution of mosquitoes, where $c_V = 1$ denotes an exponential age distribution.
`mScaleLifeExpectancy`	The scale parameter, d_V, for the Weibull distribution of mosquito mortality risk. Approximately equal to the mean life expectancy of a mosquito: $\Gamma(1 + 1/c_V)d_V$.
`bitingRate`	The average number of bites per day of a mosquito, β.
`mnBitingSuccess`	The probability of a bite resulting in transmission of the pathogen from mosquito to human, p_H.
`nmBitingSuccess`	The probability of a bite resulting in transmission of the pathogen from human to mosquito, p_V.
`recovery`	The number of days that a human is infectious, $1/\gamma$.
`mExposed`	The extrinsic incubation period, or the number of days which a mosquito is infected but not infectious, $1/\epsilon_V$.
`nExposed`	The intrinsic incubation period, or the number of days which a human is infected but not yet infectious, $1/\epsilon_H$.
`externalInfection`	The external infection rate: the number of infections per 100,000 individuals per day per strain/serotype, ι.
`longDistance`	The probability of a single transmission event being dispersed to anywhere within the lattice, ω.

`exIncPeriodRange`	The maximum deviation of the extrinsic incubation period from the mean extrinsic incubation period defined by 'mExposed', δ.
`kernelStandardDeviation`	The standard deviation of the local disease dispersal kernel. Higher values correspond to transmission events from a given community dispersing further distances within the lattice community structure, σ.

Constants

For the simulation, a frequently-used value was treated as a constant if it rarely needed altering or was not related to the mathematical description of the individual based model itself. These values were defined as constants in order to prevent hard coding of values that may in the future be adjusted. Constants were adjusted in the header file `constant.h`. Here, the constants used in the simulation.

Constant name in code	Description
`C_MAXPARARUN`	The total number of parameter sets to simulate.
`C_MAXSIMRUN`	The total number of simulations per parameter set to run.
`C_OUTPUTFOLDER`	The folder where simulation output files are saved. Note that this folder need not exist *a priori* to executing the simulation in Windows.
`C_SHUTDOWN`	Boolean for shutting down the computer at the end of the simulation. Used for large sensitivity analyses and vacations.
`C_STRAINS`	The number of strains/serotypes of the pathogen to simulate. In the case of dengue, this was set to 4.
`C_MMAXINITIALAGE`	The maximum mosquito age in days of the initialised mosquito population.
`C_NMAXINITIALAGE`	The maximum human age in year of the initialised human population.
`C_YEAR`	The number of days per year.
`C_INITIALMINTIME`	The minimum burn in period of the initial simulations. This gives outbreaks in almost entirely susceptible populations, such that R_0 can be calculated from the initial growth rate.

C_INITIALMAXTIME	The maximum burn in period for the initial simulations from which the user-defined simulations begin. This ensures that seroprevalence has reached some dynamic equilibrium.
C_NSIZERECORD	The maximum number of humans in the census to record at the end of the simulation. This prevents the large consumption of disk space in large sensitivity analyses and population sizes.
C_NAOIRECORD	The total number of ages of most recent heterotypic infections to store.
C_MAXINTROATTEMPT	The maximum number of attempts to introduce an infection from an external source (outside of the lattice). This is done in order to prevent infinite loops in situations where seroprevalence is exceptionally high (i.e. after an initial outbreak with parameter values describing high transmissibility).

C_THREADSPERBLOCK	The total number of GPU threads assigned per virtual block when calling kernels (or functions parallelised on the device). The maximum number of threads per block is limited by the GPU architecture. Generally, powers of two above and equal to 32 is recommended. Depending upon the kernel, this number can be optimised to minimise kernel run-time such that a balance is struck between shared memory reads/writes (low-level memory reads and writes within the same block) and divergence of threads within the same block (threads may take different computational pathways: i.e. one individual may be aged, the other may be killed and birthed). In order to optimise this value, experiments were done to minimise the run time of the demographic process. More information on selecting the optimal number of GPU threads per block in a CUDA kernel can be found in the NVIDIA CUDA manual.
C_THREADSPERBLOCKSUM	Total number of GPU threads assigned per virtual block when calling reduction kernels, or kernels related to summing arrays of values.

Variables

Copies of variables that stored on both the host and device were prefixed with `h_` and `d_` respectively. Otherwise, variables were only stored on the host/device in which they were created, i.e. variables without the above prefix were declared on the device in functions containing the flags `__global__` and on the host in other cases. Here, the main variables of the simulation are listed along with a brief description.

Census variables

For human and mosquito individuals, variables were prefixed with `n` and `m` respectively. Each variable corresponded to a single piece of information about an individual, such as age, or community. Each census variables was an array of values, where the index within the array mapped to the identity of an individual. Each human variable had length as a multiple of the total number of human individuals `h_nSize`, and each mosquito variable had length as a multiple of the maximum possible number of mosquitoes `h_mSize`. The census variables used during the simulation are outlined below, note whether these apply to humans (`n`) and/or mosquitoes (`m`), and the mapping of array index to individual ID.

Variable name in code	Human/mosquito	Description and mapping
`Age`	`n` & `m`	Age in days of each individuals. 1-to-1 mapping for array index → individual ID.
`Dead`	`m`	Whether each mosquito is alive or dead. This is used in keeping track of fluctuations in mosquito population size. 1-to-1 mapping for array index → individual ID.
`Exposed`	`n` & `m`	Age at which an infected individual becomes infectious. 1-to-1 mapping for array index → individual ID.

`History`	`n`	Immunological history of a human individual: (i) `0` corresponds to no previous infection, (ii) `65535` corresponds to infection at age zero, (iii) otherwise, corresponds to age of infection. The mapping from array index → individual ID depends upon the number of strains of pathogen. For dengue, this results in a 4-to-1 mapping. In general, index mod `h_nSize` = ID. This means that the immunological history is sorted by strain first, and then by individual ID, where the immunological history of strain `s` of individual `idx` is contained in entry `s*h_nSize + idx`.
`InfectStatus`	`n` & `m`	Infection status of each individual: (i) `0` corresponds to uninfected, (ii) `1` corresponds to exposed (infected but not infectious), (iii) `2` corresponds to infectious. 1-to-1 mapping for array index → individual ID.

`PLE`	`n` & `m`	Random probability assigned at birth of individual. Each simulation day, this probability is compared to the cumulative probability of death evaluated at their current age in order to determine if an individual would die. 1-to-1 mapping for array index → individual ID.
`Recovery`	`n`	Age at which an infectious individual recovers/stops being infectious. 1-to-1 mapping for array index → individual ID.
`Strain`	`n` & `m`	Current/most-recent infecting strain of pathogen. For dengue, takes 0–3. 1-to-1 mapping for array index → individual ID.
`SubPopulation`	`n` & `m`	Community, or sub-population, which the individual is resident to. 1-to-1 mapping for array index → individual ID.

Counter variables

In order to reduce unnecessary data storage (by writing census daily) or computation (by calculating population totals from each census daily), counter variables were declared which kept track of the total number of individuals which matched specific epidemiological and demographical criteria. These counter variables were increased or decreased on the fly, as individuals moved between demographical and epidemiological states. Outlined below are some of the information kept track of over time.

Counter name in code	Human/mosquito	Description
`DeadCount`	`m`	Counted the number of dead individuals per block of device threads. Allowed usage of lower-level memory (shared memory access across blocks of threads) on the device in order to get the correct mosquito population size at each time step.
`SubPopCount`	`n` & `m`	Total number of individuals within each subpopulation at the current time step only.
`InfectedSubPopCount`	`n` & `m`	Total number of infected individuals within each subpopulation of each strain at the current time step only. First sorted by strain and then by subpopulation. In other words, the number of infected individuals in subpopulation `subPop` of strain `s` is in entry `s*subPopTotal + subPop`, where `subPopTotal` is the total number of subpopulations.

`InfectedCount`	`n`	Total number of infected individuals of each strain at each time step. First sorted by time, then by strain. In other words, the total number of infection individuals of strain `s` and time `t` is in position `t*C_STRAINS + s` of the array, where `C_STRAINS` is the total number of pathogen strains/serotypes.
`OneSubPopCount`	`n`	Total number of infected individuals of each strain at each time step within a randomly chosen sub-population. This data is later used to calculate the probability of two serotypes co-circulating within the same community at any given time. Arranged as above.
`Count`	`n` & `m`	Total number of individuals at each time step.
`ReductionInfectedCount` and `ReductionCount`	`n` & `m`	Used quickly calculating the total number of (infected) individuals in the entire meta-population.

Other variables

Counter name in code	Human/mosquito	Description
`SubPopIndex`	`n` & `m`	IDs of individuals arranged by subpopulation.
`SubPopLoc`	`n` & `m`	Indices of `SubPopIndex` which correspond to the start of the next subpopulation.
`SubPopSize`	`n` & `m`	Maximum number of individuals within each community. The above three variables are used to randomly select individuals to infect within a given subpopulation.
`AgeOfInfection`	`n` & `m`	Ages of the most recent heterotypic human infections. The total number of ages for each exposure (1st, 2nd, 3rd and 4th) recorded are given by `C_NAOIRECORD`.
`metaPopCols` and `metaPopRows`	-	Dimensions of the lattice community structure.
`subPopTotal`	-	Total number of communities.
`randStates`	-	The random number generator states used to generate random numbers for demographical and epidemiological processes on the device. The number of random number generator states corresponds to the total number of possible active threads on the GPU architecture.

Appendix B

Model source code

In this appendix, all commented `.cpp` and `.cu` source files for the model in Chapter 2 are presented. All header files which only contain function declarations are excluded here.

.cpp source files

These source files are compiled with `cl` and only declare and call) functions that are executed on the host (i.e. the CPU). A description of every source file is provided at the top of each.

architecture.cpp

```
// architecture.cpp: Setups up the GPU architectural properties space with the correct
// properties of the current GPU device. This saves on relying on user input/knowledge
// of their own architecture and offers better portability between different GPUs.

#include "cuda_runtime_api.h"   // CUDA functions for getting GPU properties.
#include "architectureclass.h"  // GPU architectural properties definition.

// Gets the architectural properties of the GPU.
void setupArchitecture(Architecture* h_architecture)
{
    // CUDA device properties can only write to an int.
    int value;

    // Get the number of threads in a warp.
    cudaDeviceGetAttribute(&value, cudaDevAttrWarpSize, 0);
    h_architecture->threadsPerWarp = value;

    // Get the number of Streaming Multiprocessors (SMs) on the device.
    cudaDeviceGetAttribute(&value, cudaDevAttrMultiProcessorCount, 0);
    h_architecture->totalSM = value;

    // Get the total number of threads per SM, then calculate the number of warps per SM.
    cudaDeviceGetAttribute(&value, cudaDevAttrMaxThreadsPerMultiProcessor, 0);
    h_architecture->warpsPerSM = value / h_architecture->threadsPerWarp;
}
```

main.cpp

```
// main.cpp: the root file for the individual-based stochastic dengue model.

#include <iostream>             // Input/output to console.
#include <fstream>              // Save data to file.
#include <windows.h>            // Create output directory.
#include "censustypedef.h"      // Type definitions for census data.
#include "initial.h"            // User input for simulation.
#include "simulation.h"         // Dengue simulation.
#include "constant.h"           // Constants for simulation.
#include "parameterclass.h"     // Parameter space definition.
#include "parameter.h"          // Parameter space initialization.
#include "architectureclass.h"  // GPU architectural properties definition.
#include "architecture.h"       // GPU architectural property setup.

// Root function for initialization of simulation variables, and function
// calling to data collecting, demographic simulations and epidemiological simulations.
int main()
{
    // Get input on the number of years to simulate.
    uint32_t maxTime = C_YEAR*getYearInput();

    // Create the output file directory.
    CreateDirectoryA(C_OUTPUTFOLDER, NULL);

    // Get the architectural properties of the device.
    Architecture h_architecture;
    setupArchitecture(&h_architecture);

    // Declare the parameter space.
    Parameter h_parameter;

    // Run the simulation on a pre-compile-time defined number of parameter sets.
    for (uint32_t h_paraRun = 0; h_paraRun < C_MAXPARARUN; ++h_paraRun)
    {
        // Initialize the parameter space.
        initialParameter(&h_parameter, h_paraRun);

        // Read in the initial human population size, and the dimensions of the metapopulation
        ↪  lattice
        // from the parameter data.
        uint32_t h_nSize = static_cast<uint32_t>(h_parameter.nSize);
        uint32_t h_metaPopRows = static_cast<uint32_t>(h_parameter.metaPopRows);
        uint32_t h_metaPopCols = static_cast<uint32_t>(h_parameter.metaPopCols);

        // Initialize the total number of subpopulations.
        uint32_t h_subPopTotal{ h_metaPopRows*h_metaPopCols };

        // Initialize the maximum mosquito population size. This is the size at time zero of the
        ↪  simulation.
        uint32_t h_mSize = static_cast<uint32_t>(h_parameter.maxMosToHuman*h_nSize);

        // Initialize the total number of infected individuals at the start of the simulation.
        uint32_t nInitialInfected = 1;
        uint32_t mInitialInfected = 0;

        // Declare the GPU device variables for human census data.
        age* d_nAge;                        // Individual age (days)
        exposed* d_nExposed;                // Age at which infection becomes infectious.
        history* d_nHistory;                // Strains an individual is immune to.
        infectStatus* d_nInfectStatus;      // If the individual is susceptible, infected, or
        ↪  infectious.
        pLifeExpectancy* d_nPLE;            // Random probability determing life expectancy of the
        ↪  individual.
        recovery* d_nRecovery;              // Age at which infection ends.
        strain* d_nStrain;                  // Dengue serotype an individual is infected with.
        subPopulation* d_nSubPopulation;    // Subpopulation that the individual belongs to.

        // Declare the GPU device variables for the mosquito census data.
        age* d_mAge;                        // Individual age (days)
        dead* d_mDead;                      // Alive or dead
        exposed* d_mExposed;                // Age at which infection becomes infectious.
        infectStatus* d_mInfectStatus;      // If the individual is susceptible, infected, or
        ↪  infectious.
        pLifeExpectancy* d_mPLE;            // Random probability determing life expectancy of the
        ↪  individual.
```

```
strain* d_mStrain;                  // Dengue serotype an individual is infected with.
subPopulation* d_mSubPopulation;    // Subpopulation that the individual belongs to.

// Declare the device variables used in the initialization and demographic update.
float* d_nSurvival;                 // The human cumulative survival function.
float* d_mSurvival;                 // The mosquito cumulative survival function.
float* d_mExpectedPopSize;          // Expected population size of mosquitoes.

// Declare the GPU device variables for counting different sets of individuals.
uint16_t *d_mDeadCount;                 // Number of dead mosquitoes per GPU block.
uint32_t *d_nSubPopCount;               // Number of humans per subpopulation.
uint32_t *d_mSubPopCount;               // Number of mosquitoes per subpopulation.
uint32_t *d_nInfectedSubPopCount;       // Number of infected humans per subpopulation per
↪  strain.
uint32_t *d_mInfectedSubPopCount;       // Number of infected humans per subpopulation per
↪  strain.
uint32_t *d_nInfectedCount;             // Time series for the number of infected humans per
↪  strain.
uint32_t* d_nOneSubPopInfectedCount;    // Time series for the number of infected humans per
↪  strain for a specific subpopulation.
uint32_t *d_nCount;                     // Time series for the number of humans.
uint32_t *d_mCount;                     // Time series for the number of mosquitoes.
uint32_t *d_nReductionInfectedCount;    // Used in summing the number of infected humans per
↪  strain across all subpopulations.
uint32_t *d_nReductionCount;            // Used in summing the number of humans across all
↪  subpopulations.
uint32_t *d_mReductionCount;            // Used in summing the number of mosquitoes across
↪  all subpopulations.

// Declare the GPU device variables for disease transmission.
uint32_t *d_nSubPopIndex, *d_mSubPopIndex;              // Census indices ordered by
↪  sub-population.
uint32_t *d_nSubPopLoc, *d_mSubPopLoc;                  // Indices of the above where a new
↪  sub-population begins in the ordering.
uint32_t *d_nSubPopSize, *d_mSubPopSize;                // The maximum number of individuals
↪  per sub-population.
uint32_t *d_nTransmission, *d_mTransmission;            // Transmission numbers per
↪  subpopulation per strain.
uint32_t *d_nAgeOfInfection, *d_nAgeOfInfectionCount;   // Ages of the last few infections
↪  for each novel exposure.

// Declare the GPU device variables which are constant after user input.
uint32_t *d_nSize;          // Human population size.
uint32_t *d_mSize;          // Maximum mosquito population size.
uint32_t *d_metaPopCols;    // Number of colums in the metapopulation lattice.
uint32_t *d_metaPopRows;    // Number of rows in the metapopulation lattice.
uint32_t *d_subPopTotal;    // Total number of subpopulations.

// Declare the GPU device variables for random number generation on the GPU.
curandState_t *d_randStates;

// Initialize the number of blocks required on the GPU given the number of threads desired
↪  to be used on
// each block in order to have a thread per individual.
uint32_t nGridSize = static_cast<uint32_t>(ceil(h_nSize /
↪  static_cast<float>(C_THREADSPERBLOCK)));
uint32_t mGridSize = static_cast<uint32_t>(ceil(h_mSize /
↪  static_cast<float>(C_THREADSPERBLOCK)));

// Initialize the number of blocks such that each thread on the block is assigned to one
↪  subpopulation. This is used
// in summing count data across all subpopulations.
uint32_t reductionSize{ static_cast<uint32_t>(ceil(h_subPopTotal /
↪  static_cast<float>(C_THREADSPERBLOCKSUM))) };

// Initialize the total number of active threads on the device at any one time. This will be
↪  used for
// determining the number of random number generators to be created on the device.
uint32_t totalActiveThreads{
↪  h_architecture.totalSM*h_architecture.warpsPerSM*h_architecture.threadsPerWarp };

// Compute the maximum time between the user input and the pre-defined initial simulation
↪  length.
// This is to ensure overflow does not occur when recording time series data in the initial
↪  simulation.
uint32_t timeSeriesMaxTime = static_cast<uint32_t>(fmaxf(static_cast<float>(maxTime),
```

```
static_cast<float>(C_INITIALMAXTIME)));

// Allocate space for the device variables onto the device. Allocate (roughly)
// from the largest in size to the smallest.
cudaMalloc((void **)&d_nAge, sizeof(age)*h_nSize);
cudaMalloc((void **)&d_nExposed, sizeof(exposed)*h_nSize);
cudaMalloc((void **)&d_nHistory, sizeof(history)*h_nSize*C_STRAINS);
cudaMalloc((void **)&d_nInfectStatus, sizeof(infectStatus)*h_nSize);
cudaMalloc((void **)&d_nPLE, sizeof(pLifeExpectancy)*h_nSize);
cudaMalloc((void **)&d_nRecovery, sizeof(recovery)*h_nSize);
cudaMalloc((void **)&d_nStrain, sizeof(strain)*h_nSize);
cudaMalloc((void **)&d_nSubPopulation, sizeof(subPopulation)*h_nSize);
cudaMalloc((void **)&d_mAge, sizeof(age)*h_mSize);
cudaMalloc((void **)&d_mDead, sizeof(dead)*h_mSize);
cudaMalloc((void **)&d_mExposed, sizeof(exposed)*h_mSize);
cudaMalloc((void **)&d_mInfectStatus, sizeof(infectStatus)*h_mSize);
cudaMalloc((void **)&d_mPLE, sizeof(pLifeExpectancy)*h_mSize);
cudaMalloc((void **)&d_mStrain, sizeof(strain)*h_mSize);
cudaMalloc((void **)&d_mSubPopulation, sizeof(subPopulation)*h_mSize);
cudaMalloc((void **)&d_nSubPopIndex, sizeof(uint32_t)*h_nSize);
cudaMalloc((void **)&d_mSubPopIndex, sizeof(uint32_t)*h_mSize);
cudaMalloc((void **)&d_randStates, sizeof(curandState_t)*totalActiveThreads);
cudaMalloc((void **)&d_nInfectedSubPopCount, sizeof(uint32_t)*h_subPopTotal*C_STRAINS);
cudaMalloc((void **)&d_mInfectedSubPopCount, sizeof(uint32_t)*h_subPopTotal*C_STRAINS);
cudaMalloc((void **)&d_nTransmission, sizeof(uint32_t)*h_subPopTotal*C_STRAINS);
cudaMalloc((void **)&d_mTransmission, sizeof(uint32_t)*h_subPopTotal*C_STRAINS);
cudaMalloc((void **)&d_nAgeOfInfection, sizeof(uint32_t)*C_NAOIRECORD*C_STRAINS);
cudaMalloc((void **)&d_nSubPopCount, sizeof(uint32_t)*h_subPopTotal);
cudaMalloc((void **)&d_mSubPopCount, sizeof(uint32_t)*h_subPopTotal);
cudaMalloc((void **)&d_nSubPopLoc, sizeof(uint32_t)*h_subPopTotal);
cudaMalloc((void **)&d_mSubPopLoc, sizeof(uint32_t)*h_subPopTotal);
cudaMalloc((void **)&d_nSubPopSize, sizeof(uint32_t)*h_subPopTotal);
cudaMalloc((void **)&d_mSubPopSize, sizeof(uint32_t)*h_subPopTotal);
cudaMalloc((void **)&d_mDeadCount, sizeof(uint16_t)*mGridSize);
cudaMalloc((void **)&d_nSurvival, sizeof(float)*(C_NMAXINITIALAGE + 1));
cudaMalloc((void **)&d_mSurvival, sizeof(float)*(C_MMAXINITIALAGE + 1));
cudaMalloc((void **)&d_nReductionInfectedCount, sizeof(uint32_t)*C_STRAINS*reductionSize);
cudaMalloc((void **)&d_nReductionCount, sizeof(uint32_t)*reductionSize);
cudaMalloc((void **)&d_mReductionCount, sizeof(uint32_t)*reductionSize);
cudaMalloc((void **)&d_nOneSubPopInfectedCount,
↪  sizeof(uint32_t)*C_STRAINS*(timeSeriesMaxTime + 1));
cudaMalloc((void **)&d_nInfectedCount, sizeof(uint32_t)*C_STRAINS*(timeSeriesMaxTime + 1));
cudaMalloc((void **)&d_nCount, sizeof(uint32_t)*(timeSeriesMaxTime + 1));
cudaMalloc((void **)&d_mCount, sizeof(uint32_t)*(timeSeriesMaxTime + 1));
cudaMalloc((void **)&d_nAgeOfInfectionCount, sizeof(uint32_t)*C_STRAINS);
cudaMalloc((void **)&d_nSize, sizeof(uint32_t));
cudaMalloc((void **)&d_mSize, sizeof(uint32_t));
cudaMalloc((void **)&d_metaPopCols, sizeof(uint32_t));
cudaMalloc((void **)&d_metaPopRows, sizeof(uint32_t));
cudaMalloc((void **)&d_subPopTotal, sizeof(uint32_t));
cudaMalloc((void **)&d_mExpectedPopSize, sizeof(float));

// Copy memory from the host to the allocated space in the device.
cudaMemcpy(d_metaPopCols, &h_metaPopCols, sizeof(uint32_t), cudaMemcpyHostToDevice);
cudaMemcpy(d_metaPopRows, &h_metaPopRows, sizeof(uint32_t), cudaMemcpyHostToDevice);
cudaMemcpy(d_subPopTotal, &h_subPopTotal, sizeof(uint32_t), cudaMemcpyHostToDevice);
cudaMemcpy(d_nSize, &h_nSize, sizeof(uint32_t), cudaMemcpyHostToDevice);
cudaMemcpy(d_mSize, &h_mSize, sizeof(uint32_t), cudaMemcpyHostToDevice);

// Save to file important constants/parameters for the simulation run.
std::ofstream parameterData(static_cast<std::string>(C_OUTPUTFOLDER) + "parameterData" +
↪  std::to_string(h_paraRun) + ".csv");
parameterData << "nSize," << "maxMosToHuman," << "minMosToHuman," << "latticeRows," <<
↪  "latticeCols," << "commSize,";
parameterData << "bitingRate," << "mnBitingSuccess," << "nmBitingSuccess," << "EIPRange," <<
↪  "EIP," << "IIP," << "Recovery,";
parameterData << "kernelSD," << "EIRate," << "longDistance,";
parameterData << "mDemoScale," << "mDemoShape," << "nDemoScale," << "nDemoShape,";
parameterData << "nInfantScale," << "nInfantShape," << "nInfToDemoLoc";
parameterData << "\n";
parameterData << h_nSize << "," << h_parameter.maxMosToHuman << "," <<
↪  h_parameter.minMosToHuman << "," << h_metaPopRows << "," << h_metaPopCols << "," <<
↪  h_nSize / (static_cast<float>(h_metaPopRows*h_metaPopCols));
parameterData << "," << h_parameter.bitingRate << "," << h_parameter.mnBitingSuccess << ","
↪  << h_parameter.nmBitingSuccess << "," << h_parameter.exIncPeriodRange;
parameterData << "," << h_parameter.mExposed << "," << h_parameter.nExposed << "," <<
↪  h_parameter.recovery;
```

```
parameterData << "," << h_parameter.kernelStandardDeviation << "," <<
↪  h_parameter.externalInfection << "," << h_parameter.longDistance;
parameterData << "," << h_parameter.mScaleLifeExpectancy << "," <<
↪  h_parameter.mShapeLifeExpectancy;
parameterData << "," << h_parameter.nScaleLifeExpectancy << "," <<
↪  h_parameter.nShapeLifeExpectancy;
parameterData << "," << h_parameter.nScaleInfantMortality << "," <<
↪  h_parameter.nShapeInfantMortality << "," << h_parameter.nLocWeibull;
parameterData.close();

// Run the dengue simulation for a long period of time in order to setup
// initial conditions for future runs of the simulation. The initial simulation
// also helps with the post-program calculation of R0.
initialSimulation(d_nAge, d_nExposed, d_nHistory, d_nInfectStatus, d_nPLE, d_nRecovery,
↪  d_nStrain, d_nSubPopulation,
d_mAge, d_mDead, d_mExposed, d_mInfectStatus, d_mPLE, d_mStrain, d_mSubPopulation,
↪  d_randStates, d_nSubPopIndex, d_mSubPopIndex,
d_nInfectedSubPopCount, d_mInfectedSubPopCount, d_nTransmission, d_mTransmission,
↪  d_nAgeOfInfection, d_nAgeOfInfectionCount,
d_nSubPopCount, d_mSubPopCount, d_nSubPopLoc, d_mSubPopLoc, d_nSubPopSize, d_mSubPopSize,
↪  d_mDeadCount, d_nSurvival, d_mSurvival,
d_nReductionInfectedCount, d_nReductionCount, d_mReductionCount, d_nOneSubPopInfectedCount,
↪  d_nInfectedCount,
d_nCount, d_mCount, d_nSize, d_mSize, d_metaPopCols, d_metaPopRows, d_subPopTotal,
↪  d_mExpectedPopSize,
nInitialInfected, mInitialInfected, nGridSize, mGridSize, h_nSize, h_mSize, h_subPopTotal,
↪  h_parameter, h_architecture, h_paraRun);

// Run the dengue simulation multiple times over a user-defined period of simulation time,
↪  given
// the initial conditions as the output of the long initial simulation run.
simulation(d_nAge, d_nExposed, d_nHistory, d_nInfectStatus, d_nPLE, d_nRecovery, d_nStrain,
↪  d_nSubPopulation,
d_mAge, d_mDead, d_mExposed, d_mInfectStatus, d_mPLE, d_mStrain, d_mSubPopulation,
↪  d_randStates, d_nSubPopIndex, d_mSubPopIndex,
d_nInfectedSubPopCount, d_mInfectedSubPopCount, d_nTransmission, d_mTransmission,
↪  d_nAgeOfInfection, d_nAgeOfInfectionCount,
d_nSubPopCount, d_mSubPopCount, d_nSubPopLoc, d_mSubPopLoc, d_nSubPopSize, d_mSubPopSize,
↪  d_mDeadCount, d_nSurvival, d_mSurvival,
d_nReductionInfectedCount, d_nReductionCount, d_mReductionCount, d_nOneSubPopInfectedCount,
↪  d_nInfectedCount,
d_nCount, d_mCount, d_nSize, d_mSize, d_metaPopCols, d_metaPopRows, d_subPopTotal,
↪  d_mExpectedPopSize,
nInitialInfected, mInitialInfected, nGridSize, mGridSize, h_nSize, h_mSize, h_subPopTotal,
↪  maxTime, h_parameter, h_architecture, h_paraRun);

// Free the allocated space on the device for the device variables (this will prevent memory
↪  leaks). Furthermore,
// it was suggested to adopt a "Last In, First Out" (LIFO) strategy.
cudaFree(d_mExpectedPopSize);
cudaFree(d_subPopTotal);
cudaFree(d_metaPopRows);
cudaFree(d_metaPopCols);
cudaFree(d_mSize);
cudaFree(d_nSize);
cudaFree(d_mCount);
cudaFree(d_nCount);
cudaFree(d_nInfectedCount);
cudaFree(d_nOneSubPopInfectedCount);
cudaFree(d_mReductionCount);
cudaFree(d_nReductionCount);
cudaFree(d_nReductionInfectedCount);
cudaFree(d_nAgeOfInfectionCount);
cudaFree(d_mSurvival);
cudaFree(d_nSurvival);
cudaFree(d_mDeadCount);
cudaFree(d_mSubPopSize);
cudaFree(d_nSubPopSize);
cudaFree(d_mSubPopLoc);
cudaFree(d_nSubPopLoc);
cudaFree(d_mSubPopCount);
cudaFree(d_nSubPopCount);
cudaFree(d_nAgeOfInfection);
cudaFree(d_mTransmission);
cudaFree(d_nTransmission);
cudaFree(d_mInfectedSubPopCount);
```

```
        cudaFree(d_nInfectedSubPopCount);
        cudaFree(d_randStates);
        cudaFree(d_mSubPopIndex);
        cudaFree(d_nSubPopIndex);
        cudaFree(d_mSubPopulation);
        cudaFree(d_mStrain);
        cudaFree(d_mPLE);
        cudaFree(d_mInfectStatus);
        cudaFree(d_mExposed);
        cudaFree(d_mDead);
        cudaFree(d_mAge);
        cudaFree(d_nSubPopulation);
        cudaFree(d_nStrain);
        cudaFree(d_nRecovery);
        cudaFree(d_nPLE);
        cudaFree(d_nInfectStatus);
        cudaFree(d_nHistory);
        cudaFree(d_nExposed);
        cudaFree(d_nAge);
    }

    // Explicitly destroy and clean up all resources associated with the current device in the
    ↪  current process.
    cudaDeviceReset();

    // To overwrite the recording progress bar with "Press any key..."
    std::cout << "\b\b\b\b\b\b\b\b\b\b\b\b\b\b\b\b\b\b\b\b\b\b\b\b\b\b\b\b\b\b\b\b\b\b\b\b\b";

    // Shutdown the machine if requested.
    if (C_SHUTDOWN)
    {
    system("shutdown -s");
    }

    // Standard (no error) return with main().
    return 0;
}
```

initial.cpp

```
// inital.cpp: functions requesting user input for the number of simulation years,
// iniital number of humans, and the size of the lattice meta-population.
// Also contains the function for the initial run of the dengue simulation in order
// to setup the initial conditions for future simulation runs.

#include <iostream>             // Input/output to console.
#include <fstream>              // Writing data to file.
#include <ctime>                // Simulation timings.
#include "censustypedef.h"      // Type definitions for census data.
#include "setuprng.h"           // CUDA random number intialization.
#include "demographic.h"        // Human and mosquito demographic updating.
#include "epidemic.h"           // Simulation of transmission events.
#include "datacollect.h"        // Data collecting and collating.
#include "constant.h"           // Constants for the simulation.
#include "math_constants.h"     // Constant for computing seasonality (\pi).
#include "parameterclass.h"     // Parameter space definition.
#include "architectureclass.h"  // GPU architectural properties definition.

// Retrieves a fixed width integer for the number of simulation years.
uint32_t getYearInput()
{
    uint32_t input;
    std::cout << "Input the number of years you wish to simulate: ";
    std::cin >> input;
    return input;
}

// Runs the dengue simulation for a long time in order to setup
// initial conditions for future runs of the simulation.
void initialSimulation(age* d_nAge,
                       exposed* d_nExposed,
                       history* d_nHistory,
                       infectStatus* d_nInfectStatus,
                       pLifeExpectancy* d_nPLE,
                       recovery* d_nRecovery,
                       strain* d_nStrain,
                       subPopulation* d_nSubPopulation,
                       age* d_mAge,
                       dead* d_mDead,
                       exposed* d_mExposed,
                       infectStatus* d_mInfectStatus,
                       pLifeExpectancy* d_mPLE,
                       strain* d_mStrain,
                       subPopulation* d_mSubPopulation,
                       curandState_t* d_randStates,
                       uint32_t* d_nSubPopIndex,
                       uint32_t* d_mSubPopIndex,
                       uint32_t* d_nInfectedSubPopCount,
                       uint32_t* d_mInfectedSubPopCount,
                       uint32_t* d_nTransmission,
                       uint32_t* d_mTransmission,
                       uint32_t* d_nAgeOfInfection,
                       uint32_t* d_nAgeOfInfectionCount,
                       uint32_t* d_nSubPopCount,
                       uint32_t* d_mSubPopCount,
                       uint32_t* d_nSubPopLoc,
                       uint32_t* d_mSubPopLoc,
                       uint32_t* d_nSubPopSize,
                       uint32_t* d_mSubPopSize,
                       uint16_t* d_mDeadCount,
                       float* d_nSurvival,
                       float* d_mSurvival,
                       uint32_t* d_nReductionInfectedCount,
                       uint32_t* d_nReductionCount,
                       uint32_t* d_mReductionCount,
                       uint32_t* d_nOneSubPopInfectedCount,
                       uint32_t* d_nInfectedCount,
                       uint32_t* d_nCount,
                       uint32_t* d_mCount,
                       uint32_t* d_nSize,
                       uint32_t* d_mSize,
                       uint32_t* d_metaPopCols,
                       uint32_t* d_metaPopRows,
                       uint32_t* d_subPopTotal,
```

```
                        float* d_mExpectedPopSize,
                        const uint32_t nInitialInfected,
                        const uint32_t mInitialInfected,
                        const uint32_t nGridSize,
                        const uint32_t mGridSize,
                        const uint32_t h_nSize,
                        const uint32_t h_mSize,
                        const uint32_t h_subPopTotal,
                        const Parameter h_parameter,
                        const Architecture h_architecture,
                        const uint32_t h_paraRun)
{
    // Initialize the array containing the expected size of the mosquito population size at each
    ↪  time step.
    float* h_mExpectedPopSize = new float[C_INITIALMAXTIME + 1];

    // Initialize the array containing the number of infected humans per strain and total
    // number of individuals at each time step.
    uint32_t* h_nInfectedCount = new uint32_t[(C_INITIALMAXTIME + 1)*C_STRAINS];
    uint32_t* h_nCount = new uint32_t[C_INITIALMAXTIME + 1];

    // Set up the random number generators on the GPU.
    setupCudaRNG(d_randStates, h_architecture);

    // Initialize the start time for the simulation.
    std::clock_t start{ std::clock() };
    float duration;

    // Run the initial simulation a pre-compile-time defined number of times
    // for a much shorter amount of time, except for the last run. Output
    // the epidemiological information at the end of each run.
    for (uint32_t h_simRun = 0; h_simRun < C_MAXSIMRUN; ++h_simRun)
    {
    // Output the simulation number for the user to see progress.
    if (h_simRun < C_MAXSIMRUN - 1)
    {
        std::cout << "\rRunning short initial simulation " << h_simRun + 1
        << " of " << C_MAXSIMRUN - 1 << " for parameter set " << h_paraRun + 1 << " of " <<
        ↪  C_MAXPARARUN << " . . . ";
    }
    else
    {
        std::cout << "\nRunning final initial simulation for parameter set " << h_paraRun + 1 << "
        ↪  of " << C_MAXPARARUN << " . . . " << std::endl;
    }

    // Initialize the human and mosquito populations.
    demographicInitialization(d_nAge, d_nHistory, d_nInfectStatus, d_nPLE, d_nSubPopulation, d_mAge,
    ↪  d_mDead, d_mInfectStatus, d_mPLE, d_mSubPopulation,
    d_nSubPopIndex, d_mSubPopIndex, d_randStates, d_nSubPopCount, d_mSubPopCount, d_nSubPopLoc,
    ↪  d_mSubPopLoc, d_nSubPopSize, d_mSubPopSize,
    d_mDeadCount, d_nSurvival, d_mSurvival, d_nSize, d_mSize, d_subPopTotal, h_nSize, h_mSize,
    ↪  h_subPopTotal, h_parameter, h_architecture);

    // Initialize the infections in the human and mosquito populations.
    epidemicInitial(d_nAge, d_nHistory, d_nInfectStatus, d_nRecovery, d_nStrain, d_nSubPopulation,
    ↪  d_mInfectStatus, d_mStrain, d_mSubPopulation,
    d_randStates, d_nInfectedSubPopCount, d_mInfectedSubPopCount, d_nSize, d_subPopTotal,
    nInitialInfected, mInitialInfected, nGridSize, mGridSize, h_subPopTotal, h_parameter,
    ↪  h_architecture);

    // Declare the timestep counter t.
    uint32_t t;

    // Define the maximum number of days to run the simulation for.
    uint32_t maxTime = (h_simRun < (C_MAXSIMRUN - 1)) ? C_INITIALMINTIME : C_INITIALMAXTIME;

    // For every time step, run the simulation.
    for (t = 0; t < maxTime; ++t)
    {
        // Sum up the number of infected/total individuals across all sub-populations and store the
        ↪  results in time-series.
        dataCollect(d_nReductionInfectedCount, d_nReductionCount, d_mReductionCount,
        ↪  d_nOneSubPopInfectedCount, d_nInfectedCount,
        d_nCount, d_mCount, d_nInfectedSubPopCount, d_nSubPopCount, d_mSubPopCount, 0,
        ↪  h_subPopTotal, t);
```

```
    // Determine the expected mosquito population size. This is to model the rainfall
    ↪  seasonality of the
    // moqsuito population density.
    h_mExpectedPopSize[t + 1] = 2 * CUDART_PI_F / static_cast<float>(C_YEAR);
    h_mExpectedPopSize[t + 1] = -cos(h_mExpectedPopSize[t + 1] * (t + 1));
    h_mExpectedPopSize[t + 1] *= h_mSize*(1 - h_parameter.minMosToHuman /
    ↪  static_cast<float>(h_parameter.maxMosToHuman)) / 2.0f;
    h_mExpectedPopSize[t + 1] += h_mSize*(1 + h_parameter.minMosToHuman /
    ↪  static_cast<float>(h_parameter.maxMosToHuman)) / 2.0f;
    cudaMemcpy(d_mExpectedPopSize, &h_mExpectedPopSize[t + 1], sizeof(float),
    ↪  cudaMemcpyHostToDevice);

    // Run the human demographics on CUDA. For every individual, the human demographic function
    // will determine if an individual is due to die or not (from natural causes). If not, age
    // the individual by one day.
    nDemographic(d_nAge, d_nExposed, d_nHistory, d_nInfectStatus, d_nPLE, d_nRecovery,
    ↪  d_nStrain, d_nSubPopulation,
    d_randStates, d_nInfectedSubPopCount, d_nSize, d_subPopTotal, h_nSize, h_parameter,
    ↪  h_architecture);

    // Run the mosquito demographics on CUDA. For every individual, the mosquito demographic
    ↪  function
    // will determine if an individual is due to die or not (from natural causes). If not, age
    // the individual by one day.
    mDemographic(d_mAge, d_mDead, d_mExposed, d_mInfectStatus, d_mPLE, d_mStrain,
    ↪  d_mSubPopulation,
    d_randStates, d_mInfectedSubPopCount, d_mSubPopCount, d_mDeadCount,
    d_mSize, d_subPopTotal, h_mSize, d_mExpectedPopSize, h_parameter, h_architecture);

    // Determine the number of infections that occur from infected individuals, generate that
    ↪  many random numbers to determine
    // which subpopulation the transmission events occur in, and which individuals in those
    ↪  subpopulations they infect.
    epidemic(d_nAge, d_nExposed, d_nHistory, d_nInfectStatus, d_nRecovery, d_nStrain, d_mAge,
    ↪  d_mDead,
    d_mExposed, d_mInfectStatus, d_mStrain, d_randStates, d_nTransmission, d_mTransmission,
    d_nAgeOfInfection, d_nAgeOfInfectionCount, d_nInfectedSubPopCount, d_mInfectedSubPopCount,
    d_nSubPopCount, d_mSubPopCount, d_nInfectedCount, d_nSubPopIndex, d_nSubPopLoc,
    ↪  d_nSubPopSize,
    d_mSubPopIndex, d_mSubPopLoc, d_mSubPopSize, d_nSize, d_mSize, d_metaPopCols, d_metaPopRows,
    ↪  d_subPopTotal,
    h_subPopTotal, t, maxTime, h_parameter, h_architecture);

    // Print progress of the simulation every 10 years. Need maxTime >= 100 to prevent % error
    ↪  as
    // && reads from left to right. Else if for maxTime < 100 if population side is huge. They
    ↪  are separate
    // so that if maxTime is large, the console window isn't lagged by updates.
    if ((h_simRun == (C_MAXSIMRUN - 1)) && (maxTime >= 100) && (t % (maxTime / 100) == 0))
    {
        std::cout << "\r" << static_cast<int>(100 * (t / static_cast<float>(maxTime))) << "%
        ↪  completed . . .";
    }
    else if (maxTime < 100)
    {
        std::cout << "\r" << static_cast<int>(100 * (t / static_cast<float>(maxTime))) << "%
        ↪  completed . . .";
    }
}

// Sum up the number of infected/total individuals across all sub-populations one final time.
dataCollect(d_nReductionInfectedCount, d_nReductionCount, d_mReductionCount,
↪  d_nOneSubPopInfectedCount, d_nInfectedCount,
d_nCount, d_mCount, d_nInfectedSubPopCount, d_nSubPopCount, d_mSubPopCount, 0, h_subPopTotal,
↪  maxTime);

// Wait for device work to finish.
cudaDeviceSynchronize();

// Copy the number of infected individuals and total individuals time series to host memory.
cudaMemcpy(h_nInfectedCount, d_nInfectedCount, sizeof(uint32_t)*(maxTime + 1)*C_STRAINS,
↪  cudaMemcpyDeviceToHost);
cudaMemcpy(h_nCount, d_nCount, sizeof(uint32_t)*(maxTime + 1), cudaMemcpyDeviceToHost);

// Open human time series data files ready for recording, which saves all data which gets
```

```
    // recorded at every time step, for example population sizes, or the total
    // number of infected individuals.
    std::ofstream nInitial(static_cast<std::string>(C_OUTPUTFOLDER) + "nInitial" +
    ↪  std::to_string(h_paraRun) + "_" + std::to_string(h_simRun) + ".csv");

    // Write the headers of each column in the data file.
    uint32_t serotype{ 0 };
    nInitial << "t,popSize";
    while (serotype < C_STRAINS)
    {
        nInitial << "," << "DENV" << serotype + 1;
        ++serotype;
    }
    nInitial << "\n";

    // For every time step, record the time series data.
    for (uint32_t t = 0; t <= maxTime; ++t)
    {
        nInitial << t << "," << h_nCount[t];
        serotype = 0;
        while (serotype < C_STRAINS)
        {
            nInitial << "," << h_nInfectedCount[t*C_STRAINS + serotype];
            ++serotype;
        }
        nInitial << "\n";
    }

    // Close the data file for recording population size.
    nInitial.close();
    }

    // Output the total time for the simulation to run.
    duration = (std::clock() - start) / (float)CLOCKS_PER_SEC;
    std::cout << "\rTime for initialization: " << duration << "s" << std::endl;

    // Delete the dynamic-allocated arrays.
    delete[] h_mExpectedPopSize;
    delete[] h_nInfectedCount;
    delete[] h_nCount;
}
```

parameter.cpp

```
// parameter.cpp: contains function for the initialization of
// the different parameter sets for the dengue simulation (parameter sweep).

#include "parameterclass.h"     // Parameter space definition.
#include "constant.h"           // Simulation constants.

// Initializes different parameter sets fro dengue simulation (parameter sweep).
void initialParameter(Parameter* h_parameter,
uint32_t h_paraRun)
{
    // Non-epidemiological parameters.
    h_parameter->nSize = 5000000.0f;                                    // Number of human individuals
    ↪  in the metapopulation.
    h_parameter->metaPopRows = 125.0f;                                  // Number of rows in the
    ↪  metapopualtion lattice.
    h_parameter->metaPopCols = 125.0f;                                  // Number of columns in the
    ↪  metapopulation lattice.
    h_parameter->maxMosToHuman = 1.2f;                                  // Maximum mosquito to human
    ↪  ratio.
    h_parameter->minMosToHuman = 0.8f;                                  // Minimum mosquito to human
    ↪  ratio.
    h_parameter->nShapeInfantMortality = 0.4f;                          // Human life-expectancy
    ↪  bi-weibull scale parameter (burn in).
    h_parameter->nScaleInfantMortality = 1.0f / 100000.0f / C_YEAR;  // Human life-expectancy
    ↪  bi-weibull shape parameter (burn in).
    h_parameter->nScaleLifeExpectancy = 75.0f*C_YEAR;                   // Second (decay) human
    ↪  bi-weibull scale parameter. "Close to" life expectency.
    h_parameter->nShapeLifeExpectancy = 6.0f;                           // Second (decay)
    ↪  human-bi-weibull shape parameter.
    h_parameter->nLocWeibull = 8.0f*C_YEAR;                             // Age at which human
    ↪  life-expectancy that burn in distribution becomes decay out.
    h_parameter->mScaleLifeExpectancy = 23.0f;                          // Mosquito life-expectancy
    ↪  Weibull scale parameter.
    h_parameter->mShapeLifeExpectancy = 4.0f;                           // Mosquito life-expectancy
    ↪  Weibull shape parameter.

    // Epidemiological parameters.
    h_parameter->bitingRate = 0.6f;                 // The per day biting rate of mosquitoes.
    h_parameter->mnBitingSuccess = 0.5f;            // The probability of virus being transmitted
    ↪  from an infectious individual given a bite.
    h_parameter->nmBitingSuccess = 0.5f;            // The probability of virus being transmitted
    ↪  from an infectious individual given a bite.
    h_parameter->recovery = 4.0f;                   // The number of days humans are infectious.
    h_parameter->mExposed = 7.0f;                   // The number of days mosquitoes are infected,
    ↪  but not infectious (EIP).
    h_parameter->nExposed = 6.0f;                   // The number of days humans are infected , but
    ↪  not infectious.
    h_parameter->externalInfection = 0.01f;         // Imported infections per day per strain.
    h_parameter->longDistance = 0.0001f;            // The probability of a single infectious
    ↪  causing long distance transmission.
    h_parameter->exIncPeriodRange = 2.0f;           // Maximum difference in mean EIP in
    ↪  off/on-season with the mid-season.
    h_parameter->kernelStandardDeviation = 4.0f;    // The standard deviation of the
    ↪  normally-distributed disease dispersal kernel.
    // used in modelling spread of disease from a given subpopulation to surrounding subpopulations.

    // Define the initial parameter you will be changing
    if (C_MAXPARARUN > 1)
    {
        float* sweepPara = &h_parameter->metaPopRows;    // The parameter to be sweeped over.
        float myArray[8] = { 1.0f, 2.0f, 4.0f, 8.0f, 16.0f, 32.0f, 64.0f, 128.0f };
        *sweepPara = myArray[h_paraRun];
        h_parameter->metaPopCols = h_parameter->metaPopRows;
    }
}
```

simulation.cpp

```
// simulation.cpp: contains the code for running the dengue simulation
// multiple times over a user-defined period of time given initial conditions from
// a long initial simulation run.

#include <iostream>             // Input/output to console.
#include <fstream>              // Save data to file.
#include <random>               // (Pseudo-)random number generation.
#include <chrono>               // Seed for RNG.
#include <ctime>                // Simulation timings.
#include "censustypedef.h"      // Type definitions for census data.
#include "setuprng.h"           // CUDA random number initialization.
#include "demographic.h"        // Human and mosquito demographic updating.
#include "epidemic.h"           // Simulation of transmission events.
#include "datacollect.h"        // Data collecting and collating.
#include "constant.h"           // Non-epidemiological parameters.
#include "math_constants.h"     // Constant for computing seasonality (\pi).
#include "parameterclass.h"     // Parameter space definition.
#include "architectureclass.h"  // GPU architectural properties definition.

// Runs the dengue simulation a pre-defined number of times over
// a user-defined period of time given initial conditions from a long initial simulation run.
void simulation(age* d_nAge,
                exposed* d_nExposed,
                history* d_nHistory,
                infectStatus* d_nInfectStatus,
                pLifeExpectancy* d_nPLE,
                recovery* d_nRecovery,
                strain* d_nStrain,
                subPopulation* d_nSubPopulation,
                age* d_mAge,
                dead* d_mDead,
                exposed* d_mExposed,
                infectStatus* d_mInfectStatus,
                pLifeExpectancy* d_mPLE,
                strain* d_mStrain,
                subPopulation* d_mSubPopulation,
                curandState_t* d_randStates,
                uint32_t* d_nSubPopIndex,
                uint32_t* d_mSubPopIndex,
                uint32_t* d_nInfectedSubPopCount,
                uint32_t* d_mInfectedSubPopCount,
                uint32_t* d_nTransmission,
                uint32_t* d_mTransmission,
                uint32_t* d_nAgeOfInfection,
                uint32_t* d_nAgeOfInfectionCount,
                uint32_t* d_nSubPopCount,
                uint32_t* d_mSubPopCount,
                uint32_t* d_nSubPopLoc,
                uint32_t* d_mSubPopLoc,
                uint32_t* d_nSubPopSize,
                uint32_t* d_mSubPopSize,
                uint16_t* d_mDeadCount,
                float* d_nSurvival,
                float* d_mSurvival,
                uint32_t* d_nReductionInfectedCount,
                uint32_t* d_nReductionCount,
                uint32_t* d_mReductionCount,
                uint32_t* d_nOneSubPopInfectedCount,
                uint32_t* d_nInfectedCount,
                uint32_t* d_nCount,
                uint32_t* d_mCount,
                uint32_t* d_nSize,
                uint32_t* d_mSize,
                uint32_t* d_metaPopCols,
                uint32_t* d_metaPopRows,
                uint32_t* d_subPopTotal,
                float* d_mExpectedPopSize,
                const uint32_t nInitialInfected,
                const uint32_t mInitialInfected,
                const uint32_t nGridSize,
                const uint32_t mGridSize,
                const uint32_t h_nSize,
                const uint32_t h_mSize,
                const uint32_t h_subPopTotal,
```

```
                const uint32_t maxTime,
                const Parameter h_parameter,
                const Architecture h_architecture,
                const uint32_t h_paraRun)
{
    // Declare the CPU counters for the number of infected humans in one specific subpopulation,
    // the total number of infected humans per strain, and the total number of human and mosquitoes,
    // at each time step of the simulation.
    uint32_t* h_nOneSubPopInfectedCount = new uint32_t[(maxTime + 1)*C_STRAINS];
    uint32_t* h_nInfectedCount = new uint32_t[(maxTime + 1)*C_STRAINS];
    uint32_t *h_nCount = new uint32_t[maxTime + 1];
    uint32_t *h_mCount = new uint32_t[maxTime + 1];

    // Initialize the array containing the expected size of the mosquito population size at each
    ↪  time step.
    float* h_mExpectedPopSize = new float[maxTime + 1]{ static_cast<float>(h_mSize) };

    // Declare arrays for storing the initial conditions of each simulation run.
    age* h_nAge = new age[h_nSize];
    exposed* h_nExposed = new exposed[h_nSize];
    history* h_nHistory = new history[h_nSize*C_STRAINS];
    infectStatus* h_nInfectStatus = new infectStatus[h_nSize];
    pLifeExpectancy* h_nPLE = new pLifeExpectancy[h_nSize];
    recovery* h_nRecovery = new recovery[h_nSize];
    strain* h_nStrain = new strain[h_nSize];
    age* h_mAge = new age[h_mSize];
    dead* h_mDead = new dead[h_mSize];
    exposed* h_mExposed = new exposed[h_mSize];
    infectStatus* h_mInfectStatus = new infectStatus[h_mSize];
    pLifeExpectancy* h_mPLE = new pLifeExpectancy[h_mSize];
    strain* h_mStrain = new strain[h_mSize];
    uint32_t* h_nInfectedSubPopCount = new uint32_t[h_subPopTotal*C_STRAINS];
    uint32_t* h_mInfectedSubPopCount = new uint32_t[h_subPopTotal*C_STRAINS];
    uint32_t* h_nSubPopCount = new uint32_t[h_subPopTotal];
    uint32_t* h_mSubPopCount = new uint32_t[h_subPopTotal];
    uint16_t* h_mDeadCount = new uint16_t[mGridSize];

    // Record the data for alive humans (used in computing R0 from immunity landscape).
    // Declare memory for humans age and immunological history.
    uint32_t h_nSizeRecord = static_cast<uint32_t>(fminf(C_NSIZERECORD, h_nSize));
    age* h_nAgeRecord = new age[h_nSizeRecord];
    history* h_nHistoryRecord = new history[h_nSize*C_STRAINS];
    age* h_mAgeRecord = new age[h_mSize];
    infectStatus* h_mInfectStatusRecord = new infectStatus[h_mSize];

    // Declare memory for the most recent ages of infection for each novel exposure.
    uint32_t* h_nAgeOfInfection = new uint32_t[C_NAOIRECORD*C_STRAINS];

    // Copy the device data from the initial simulation run into host memory.
    cudaMemcpy(h_nAge, d_nAge, sizeof(age)*h_nSize, cudaMemcpyDeviceToHost);
    cudaMemcpy(h_nExposed, d_nExposed, sizeof(exposed)*h_nSize, cudaMemcpyDeviceToHost);
    cudaMemcpy(h_nHistory, d_nHistory, sizeof(history)*h_nSize*C_STRAINS, cudaMemcpyDeviceToHost);
    cudaMemcpy(h_nInfectStatus, d_nInfectStatus, sizeof(infectStatus)*h_nSize,
    ↪  cudaMemcpyDeviceToHost);
    cudaMemcpy(h_nPLE, d_nPLE, sizeof(pLifeExpectancy)*h_nSize, cudaMemcpyDeviceToHost);
    cudaMemcpy(h_nRecovery, d_nRecovery, sizeof(recovery)*h_nSize, cudaMemcpyDeviceToHost);
    cudaMemcpy(h_nStrain, d_nStrain, sizeof(strain)*h_nSize, cudaMemcpyDeviceToHost);
    cudaMemcpy(h_mAge, d_mAge, sizeof(age)*h_mSize, cudaMemcpyDeviceToHost);
    cudaMemcpy(h_mDead, d_mDead, sizeof(dead)*h_mSize, cudaMemcpyDeviceToHost);
    cudaMemcpy(h_mExposed, d_mExposed, sizeof(exposed)*h_mSize, cudaMemcpyDeviceToHost);
    cudaMemcpy(h_mInfectStatus, d_mInfectStatus, sizeof(infectStatus)*h_mSize,
    ↪  cudaMemcpyDeviceToHost);
    cudaMemcpy(h_mPLE, d_mPLE, sizeof(pLifeExpectancy)*h_mSize, cudaMemcpyDeviceToHost);
    cudaMemcpy(h_mStrain, d_mStrain, sizeof(strain)*h_mSize, cudaMemcpyDeviceToHost);
    cudaMemcpy(h_nInfectedSubPopCount, d_nInfectedSubPopCount,
    ↪  sizeof(uint32_t)*h_subPopTotal*C_STRAINS, cudaMemcpyDeviceToHost);
    cudaMemcpy(h_mInfectedSubPopCount, d_mInfectedSubPopCount,
    ↪  sizeof(uint32_t)*h_subPopTotal*C_STRAINS, cudaMemcpyDeviceToHost);
    cudaMemcpy(h_nSubPopCount, d_nSubPopCount, sizeof(uint32_t)*h_subPopTotal,
    ↪  cudaMemcpyDeviceToHost);
    cudaMemcpy(h_mSubPopCount, d_mSubPopCount, sizeof(uint32_t)*h_subPopTotal,
    ↪  cudaMemcpyDeviceToHost);
    cudaMemcpy(h_mDeadCount, d_mDeadCount, sizeof(uint16_t)*mGridSize, cudaMemcpyDeviceToHost);

    // Run the simulation a pre-compile-time defined number of times and output
    // demographical and epidemiological data at the end of each run.
```

```
for (uint32_t simRun = 0; simRun < C_MAXSIMRUN; ++simRun)
{
    // Choose a random subpopulation as the "special subpopulation". This
    // subpopulation will provide information on local serotype co-circulation.
    std::mt19937 rng(static_cast<unsigned
    ↪  int>(std::chrono::system_clock::now().time_since_epoch().count()));
    uint32_t h_specialSubPop = static_cast<uint32_t>(rng() % h_subPopTotal);

    // Set up the random number generators on the GPU.
    setupCudaRNG(d_randStates, h_architecture);

    // Output the simulation number for the user to see progress.
    std::cout << "\rRunning simulation " << simRun + 1
        << " of " << C_MAXSIMRUN << " for parameter set " << h_paraRun + 1 << " of " <<
        ↪  C_MAXPARARUN << " . . . " << std::endl;

    // Initialize the start time for the simulation.
    std::clock_t start{ std::clock() };
    float duration;

    // For every time step, run the simulation.
    for (uint32_t t = 0; t < maxTime; ++t)
    {
        // Sum up the number of infected/total individuals across all sub-populations and store
        ↪  the results in time-series. Also record
        // the special subpopulations infected totals.
        dataCollect(d_nReductionInfectedCount, d_nReductionCount, d_mReductionCount,
        ↪  d_nOneSubPopInfectedCount, d_nInfectedCount,
            d_nCount, d_mCount, d_nInfectedSubPopCount, d_nSubPopCount, d_mSubPopCount,
            ↪  h_specialSubPop, h_subPopTotal, t);

        // Determine the expected mosquito population size. This is to model the rainfall
        ↪  seasonality of the
        // moqsuito population density.
        h_mExpectedPopSize[t + 1] = 2 * CUDART_PI_F / static_cast<float>(C_YEAR);
        h_mExpectedPopSize[t + 1] = -cos(h_mExpectedPopSize[t + 1] * (t + 1));
        h_mExpectedPopSize[t + 1] *= h_mSize*(1 - h_parameter.minMosToHuman /
        ↪  static_cast<float>(h_parameter.maxMosToHuman)) / 2.0f;
        h_mExpectedPopSize[t + 1] += h_mSize*(1 + h_parameter.minMosToHuman /
        ↪  static_cast<float>(h_parameter.maxMosToHuman)) / 2.0f;
        cudaMemcpy(d_mExpectedPopSize, &h_mExpectedPopSize[t + 1], sizeof(float),
        ↪  cudaMemcpyHostToDevice);

        // Run the human demographics on CUDA. For every individual, the human demographic
        ↪  function
        // will determine if an individual is due to die or not (from natural causes). If not,
        ↪  age
        // the individual by one day.
        nDemographic(d_nAge, d_nExposed, d_nHistory, d_nInfectStatus, d_nPLE, d_nRecovery,
        ↪  d_nStrain, d_nSubPopulation,
            d_randStates, d_nInfectedSubPopCount, d_nSize, d_subPopTotal, h_nSize, h_parameter,
            ↪  h_architecture);

        // Run the mosquito demographics on CUDA. For every individual, the mosquito demographic
        ↪  function
        // will determine if an individual is due to die or not (from natural causes). If not,
        ↪  age
        // the individual by one day.
        mDemographic(d_mAge, d_mDead, d_mExposed, d_mInfectStatus, d_mPLE, d_mStrain,
        ↪  d_mSubPopulation,
            d_randStates, d_mInfectedSubPopCount, d_mSubPopCount, d_mDeadCount,
            d_mSize, d_subPopTotal, h_mSize, d_mExpectedPopSize, h_parameter, h_architecture);

        // Determine the number of infections that occur from infected individuals, generate
        ↪  that many random numbers to determine
        // which subpopulation the transmission events occur in, and which individuals in those
        ↪  subpopulations they infect.
        epidemic(d_nAge, d_nExposed, d_nHistory, d_nInfectStatus, d_nRecovery, d_nStrain,
        ↪  d_mAge, d_mDead,
            d_mExposed, d_mInfectStatus, d_mStrain, d_randStates, d_nTransmission,
            ↪  d_mTransmission,
            d_nAgeOfInfection, d_nAgeOfInfectionCount, d_nInfectedSubPopCount,
            ↪  d_mInfectedSubPopCount,
            d_nSubPopCount, d_mSubPopCount, d_nInfectedCount, d_nSubPopIndex, d_nSubPopLoc,
            ↪  d_nSubPopSize,
            d_mSubPopIndex, d_mSubPopLoc, d_mSubPopSize, d_nSize, d_mSize, d_metaPopCols,
            ↪  d_metaPopRows, d_subPopTotal,
```

```
        h_subPopTotal, t, maxTime, h_parameter, h_architecture);

    // Print progress of the simulation every 10 years. Need maxTime >= 100 to prevent %
    ↪  error as
    // && reads from left to right. Else if for maxTime < 100 if population side is huge.
    ↪  They are separate
    // so that if maxTime is large, the console window isn't lagged by updates.
    if ((maxTime >= 100) && (t % (maxTime / 100) == 0))
    {
        std::cout << "\b\b\b\b\b\b\b\b\b\b\b\b\b\b\b\b\b\b\b\b\b\b\b\b\b\b\b\b\b\b\b"
            << static_cast<int>(100 * (t / static_cast<float>(maxTime))) << "% completed . .
            ↪  .";
    }
    else if (maxTime < 100)
    {
        std::cout << "\b\b\b\b\b\b\b\b\b\b\b\b\b\b\b\b\b\b\b\b\b\b\b\b\b\b\b\b\b\b\b"
            << static_cast<int>(100 * (t / static_cast<float>(maxTime))) << "% completed . .
            ↪  .";
    }
}

// Do the final sum of the number of infected/total individuals across all subpopulations
↪  and store into time-series data.
dataCollect(d_nReductionInfectedCount, d_nReductionCount, d_mReductionCount,
↪  d_nOneSubPopInfectedCount, d_nInfectedCount,
    d_nCount, d_mCount, d_nInfectedSubPopCount, d_nSubPopCount, d_mSubPopCount,
    ↪  h_specialSubPop, h_subPopTotal, maxTime);

// Wait for device work to finish.
cudaDeviceSynchronize();

// Output the total time for the simulation to run.
duration = (std::clock() - start) / (float)CLOCKS_PER_SEC;
std::cout << "\b\b\b\b\b\b\b\b\b\b\b\b\b\b\b\b\b\b\b\b\b\b\b\bTime for simulation: " << duration <<
↪  "s" << std::endl;

// Copy to host memory the time series data for the number of infected/total individuals,
// as well as the time series for the number of infected individuals in a specific
↪  subpopulation,
// and the ages of infection of novel exposure.
cudaMemcpy(h_nOneSubPopInfectedCount, d_nOneSubPopInfectedCount, sizeof(uint32_t)*(maxTime +
↪  1)*C_STRAINS, cudaMemcpyDeviceToHost);
cudaMemcpy(h_nInfectedCount, d_nInfectedCount, sizeof(uint32_t)*(maxTime + 1)*C_STRAINS,
↪  cudaMemcpyDeviceToHost);
cudaMemcpy(h_nCount, d_nCount, sizeof(uint32_t)*(maxTime + 1), cudaMemcpyDeviceToHost);
cudaMemcpy(h_mCount, d_mCount, sizeof(uint32_t)*(maxTime + 1), cudaMemcpyDeviceToHost);
cudaMemcpy(h_nAgeOfInfection, d_nAgeOfInfection, sizeof(uint32_t) * C_NAOIRECORD *
↪  C_STRAINS, cudaMemcpyDeviceToHost);

// Open human and mosquito time series data files ready for recording, which saves all data
↪  which gets
// recorded at every time step, for example human and mosquito population sizes, or the
↪  total
// number of infected individuals.
std::ofstream nData(static_cast<std::string>(C_OUTPUTFOLDER) + "nData" +
↪  std::to_string(h_paraRun) + "_" + std::to_string(simRun) + ".csv");
std::cout << "\b\b\b\b\b\b\b\b\b\b\b\b\b\b\b\b\b\b\b\b\b\b\b\bWriting time series data to file . . ."
↪  << std::endl;

// Write the headers of each column in the data file.
uint32_t serotype{ 0 };
nData << "t,popSize";
while (serotype < C_STRAINS)
{
    nData << "," << "DENV" << serotype + 1;
    ++serotype;
}
serotype = 0;
while (serotype < C_STRAINS)
{
    nData << "," << "oneSubPop.DENV" << serotype + 1;
    ++serotype;
}
nData << "\n";

// For every time step, record the time series data.
```

```
for (uint32_t t = 0; t <= maxTime; ++t)
{
    nData << t << "," << h_nCount[t];
    serotype = 0;
    while (serotype < C_STRAINS)
    {
        nData << "," << h_nInfectedCount[t*C_STRAINS + serotype];
        ++serotype;
    }
    serotype = 0;
    while (serotype < C_STRAINS)
    {
        nData << "," << h_nOneSubPopInfectedCount[t*C_STRAINS + serotype];
        ++serotype;
    }
    nData << "\n";

    // Print progress of writing periodically. Need humanSize > 100 to prevent % error as
    // && reads from left to right.
    if ((maxTime >= 100) && (t % (maxTime / 100) == 0))
    {
        std::cout << "\b\b\b\b\b\b\b\b\b\b\b\b\b\b\b\b\b\b\b\b\b\b\b\b\b"
            << static_cast<int>(100 * (t / static_cast<float>(maxTime))) << "% completed . .
            ↪  .";
    }
}

// Close the data file for recording population size.
nData.close();

// Record the data for alive mosquitoes (used in computing mean age).
std::cout << "\b\b\b\b\b\b\b\b\b\b\b\b\b\b\b\b\b\b\b\b\b\b\bWriting human census data to file . .
↪  ." << std::endl;

// Copy from the graphics cards memory to the host.
cudaMemcpy(h_nAgeRecord, d_nAge, sizeof(age)*h_nSizeRecord, cudaMemcpyDeviceToHost);
cudaMemcpy(h_nHistoryRecord, d_nHistory, sizeof(history)*h_nSize*C_STRAINS,
↪  cudaMemcpyDeviceToHost);

// Open the file and record headers.
std::ofstream nAlive(static_cast<std::string>(C_OUTPUTFOLDER) + "nAlive" +
↪  std::to_string(h_paraRun) + "_" + std::to_string(simRun) + ".csv");
nAlive << "age";
for (uint32_t s = 0; s < C_STRAINS; ++s)
{
    nAlive << ",history" << s + 1;
}

// Record the data.
for (uint32_t i = 0; i < h_nSizeRecord; ++i)
{
    nAlive << "\n" << h_nAgeRecord[i];
    for (uint32_t s = 0; s < C_STRAINS; ++s)
    {
        nAlive << "," << static_cast<uint32_t>(h_nHistoryRecord[s*h_nSize + i]);
    }

    // Print progress of writing periodically. Need mosquito and human pop size > 100 to
    // prevent % error as && reads from left to right.
    if ((h_nSizeRecord > 100) && (i % ((h_nSizeRecord) / 100) == 0))
    {
        std::cout << "\r" << static_cast<int>(100 * (i / static_cast<float>(h_nSize))) << "%
        ↪  completed . . .";
    }
}
nAlive.close();

// Copy from the graphics cards memory to the host.
cudaMemcpy(h_mAgeRecord, d_mAge, sizeof(age)*h_mSize, cudaMemcpyDeviceToHost);
cudaMemcpy(h_mInfectStatusRecord, d_mInfectStatus, sizeof(infectStatus)*h_mSize,
↪  cudaMemcpyDeviceToHost);

// Open the file and record headers.
std::ofstream mAlive(static_cast<std::string>(C_OUTPUTFOLDER) + "mAlive" +
↪  std::to_string(h_paraRun) + "_" + std::to_string(simRun) + ".csv");
mAlive << "age,infectStatus";
```

```
    // Record the data.
    for (uint32_t i = 0; i < h_mSize; ++i)
    {
        mAlive << "\n" << h_mAgeRecord[i] << "," <<
        ↪  static_cast<uint32_t>(h_mInfectStatusRecord[i]);

        // Print progress of writing peeriodically. Need mosquito and human pop size > 100 to
        // prevent % error as && reads from left to right.
        if ((h_mSize > 100) && (i % ((h_mSize) / 100) == 0))
        {
            std::cout << "\r" << static_cast<int>(100 * (i / static_cast<float>(h_mSize))) << "%
            ↪  completed . . .";
        }
    }
    mAlive.close();

    // Open a file for recording the most recent ages of infections of each novel exposure.
    std::ofstream nAOI(static_cast<std::string>(C_OUTPUTFOLDER) + "nAOI" +
    ↪  std::to_string(h_paraRun) + "_" + std::to_string(simRun) + ".csv");

    // Record the headers of the file.
    nAOI << "first,second,third,fourth";

    // For each record, save the age of infection for each novel exposure.
    for (uint32_t record = 0; record < C_NAOIRECORD; ++record)
    {
        nAOI << "\n" << h_nAgeOfInfection[record];
        for (uint32_t s = 1; s < C_STRAINS; ++s)
        {
            nAOI << "," << h_nAgeOfInfection[s*C_NAOIRECORD + record];
        }
    }

    // Close the file for recording ages of infection.
    nAOI.close();

    // Copy the initial conditions from the host into device memory for the next simulation run.
    if (simRun != C_MAXSIMRUN)
    {
        cudaMemcpy(d_nAge, h_nAge, sizeof(age)*h_nSize, cudaMemcpyHostToDevice);
        cudaMemcpy(d_nExposed, h_nExposed, sizeof(exposed)*h_nSize, cudaMemcpyHostToDevice);
        cudaMemcpy(d_nHistory, h_nHistory, sizeof(history)*h_nSize*C_STRAINS,
        ↪  cudaMemcpyHostToDevice);
        cudaMemcpy(d_nInfectStatus, h_nInfectStatus, sizeof(infectStatus)*h_nSize,
        ↪  cudaMemcpyHostToDevice);
        cudaMemcpy(d_nPLE, h_nPLE, sizeof(pLifeExpectancy)*h_nSize, cudaMemcpyHostToDevice);
        cudaMemcpy(d_nRecovery, h_nRecovery, sizeof(recovery)*h_nSize, cudaMemcpyHostToDevice);
        cudaMemcpy(d_nStrain, h_nStrain, sizeof(strain)*h_nSize, cudaMemcpyHostToDevice);
        cudaMemcpy(d_mAge, h_mAge, sizeof(age)*h_mSize, cudaMemcpyHostToDevice);
        cudaMemcpy(d_mDead, h_mDead, sizeof(dead)*h_mSize, cudaMemcpyHostToDevice);
        cudaMemcpy(d_mExposed, h_mExposed, sizeof(exposed)*h_mSize, cudaMemcpyHostToDevice);
        cudaMemcpy(d_mInfectStatus, h_mInfectStatus, sizeof(infectStatus)*h_mSize,
        ↪  cudaMemcpyHostToDevice);
        cudaMemcpy(d_mPLE, h_mPLE, sizeof(pLifeExpectancy)*h_mSize, cudaMemcpyHostToDevice);
        cudaMemcpy(d_mStrain, h_mStrain, sizeof(strain)*h_mSize, cudaMemcpyHostToDevice);
        cudaMemcpy(d_nInfectedSubPopCount, h_nInfectedSubPopCount,
        ↪  sizeof(uint32_t)*h_subPopTotal*C_STRAINS, cudaMemcpyHostToDevice);
        cudaMemcpy(d_mInfectedSubPopCount, h_mInfectedSubPopCount,
        ↪  sizeof(uint32_t)*h_subPopTotal*C_STRAINS, cudaMemcpyHostToDevice);
        cudaMemcpy(d_nSubPopCount, h_nSubPopCount, sizeof(uint32_t)*h_subPopTotal,
        ↪  cudaMemcpyHostToDevice);
        cudaMemcpy(d_mSubPopCount, h_mSubPopCount, sizeof(uint32_t)*h_subPopTotal,
        ↪  cudaMemcpyHostToDevice);
        cudaMemcpy(d_mDeadCount, h_mDeadCount, sizeof(uint16_t)*mGridSize,
        ↪  cudaMemcpyHostToDevice);
    }
}

// Delete all dynamic array space declared on the host.
delete[] h_nAgeOfInfection;
delete[] h_mInfectStatusRecord;
delete[] h_mAgeRecord;
delete[] h_nHistoryRecord;
delete[] h_nAgeRecord;
delete[] h_mDeadCount;
```

```
    delete[] h_mSubPopCount;
    delete[] h_nSubPopCount;
    delete[] h_mInfectedSubPopCount;
    delete[] h_nInfectedSubPopCount;
    delete[] h_mStrain;
    delete[] h_mPLE;
    delete[] h_mInfectStatus;
    delete[] h_mExposed;
    delete[] h_mDead;
    delete[] h_mAge;
    delete[] h_nStrain;
    delete[] h_nRecovery;
    delete[] h_nPLE;
    delete[] h_nInfectStatus;
    delete[] h_nHistory;
    delete[] h_nExposed;
    delete[] h_nAge;
    delete[] h_mExpectedPopSize;
    delete[] h_mCount;
    delete[] h_nCount;
    delete[] h_nInfectedCount;
    delete[] h_nOneSubPopInfectedCount;
}
```

.cu source files

These source files are compiled with `nvcc` and define `__global__` or `__device__` functions that are executed on the device (i.e. GPU). Device functions are started by the host. A description of every source file is provided at the top of each.

datacollect.cu

```
// datacollect.cu: contains the host code which invokes CUDA kernels for summing up
// the number of infected/total individuals across all sub-populations, and kernel code
// for storing the result in time series data.

#include "device_launch_parameters.h"   // CUDA thread and block index.
#include "reduction.cuh"                // Reduction/summation CUDA kernel.
#include "constant.h"                   // Number of threads per block in summation.

// CUDA kernel for copying reduced infected data into the time series device memory.
__global__ void saveInfected(uint32_t* d_nInfectedCount,
                             const uint32_t* d_nReductionInfectedCount,
                             const uint32_t t)
{
    // Initialize the strain the thread is resposible for.
    uint32_t strain = threadIdx.x + blockIdx.x*blockDim.x;

    // Check that the strain is valid.
    if (strain < C_STRAINS)
    {
        // Copy the total number of infected individuals across to the time series data.
        d_nInfectedCount[t*C_STRAINS + strain] = d_nReductionInfectedCount[strain];
    }
}

// CUDA kernel for copying a specific sub-populations infected data into the
// time series device memory. This will be used in determining local serotype
// co-ciculation levels.
__global__ void saveSpecialInfected(uint32_t* d_nOneSubPopInfectedCount,
                                    const uint32_t* d_nInfectedSubPopCount,
                                    const uint32_t h_specialSubPop,
                                    const uint32_t h_subPopTotal,
                                    const uint32_t t)
{
    // Initialize the strain the thread is resposible for.
    uint32_t strain = threadIdx.x + blockIdx.x*blockDim.x;

    // Check that the strain is valid.
    if (strain < C_STRAINS)
    {
        // Copy the total number of infected individuals across to the time series data.
        d_nOneSubPopInfectedCount[t*C_STRAINS + strain] =
            d_nInfectedSubPopCount[strain*h_subPopTotal + h_specialSubPop];
    }
}

// CUDA kernel for copying reduced mosquito and human population size data into
// the time series device memory.
__global__ void saveCount(uint32_t* d_nCount,
                          uint32_t* d_mCount,
                          const uint32_t* d_nReductionCount,
                          const uint32_t* d_mReductionCount,
                          const uint32_t t)
{
    // Copy the total number of human and mosquito individuals across to the time series data.
    d_nCount[t] = d_nReductionCount[0];
    d_mCount[t] = d_mReductionCount[0];
}

// Host code which invokes CUDA kernels for summing up the number of infected/total
// individuals across all sub-populations, and kernel code for storing the result
// in time series data.
__host__ void dataCollect(uint32_t* d_nReductionInfectedCount,
```

```
                        uint32_t* d_nReductionCount,
                        uint32_t* d_mReductionCount,
                        uint32_t* d_nOneSubPopInfectedCount,
                        uint32_t* d_nInfectedCount,
                        uint32_t* d_nCount,
                        uint32_t* d_mCount,
                        const uint32_t* d_nInfectedSubPopCount,
                        const uint32_t* d_nSubPopCount,
                        const uint32_t* d_mSubPopCount,
                        const uint32_t h_specialSubPop,
                        const uint32_t h_subPopTotal,
                        const uint32_t t)
{
    // Compute the number of blocks required to sum the total number of individuals
    // across all sub-populations.
    uint32_t reductionSize{static_cast<uint32_t>(ceil(h_subPopTotal /
    ↪  static_cast<float>(C_THREADSPERBLOCKSUM)))};
    reductionSize = static_cast<uint32_t>(ceil(reductionSize / 2.0f));
    blockSingleSum<C_THREADSPERBLOCKSUM, uint32_t>
        <<< reductionSize, C_THREADSPERBLOCKSUM, sizeof(uint32_t)*C_THREADSPERBLOCKSUM >>>
        (d_nReductionCount, d_nSubPopCount, h_subPopTotal, reductionSize);
    blockSingleSum<C_THREADSPERBLOCKSUM, uint32_t>
        <<< reductionSize, C_THREADSPERBLOCKSUM, sizeof(uint32_t)*C_THREADSPERBLOCKSUM >>>
        (d_mReductionCount, d_mSubPopCount, h_subPopTotal, reductionSize);

    // The sum kernel can only sum at most 2*C_THREADSPERBLOCKSUM
    // old blocks together at a time. Keep calling the kernel, until all old blocks
    // (oldReductionSize) can be summed on one new block (reductionSize).
    while (reductionSize > 1)
    {
        uint32_t oldReductionSize = reductionSize;
        reductionSize = static_cast<uint32_t>(ceil(reductionSize /
        ↪  static_cast<float>(C_THREADSPERBLOCKSUM)));
        reductionSize = static_cast<uint32_t>(ceil(reductionSize / 2.0f));
        blockSingleSum<C_THREADSPERBLOCKSUM, uint32_t>
            <<< reductionSize, C_THREADSPERBLOCKSUM, sizeof(uint32_t)*C_THREADSPERBLOCKSUM >>>
            (d_nReductionCount, d_nReductionCount, oldReductionSize, reductionSize);
        blockSingleSum<C_THREADSPERBLOCKSUM, uint32_t>
            <<< reductionSize, C_THREADSPERBLOCKSUM, sizeof(uint32_t)*C_THREADSPERBLOCKSUM >>>
            (d_mReductionCount, d_mReductionCount, oldReductionSize, reductionSize);
    }

    // Compute the number of blocks required to sum the total number of infected individuals
    ↪  across all sub-populations.
    reductionSize = static_cast<uint32_t>(ceil(h_subPopTotal /
    ↪  static_cast<float>(C_THREADSPERBLOCKSUM)));
    reductionSize = static_cast<uint32_t>(ceil(reductionSize / 2.0f));
    uint32_t infectedSize{ reductionSize*C_STRAINS };
    blockSingleSum<C_THREADSPERBLOCKSUM, uint32_t>
        <<< infectedSize, C_THREADSPERBLOCKSUM, sizeof(uint32_t)*C_THREADSPERBLOCKSUM >>>
        (d_nReductionInfectedCount, d_nInfectedSubPopCount, h_subPopTotal, reductionSize);

    // The sum kernel can only sum at most 2*C_THREADSPERBLOCKSUM
    // old blocks together at a time. Keep calling the kernel, until all old blocks
    // (oldReductionSize) can be summed on one new block (reductionSize).
    while (reductionSize > 1)
    {
        uint32_t oldReductionSize = reductionSize;
        reductionSize = static_cast<uint32_t>(ceil(reductionSize /
        ↪  static_cast<float>(C_THREADSPERBLOCKSUM)));
        reductionSize = static_cast<uint32_t>(ceil(reductionSize / 2.0f));
        infectedSize = reductionSize*C_STRAINS;
        blockSingleSum<C_THREADSPERBLOCKSUM, uint32_t>
            <<< infectedSize, C_THREADSPERBLOCKSUM, sizeof(uint32_t)*C_THREADSPERBLOCKSUM >>>
            (d_nReductionInfectedCount, d_nReductionInfectedCount, oldReductionSize,
            ↪  reductionSize);
    }

    // Copy the reduced infected data into time series memory on the device.
    uint32_t infectedGridSize = static_cast<uint32_t>(ceil(C_STRAINS /
    ↪  static_cast<float>(C_THREADSPERBLOCKSUM)));
    saveInfected<<< infectedGridSize, C_THREADSPERBLOCKSUM >>>
        (d_nInfectedCount, d_nReductionInfectedCount, t);

    // Copy the single subpopulations infected data into time series memory on the device.
    uint32_t specialInfectedGridSize = static_cast<uint32_t>(ceil(C_STRAINS /
    ↪  static_cast<float>(C_THREADSPERBLOCKSUM)));
```

```
    saveSpecialInfected<<< specialInfectedGridSize, C_THREADSPERBLOCKSUM >>>
        (d_nOneSubPopInfectedCount, d_nInfectedSubPopCount, h_specialSubPop, h_subPopTotal, t);

    // Copy the reduced population counts for human and mosquitoes into time series device
    ↪  memory.
    saveCount<<< 1, 1 >>>(d_nCount, d_mCount, d_nReductionCount, d_mReductionCount, t);
}
```

demographicinitial.cu

```
// demographicintial.cu: host function and CUDA kernels for the intialization of the human
// and mosquito population (disease-free).

#include <cstdint>                      // Fixed width integers.
#include "curand_kernel.h"              // Device functions for CUDA random number generation.
#include "cudaidentities.h"             // Identities for determining the SM, warp, and lane id of a
↪  GPU thread.
#include "device_launch_parameters.h"   // CUDA block id, thread id.
#include "constant.h"                   // Constants for simulation.
#include "censustypedef.h"              // Type definitions for census data.
#include "parameterclass.h"             // Parameter space definition.
#include "architectureclass.h"          // GPU architectural properties definition.

// The cumulative life expectancy bi-Weibull distribution for humans. This distribution
// gives the most flexibility in age-specific mortality rates.
static __forceinline__ __device__ float nLifeExpectancy(age local_age, Parameter h_parameter)
{
    // Initialize as the argument of the burn-in exponential.
    float cumulativeLifeExpectancy = powf(static_cast<float>(local_age *
    ↪  h_parameter.nScaleInfantMortality), h_parameter.nShapeInfantMortality);

    // Need more terms if beyond the location where the second (decay out) Weibull distribution
    ↪  begins.
    if (local_age < h_parameter.nLocWeibull)
    {
        cumulativeLifeExpectancy = 1 - exp(-cumulativeLifeExpectancy);
    }
    else
    {
        cumulativeLifeExpectancy = cumulativeLifeExpectancy +
            powf(static_cast<float>(local_age - h_parameter.nLocWeibull) /
            ↪  h_parameter.nScaleLifeExpectancy, h_parameter.nShapeLifeExpectancy);
        cumulativeLifeExpectancy = 1 - exp(-cumulativeLifeExpectancy);
    }

    return cumulativeLifeExpectancy;
}

// The cumulative life expectancy Weibull distribution for mosquitos. This distribution
// gives the most flexibility in age-specific mortality rates.
static __forceinline__ __device__ float mLifeExpectancy(age local_age, Parameter h_parameter)
{
    // The cumulative weibull distribution is given by 1 - exp(-(x / scale)^shape)
    float cumulativeLifeExpectancy{ 1 - exp(-powf(local_age / h_parameter.mScaleLifeExpectancy,
        h_parameter.mShapeLifeExpectancy)) };

    return cumulativeLifeExpectancy;
}

// The CUDA kernel for the human cumulative survival function. This gives an
// estimate for the expected value of the human life expectancy distribution.
// Upper Rieman Sum of the survival function, overestimates expected value.
__global__ void nCumulativeSurvival(float* d_nSurvival, Parameter h_parameter)
{
    // Define the first probability of surviving (at age 0 days).
    d_nSurvival[0] = static_cast<float>(1 - nLifeExpectancy(0, h_parameter));

    // Initialize the age counter (in years)
    uint16_t ageCount = 1;

    // For every age, calculate the probability of surviving up until that age,
    // and sum with all probabilities of surviving up until all previous ages.
    while (ageCount <= C_NMAXINITIALAGE)
    {
        d_nSurvival[ageCount] = d_nSurvival[ageCount - 1] + (1 - nLifeExpectancy(ageCount*C_YEAR,
        ↪  h_parameter));
        ++ageCount;
    }
}

// The CUDA kernel for the human cumulative survival function. This gives an
// estimate for the expected value of the mosquito life expectancy distribution.
// Upper Rieman Sum of the survival function, overestimates expected value.
__global__ void mCumulativeSurvival(float* d_mSurvival, Parameter h_parameter)
```

```
{
    // Define the first probability of surviving (at age 0 days).
    d_mSurvival[0] = static_cast<float>(1 - mLifeExpectancy(0, h_parameter));

    // Initialize the age counter (in days)
    uint16_t ageCount = 1;

    // For every age, calculate the probability of surviving up until that age,
    // and sum with all probabilities of surviving up until all previous ages.
    while (ageCount <= C_MMAXINITIALAGE)
    {
        d_mSurvival[ageCount] = d_mSurvival[ageCount - 1] + (1 - mLifeExpectancy(ageCount,
        ↪  h_parameter));
        ++ageCount;
    }
}

// The CUDA kernel for the initialization of the human population.
__global__ void nDemographicInitialization(age* d_nAge,
                                           history* d_nHistory,
                                           infectStatus* d_nInfectStatus,
                                           pLifeExpectancy* d_nPLE,
                                           curandState_t* d_randStates,
                                           const float* d_nSurvival,
                                           const uint32_t* d_nSize,
                                           const Parameter h_parameter,
                                           const Architecture h_architecture)
{
    // Initalize the human index which determines the human in the census data
    // that is assigned to the GPU thread.
    uint16_t threadid = threadIdx.x;
    uint32_t blockid = blockIdx.x;
    uint32_t nIndex = blockid*blockDim.x + threadid;

    // Determine which active thread the individual is being run on. This is used
    // for random number generation.
    uint32_t activeThreadId = smId()*h_architecture.warpsPerSM*h_architecture.threadsPerWarp
        + warpId()*h_architecture.threadsPerWarp + laneId();

    // Create a human if the human index is within the requested size of the human
    // population.
    uint32_t local_nSize = __ldg(&d_nSize[0]);
    if (nIndex < local_nSize)
    {
        // Load the random number generator state from global memory to local memory.
        curandState_t localState = d_randStates[activeThreadId];

        // Declare the age variable for the human.
        age local_age;

        // Determine the age of an individual over the bi-Weibull distribution's cumulative survival
        ↪  function.
        // Choose a random value between zero and the final cumulative survival function.
        float randCSF = d_nSurvival[C_NMAXINITIALAGE] * curand_uniform(&localState);

        // Starting at age zero, scan through the cumulative survival function and determine the age
        ↪  that
        // the individual has survived until. Once an age has been assigned, exit the loop.
        // In other words, determine what age in the cumulative survival function gives randCSF.
        uint16_t ageCount = 0;
        while (ageCount <= C_NMAXINITIALAGE)
        {
            if (randCSF <= d_nSurvival[ageCount])
            {
                // Randomly assign a day of their birth within the year they were born.
                local_age = static_cast<uint16_t>(C_YEAR*ageCount + (curand(&localState) % C_YEAR));
                break;
            }
            else
            {
                ++ageCount;
            }
        }

        // Assign a life expectancy probability to an individual such that it is greater than the
        ↪  life
```

```
        // expectancy probability of their current age (scale, and move up uniform disitribution).
        ↪  However,
        // to avoid the small blip in the first run through of the simulation, where no-one dies,
        ↪  step age
        // back by one when calculating the life probability (unless already zero).
        if (local_age != 0)
        {
            d_nPLE[nIndex] = (1 - nLifeExpectancy(local_age - 1, h_parameter)) *
            ↪  curand_uniform(&localState) +
                nLifeExpectancy(local_age - 1, h_parameter);
        }
        else
        {
            d_nPLE[nIndex] = (1 - nLifeExpectancy(local_age, h_parameter)) *
            ↪  curand_uniform(&localState) +
                nLifeExpectancy(local_age, h_parameter);
        }

        // Initially assume that all individuals are not infected.
        d_nInfectStatus[nIndex] = 0;

        // Set some kind of immunity profile in the human population.
        uint8_t strain{ 0 };
        while (strain < C_STRAINS)
        {
            d_nHistory[strain*local_nSize + nIndex] = 0;
            ++strain;
        }

        // Record the random number generator state back to global memory.
        d_randStates[activeThreadId] = localState;

        // Record the human from local memory to global memory.
        d_nAge[nIndex] = local_age;
    }
}

// The CUDA kernel for the initialization of the mosquito population.
__global__ void mDemographicInitialization(age* d_mAge,
                                           dead* d_mDead,
                                           infectStatus* d_mInfectStatus,
                                           pLifeExpectancy* d_mPLE,
    subPopulation* d_mSubPopulation,
                                           curandState_t* d_randStates,
                                           uint16_t* d_mDeadCount,
    uint32_t* d_mSubPopCount,
                                           const float* d_mSurvival,
                                           const uint32_t* d_mSize,
                                           const Parameter h_parameter,
                                           const Architecture h_architecture)
{
    // Initalize the mosquito index which determines the human in the census data
    // that is assigned to the GPU thread.
    uint16_t threadIndex = threadIdx.x;
    uint32_t blockIndex = blockIdx.x;
    uint32_t mIndex = blockIndex*blockDim.x + threadIndex;

    // Determine which active thread the individual is being run on. This is used
    // for random number generation.
    uint32_t activeThreadId = smId()*h_architecture.warpsPerSM*h_architecture.threadsPerWarp
        + warpId()*h_architecture.threadsPerWarp + laneId();

    // Declare the shared variables.
    __shared__ uint32_t shared_deadCount;         // The number of dead individuals on the GPU
    ↪  block.

    // Initialize the counter for the total number of dead individuals on a block.
    if (threadIndex == 0)
    {
        shared_deadCount = 0;
    }

    // Synchronize threads across a block.
    __syncthreads();

    // Create a human if the human index is within the requested size of the human
```

```
// population.
if (mIndex < d_mSize[0])
{
    // Load the random number generator state from global memory to local memory.
    curandState_t localState = d_randStates[activeThreadId];

    // Determine if the individual is alive or not.
    if (curand_uniform(&localState) >= (h_parameter.minMosToHuman / h_parameter.maxMosToHuman))
    {
        d_mDead[mIndex] = 1;
        atomicAdd(&shared_deadCount, 1);
        atomicSub(&d_mSubPopCount[d_mSubPopulation[mIndex]], 1);
    }
    else
    {
        // Determine the age of an individual over the bi-Weibull distribution's cumulative
        ↪  survival function.
        // Choose a random value between zero and the final cumulative survival function.
        float randCSF = d_mSurvival[C_MMAXINITIALAGE] * curand_uniform(&localState);

        // Starting at age zero, scan through the cumulative survival function and determine the
        ↪  age that
        // the individual has survived until. Once an age has been assigned, exit the loop.
        // In other words, determine what age in the cumulative survival function gives randCSF.
        age local_age;
        uint16_t ageCount = 0;
        bool ageAssigned = false;
        while ((ageCount <= C_MMAXINITIALAGE) && (!ageAssigned))
        {
            if (randCSF <= d_mSurvival[ageCount])
            {
                // Assign the age which the mosquito has thus far survived until.
                local_age = ageCount;
                ageAssigned = true;
            }
            else
            {
                ++ageCount;
            }
        }

        // Individuals are alive when created.
        d_mDead[mIndex] = 0;

        // Assign a life expectancy probability to an individual such that it is greater than
        ↪  the life
        // expectancy probability of their current age (scale, and move up uniform
        ↪  disitribution). However,
        // to avoid the small blip in the first run through of the simulation, where no-one
        ↪  dies, step age
        // back by one when calculating the life probability (unless already zero).
        if (local_age != 0)
        {
            d_mPLE[mIndex] = (1 - mLifeExpectancy(local_age - 1, h_parameter)) *
            ↪  curand_uniform(&localState) +
                mLifeExpectancy(local_age - 1, h_parameter);
        }
        else
        {
            d_mPLE[mIndex] = (1 - mLifeExpectancy(local_age, h_parameter)) *
            ↪  curand_uniform(&localState) +
                mLifeExpectancy(local_age, h_parameter);
        }

        // Initially assume that all individuals are not infected. Not necessary to set a
        ↪  strain.
        d_mInfectStatus[mIndex] = 0;

        // Record the mosquito from local memory to global memory.
        d_mAge[mIndex] = local_age;
    }

    // Record the random number generator state back to global memory.
    d_randStates[activeThreadId] = localState;
}
```

```
    // Ensure that all necessary deaths have occured.
    __syncthreads();

    // Save the number of dead mosquitoes on the block. Required for
    // mosquito demographic simulation.
    if (threadIndex == 0)
    {
        d_mDeadCount[blockIdx.x] = shared_deadCount;
    }
}

// CUDA kernel which computes how many individuals are in each subpopulation
// (reserved individuals and initialized individuals).
__global__ void subPopCount(uint32_t* d_nSubPopCount,
                            uint32_t* d_mSubPopCount,
                            uint32_t* d_nSubPopSize,
                            uint32_t* d_mSubPopSize,
                            const uint32_t* d_subPopTotal)
{
    // Initialize the subpopulation index.
    uint32_t subPop = blockIdx.x*blockDim.x + threadIdx.x;

    // If sub-population index is well defined, count individuals over uniform distribution.
    if (subPop < d_subPopTotal[0])
    {
        d_nSubPopCount[subPop] = 0;
        d_mSubPopCount[subPop] = 0;
        d_nSubPopSize[subPop] = 0;
        d_mSubPopSize[subPop] = 0;
    }
}

// CUDA kernel which initializes the subpopulation all individuals belong to.
__global__ void subPopInitial(subPopulation* d_subPopulation,
                              curandState_t* d_randStates,
                              uint32_t* d_subPopCount,
                              uint32_t* d_subPopSize,
                              const uint32_t* d_size,
                              const uint32_t* d_subPopTotal,
                              const Architecture h_architecture)
{
    // Determine the individuals census index.
    uint32_t censusIndex = blockIdx.x*blockDim.x + threadIdx.x;

    // Get the active thread identity of the current thread.
    uint32_t activeThreadId = smId()*h_architecture.warpsPerSM*h_architecture.threadsPerWarp
        + warpId()*h_architecture.threadsPerWarp + laneId();

    // For all individuals defined in the census, randomly assign a subpopulation and
    // increment the subpopulation counter (variable) and size (fixed).
    if (censusIndex < __ldg(&d_size[0]))
    {
        // Record the subpopulation to global memory.
        subPopulation local_subPopulation = curand(&d_randStates[activeThreadId]) %
        ↪  d_subPopTotal[0];
        atomicAdd(&d_subPopSize[local_subPopulation], 1);
        atomicAdd(&d_subPopCount[local_subPopulation], 1);
        d_subPopulation[censusIndex] = local_subPopulation;
    }
}

// "Inefficient" CUDA kernel which cumulatively sums the number of individuals assigned
// to each subpopulation, to give the locations of where census indices of the individuals
// of that subpopulation begin in the subpopulation array. Note that
// the first element of the index array is zero.
__global__ void subPopCumulativeSum(uint32_t* d_nSubPopLoc,
                                    uint32_t* d_mSubPopLoc,
                                    const uint32_t* d_nSubPopSize,
                                    const uint32_t* d_mSubPopSize,
                                    const uint32_t* d_subPopTotal)
{
    // Declare sub-population index.
    uint32_t subPop;

    // Initialize the position of subpopulation zero.
    d_nSubPopLoc[0] = 0;
```

```
    d_mSubPopLoc[0] = 0;

    // Compute the position for accessing each subpopulations members.
    for (subPop = 1; subPop < d_subPopTotal[0]; ++subPop)
    {
        d_nSubPopLoc[subPop] = d_nSubPopLoc[subPop - 1] + d_nSubPopSize[subPop - 1];
        d_mSubPopLoc[subPop] = d_mSubPopLoc[subPop - 1] + d_mSubPopSize[subPop - 1];
    }
}

// CUDA kernel which takes the census index of an individual and records it in the appropriate
// subpopulation position in the array which contains census indices for individuals for
// every subpopulation combination.
__global__ void subPopIndexing(uint32_t* d_subPopIndex,
                               uint32_t* d_subPopLoc,
                               const subPopulation* d_subPopulation,
                               const uint32_t* d_reserveSize)
{
    // Determine the individuals census index.
    uint32_t censusIndex = blockIdx.x*blockDim.x + threadIdx.x;

    // If the census index is valid, find the location of the sub-population ordered
    // census index array for the individuals sub-population.
    if (censusIndex < d_reserveSize[0])
    {
        uint32_t locationIndex = atomicAdd(&d_subPopLoc[d_subPopulation[censusIndex]], 1);
        d_subPopIndex[locationIndex] = censusIndex;
    }
}

// Host function which calls the CUDA kernels for the cumulative survival functions, and initializes
// the human and mosquito populations, and creates the sub-population ordered array of population
// indices used during infection.
__host__ void demographicInitialization(age* d_nAge,
                                        history* d_nHistory,
                                        infectStatus* d_nInfectStatus,
                                        pLifeExpectancy* d_nPLE,
                                        subPopulation* d_nSubPopulation,
                                        age* d_mAge,
                                        dead* d_mDead,
                                        infectStatus* d_mInfectStatus,
                                        pLifeExpectancy* d_mPLE,
                                        subPopulation* d_mSubPopulation,
                                        uint32_t* d_nSubPopIndex,
                                        uint32_t* d_mSubPopIndex,
                                        curandState_t* d_randStates,
                                        uint32_t* d_nSubPopCount,
                                        uint32_t* d_mSubPopCount,
                                        uint32_t* d_nSubPopLoc,
                                        uint32_t* d_mSubPopLoc,
                                        uint32_t* d_nSubPopSize,
                                        uint32_t* d_mSubPopSize,
                                        uint16_t* d_mDeadCount,
                                        float* d_nSurvival,
                                        float* d_mSurvival,
                                        const uint32_t* d_nSize,
                                        const uint32_t* d_mSize,
                                        const uint32_t* d_subPopTotal,
                                        const uint32_t h_nSize,
                                        const uint32_t h_mSize,
                                        const uint32_t h_subPopTotal,
                                        const Parameter h_parameter,
                                        const Architecture h_architecture)
{
    // Call the CUDA kernel which initializes the subpopulation counters for the human and
    // mosquito populations.
    uint32_t subPopGridSize = static_cast<uint32_t>(ceil(h_subPopTotal /
    ↪  static_cast<float>(C_THREADSPERBLOCK)));
    subPopCount<<< subPopGridSize, C_THREADSPERBLOCK >>>
        (d_nSubPopCount, d_mSubPopCount, d_nSubPopSize, d_mSubPopSize, d_subPopTotal);

    // Initialize the number of blocks (gridSize) required on the GPU given the number of threads
    ↪  desired to be used on
    // each block. Need as many threads as individuals.
    uint32_t nGridSize = static_cast<uint32_t>(ceil(h_nSize /
    ↪  static_cast<float>(C_THREADSPERBLOCK)));
```

```
    uint32_t mGridSize = static_cast<uint32_t>(ceil(h_mSize /
    ↪  static_cast<float>(C_THREADSPERBLOCK)));

    // Call the CUDA kernels for initialization of human and mosquito subpopulation assignment, and
    ↪  increment
    // the counters for the number of individuals of each subpopulation.
    subPopInitial<<< nGridSize, C_THREADSPERBLOCK >>>
        (d_nSubPopulation, d_randStates, d_nSubPopCount, d_nSubPopSize, d_nSize, d_subPopTotal,
        ↪  h_architecture);
    subPopInitial<<< mGridSize, C_THREADSPERBLOCK >>>
        (d_mSubPopulation, d_randStates, d_mSubPopCount, d_mSubPopSize, d_mSize, d_subPopTotal,
        ↪  h_architecture);

    // Call the CUDA kernel which contains the indices of where each subpopulation begins
    // in the sub-population ordered census index array.
    subPopCumulativeSum<<< 1, 1 >>>(d_nSubPopLoc, d_mSubPopLoc, d_nSubPopSize, d_mSubPopSize,
    ↪  d_subPopTotal);

    // Use the array of subpopulation locations to record the census index of individuals
    // into the sub-population ordered census index arary.
    subPopIndexing<<< nGridSize, C_THREADSPERBLOCK >>>
        (d_nSubPopIndex, d_nSubPopLoc, d_nSubPopulation, d_nSize);
    subPopIndexing<<< mGridSize, C_THREADSPERBLOCK >>>
        (d_mSubPopIndex, d_mSubPopLoc, d_mSubPopulation, d_mSize);

    // Call the CUDA kernel again which contains the indices of where each subpopulation begins
    // in the sub-population ordered census index array (overrode in the previous kernel calls).
    subPopCumulativeSum<<< 1, 1 >>>(d_nSubPopLoc, d_mSubPopLoc, d_nSubPopSize, d_mSubPopSize,
    ↪  d_subPopTotal);

    // Initialize the cumulative survival functions for human and mosquito life expectancy.
    nCumulativeSurvival<<< 1, 1 >>>(d_nSurvival, h_parameter);
    mCumulativeSurvival<<< 1, 1 >>>(d_mSurvival, h_parameter);

    // Call the CUDA kernels for initialization of the human and mosquito populations.
    nDemographicInitialization<<< nGridSize, C_THREADSPERBLOCK >>>
        (d_nAge, d_nHistory, d_nInfectStatus, d_nPLE, d_randStates, d_nSurvival, d_nSize,
        ↪  h_parameter, h_architecture);
    mDemographicInitialization<<< mGridSize, C_THREADSPERBLOCK >>>
        (d_mAge, d_mDead, d_mInfectStatus, d_mPLE, d_mSubPopulation, d_randStates, d_mDeadCount,
        ↪  d_mSubPopCount, d_mSurvival, d_mSize, h_parameter, h_architecture);
}
```

epidemic.cu

```
// epidemic.cu: contains the host function which calls the CUDA disease transmission kernels.

#include "censustypedef.h"              // Type definitions for census data.
#include "curand_kernel.h"              // Device functions for CUDA random number generation.
#include "constant.h"                   // Constants for the simulation.
#include "nepidemic.h"                  // To human transmission.
#include "mepidemic.h"                  // To mosquito transmission.
#include "device_launch_parameters.h"   // Thread ID and block ID.
#include "parameter.h"                  // Parameter space definitions.
#include "architectureclass.h"          // GPU architectural properties definition.

// CUDA kernel which intializes the number of transmission events between species
// for every subpopulation and strain.
__global__ void initialTransmission(uint32_t* d_nTransmission,
                                    uint32_t* d_mTransmission,
                                    const uint32_t* d_subPopTotal)
{
    // Initialize the strain, subpopulation combination of the thread.
    uint32_t strainSubPop = blockIdx.x*blockDim.x + threadIdx.x;

    // Load in the total number of subpopulation from global memory to L1 cache.
    uint32_t subPopTotal = __ldg(&d_subPopTotal[0]);

    // Initialize the number of transmission events that occur to individuals of every subpopulation
    // and strain combination.
    if (strainSubPop < subPopTotal*C_STRAINS)
    {
        d_nTransmission[strainSubPop] = 0;
        d_mTransmission[strainSubPop] = 0;
    }
}

// Host function which calls the CUDA kernels for simulating disease transmission.
__host__ void epidemic(age* d_nAge,
                       exposed* d_nExposed,
                       history* d_nHistory,
                       infectStatus* d_nInfectStatus,
                       recovery* d_nRecovery,
                       strain* d_nStrain,
                       age* d_mAge,
                       dead* d_mDead,
                       exposed* d_mExposed,
                       infectStatus* d_mInfectStatus,
                       strain* d_mStrain,
                       curandState_t* d_randStates,
                       uint32_t* d_nTransmission,
                       uint32_t* d_mTransmission,
                       uint32_t* d_nAgeOfInfection,
                       uint32_t* d_nAgeOfInfectionCount,
                       const uint32_t* d_nInfectedSubPopCount,
                       const uint32_t* d_mInfectedSubPopCount,
                       const uint32_t* d_nSubPopCount,
                       const uint32_t* d_mSubPopCount,
                       const uint32_t* d_nInfectedCount,
                       const uint32_t* d_nSubPopIndex,
                       const uint32_t* d_nSubPopLoc,
                       const uint32_t* d_nSubPopSize,
                       const uint32_t* d_mSubPopIndex,
                       const uint32_t* d_mSubPopLoc,
                       const uint32_t* d_mSubPopSize,
                       const uint32_t* d_nSize,
                       const uint32_t* d_mSize,
                       const uint32_t* d_metaPopCols,
                       const uint32_t* d_metaPopRows,
                       const uint32_t* d_subPopTotal,
                       const uint32_t h_subPopTotal,
                       const uint32_t t,
                       const uint32_t maxTime,
                       const Parameter h_parameter,
                       const Architecture h_architecture)
{
    // Call the CUDA kernel which initializes transmission numbers.
    uint32_t infectedGridSize = static_cast<uint32_t>(ceil(h_subPopTotal*C_STRAINS /
    ↪  static_cast<float>(C_THREADSPERBLOCK)));
```

```
    initialTransmission<<< infectedGridSize, C_THREADSPERBLOCK >>>
        (d_nTransmission, d_mTransmission, d_subPopTotal);

    // Call the CUDA kernel which determines transmission numbers from each sub-population to
    ↪  another subpopulation.
    nVisitingInfected<<< infectedGridSize, C_THREADSPERBLOCK >>>
    (d_randStates, d_nTransmission, d_nInfectedSubPopCount, d_nSubPopCount, d_mSubPopCount,
    ↪  d_metaPopCols, d_metaPopRows, d_subPopTotal, h_parameter, h_architecture);
    mVisitingInfected<<< infectedGridSize, C_THREADSPERBLOCK >>>
    (d_randStates, d_mTransmission, d_mInfectedSubPopCount, d_metaPopCols, d_metaPopRows,
    ↪  d_subPopTotal, h_parameter, h_architecture);

    // Call the CUDA kernel which infects human and mosquitoes based on the transmission numbers
    ↪  computed
    // in the previous kernel invokation.
    uint32_t subPopGridSize = static_cast<uint32_t>(ceil(h_subPopTotal /
    ↪  static_cast<float>(C_THREADSPERBLOCK)));
    nmTransmission<<< subPopGridSize, C_THREADSPERBLOCK >>>
        (d_mAge, d_mDead, d_mExposed, d_mInfectStatus, d_mStrain, d_randStates,
            d_mSubPopIndex, d_mSubPopLoc, d_mSubPopSize, d_nTransmission, d_mSubPopCount, d_mSize,
            ↪  d_subPopTotal, t, h_parameter, h_architecture);
    mnTransmission<<< subPopGridSize, C_THREADSPERBLOCK >>>
        (d_nAge, d_nExposed, d_nHistory, d_nInfectStatus, d_nRecovery, d_nStrain, d_randStates,
        ↪  d_nAgeOfInfection, d_nAgeOfInfectionCount,
            d_nSubPopIndex, d_nSubPopLoc, d_nSubPopSize, d_mTransmission, d_nSize, d_subPopTotal,
            ↪  maxTime - t, h_parameter, h_architecture);
}
```

epidemicinitial.cu

```
// epidemicinitial.cu: this CUDA file contains the host functions and device kernels
// for the initialization of infection into the human and mosquito populations.

#include "constant.h"                   // Constants for the simulation.
#include "censustypedef.h"              // Type definitions for census data.
#include "curand_kernel.h"              // CUDA random number generation.
#include "cudaidentities.h"             // Warp ID, streaming multiprocessor ID and laneID.
#include "device_launch_parameters.h"   // Block ID and thread ID.
#include "parameterclass.h"             // Parameter space definition.
#include "architectureclass.h"          // GPU architectural properties definition.

// Device function which determines if a float should be rounded up or down, dependent
// on uniform random number generation.
__device__ __forceinline__ uint32_t decimalResolve(curandState_t* state, float floating)
{
    float frac = floating - floorf(floating);
    if (curand_uniform(state) <= frac)
    {
        return static_cast<uint32_t>(ceilf(floating));
    }
    else
    {
        return static_cast<uint32_t>(floorf(floating));
    }
}

// CUDA kernel initializing the infection in the human population.
__global__ void nEpidemicInitial(age* d_nAge,
                                 history* d_nHistory,
                                 infectStatus* d_nInfectStatus,
                                 recovery* d_nRecovery,
                                 strain* d_nStrain,
                                 subPopulation* d_nSubPopulation,
                                 curandState_t* d_randStates,
                                 uint32_t* d_nInfectedCount,
                                 const uint32_t* d_nSize,
                                 const uint32_t* d_subPopTotal,
                                 const uint32_t nInitialInfected,
                                 const uint32_t nGridSize,
                                 const Parameter h_parameter,
                                 const Architecture h_architecture)
{
    // Initialize the human index for census data access.
    uint32_t nIndex = blockIdx.x*blockDim.x + threadIdx.x;

    // Determine which active thread is being run (for random number generation).
    uint32_t activeThreadId = smId()*h_architecture.warpsPerSM*h_architecture.threadsPerWarp
        + warpId()*h_architecture.threadsPerWarp + laneId();

    // Infect the first nInitialInfected humans in the census.
    if (nIndex < nInitialInfected)
    {
        // Load the random number generator state from global memory.
        curandState_t local_state = d_randStates[activeThreadId];

        // Randomly assign the individual on the census a strain of the virus. Alter the individuals
        // immunological history, and the age at which they recover from the infection.
        d_nInfectStatus[nIndex] = 2;
        strain local_strain = static_cast<strain>(curand(&local_state) % C_STRAINS);
        d_nStrain[nIndex] = local_strain;
        d_nHistory[local_strain*d_nSize[0] + nIndex] = 1;

        // Age at which individual stops being infected.
        d_nRecovery[nIndex] = static_cast<recovery>(decimalResolve(&local_state,
        ↪  static_cast<float>(d_nAge[nIndex] + h_parameter.recovery)));

        // Increment the number of infected individuals of that strain in the individuals
        ↪  sub-population.
        atomicAdd(&d_nInfectedCount[local_strain* d_subPopTotal[0] + d_nSubPopulation[nIndex]], 1);
    }
}

// CUDA kernel initializing infection in the mosquito population.
__global__ void mEpidemicInitial(infectStatus* d_mInfectStatus,
```

```
                                strain* d_mStrain,
                                subPopulation* d_mSubPopulation,
                                curandState_t* d_randStates,
                                uint32_t* d_mInfectedCount,
                                const uint32_t* d_subPopTotal,
                                const uint32_t mInitialInfected,
                                const uint32_t mGridSize,
                                const Architecture h_architecture)
{
    // Initialize the mosquito index for census data access.
    uint32_t mIndex = blockIdx.x*blockDim.x + threadIdx.x;

    // Determine which active thread is being run (for random number generation).
    uint32_t activeThreadId = smId()*h_architecture.warpsPerSM*h_architecture.threadsPerWarp
        + warpId()*h_architecture.threadsPerWarp + laneId();

    // Infect the first mInitialInfected mosquitoes in the census.
    if (mIndex < mInitialInfected)
    {
        // Randomly assign the individual on the census a strain of the virus.
        d_mInfectStatus[mIndex] = 2;
        strain local_strain = static_cast<strain>(curand(&d_randStates[activeThreadId]) %
        ↪  C_STRAINS);
        d_mStrain[mIndex] = local_strain;

        // Increment the number of infected individuals of that strain in the individuals
        ↪  sub-population.
        atomicAdd(&d_mInfectedCount[local_strain*d_subPopTotal[0] +
        ↪  __ldg(&d_mSubPopulation[mIndex])], 1);
    }
}

// CUDA Kernel which intiializes the infected counters for each subpopulation for each species.
__global__ void infectCountInitial(uint32_t* d_nInfectedSubPopCount,
                                   uint32_t* d_mInfectedSubPopCount,
                                   const uint32_t* d_subPopTotal)
{
    // Determine the strain, subpopulation combination the thread is responsible for.
    uint32_t strainSubPop = threadIdx.x + blockIdx.x*blockDim.x;

    // Ensure the combination is well defined.
    if (strainSubPop < d_subPopTotal[0]*C_STRAINS)
    {
        d_nInfectedSubPopCount[strainSubPop] = 0;
        d_mInfectedSubPopCount[strainSubPop] = 0;
    }
}

// Host function which calls CUDA kernels to initial the epidemics of the
// disease in the human and mosquito populations and initialize the counter for
// all strain, subpopulation combinations.
__host__ void epidemicInitial(age* d_nAge,
                              history* d_nHistory,
                              infectStatus* d_nInfectStatus,
                              recovery* d_nRecovery,
                              strain* d_nStrain,
                              subPopulation* d_nSubPopulation,
                              infectStatus* d_mInfectStatus,
                              strain* d_mStrain,
                              subPopulation* d_mSubPopulation,
                              curandState_t* d_randStates,
                              uint32_t* d_nInfectedSubPopCount,
                              uint32_t* d_mInfectedSubPopCount,
                              const uint32_t* d_nSize,
                              const uint32_t* d_subPopTotal,
                              const uint32_t nInitialInfected,
                              const uint32_t mInitialInfected,
                              const uint32_t nGridSize,
                              const uint32_t mGridSize,
                              const uint32_t h_subPopTotal,
                              const Parameter h_parameter,
                              const Architecture h_architecture)
{
    // Initialize the infected counters.
    uint32_t infectSubPopGridSize{ static_cast<uint32_t>(ceil(C_STRAINS*h_subPopTotal /
        static_cast<float>(C_THREADSPERBLOCK))) };
```

```
    infectCountInitial<<< infectSubPopGridSize, C_THREADSPERBLOCK >>>
        (d_nInfectedSubPopCount, d_mInfectedSubPopCount, d_subPopTotal);

    // Determine the grid size to use on the GPU for epidemic initialization with a pre-determined
    ↪  number of threads
    // per block on the grid (must be a whole number).
    uint32_t nInfectGridSize{ static_cast<uint32_t>(ceil(nInitialInfected /
                                                         static_cast<float>(C_THREADSPERBLOCK))) };
    uint32_t mInfectGridSize{ static_cast<uint32_t>(ceil(mInitialInfected /
                                                         static_cast<float>(C_THREADSPERBLOCK))) };

    // Infect the first nInitialInfected/mInitialInfected humans/mosquitoes in their respective
    ↪  census data.
    // Keep track of the number of infected individuals for each subpopulation and virus strain
    ↪  combination.
    nEpidemicInitial<<< nInfectGridSize, C_THREADSPERBLOCK >>>
        (d_nAge, d_nHistory, d_nInfectStatus, d_nRecovery, d_nStrain, d_nSubPopulation,
        d_randStates, d_nInfectedSubPopCount, d_nSize, d_subPopTotal, nInitialInfected, nGridSize,
        ↪  h_parameter, h_architecture);
    mEpidemicInitial<<< mInfectGridSize, C_THREADSPERBLOCK >>>
        (d_mInfectStatus, d_mStrain, d_mSubPopulation,
        d_randStates, d_mInfectedSubPopCount, d_subPopTotal, mInitialInfected, mGridSize,
        ↪  h_architecture);
}
```

mdemographic.cu

```
// mdemographics.cu: This file contains the host code which invokes CUDA kernels
// for the simulation of the mosquito demographics.

#include "censustypedef.h"              // Type definitions for census data.
#include "constant.h"                   // Constants for simulation.
#include "curand_kernel.h"              // Required for random number generation in CUDA.
#include "cudaidentities.h"             // Streaming multiprocessor ID, warp ID, and lane ID.
#include "device_launch_parameters.h"   // Thread ID and block ID.
#include "parameterclass.h"             // Parameter space definition.
#include "architectureclass.h"          // GPU architectural properties definition.

// The cumulative life expectancy Weibull distribution for mosquitos. This distribution
// gives the most flexibility in age-specific mortality rates.
__device__ __forceinline__ float device_mLifeExpectancy(const age local_age, const Parameter
↪  h_parameter)
{
    // The cumulative weibull distribution is given by 1 - exp(-(x / scale)^shape)
    return (1.0f - expf(-(powf(local_age / h_parameter.mScaleLifeExpectancy,
    ↪  h_parameter.mShapeLifeExpectancy))));
}

// Determines if an individual dies at the current time step and if a deceased individual
// needs to be replaced by a new individual. Keeps track of the number of dead individuals
// on the GPU block.
__global__ void mBlockBirthDeath(age* d_mAge,
                                 dead* d_mDead,
                                 exposed* d_mExposed,
                                 infectStatus* d_mInfectStatus,
                                 pLifeExpectancy* d_mPLE,
                                 strain* d_mStrain,
                                 subPopulation* d_mSubPopulation,
                                 curandState_t* d_randStates,
                                 uint32_t* d_infectedCount,
                                 uint32_t* d_subPopCount,
                                 uint16_t* d_deadCount,
                                 const uint32_t* d_size,
                                 const uint32_t* d_subPopTotal,
                                 const float* d_mExpectedPopSize,
                                 const Parameter h_parameter,
                                 const Architecture h_architecture)
{
    // Initialize the mosquito index for the thread.
    uint32_t mIndex = threadIdx.x + blockDim.x * blockIdx.x;

    // Initialize the active thread index for random number generation.
    uint32_t activeThreadId = smId()*h_architecture.warpsPerSM*h_architecture.threadsPerWarp
        + warpId()*h_architecture.threadsPerWarp + laneId();

    // Declare the shared variables and fast read in the current population size with __ldg.
    __shared__ uint32_t shared_deadCount;         // The number of dead individuals on the GPU
    ↪  block.
    __shared__ int32_t shared_extraBirthCount;    // The number of new birth required on the GPU
    ↪  block.
    uint32_t local_size = __ldg(d_size);

    // Initialize the dead counter shared variables.
    if (threadIdx.x == 0)
    {
        shared_deadCount = d_deadCount[blockIdx.x];
    }

    // Ensure shared memory has been initialized before proceeding.
    __syncthreads();

    // Initialize the varaible which accounts for the net birth/death ratio of the thread.
    // This saves on atomic incrementation of subpopulation count data.
    int32_t netThreadBirthDeath{ 0 };

    // If the individual is well defined, update the individuals demographics.
    if ((mIndex < local_size) && (!d_mDead[mIndex]))
    {
        // Load in the individuals age and if they are susceptible, infected, or infectious.
        age local_age = (++d_mAge[mIndex]);
        infectStatus local_infectStatus = d_mInfectStatus[mIndex];
```

```
    // Determine if the mosquito dies from the cumulative life expectancy distribution and the
    // individuals pre-determined life expectancy probability.
    if (__ldg(&d_mPLE[mIndex]) <= device_mLifeExpectancy(local_age, h_parameter))
    {
        // If the mosquito was infected with a virus when they died, decrease counter containing
        ↪  the total
        // number of infected individuals for the subpopulation they belong to, for that strain.
        if ((local_infectStatus == 2) && (!d_mDead[mIndex]))
        {
            atomicSub(&d_infectedCount[d_mStrain[mIndex] * __ldg(&d_subPopTotal[0]) +
            ↪  __ldg(&d_mSubPopulation[mIndex])], 1);
        }

        // Record that the individual is dead.
        d_mDead[mIndex] = 1;
        --netThreadBirthDeath;

        // Increase the counter for dead individuals on the block.
        atomicAdd(&shared_deadCount, 1);
    }
    else
    {
        // If the individual is infected but not infectious, determine if the individual
        // needs to be come infectious.
        if ((local_infectStatus == 1) && (d_mExposed[mIndex] < local_age))
        {
            atomicAdd(&d_infectedCount[d_mStrain[mIndex] * __ldg(&d_subPopTotal[0]) +
            ↪  __ldg(&d_mSubPopulation[mIndex])], 1);
            d_mInfectStatus[mIndex] = 2;
        }
    }
}

// Ensure that all necessary deaths have occured at the time step before
// computing the number of births required in order to give the correct population size.
__syncthreads();

// Compute the number of births needed to give the correct population size of the block.
if (threadIdx.x == 0)
{
    // Determine the maximum number of individuals on the block.
    float extraBirthCount = fminf(local_size - mIndex, blockDim.x);

    // Determine the number of births required on the block in order to get the expected
    ↪  population size of the block.
    extraBirthCount = (__ldg(&d_mExpectedPopSize[0])*extraBirthCount) /
    ↪  static_cast<float>(local_size)
        - extraBirthCount + shared_deadCount;

    // Get the fractional part of the expected number of births on the block, and randomly
    ↪  generate a probability in
    // order to determine if the expected number of births should be rounded up or down.
    float frac_extraBirthCount = extraBirthCount - floorf(extraBirthCount);
    if (curand_uniform(&d_randStates[activeThreadId]) <= frac_extraBirthCount)
    {
        shared_extraBirthCount = static_cast<int32_t>(ceilf(extraBirthCount));
    }
    else
    {
        shared_extraBirthCount = static_cast<int32_t>(floorf(extraBirthCount));
    }

    // All births are guaranteed to occur, so decrease the number of dead individuals on the
    ↪  block
    // and store this back to global memory.
    d_deadCount[blockIdx.x] = shared_deadCount - shared_extraBirthCount;
}

// The extra number of births required at the current timestep is required on all threads in the
↪  block before continuing.
__syncthreads();

// If the individual is dead and a birth is required, birth a new individual.
if ((mIndex < local_size) && (d_mDead[mIndex]) && (atomicSub(&shared_extraBirthCount, 1) > 0))
{
```

```
        d_mAge[mIndex] = 0;
        d_mDead[mIndex] = 0;
        d_mInfectStatus[mIndex] = 0;
        d_mPLE[mIndex] = curand_uniform(&d_randStates[activeThreadId]);
        ++netThreadBirthDeath;
    }

    // If an individual only died, or was only birthed during the current time step, record
    // this change in the subpopulation size.
    if (netThreadBirthDeath != 0)
    {
        atomicAdd(&d_subPopCount[d_mSubPopulation[mIndex]], netThreadBirthDeath);
    }
}

// The CPU code which calls all CUDA kernels involved in the update of mosquito demographics.
__host__ void mDemographic(age* d_mAge,
                           dead* d_mDead,
                           exposed* d_mExposed,
                           infectStatus* d_mInfectStatus,
                           pLifeExpectancy* d_mPLE,
                           strain* d_mStrain,
                           subPopulation* d_mSubPopulation,
                           curandState_t* d_randStates,
                           uint32_t* d_infectedCount,
                           uint32_t* d_subPopCount,
                           uint16_t *d_deadCount,
                           uint32_t* d_size,
                           const uint32_t* d_subPopTotal,
                           const uint32_t sizeReserve,
                           const float* d_mExpectedPopSize,
                           const Parameter h_parameter,
                           const Architecture h_architecture)
{
    // Determine the size of the grid of blocks required on the GPU to do the update of mosquito
    ↪  demographics.
    uint32_t gridSize = static_cast<uint32_t>(ceil(sizeReserve /
    ↪  static_cast<float>(C_THREADSPERBLOCK)));

    // Call the kernel which determines if a mosquito dies or not, and if they are to be replaced by
    ↪  a new individual.
    mBlockBirthDeath<<< gridSize, C_THREADSPERBLOCK >>>
        (d_mAge, d_mDead, d_mExposed, d_mInfectStatus, d_mPLE, d_mStrain, d_mSubPopulation,
        ↪  d_randStates,
        d_infectedCount, d_subPopCount, d_deadCount, d_size, d_subPopTotal, d_mExpectedPopSize,
        ↪  h_parameter, h_architecture);
}
```

mepidemic.cu

```
// mepidemic.cu: CUDA kernels for determining the number of transmission events that should occur
// to the mosquitoes of each subpopulation and infecting mosquitoess.

#include "censustypedef.h"              // Type definitions for census data.
#include "constant.h"                   // Constants for simulation.
#include "parameterclass.h"             // Parameter space definition.
#include "curand_kernel.h"              // Required for random number generation in CUDA.
#include "cudaidentities.h"             // Streaming multiprocessor ID, warp ID, and lane ID.
#include "device_launch_parameters.h"   // Thread ID and block ID.
#include "architectureclass.h"          // GPU architectural properties definition.

// Device function which determines if a float should be rounded up or down, dependent
// on uniform random number generation.
__device__ __forceinline__ uint32_t decimalResolve(curandState_t* state, float floating)
{
    float frac = floating - floorf(floating);
    if (curand_uniform(state) <= frac)
    {
        return static_cast<uint32_t>(ceilf(floating));
    }
    else
    {
        return static_cast<uint32_t>(floorf(floating));
    }
}

// CUDA kernel which determines the number of transmission events that should occur to
// mosquitoes in every subpopulation in a system where the human to mosquito population ratio is 1
↪  to 1.
__global__ void mVisitingInfected(curandState_t* d_randStates,
                                  uint32_t* d_mTransmission,
                                  const uint32_t* d_infectedCount,
                                  const uint32_t* d_metaPopCols,
                                  const uint32_t* d_metaPopRows,
                                  const uint32_t* d_subPopTotal,
                                  const Parameter h_parameter,
                                  const Architecture h_architecture)
{
    // Initialize the strain, subpopulation combination of the thread.
    uint32_t strainSubPop = blockIdx.x*blockDim.x + threadIdx.x;

    // Load in the total number of subpopulation from global memory to L1 cache.
    uint32_t subPopTotal = __ldg(&d_subPopTotal[0]);

    // Check if the strain, subpopulation combination is within limits.
    if (strainSubPop < subPopTotal*C_STRAINS)
    {
        // Load the number of infected mosquitoes in that subpopulation, strain combination.
        uint32_t infectedCount = __ldg(&d_infectedCount[strainSubPop]);

        // If there are infected individuals, then compute the number of transmission events
        // that they would cause.
        if (infectedCount > 0)
        {
            // Load in the number of columns and rows into the L1 cache from global memory.
            uint32_t metaPopCols = __ldg(&d_metaPopCols[0]);
            uint32_t metaPopRows = __ldg(&d_metaPopRows[0]);

            // Determine the row and column number of the source subpopulation.
            uint32_t fromSubPop = strainSubPop % subPopTotal;
            int32_t fromCol = fromSubPop % metaPopCols;
            int32_t fromRow = fromSubPop / metaPopCols;

            // Initialize the active thread index used for random number generation.
            uint32_t activeThreadId = smId()*h_architecture.warpsPerSM*h_architecture.threadsPerWarp
                + warpId()*h_architecture.threadsPerWarp + laneId();

            // Load in the random number generator into local memory.
            curandState_t local_state = d_randStates[activeThreadId];

            // Determine the total number of transmission events that the infected humans would
            ↪  cause
            // on a mosquito population of the same size as the human population, and then determine
            // how many of those are long-distance transmissions and local transmissions.
```

```
            uint32_t transmissions = decimalResolve(&local_state,
            ↪  static_cast<float>(h_parameter.bitingRate*h_parameter.mnBitingSuccess*infectedCount));
            uint32_t forcedLongDistance = decimalResolve(&local_state,
            ↪  transmissions*h_parameter.longDistance);
            transmissions -= forcedLongDistance;

            // Initialize the location in global memory where transmission numbers to individuals
            // of every subpopulation begins for each strain.
            uint32_t strainStartIndex = subPopTotal*(strainSubPop / subPopTotal);

            // Declare the row and column for the subpopulation that each transmission
            // event will be "sent" to.
            int32_t toRow, toCol;

            // Declare the subpopulation that each transmission event is sent to, and the
            // distance from the subpopulation that the infected mosquitoes originate.
            int32_t toSubPop;
            float2 distance;

            // Provided there are still local transmission numbers to be made.
            while (transmissions > 0)
            {
                // Randomly generate a distance in the x and y direction to send
                // the transmission event.
                distance = curand_normal2(&local_state);
                distance.x = roundf(h_parameter.kernelStandardDeviation*distance.x);
                distance.y = roundf(h_parameter.kernelStandardDeviation*distance.y);

                // Determine the row and column index of that the transmission event is dispersed
                ↪  to.
                toRow = fromRow + distance.y;
                toCol = fromCol + distance.x;

                // Check the the transmission event makes sense (i.e. the destination is within the
                ↪  lattice).
                // If not, keep generating until it does.
                while ((toRow < 0) || (toRow >= metaPopRows))
                {
                    distance.y = curand_normal(&local_state);
                    distance.y = roundf(h_parameter.kernelStandardDeviation*distance.y);
                    toRow = fromRow + distance.y;
                }
                while ((toCol < 0) || (toCol >= metaPopCols))
                {
                    distance.x = curand_normal(&local_state);
                    distance.x = roundf(h_parameter.kernelStandardDeviation*distance.x);
                    toCol = fromCol + distance.x;
                }

                // Determine the destination subpopulation for the transmission event and
                // increment the number of transmissions that should be placed upon individuals
                // of the destination subpopulation.
                toSubPop = toRow*metaPopCols + toCol;
                atomicAdd(&d_mTransmission[strainStartIndex + toSubPop], 1);

                // Decrease the number of local transmission events to disperse.
                --transmissions;
            }

            // Provided there are long distance transmission events to make, randomly choose
            // a subpopulation within the lattice, and disperse the transmission event to the
            // individuals of that subpopulation.
            while (forcedLongDistance > 0)
            {
                toSubPop = static_cast<uint32_t>(curand(&local_state) % subPopTotal);
                atomicAdd(&d_mTransmission[strainStartIndex + toSubPop], 1);
                --forcedLongDistance;
            }

            // Store the random number generator state back to global memory.
            d_randStates[activeThreadId] = local_state;
        }
    }
}

// CUDA kernel which infects the mosquitoes of every subpopulation given the number
```

```
// of transmission events that should occur in a subpopulation with a one to one human
// to mosquito population ratio.
__global__ void nmTransmission(age* d_mAge,
                               dead* d_mDead,
                               exposed* d_mExposed,
                               infectStatus* d_mInfectStatus,
                               strain* d_mStrain,
                               curandState_t* d_randStates,
                               const uint32_t* d_subPopIndex,
                               const uint32_t* d_subPopLoc,
                               const uint32_t* d_subPopSize,
                               const uint32_t* d_nmTransmission,
                               const uint32_t* d_mSubPopCount,
                               const uint32_t* d_size,
                               const uint32_t* d_subPopTotal,
                               const uint32_t t,
                               const Parameter h_parameter,
                               const Architecture h_architecture)
{
    // Initialize the subpopulation of the thread.
    uint32_t subPop = blockIdx.x*blockDim.x + threadIdx.x;

    // Load in the total number of subpopulation from global memory to L1 cache.
    uint32_t subPopTotal = __ldg(&d_subPopTotal[0]);

    // Initialize the active thread index used for random number generation.
    uint32_t activeThreadId = smId()*h_architecture.warpsPerSM*h_architecture.threadsPerWarp
        + warpId()*h_architecture.threadsPerWarp + laneId();

    // Check if the subpopulation of the thread is within limits.
    if (subPop < subPopTotal)
    {
        // Load in the maximum number of individuals in the subpopulation.
        uint32_t subPopSize = __ldg(&d_subPopSize[subPop]);

        // Load in the total number of alive individuals in the subpopulation.
        uint32_t subPopCount = __ldg(&d_mSubPopCount[subPop]);

        // Check that individuals can exist in the subpopulation
        if (subPopCount > 0)
        {
            // Read in the random number generator state into global memory.
            curandState_t local_state = d_randStates[activeThreadId];

            // Load in the location where to begin in the sub-population ordered census
            // indices for the human population for the subpopulation of the thread.
            uint32_t subPopLoc = __ldg(&d_subPopLoc[subPop]);

            // Load in the total number of humans alive in the entire population.
            uint32_t local_size = __ldg(&d_size[0]);

            // Randomly choose a strain of the virus to begin infecting individuals of the
            // subpopulation with.
            uint32_t local_strain = curand(&local_state) % C_STRAINS;

            // Cycle through all the strains, infecting humans give the total number of transmission
            ↪  events that
            // occur to individuals of the subpopulation per strain.
            for (uint32_t strainCount = 0; strainCount < C_STRAINS; ++strainCount)
            {
                // Load in the number of transmission events of a one to one mosquito to human
                ↪  population ratio,
                // and multiply by the actual mosquito to human ratio to get the total number of
                ↪  transmission events
                // that act on mosquitoes in the subpopulation.
                uint32_t transmissionCount = d_nmTransmission[local_strain*subPopTotal + subPop];

                // Provided there are transmission events:
                while (transmissionCount > 0)
                {
                    // Choose a random individual in the sub-population.
                    uint32_t censusIndex = curand(&local_state) % subPopSize;

                    // Find their index within the census data by using the sub-population ordered
                    // census indices.
                    censusIndex = d_subPopIndex[subPopLoc + censusIndex];
```

```
                    // If their census index is valid (which it should be by construction anyway),
                    // and the mosquito is actually alive, continue with transmission.
                    if ((censusIndex < local_size) && (d_mDead[censusIndex] == 0))
                    {
                        // The virus is transmitted to the individual, so decrease the number of
                        // remaining transmission events to make.
                        --transmissionCount;

                        // The individual will be infected if it is not already infected.
                        if (d_mInfectStatus[censusIndex] == 0)
                        {
                            // Infected, not infectious.
                            d_mInfectStatus[censusIndex] = 1;
                            d_mStrain[censusIndex] = local_strain;

                            // Age at which individual becomes infectious.
                            d_mExposed[censusIndex] =
                            ↪  static_cast<exposed>(decimalResolve(&local_state,
                                static_cast<float>(d_mAge[censusIndex] + h_parameter.mExposed +
                                    h_parameter.exIncPeriodRange*cosf(2 * CR_CUDART_PI*t /
                                    ↪  static_cast<float>(C_YEAR)))));
                        }
                    }
                }

                // Move onto the transmission events of the next strain.
                local_strain = (local_strain + 1) % C_STRAINS;
            }

            // Store the random number generator back to global memory.
            d_randStates[activeThreadId] = local_state;

        }
    }
}
```

ndemographic.cu

```
// ndemographics.cu: This file contains the host code which invokes CUDA kernels
// for the simulation of the humans demographics.

#include "censustypedef.h"              // Type definitions for census data.
#include "constant.h"                   // Constants for simulation.
#include "curand_kernel.h"              // Required for random number generation in CUDA.
#include "cudaidentities.h"             // Streaming multiprocessor ID, warp ID, and lane ID.
#include "device_launch_parameters.h"   // Thread ID and block ID.
#include "parameterclass.h"             // Parameter space definition.
#include "architectureclass.h"          // GPU architectural properties definition.

// Gives the cumulative bi-Weibull human life expectancy distribution probability. This disitrubion
// gives the most flexibility in age-specific mortality rates.
__device__ __forceinline__ float device_nLifeExpectancy(const age local_age, const Parameter
↪  h_parameter)
{
    // Initialize as -(x / scale)^shape
    float dLifeExpectancy = powf(local_age * static_cast<float>(h_parameter.nScaleInfantMortality),
    ↪  h_parameter.nShapeInfantMortality);

    // Need more terms if passed the location where the second (burn out) Weibull distribution
    ↪  begins
    if (local_age < h_parameter.nLocWeibull)
    {
        return (1.0f - expf(-dLifeExpectancy));
    }
    else
    {
        dLifeExpectancy = dLifeExpectancy +
            powf((local_age - h_parameter.nLocWeibull) / h_parameter.nScaleLifeExpectancy,
            ↪  h_parameter.nShapeLifeExpectancy);
        return (1.0f - expf(-dLifeExpectancy));
    }
}

// CUDa kernel determining if an individual dies at the current time step.
__global__ void nBlockBirthDeath(age* d_nAge,
                                 exposed* d_nExposed,
                                 history* d_nHistory,
                                 infectStatus* d_nInfectStatus,
                                 pLifeExpectancy* d_nPLE,
                                 recovery* d_nRecovery,
                                 strain* d_nStrain,
                                 subPopulation* d_nSubPopulation,
                                 curandState_t* d_randStates,
                                 uint32_t* d_nInfectedCount,
                                 const uint32_t* d_nSize,
                                 const uint32_t* d_subPopTotal,
                                 const Parameter h_parameter,
                                 const Architecture h_architecture)
{
    // Determine which human belongs to each thread.
    uint32_t nIndex = threadIdx.x + blockDim.x * blockIdx.x;

    // Read in the number of humans in the population.
    uint32_t local_nSize = __ldg(&d_nSize[0]);

    // If the human belonging to the thread is valid.
    if (nIndex < local_nSize)
    {
        // Load in the individuals age and if they are susceptible, infected, or infectious.
        age local_age = (++d_nAge[nIndex]);
        infectStatus local_infectStatus = d_nInfectStatus[nIndex];

        // Determine if the human dies from the cumulative life expectancy distribution and the
        // individuals pre-determined life expectancy probability.
        if (__ldg(&d_nPLE[nIndex]) > device_nLifeExpectancy(local_age, h_parameter))
        {
            // Check if an infected, but not infectious individual is due to become infectious, and
            // check if an infectious individual has reached age of recovery from the disease.
            if ((local_infectStatus == 2) && (d_nRecovery[nIndex] < local_age))
            {
                atomicSub(&d_nInfectedCount[d_nStrain[nIndex] * __ldg(&d_subPopTotal[0]) +
                ↪  __ldg(&d_nSubPopulation[nIndex])], 1);
```

```
                d_nInfectStatus[nIndex] = 0;
            }
            else if ((local_infectStatus == 1) && (d_nExposed[nIndex] < local_age))
            {
                atomicAdd(&d_nInfectedCount[d_nStrain[nIndex] * __ldg(&d_subPopTotal[0]) +
                ↪  __ldg(&d_nSubPopulation[nIndex])], 1);
                d_nInfectStatus[nIndex] = 2;
            }
        }
        else
        {
            // If the human was infected with a virus when they died, decrease counter containing
            ↪  the total
            // number of infected individuals for the subpopulation they belong to, for that strain.
            if (local_infectStatus == 2)
            {
                atomicSub(&d_nInfectedCount[d_nStrain[nIndex] * __ldg(&d_subPopTotal[0]) +
                ↪  +__ldg(&d_nSubPopulation[nIndex])], 1);
            }

            // Initialize the active thread ID used for random number generation.
            uint32_t activeThreadId = smId()*h_architecture.warpsPerSM*h_architecture.threadsPerWarp
                + warpId()*h_architecture.threadsPerWarp + laneId();

            // Create a new human in place of the human that died.
            d_nAge[nIndex] = 0;
            d_nInfectStatus[nIndex] = 0;
            d_nPLE[nIndex] = curand_uniform(&d_randStates[activeThreadId]);

            // Reset the humans immunological history.
            strain local_strain{ 0 };
            while (local_strain < C_STRAINS)
            {
                d_nHistory[local_strain*local_nSize + nIndex] = 0;
                ++local_strain;
            }
        }
    }
}

// The CPU code which calls all CUDA kernels involved in the update of human demographics.
__host__ void nDemographic(age* d_nAge,
                           exposed* d_nExposed,
                           history* d_nHistory,
                           infectStatus* d_nInfectStatus,
                           pLifeExpectancy* d_nPLE,
                           recovery* d_nRecovery,
                           strain* d_nStrain,
                           subPopulation* d_nSubPopulation,
                           curandState_t* d_randStates,
                           uint32_t* d_nInfectedCount,
                           uint32_t* d_nSize,
                           const uint32_t* d_subPopTotal,
                           const uint32_t h_nSize,
                           const Parameter h_parameter,
                           const Architecture h_architecture)
{
    // Determine the size of the grid of blocks required on the GPU to do the update of human
    ↪  demographics.
    uint32_t gridSize = static_cast<uint32_t>(ceil(h_nSize /
    ↪  static_cast<float>(C_THREADSPERBLOCK)));

    // Call the kernel which determines if a human dies or not (and is replaced if they do die).
    nBlockBirthDeath<<< gridSize, C_THREADSPERBLOCK >>>
        (d_nAge, d_nExposed, d_nHistory, d_nInfectStatus, d_nPLE, d_nRecovery, d_nStrain,
        ↪  d_nSubPopulation,
        d_randStates, d_nInfectedCount, d_nSize, d_subPopTotal, h_parameter, h_architecture);
}
```

nepidemic.cu

```
// nepidemic.cu: CUDA kernels for determining the number of transmission events that should occur
// to the individuals of each subpopulation and infecting humans.

#include "censustypedef.h"              // Type definitions for census data.
#include "constant.h"                   // Constants for simulation.
#include "parameterclass.h"             // Parameter space definition.
#include "curand_kernel.h"              // Required for random number generation in CUDA.
#include "cudaidentities.h"             // Streaming multiprocessor ID, warp ID, and lane ID.
#include "device_launch_parameters.h"   // Thread ID and block ID.
#include "architectureclass.h"          // GPU architectural properties definition.

// Device function which determines if a float should be rounded up or down, dependent
// on uniform random number generation.
__device__ __forceinline__ uint32_t decimalResolve(curandState_t* state, float floating)
{
    float frac = floating - floorf(floating);
    if (curand_uniform(state) <= frac)
    {
        return static_cast<uint32_t>(ceilf(floating));
    }
    else
    {
        return static_cast<uint32_t>(floorf(floating));
    }
}

// CUDA kernel which determines the number of transmission events that should occur to
// individuals in every subpopulation.
__global__ void nVisitingInfected(curandState_t* d_randStates,
                                  uint32_t* d_nTransmission,
                                  const uint32_t* d_infectedCount,
                                  const uint32_t* d_nSubPopCount,
                                  const uint32_t* d_mSubPopCount,
                                  const uint32_t* d_metaPopCols,
                                  const uint32_t* d_metaPopRows,
                                  const uint32_t* d_subPopTotal,
                                  const Parameter h_parameter,
                                  const Architecture h_architecture)
{
    // Initialize the strain, subpopulation combination of the thread.
    uint32_t strainSubPop = blockIdx.x*blockDim.x + threadIdx.x;

    // Load in the total number of subpopulation from global memory to L1 cache.
    uint32_t subPopTotal = __ldg(&d_subPopTotal[0]);

    // Check if the strain, subpopulation combination is within limits.
    if (strainSubPop < subPopTotal*C_STRAINS)
    {
        // Load the number of infected mosquitoes in that subpopulation, strain combination.
        uint32_t infectedCount = __ldg(&d_infectedCount[strainSubPop]);

        // If there are infected individuals, then compute the number of transmission events
        // that they would cause.
        if (infectedCount > 0)
        {
            // Load in the number of columns and rows into the L1 cache from global memory.
            uint32_t metaPopCols = __ldg(&d_metaPopCols[0]);
            uint32_t metaPopRows = __ldg(&d_metaPopRows[0]);

            // Determine the row and column number of the source subpopulation.
            uint32_t fromSubPop = strainSubPop % subPopTotal;
            int32_t fromCol = fromSubPop % metaPopCols;
            int32_t fromRow = fromSubPop / metaPopCols;

            // Initialize the active thread index used for random number generation.
            uint32_t activeThreadId = smId()*h_architecture.warpsPerSM*h_architecture.threadsPerWarp
                + warpId()*h_architecture.threadsPerWarp + laneId();

            // Load in the random number generator into local memory.
            curandState_t local_state = d_randStates[activeThreadId];

            // Initialize the ratio between the number of mosquitoes and the number of humans.
            float mosquitoHumanRatio = d_mSubPopCount[fromSubPop] /
            ↪  static_cast<float>(d_nSubPopCount[fromSubPop]);
```

```
    // Determine the total number of transmission events that the mosquitoes would cause,
    // and then determine how many of those are long-distance transmissions and local
    ↪  transmissions.
    uint32_t transmissions = decimalResolve(&local_state,
    ↪  static_cast<float>(mosquitoHumanRatio*h_parameter.bitingRate*h_parameter.nmBitingSuccess*infectedCount));
    uint32_t forcedLongDistance = decimalResolve(&local_state,
    ↪  transmissions*h_parameter.longDistance);
    transmissions -= forcedLongDistance;

    // Initialize the location in global memory where transmission numbers to individuals
    // of every subpopulation begins for each strain.
    uint32_t strainStartIndex = subPopTotal*(strainSubPop / subPopTotal);

    // Declare the row and column for the subpopulation that each transmission
    // event will be "sent" to.
    int32_t toRow, toCol;

    // Declare the subpopulation that each transmission event is sent to, and the
    // distance from the subpopulation that the infected mosquitoes originate.
    int32_t toSubPop;
    float2 distance;

    // Provided there are still local transmission numbers to be made.
    while (transmissions > 0)
    {
        // Randomly generate a distance in the x and y direction to send
        // the transmission event.
        distance = curand_normal2(&local_state);
        distance.x = roundf(h_parameter.kernelStandardDeviation*distance.x);
        distance.y = roundf(h_parameter.kernelStandardDeviation*distance.y);

        // Determine the row and column index of that the transmission event is dispersed
        ↪  to.
        toRow = fromRow + distance.y;
        toCol = fromCol + distance.x;

        // Check the the transmission event makes sense (i.e. the destination is within the
        ↪  lattice).
        // If not, keep generating until it does.
        while ((toRow < 0) || (toRow >= metaPopRows))
        {
            distance.y = curand_normal(&local_state);
            distance.y = roundf(h_parameter.kernelStandardDeviation*distance.y);
            toRow = fromRow + distance.y;
        }
        while ((toCol < 0) || (toCol >= metaPopCols))
        {
            distance.x = curand_normal(&local_state);
            distance.x = roundf(h_parameter.kernelStandardDeviation*distance.x);
            toCol = fromCol + distance.x;
        }

        // Determine the destination subpopulation for the transmission event and
        // increment the number of transmissions that should be placed upon individuals
        // of the destination subpopulation.
        toSubPop = toRow*metaPopCols + toCol;
        atomicAdd(&d_nTransmission[strainStartIndex + toSubPop], 1);

        // Decrease the number of local transmission events to disperse.
        --transmissions;
    }

    // Provided there are long distance transmission events to make, randomly choose
    // a subpopulation within the lattice, and disperse the transmission event to the
    // individuals of that subpopulation.
    while (forcedLongDistance > 0)
    {
        toSubPop = static_cast<uint32_t>(curand(&local_state) % subPopTotal);
        atomicAdd(&d_nTransmission[strainStartIndex + toSubPop], 1);
        --forcedLongDistance;
    }

    // Store the random number generator state back to global memory.
    d_randStates[activeThreadId] = local_state;
}
```

```
    }
}

// CUDA kernel which infects the humans of every subpopulation given the number
// of transmission events that should occur to humans in that subpopulation.
__global__ void mnTransmission(age* d_nAge,
                               exposed* d_nExposed,
                               history* d_nHistory,
                               infectStatus* d_nInfectStatus,
                               recovery* d_nRecovery,
                               strain* d_nStrain,
                               curandState_t* d_randStates,
                               uint32_t* d_nAgeOfInfection,
                               uint32_t* d_nAgeOfInfectionCount,
                               const uint32_t* d_nSubPopIndex,
                               const uint32_t* d_nSubPopLoc,
                               const uint32_t* d_nSubPopSize,
                               const uint32_t* d_mnTransmission,
                               const uint32_t* d_nSize,
                               const uint32_t* d_subPopTotal,
                               const uint32_t timeLeft,
                               const Parameter h_parameter,
                               const Architecture h_architecture)
{
    // Initialize the subpopulation of the thread.
    uint32_t subPop = blockIdx.x*blockDim.x + threadIdx.x;

    // Load in the total number of subpopulation from global memory to L1 cache.
    uint32_t subPopTotal = __ldg(&d_subPopTotal[0]);

    // Initialize the active thread index used for random number generation.
    uint32_t activeThreadId = smId()*h_architecture.warpsPerSM*h_architecture.threadsPerWarp
        + warpId()*h_architecture.threadsPerWarp + laneId();

    // Check if the subpopulation of the thread is within limits.
    if (subPop < subPopTotal)
    {
        // Load in the maximum number of individuals in the subpopulation.
        uint32_t subPopSize = __ldg(&d_nSubPopSize[subPop]);

        // Check that individuals exist in the subpopulation.
        if (subPopSize > 0)
        {
            // Read in the random number generator state into global memory.
            curandState_t local_state = d_randStates[activeThreadId];

            // Load in the location where to begin in the sub-population ordered census
            // indices for the human population for the subpopulation of the thread.
            uint32_t subPopLoc = __ldg(&d_nSubPopLoc[subPop]);

            // Load in the total number of humans alive in the subpopulation.
            uint32_t local_nSize = __ldg(&d_nSize[0]);

            // Randomly choose a strain of the virus to begin infecting individuals of the
            // subpopulation with.
            uint32_t local_strain = curand(&local_state) % C_STRAINS;

            // Cycle through all the strains, infecting humans give the total number of transmission
            ↪  events that
            // occur to individuals of the subpopulation per strain.
            for (uint32_t strainCount = 0; strainCount < C_STRAINS; ++strainCount)
            {
                // Load in the number of transmission events.
                uint32_t transmissionCount = d_mnTransmission[local_strain*subPopTotal + subPop];

                // Provided there are transmission events:
                while (transmissionCount > 0)
                {
                    // Choose a random individual in the sub-population.
                    uint32_t censusIndex = curand(&local_state) % subPopSize;

                    // Find their index within the census data by using the sub-population ordered
                    // census indices.
                    censusIndex = d_nSubPopIndex[subPopLoc + censusIndex];

                    // If their census index is valid (which it should be by construction anyway),
```

```
    // continue with transmission.
    if (censusIndex < local_nSize)
    {
        // The virus is transmitted to the individual, so decrease the number of
        // remaining transmission events to make.
        --transmissionCount;

        // The individual will be infected if it is not already infected, or
        // if it is not immune to the strain of the virus.
        if ((!d_nInfectStatus[censusIndex]) && (!d_nHistory[local_strain*local_nSize
        ↪  + censusIndex]))
        {
            // If the simulation is within the last 10 years, record the age of the
            ↪  individual being infected.
            if (timeLeft <= (10 * C_YEAR))
            {
                // Compute the number of previous infections the individual has.
                uint32_t prevInfect = 0;
                for (uint32_t s = 0; s < C_STRAINS; ++s)
                {
                    prevInfect += ((d_nHistory[s*local_nSize + censusIndex] == 0) ?
                    ↪  0 : 1);
                }

                // Increment the appropriate number of ages recorded for the first,
                ↪  second, third and fourth infection.
                uint32_t infNumber = atomicInc(&d_nAgeOfInfectionCount[prevInfect],
                ↪  C_NAOIRECORD);

                // Save the age of exposure.
                d_nAgeOfInfection[prevInfect*C_NAOIRECORD + infNumber] =
                ↪  d_nAge[censusIndex];
            }

            // Infect the individual.
            d_nInfectStatus[censusIndex] = 1;                           // Infected,
            ↪  not infectious.
            d_nStrain[censusIndex] = local_strain;
            d_nHistory[local_strain*local_nSize + censusIndex] =        // Immune to
            ↪  all future infections of strain.
                !d_nAge[censusIndex] ? 65535 : d_nAge[censusIndex];     // Record
                ↪  age of exposure (65535 is age zero days).

            // Age at which individual becomes infectious.
            d_nExposed[censusIndex] =
            ↪  static_cast<exposed>(decimalResolve(&local_state,
                static_cast<float>(d_nAge[censusIndex] + h_parameter.nExposed)));

            // Age at which individual stops being infected.
            d_nRecovery[censusIndex] =
            ↪  static_cast<recovery>(decimalResolve(&local_state,
                static_cast<float>(d_nExposed[censusIndex] +
                ↪  h_parameter.recovery)));
        }
    }
}

// Create a small chance of a transmission event occuring from external sources.
// Introduction rate given as per day per strain, so need to convert to per day per
↪  strain per subpopulation.
uint32_t extIntroduction = decimalResolve(&local_state,
    static_cast<float>(h_parameter.externalInfection) /
    ↪  static_cast<float>(subPopTotal));

// Define the number of attempts that have been made to externally introduce the
↪  virus.
uint32_t introAttempt = 0;

// Provided there is an introduction:
while ((extIntroduction > 0) && (introAttempt < C_MAXINTROATTEMPT))
{
    // Increase the number of attempts that have been made to introduce the virus.
    ++introAttempt;

    // Choose a random individual in the sub-population.
    uint32_t censusIndex = curand(&local_state) % subPopSize;
```

```
                    // Find their index within the census data by using the sub-population ordered
                    // census indices.
                    censusIndex = d_nSubPopIndex[subPopLoc + censusIndex];

                    // If their census index is valid (which it should be by construction anyway),
                    // continue with transmission.
                    if (censusIndex < local_nSize)
                    {
                        // The individual will be infected if it is not already infected, or
                        // if it is not immune to the strain of the virus.
                        if ((!d_nInfectStatus[censusIndex]) && (!d_nHistory[local_strain*local_nSize
                        ↪  + censusIndex]))
                        {
                            // The virus is transmitted to the individual, so decrease the number of
                            // remaining introduction events to make.
                            --extIntroduction;

                            // Infect the individual.
                            d_nInfectStatus[censusIndex] = 1;                           // Infected,
                            ↪  not infectious.
                            d_nStrain[censusIndex] = local_strain;
                            d_nHistory[local_strain*local_nSize + censusIndex] =        // Immune to
                            ↪  all future infections of strain.
                                !d_nAge[censusIndex] ? 65535 : d_nAge[censusIndex];     // Record
                                ↪  age of exposure (65535 is age zero days).

                            // Age at which individual becomes infectious.
                            d_nExposed[censusIndex] =
                            ↪  static_cast<exposed>(decimalResolve(&local_state,
                                static_cast<float>(d_nAge[censusIndex] + h_parameter.nExposed)));

                            // Age at which individual stops being infected.
                            d_nRecovery[censusIndex] =
                            ↪  static_cast<recovery>(decimalResolve(&local_state,
                                static_cast<float>(d_nExposed[censusIndex] +
                                ↪  h_parameter.recovery)));
                        }
                    }
                }

                // Move onto the transmission events of the next strain.
                local_strain = (local_strain + 1) % C_STRAINS;
            }

            // Store the random number generator back to global memory.
            d_randStates[activeThreadId] = local_state;
        }
    }
}
```

setuprng.cu

```
// setuprng.cu :: Initializes the random states for CUDA random number generation.
// There is one RNG for each possible active thread on the GPU.

#include "curand_kernel.h"              // CUDA random number generation.
#include "device_launch_parameters.h"   // Block ID and thread ID.
#include "constant.h"                   // Constants for simulation.
#include <chrono>                       // System clock for seed generation.
#include "cudaidentities.h"             // Identities for determining the SM, warp, and lane id of a
↪  GPU thread.
#include "architectureclass.h"          // GPU architectural properties definition.

// CUDA kernel which initializes the random number generator states.
__global__ void setupRandStates(curandState_t* d_randStates,
                                unsigned int seed,
                                uint32_t totalActiveThreads)
{
    // Initialize the active thread index for the CUDA thread.
    uint32_t activeThreadId = blockIdx.x*blockDim.x + threadIdx.x;

    // Provided the active thread index is valid, initialize the
    // random number generator state.
    if (activeThreadId < totalActiveThreads)
    {
        curand_init(seed, activeThreadId, 0, &d_randStates[activeThreadId]);
    }
}

// Host function which calls the kernel which initializes the CUDA random number generator states.
__host__ void setupCudaRNG(curandState_t* d_randStates,
                           Architecture h_architecture)
{
    // Initialize random number generator seed from the system clock time.
    unsigned int seed{ static_cast<unsigned
    ↪  int>(std::chrono::system_clock::now().time_since_epoch().count()) };

    // Initialize the total possible number of active threads on the GPU.
    // Initialize the grid size to use in initialization of random number generators.
    uint32_t totalActiveThreads{
    ↪  h_architecture.totalSM*h_architecture.warpsPerSM*h_architecture.threadsPerWarp };
    uint32_t setupRandGridSize{ static_cast<uint32_t>(ceil(totalActiveThreads /
    ↪  static_cast<float>(C_THREADSPERBLOCK))) };

    // Call the kernel which initializes the random number generator states.
    setupRandStates<<< setupRandGridSize, C_THREADSPERBLOCK >>>(d_randStates, seed,
    ↪  totalActiveThreads);
}
```

.cuh and .h header files

Here, only header files which contain class and function **definitions** are presented. `.cuh` header files must be processed by `nvcc`.

Architectureclass.h

```
// Architectureclass.h: This header file contains the custom class for GPU architecture properties.

#include <cstdint>      // Fixed-width integers.

// Start of the header guard.
#ifndef ARCHITECTURECLASS_H
#define ARCHITECTURECLASS_H

class Architecture
{
public:
    uint32_t threadsPerWarp;    // Number of threads per warp.
    uint32_t warpsPerSM;        // Number of warps per Streaming Multiprocessor (SM).
    uint32_t totalSM;           // Total number of SMs.
};

#endif
```

censustypedef.h

```
// censustypedef.h: Header file for the type declarations of the human and mosquito censuses.
#include <cstdint>              // Fixed width integers.

// Start of header guard
#ifndef CENSUSTYPEDEF_H
#define CENSUSTYPEDEF_H

typedef uint16_t age;           // Inidividual age in days unlikely to exceed 65535 days.
typedef uint8_t dead;           // Boolean for if individual is dead or alive (mosquito only).
typedef uint16_t exposed;       // Age at which infection becomes infectious.
typedef uint16_t history;        // Strains an individual has been infected with (human only).
typedef uint8_t infectStatus;   // Disease susceptibility (= 0), infected ( = 1) and infectiousness
↪  (= 2).
typedef float pLifeExpectancy;  // A random probability used in determining the life expectancy of
↪  individual.
typedef uint16_t recovery;      // Age at which infection ends.
typedef uint8_t strain;         // Dengue serotype the individual is infected with.
typedef uint32_t subPopulation; // The subpopulation that the individual belongs to.

#endif
```

constant.h

```
// The header file containing the name space for all constants of the model.

#include <string>           // String library.
#include <cstdint>          // Fixed-width integers.
#include "host_defines.h"   // Allows variables to be defined on the device.

// Start of header guard.
#ifndef CONSTANT_H
#define CONSTANT_H

// Constants external to simulation.
#define C_MAXPARARUN 1                          // Total number of parameter values to test.
#define C_MAXSIMRUN 25                          // Total number of simulation runs.
#define C_OUTPUTFOLDER "../../Output files/"    // Define an output folder for data.
#define C_SHUTDOWN 0                            // Shut down the computer at the end of the
↪  simulation.

// Constants explicitly internal to simulation.
#define C_STRAINS 4                             // Number of serotypes.
```

```
#define C_MMAXINITIALAGE 300                    // Maximum mosquito age in days.
#define C_NMAXINITIALAGE 150                    // Maximum human age in years.
#define C_YEAR 365                              // Number of days in a year.
#define C_INITIALMINTIME 5*C_YEAR               // Simulation time for obtaining initial outbreak
↪  repeats.
#define C_INITIALMAXTIME 100*C_YEAR             // Simulation time for setting up initial conditions
↪  for subsequent runs.
#define C_NSIZERECORD 100000                    // Maximum number of humans to output in a census.
#define C_NAOIRECORD 10000                      // The maximum number of ages of infection to record
↪  for each novel exposure.
#define C_MAXINTROATTEMPT 1000                  // Maximum number of external introduction attempts
↪  on susceptible individuals.

// GPU-specific constants.
#define C_THREADSPERBLOCK 128                   // Number of GPU threads per GPU block per kernel
↪  call (except summing across a vector).
#define C_THREADSPERBLOCKSUM 128                // Number of threads per block per summing kernel
↪  call.

#endif
```

cudaidentities.h

```
// cudaidentities.h: Header file containing CUDA intrinsics for determining the warp ID,
// SM ID (streaming multi-processor), and lane ID of a thread. This is used
// to compute the active thread index for a thread.

#include "host_defines.h"       // Definition of device-only functions.
#include <cstdint>              // Fixed-width integers.

// Start of the header guard.
#ifndef CUDAIDENTITIES_H
#define CUDAIDENTITIES_H

// Device function outputting the warp ID of a thread. This is the
// identity of the warp on each streaming multi-processor.
static __device__ __inline__ uint32_t warpId()
{
    uint32_t warpid;
    asm("mov.u32 %0, %warpid;" : "=r"(warpid));
    return warpid;
}

// Device function outputting the stream-multiprocessor ID of a thread.
// There are typically eight streaming-multiprocessors.
static __device__ __inline__ uint32_t smId()
{
    uint32_t smid;
    asm("mov.u32 %0, %smid;" : "=r"(smid));
    return smid;
}

// Device function outputting the lane ID of a thread. This is the
// thread ID of the warp.
static __device__ __inline__ uint32_t laneId()
{
    uint32_t laneid;
    asm("mov.u32 %0, %laneid;" : "=r"(laneid));
    return laneid;
}

#endif
```

Parameterclass.h

```
// Parameterclass.h: This header file contains the class for parameters for the dengue simulation.

#include <cstdint>      // Fixed-width integers.

// Start of the header guard.
#ifndef PARAMETERCLASS_H
#define PARAMETERCLASS_H

class Parameter
{
public:
    // Non-epidemiological parameters.
    float nSize{ 0.0f };                    // Number of human individuals in the metapopulation.
    float metaPopRows{ 0.0f };              // Number of rows in the metapopulation lattice.
    float metaPopCols{ 0.0f };              // Number of columns in the metapopulation lattice.
    float maxMosToHuman{ 0.0f };            // Maximum mosquito to human ratio.
    float minMosToHuman{ 0.0f };            // Minimum mosquito to human ratio.
    float nShapeInfantMortality{ 0.0f };    // Human life-expectancy bi-weibull scale parameter
    ↪  (burn in).
    float nScaleInfantMortality{ 0.0f };    // Human life-expectancy bi-weibull shape parameter
    ↪  (burn in).
    float nScaleLifeExpectancy{ 0.0f };     // Second (decay) human bi-weibull scale parameter.
    ↪  "Close to" life expectency.
    float nShapeLifeExpectancy{ 0.0f };     // Second (decay) human-bi-weibull shape parameter.
    float nLocWeibull{ 0.0f };              // Age at which human life-expectancy that burn in
    ↪  distribution becomes decay out.
    float mScaleLifeExpectancy{ 0.0f };     // Mosquito life-expectancy Weibull scale parameter.
    float mShapeLifeExpectancy{ 0.0f };     // Mosquito life-expectancy Weibull shape parameter.

    // Epidemiological parameters.
    float bitingRate{ 0.0f };               // The per day biting rate of mosquitoes.
    float mnBitingSuccess{ 0.0f };           // The probability of virus being transmitted from an
    ↪  infectious individual given a bite.
    float nmBitingSuccess{ 0.0f };            // The probability of virus being transmitted from an
    ↪  infectious individual given a bite.
    float recovery{ 0.0f };                // The number of days humans are infectious.
    float mExposed{ 0.0f };                // The number of days mosquitoes are infected, but not
    ↪  infectious (EIP).
    float nExposed{ 0.0f };                // The number of days humans are infected , but not
    ↪  infectious.
    float externalInfection{ 0.0f };        // Infections per 100,000 per day per strain.
    float longDistance{ 0.0f };             // The probability of a single infectious causing long
    ↪  distance transmission .
    float exIncPeriodRange{ 0.0f };         // Maximum difference in mean EIP in off/on-season with
    ↪  the mid-season.
    float kernelStandardDeviation{ 1.0f };  // The standard deviation of the normally-distributed
    ↪  disease dispersal kernel
                                            // used in modelling spread of disease from a given
                                            ↪  subpopulation to surrounding subpopulations.
};

#endif
```

reduction.cuh

```
// reduction.cuh: This CUDA header file contains the CUDA reduction kernels which compute the sum
// across input count arrays. For example, the input count array may countain counts per GPU block,
// yielding an output that sums the counts across all GPU blocks.

#include <cstdint>              // Fixed-width integers
#include "host_defines.h"       // Allows definition of CUDA kernels.

#ifndef REDUCTION_CUH
#define REDUCTION_CUH

// blockSingleSum is the CUDA kernel which sums across a vector of input count data. The kernel can
↪  sum at most
// twice the size of a GPU block number of elements at once. The kernel is called multiple times in
↪  cases where
// the number of elements exceeds twice the size of a GPU block. The kernel can also sum across
↪  multiple sets of count
// data concurrently provided the multiple sets are ordered within the input data (e.g. for each
↪  strain), in this case
// invoke the kernel over an integer multiple the reduction grid size.
template <uint32_t blockSize, typename T>
__global__ void blockSingleSum(T* d_outputCount,
                               const T* d_inputCount,
                               const uint32_t oldReductionSize,
                               const uint32_t reductionSize)
{
    // Declare all the shared memory required on the GPU block. The amount of shared memory is
    ↪  dependent
    // on the number of threads in a block, thus the size of the shared memory is defined in the
    ↪  kernel call itself.
    extern __shared__ uint32_t sharingIsCaring[];
    uint32_t* countData = (uint32_t*)(&sharingIsCaring[0]);

    // Initialize the reduction index. This is the (local) index for one set of count data.
    // reductionSize gives the number of blocks in the GPU grid required to reduce one set of count
    ↪  data.
    // Multiply by two here, since a single GPU block can sum up to two blocks worth of elements.
    uint16_t threadid = threadIdx.x;
    uint32_t redIndex = (blockIdx.x % reductionSize)*blockDim.x*2 + threadid;

    // Initialize the index for loading in from the input count data (of the previous reduction).
    uint32_t loadIndex = (blockIdx.x / reductionSize)*oldReductionSize + redIndex;

    // Initialize the shared variables to zero.
    countData[threadid] = 0;

    // If the reduction index is small enough such that a data element exists within that particular
    // set of input data, then load it in using the loading index. If the reduction index is small
    ↪  enough
    // such that the corresponding data element and a data element a block-width further in memory
    ↪  is within the same
    // set of input data, then load both in and "pre-"sum.
    if ((redIndex + blockDim.x) < oldReductionSize)
    {
        countData[threadid] = static_cast<uint32_t>(d_inputCount[loadIndex] + d_inputCount[loadIndex
        ↪  + blockDim.x]);
    }
    else if (redIndex < oldReductionSize)
    {
        countData[threadid] = static_cast<uint32_t>(d_inputCount[loadIndex]);
    }

    // Ensure all data has been loaded onto the shared variables before proceeding
    // with the summation/reduction.
    __syncthreads();

    // Depending on the size of each GPU block, add the elements half a block-width away until
    ↪  reaching only the
    // first 32 elements of the reduced count data are left to be summed (this is the same of a
    ↪  warp).
    if (blockSize >= 1024){ if (threadid < 512) { countData[threadid] += countData[threadid + 512];
    ↪  } __syncthreads(); }
    if (blockSize >= 512){ if (threadid < 256) { countData[threadid] += countData[threadid + 256]; }
    ↪  __syncthreads(); }
    if (blockSize >= 256){ if (threadid < 128) { countData[threadid] += countData[threadid + 128]; }
    ↪  __syncthreads(); }
```

```
    if (blockSize >= 128){ if (threadid < 64) { countData[threadid] += countData[threadid + 64]; }
    ↪  __syncthreads(); }

    // Since the reduction is now occuring only on one warp, and instructions are issued per warp,
    ↪  the latter if
    // statements are no-longer required. If countData is declared as "volatile", the threadfence
    ↪  blocks are no longer
    // required either since operating on the shared data would force threads to store the result
    ↪  back.
    if (threadid < 32)
    {
        if (blockSize >= 64){ countData[threadid] += countData[threadid + 32];
        ↪  __threadfence_block(); };
        if (blockSize >= 32){ countData[threadid] += countData[threadid + 16];
        ↪  __threadfence_block(); };
        if (blockSize >= 16){ countData[threadid] += countData[threadid + 8]; __threadfence_block();
        ↪  };
        if (blockSize >= 8){ countData[threadid] += countData[threadid + 4]; __threadfence_block();
        ↪  };
        if (blockSize >= 4){ countData[threadid] += countData[threadid + 2]; __threadfence_block();
        ↪  };
        if (blockSize >= 2){ countData[threadid] += countData[threadid + 1]; };
    }

    // Write the summed result back to global memory such that consecutive blocks store
    // in consecutive indices of the output count data.
    if (threadid == 0)
    {
        d_outputCount[blockIdx.x] = static_cast<T>(countData[0]);
    }
}

#endif
```

Glossary

Antibody dependent enhancement	The mechanism by which subsequent heterotypic infection following a primary infect is immunologically enhanced.
Basic reproduction number	In epidemiology, defined as the total number of secondary infections caused by a single primary infection in an entirely susceptible population.
Burn-in period	The number of iterations to allow for a Markov chain to converge to a stationary distribution.
Central processing unit	Executes computational instructions on a typical desktop machine.
Graphics processing unit	An optional component of a desktop machine, the graphics processing unit can execute computational instructures and has high arithmetic throughput but low memory bandwidth.
Hyper-endemic	Region has multiple co-circulating dengue serotypes.
Endemic	A disease is endemic if it is found to regularly circulate a population of humans within a given area.
Epidemic	A large outbreak of a disease.
Extrinsic incubation period	The period of time for an infected disease vector to become infectious.
Incidence	The number of new cases during a given time interval.
Individual based model	In epidemiology, a mathematical description of explicit processes which each individual undergoes over time.
Intrinsic incubation period	The time period for an infected human to become infectious.
Local polynomial regression	A method for fitting a line through some data.

Ordinary differential equation	In epidemiology, a mathematical description of the rate of change in the number of individuals in a particular state, e.g. susceptible, infected.
Peri-urban	Landscape between urban and rural areas.
Posterior distribution	A statistical distribution of the probability of a random event occurring conditional on some data.
Prevalence	The number of infected individuals at a given time point.
Prior distribution	A statistical distribution of the probability of a random event occurring before the consideration of some data.
Seroconversion	Changing from seronegative to seropositive.
Seronegative	Individual has not previously been exposed to any dengue serotype.
Seropositive	Individual has previously been exposed to any dengue serotype.
Seroprevalence	The proportion of a given population that has been exposed to at least one dengue serotype.
Serotype	A serologically distinguishable strain of a particular pathogen
Sylvatic	A disease is sylvatic if its transmission cycle involves wild animals.
Temporary cross immunity	The mechanism by which infection protects against the immuno-invasion of other pathogens or serotypes.
Vectorial capacity	The ability of the disease vector to spread the pathogen.
Vectorial competency	Susceptibility of the disease vector to the pathogen.

Bibliography

Acevedo, M. A., Prosper, O., Lopiano, K., Ruktanonchai, N., Caughlin, T. T., et al. (2015). Spatial heterogeneity, host movement and mosquito-borne disease transmission. *PLoS ONE*, 10(6):1–15.

Adams, B. and Kapan, D. D. (2009). Man bites mosquito: Understanding the contribution of human movement to vector-borne disease dynamics. *PLoS ONE*, 4(8).

Adams, B., Holmes, E. C., Zhang, C., Mammen, M. P., Nimmannitya, S., et al. (2006). Cross-protective immunity can account for the alternating epidemic pattern of dengue virus serotypes circulating in Bangkok. *Proceedings of the National Academy of Sciences*, 103(38):14234–9.

Aguiar, M., Stollenwerk, N., and Halstead, S. B. (2016). The risks behind Dengvaxia recommendation. *The Lancet Infectious Diseases*, 16(8):882–883.

Aguiar, M., Halstead, S. B., and Stollenwerk, N. (2017). Consider stopping dengvaxia administration without immunological screening. *Expert Review of Vaccines*, 16(4): 301–302.

Alera, M. T., Srikiatkhachorn, A., Velasco, J. M., Tac-An, I. A., Lago, C. B., et al. (2016). Incidence of Dengue Virus Infection in Adults and Children in a Prospective Longitudinal Cohort in the Philippines. *PLOS Neglected Tropical Diseases*, 10(2):e0004337.

Alto, B. W. and Bettinardi, D. (2013). Temperature and dengue virus infection in mosquitoes: Independent effects on the immature and adult stages. *American Journal of Tropical Medicine and Hygiene*, 88(3):497–505.

Alto, B. W. and Juliano, S. A. (2001). Precipitation and temperature effects on populations of Aedes albopictus (Diptera: Culicidae): Implications for range expansion. *Journal of Medical Entomology*, 38(5):646–656.

AnandaRao, R., Swaminathan, S., Fernando, S., Jana, A. M., and Khanna, N. (2006). Recombinant Multiepitope Protein for Early Detection of Dengue Infections. *Clinical and Vaccine Immunology*, 13(1):59–67.

Andersson, N., Nava-Aguilera, E., Arosteguí, J., Morales-Perez, A., Suazo-Laguna, H., et al. (2015). Evidence based community mobilization for dengue prevention in Nicaragua and Mexico (Camino Verde, the Green Way): Cluster randomized controlled trial. *BMJ (Online)*, 351.

Andersson, N., Arostegui, J., Nava-Aguilera, E., Harris, E., and Ledogar, R. J. (2017). Camino Verde (The Green Way): evidence-based community mobilisation for dengue control in Nicaragua and Mexico: feasibility study and study protocol for a randomised controlled trial. *BMC Public Health*, 17(S1):407.

Andrieu, C. and Roberts, G. O. (2009). The pseudo-marginal approach for efficient Monte Carlo computations. *The Annals of Statistics*, 37(2):697–725.

Andrieu, C., Doucet, A., and Holenstein, R. (2010). Particle Markov chain Monte Carlo methods. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 72 (3):269–342.

Aparicio, J. P. and Pascual, M. (2007). Building epidemiological models from R0: an implicit treatment of transmission in networks. *Proceedings of the Royal Society B: Biological Sciences*, 274(1609):505–512.

Aranda, C., Martínez, M. J., Montalvo, T., Eritja, R., Navero-Castillejos, J., et al. (2018). Arbovirus surveillance: first dengue virus detection in local Aedes albopictus mosquitoes in Europe, Catalonia, Spain, 2015. *Eurosurveillance*, 23(47).

Barmak, D. H., Dorso, C. O., Otero, M., and Solari, H. G. (2011). Dengue epidemics and human mobility. *Physical Review E - Statistical, Nonlinear, and Soft Matter Physics*, 84 (1):1–11.

Barmak, D. H., Dorso, C. O., and Otero, M. (2016). Modelling dengue epidemic spreading with human mobility. *Physica A: Statistical Mechanics and its Applications*, 447:129–140.

Barrera, R., Delgado, N., Jiménez, M., Villalobos, I., and Romero, I. (2000). Stratification of a hyperendemic city in hemorrhagic dengue. *Revista panamericana de salud publica = Pan American journal of public health*, 8(4):225–33.

Beaumont, M. A. (2003). Estimation of population growth or decline in genetically monitored populations. *Genetics*, 164(3):1139–1160.

Beck-Johnson, L. M., Nelson, W. A., Paaijmans, K. P., Read, A. F., Thomas, M. B., and Bjørnstad, O. N. (2013). The effect of temperature on Anopheles mosquito population dynamics and the potential for malaria transmission. *PLoS ONE*, 8(11).

Beckett, C. G., Kosasih, H., Faisal, I., Nurhayati, Tan, R., et al. (2005). Early detection of dengue infections using cluster sampling around index cases. *The American journal of tropical medicine and hygiene*, 72(6):777–82.

Bellan, S. E. (2010). The importance of age dependent mortality and the extrinsic incubation period in models of mosquito-borne disease transmission and control. *PLoS ONE*, 5(4).

Bennett, S. N., Holmes, E. C., Chirivella, M., Rodriguez, D. M., Beltran, M., et al. (2003). Selection-driven evolution of emergent dengue virus. *Molecular Biology and Evolution*, 20(10):1650–1658.

Bhamarapravati, N. and Sutee, Y. (2000). Live attenuated tetravalent dengue vaccine. *Vaccine*, 18(SUPPL. 2):44–47.

Bharti, N., Tatem, A. J., Ferrari, M. J., Grais, R. F., Djibo, A., and Grenfell, B. T. (2011). Explaining Seasonal Fluctuations of Measles in Niger Using Nighttime Lights Imagery. *Science*, 334(6061):1424–1427.

Bhatt, S., Gething, P. W., Brady, O. J., Messina, J. P., Farlow, A. W., et al. (2013). The global distribution and burden of dengue. *Nature*, 496(7446):504–507.

Bousema, T., Griffin, J. T., Sauerwein, R. W., Smith, D. L., Churcher, T. S., et al. (2012). Hitting hotspots: Spatial targeting of malaria for control and elimination. *PLoS Medicine*, 9(1):1–7.

Brady, O. J., Gething, P. W., Bhatt, S., Messina, J. P., Brownstein, J. S., et al. (2012). Refining the Global Spatial Limits of Dengue Virus Transmission by Evidence-Based Consensus. *PLoS Neglected Tropical Diseases*, 6(8).

Brady, O. J., Johansson, M. a., Guerra, C. a., Bhatt, S., Golding, N., et al. (2013). Modelling adult Aedes aegypti and Aedes albopictus survival at different temperatures in laboratory and field settings. *Parasites & vectors*, 6:351.

Brady, O. J., Golding, N., Pigott, D. M., Kraemer, M. U. G., Messina, J. P., et al. (2014). Global temperature constraints on Aedes aegypti and Ae. albopictus persistence and competence for dengue virus transmission. *Parasites & vectors*, 7(1):338.

Brand, S. P., Rock, K. S., and Keeling, M. J. (2016). The Interaction between Vector Life History and Short Vector Life in Vector-Borne Disease Transmission and Control. *PLoS Computational Biology*, 12(4):1–21.

Bravo, L., Roque, V. G., Brett, J., Dizon, R., and L'Azou, M. (2014). Epidemiology of Dengue Disease in the Philippines (2000–2011): A Systematic Literature Review. *PLoS Neglected Tropical Diseases*, 8(11).

Breban, R., Vardavas, R., and Blower, S. (2007). Theory versus data: How to calculate R0? *PLoS ONE*, 2(3):0–3.

Britton, T. (2010). Stochastic epidemic models: A survey. *Mathematical Biosciences*, 225 (1):24–35.

Broido, A. D. and Clauset, A. (2019). Scale-free networks are rare. *Nature Communications*, 10(1):1017.

Burke, D. S., Nisalak, A., and Ussery, M. A. (1982). Antibody capture immunoassay detection of Japanese encephalitis virus immunoglobulin M and G antibodies in cerebrospinal fluid. *Journal of Clinical Microbiology*, 16(6):1034–1042.

Cairns, A. J. (1991). Model fitting and projection of the AIDS epidemic. *Mathematical biosciences*, 107(2):451–89.

Caminade, C., Turner, J., Metelmann, S., Hesson, J. C., Blagrove, M. S. C., et al. (2017). Global risk model for vector-borne transmission of Zika virus reveals the role of El Niño 2015. *Proceedings of the National Academy of Sciences*, 114(1):119–124.

Canyon, D. V., Muller, R., and Hii, J. L. (2013). Aedes aegypti disregard humidity-related conditions with adequate nutrition. *Tropical Biomedicine*, 30(1):1–8.

Carbajo, A. E., Schweigmann, N., Curto, S. I., de Garin, A., and Bejaran, R. (2001). Dengue transmission risk maps of Argentina. *Tropical Medicine and International Health*, 6(3):170–183.

Carey, D. (1971). DENGUE VIRUSES FROM FEBRILE PATIENTS IN NIGERIA, 1964-68. *The Lancet*, 297(7690):105–106.

Carrington, L. B., Seifert, S. N., Armijos, M. V., Lambrechts, L., and Scott, T. W. (2013). Reduction of Aedes aegypti vector competence for dengue virus under large temperature fluctuations. *American Journal of Tropical Medicine and Hygiene*, 88(4):689–697.

Carter, R., Mendis, K. N., and Roberts, D. (2000). Spatial targeting of interventions against malaria. *Bulletin of the World Health Organization*, 78(12):1401–11.

Cauchemez, S., Bhattarai, A., Marchbanks, T. L., Fagan, R. P., Ostroff, S., et al. (2011). Role of social networks in shaping disease transmission during a community outbreak of 2009 H1N1 pandemic influenza. *Proceedings of the National Academy of Sciences*, 108 (7):2825–2830.

Cauchemez, S., Besnard, M., Bompard, P., Dub, T., Guillemette-Artur, P., et al. (2016). Association between Zika virus and microcephaly in French Polynesia, 2013-15: A retrospective study. *The Lancet*, 387(10033):2125–2132.

Centers for Disease Control and Prevention (CDC). (2010). Locally acquired Dengue–Key West, Florida, 2009-2010. *MMWR. Morbidity and mortality weekly report*, 59(19):577–81.

Chakravarti, A., Arora, R., and Luxemburger, C. (2012). Fifty years of dengue in India. *Transactions of the Royal Society of Tropical Medicine and Hygiene*, 106(5):273–282.

Chan, M. and Johansson, M. A. (2012). The Incubation Periods of Dengue Viruses. *PLoS ONE*, 7(11):1–7.

Chao, D. L., Halstead, S. B., Halloran, M. E., and Longini, I. M. (2012). Controlling Dengue with Vaccines in Thailand. *PLoS Neglected Tropical Diseases*, 6(10).

Chao, D. L., Longini, I. M., and Halloran, M. E. (2013). The effects of vector movement and distribution in a mathematical model of dengue transmission. *PloS one*, 8(10): e76044.

Chen, M.-J., Lin, C.-Y., Wu, Y.-T., Wu, P.-C., Lung, S.-C., and Su, H.-J. (2012). Effects of Extreme Precipitation to the Distribution of Infectious Diseases in Taiwan, 1994–2008. *PLoS ONE*, 7(6):e34651.

Chowell, G., Hengartner, N. W., Castillo-Chavez, C., Fenimore, P. W., and Hyman, J. M. (2004). The basic reproductive number of Ebola and the effects of public health measures: The cases of Congo and Uganda. *Journal of Theoretical Biology*, 229(1):119–126.

Chowell, G., Diaz-Dueñas, P., Miller, J. C., Alcazar-Velazco, A., Hyman, J. M., et al. (2007). Estimation of the reproduction number of dengue fever from spatial epidemic data. *Mathematical Biosciences*, 208(2):571–589.

Ciota, A. T., Matacchiero, A. C., Kilpatrick, A. M., and Kramer, L. D. (2014). The effect of temperature on life history traits of Culex mosquitoes. *Journal of medical entomology*, 51(1):55–62.

Clancy, D. and O'Neill, P. D. (2008). Bayesian estimation of the basic reproduction number in stochastic epidemic models. *Bayesian Analysis*, 3(4):737–758.

Clements, A. N. and Paterson, G. D. (1981). The Analysis of Mortality and Survival Rates in Wild Populations of Mosquitoes. *The Journal of Applied Ecology*, 18(2):373.

Cobra, C., Rigau-pérez, J. G., Kuno, G., and Vomdam, V. (1995). Symptoms of dengue fever in relation to host immunologic response and virus serotype, puerto rico, 1990-1991. *American Journal of Epidemiology*, 142(11):1204–1211.

Cori, A., Ferguson, N. M., Fraser, C., and Cauchemez, S. (2013). A new framework and software to estimate time-varying reproduction numbers during epidemics. *American Journal of Epidemiology*, 178(9):1505–1512.

Cosner, C., Beier, J. C., Cantrell, R. S., Impoinvil, D., Kapitanski, L., et al. (2009). The effects of human movement on the persistence of vector-borne diseases. *Journal of Theoretical Biology*, 258(4):550–560.

Coudeville, L. and Garnett, G. P. (2012). Transmission Dynamics of the Four Dengue Serotypes in Southern Vietnam and the Potential Impact of Vaccination. *PLoS ONE*, 7 (12).

Cromwell, E. A., Stoddard, S. T., Barker, C. M., Van Rie, A., Messer, W. B., et al. (2017). The relationship between entomological indicators of Aedes aegypti abundance and dengue virus infection. *PLoS Neglected Tropical Diseases*.

Csardi, G. and Nepusz, T. (2006). The igraph software package for complex network research. *InterJournal*, Complex Sy:1695.

Cummings, D. A., Iamsirithaworn, S., Lessler, J. T., McDermott, A., Prasanthong, R., et al. (2009). The impact of the demographic transition on dengue in Thailand: Insights from a statistical analysis and mathematical modeling. *PLoS Medicine*, 6(9).

Cummings, D. A. T., Schwartz, I. B., Billings, L., Shaw, L. B., and Burke, D. S. (2005). Dynamic effects of antibody-dependent enhancement on the fitness of viruses. *Proceedings of the National Academy of Sciences*, 102(42):15259–15264.

Cummins, B., Cortez, R., Foppa, I. M., Walbeck, J., and Hyman, J. M. (2012). A Spatial Model of Mosquito Host-Seeking Behavior. *PLoS Computational Biology*, 8(5):e1002500.

Cuong, H. Q., Hien, N. T., Duong, T. N., Phong, T. V., Cam, N. N., et al. (2011). Quantifying the emergence of dengue in Hanoi, Vietnam: 1998-2009. *PLoS Neglected Tropical Diseases*, 5(9):1–7.

Cuong, H. Q., Vu, N. T., Cazelles, B., Boni, M. F., Thai, K. T., et al. (2013). Spatiotemporal Dynamics of Dengue Epidemics, Southern Vietnam. *Emerging Infectious Diseases*, 19 (6):945–953.

Da Cruz Ferreira, D. A., Degener, C. M., De Almeida Marques-Toledo, C., Bendati, M. M., Fetzer, L. O., et al. (2017). Meteorological variables and mosquito monitoring are good predictors for infestation trends of Aedes aegypti, the vector of dengue, chikungunya and Zika. *Parasites and Vectors*, 10(1):1–11.

Dawes, E. J., Churcher, T. S., Zhuang, S., Sinden, R. E., and Basáñez, M.-G. (2009). Anopheles mortality is both age- and Plasmodium-density dependent: implications for malaria transmission. *Malaria Journal*, 8(1):228.

De Simone, T. S., Nogueira, R. M. R., Araújo, E. S. M., Guimarães, F. R., Santos, F. B., et al. (2004). Dengue virus surveillance: The co-circulation of DENV-1, DENV-2 and DENV-3 in the State of Rio de Janeiro, Brazil. *Transactions of the Royal Society of Tropical Medicine and Hygiene*, 98(9):553–562.

Dean-Ben, X. L., Ozbek, A., and Razansky, D. (2013). Volumetric real-time tracking of peripheral human vasculature with GPU-accelerated three-dimensional optoacoustic tomography. *IEEE Transactions on Medical Imaging*, 32(11):2050–2055.

Dejnirattisai, W., Jumnainsong, A., Onsirisakul, N., Fitton, P., Vasanawathana, S., et al. (2010). Cross-Reacting Antibodies Enhance Dengue Virus Infection in Humans. *Science*, 328(5979):745–748.

Dejnirattisai, W., Supasa, P., Wongwiwat, W., Rouvinski, A., Barba-Spaeth, G., et al.

(2016). Dengue virus sero-cross-reactivity drives antibody-dependent enhancement of infection with zika virus. *Nature Immunology*, 17(9):1102–1108.

Delatte, H., Desvars, A., Bouétard, A., Bord, S., Gimonneau, G., et al. (2010). Blood-Feeding Behavior of Aedes albopictus , a Vector of Chikungunya on La Réunion. *Vector-Borne and Zoonotic Diseases*, 10(3):249–258.

Depradine, C. A. and Lovell, E. H. (2004). Climatological variables and the incidence of Dengue fever in Barbados. *International Journal of Environmental Health Research*, 14 (6):429–441.

Descloux, E., Mangeas, M., Menkes, C. E., Lengaigne, M., Leroy, A., et al. (2012). Climate-Based Models for Understanding and Forecasting Dengue Epidemics. *PLoS Neglected Tropical Diseases*, 6(2):e1470.

Diallo, M., Sall, A. A., Moncayo, A. C., Ba, Y., Fernandez, Z., et al. (2005). Potential role of sylvatic and domestic African mosquito species in dengue emergence. *The American journal of tropical medicine and hygiene*, 73(2):445–9.

Diallo, M., Ba, Y., Faye, O., Soumare, M. L., Dia, I., and Sall, A. A. (2008). Vector competence of Aedes aegypti populations from Senegal for sylvatic and epidemic dengue 2 virus isolated in West Africa. *Transactions of the Royal Society of Tropical Medicine and Hygiene*, 102(5):493–498.

Díaz-Quijano, F. A. and Waldman, E. A. (2012). Factors associated with dengue mortality in Latin America and the Caribbean, 1995-2009: An ecological study. *American Journal of Tropical Medicine and Hygiene*, 86(2):328–334.

Dick, O. B., San Martín, J. L., Montoya, R. H., Del Diego, J., Zambrano, B., and Dayan, G. H. (2012). Review: The history of dengue outbreaks in the Americas. *American Journal of Tropical Medicine and Hygiene*, 87(4):584–593.

Diekmann, O., Heesterbeek, J. A. P., and Roberts, M. G. (2010). The construction of next-generation matrices for compartmental epidemic models. *Journal of The Royal Society Interface*, 7(47):873–885.

Dietz, K. (1993). The estimation of the basic reproduction number for infectious diseases. *Statistical Methods in Medical Research*, 2(1):23–41.

Doucet, A., Pitt, M. K., Deligiannidis, G., and Kohn, R. (2015). Efficient implementation of Markov chain Monte Carlo when using an unbiased likelihood estimator. *Biometrika*, 102(2):295–313.

Duffy, M. R., Chen, T.-H., Hancock, W. T., Powers, A. M., Kool, J. L., et al. (2009). Zika Virus Outbreak on Yap Island, Federated States of Micronesia. *New England Journal of Medicine*, 360(24):2536–2543.

Duong, V., Lambrechts, L., Paul, R. E., Ly, S., Lay, R. S., et al. (2015). Asymptomatic humans transmit dengue virus to mosquitoes. *Proceedings of the National Academy of Sciences*, 112(47):14688–14693.

Effler, P. V., Pang, L., Kitsutani, P., Vorndam, V., Nakata, M., et al. (2005). Dengue fever, Hawaii, 2001-2002. *Emerging infectious diseases*, 11(5):742–9.

Endy, T. P., Chunsuttiwat, S., Nisalak, A., Libraty, D. H., Green, S., et al. (2002). Epidemiology of inapparent and symptomatic acute dengue virus infection: A prospective study of primary school children in Kamphaeng Phet, Thailand. *American Journal of Epidemiology*, 156(1):40–51.

Endy, T. P., Anderson, K. B., Nisalak, A., Yoon, I.-K., Green, S., et al. (2011). Determinants of Inapparent and Symptomatic Dengue Infection in a Prospective Study of Primary School Children in Kamphaeng Phet, Thailand. *PLoS Neglected Tropical Diseases*, 5(3): e975.

Erdős, P. and Rényi, A. (1959). On random graphs I. *Publ. Math. Debrecen*, 6:290–297.

Erlanger, T. E., Weiss, S., Keiser, J., Utzinger, J., and Wiedenmayer, K. (2009). Past, present, and future of Japanese encephalitis. *Emerging Infectious Diseases*, 15(1):1–7.

Espinoza-Gómez, F., Hernández-Suárez, C. M., and Coll-Cárdenas, R. (2002). Educational campaign versus malathion spraying for the control of Aedes aegypti in Colima, Mexico. *Journal of epidemiology and community health*, 56(2):148–52.

European Medicines Agency. Dengvaxia, (2018). https://www.ema.europa.eu/en/medicines/human/EPAR/dengvaxia. [Accessed on: 2019-03-27].

Falcón-Lezama, J. A., Santos-Luna, R., Román-Pérez, S., Martínez-Vega, R. A., Herrera-Valdez, M. A., et al. (2017). Analysis of spatial mobility in subjects from a Dengue endemic urban locality in Morelos State, Mexico. *PLoS One*, 12(2):e0172313.

Faria, N. R., Quick, J., Claro, I., Thézé, J., de Jesus, J. G., et al. (2017). Establishment and cryptic transmission of Zika virus in Brazil and the Americas. *Nature*, 546(7658): 406–410.

Farrugia, J.-p., Horain, P., Guehenneux, E., Alusse, Y., Int, G. E. T., et al. (2006). Gpucv : a Framework for Image Processing Acceleration. *Image (Rochester, N.Y.)*, 585–588.

Fatima, K. and Syed, N. I. (2018). Dengvaxia controversy: impact on vaccine hesitancy. *Journal of Global Health*, 8(2):8–10.

Favier, C., Schmit, D., Muller-Graf, C. D., Cazelles, B., Degallier, N., et al. (2005). Influence of spatial heterogeneity on an emerging infectious disease: the case of dengue epidemics. *Proceedings of the Royal Society B: Biological Sciences*, 272(1568):1171–1177.

Favier, C., Degallier, N., Rosa-Freitas, M. G., Boulanger, J. P., Costa Lima, J. R., et al. (2006). Early determination of the reproductive number for vector-borne diseases: The case of dengue in Brazil. *Tropical Medicine and International Health*, 11(3):332–340.

Felix, A. C., Souza, N. C., Figueiredo, W. M., Costa, A. A., Inenami, M., et al. (2017). Cross reactivity of commercial anti-dengue immunoassays in patients with acute Zika virus infection. *Journal of Medical Virology*, 89(8):1477–1479.

Ferguson, N., Anderson, R., and Gupta, S. (1999a). The effect of antibody-dependent enhancement on the transmission dynamics and persistence of multiple-strain pathogens. *Proceedings of the National Academy of Sciences*, 96(2):790–794.

Ferguson, N. M., Donnelly, C. A., and Anderson, R. M. (1999b). Transmission dynamics and epidemiology of dengue: insights from age-stratified sero-prevalence surveys. *Philosophical Transactions of the Royal Society B: Biological Sciences*, 354(1384):757–768.

Ferguson, N. M., Rodríguez-Barraquer, I., Dorigatti, I., Mier-y Teran-Romero, L., Laydon, D. J., and Cummings, D. A. T. (2016). Benefits and risks of the Sanofi-Pasteur dengue vaccine: Modeling optimal deployment. *Science*, 353(6303):1033 LP – 1036.

Fernandez-Garcia, M. D., Mazzon, M., Jacobs, M., and Amara, A. (2009). Pathogenesis of Flavivirus Infections: Using and Abusing the Host Cell. *Cell Host and Microbe*, 5(4): 318–328.

Flasche, S., Jit, M., Rodríguez-Barraquer, I., Coudeville, L., Recker, M., et al. (2016). The Long-Term Safety, Public Health Impact, and Cost-Effectiveness of Routine Vaccination with a Recombinant, Live-Attenuated Dengue Vaccine (Dengvaxia): A Model Comparison Study. *PLoS Medicine*, 13(11):e1002181.

Focks, D. A., Haile, D. G., Daniels, E., and Mount, G. A. (1993). Dynamic life table model for Aedes aegypti (Diptera: Culicidae): analysis of the literature and model development. *J Med Entomol*, 30(6):1003–1017.

Focks, D. A., Daniels, E., Haile, D. G., and Keesling, J. E. (1995). A simulation model of the epidemiology of urban dengue fever: Literature analysis, model development, preliminary validation, and samples of simulation results. *American Journal of Tropical Medicine and Hygiene*, 53(5):489–506.

Forbes, C., Evans, M., Hastings, N., and Peacock, B. *Statistical Distributions*. John Wiley & Sons, Inc., Hoboken, NJ, USA, 4th edition, (2010).

Franco, L., Palacios, G., Martinez, J. A., Vázquez, A., Savji, N., et al. (2011). First Report of Sylvatic DENV-2-Associated Dengue Hemorrhagic Fever in West Africa. *PLoS Neglected Tropical Diseases*, 5(8):e1251.

Galvão Filho, A. R., Martins De Paula, L. C., Coelho, C. J., De Lima, T. W., and Da Silva Soares, A. (2016). CUDA parallel programming for simulation of epidemiological models based on individuals. *Mathematical Methods in the Applied Sciences*, 39(3):405–411.

Gardner, L. and Sarkar, S. (2013). A Global Airport-Based Risk Model for the Spread of Dengue Infection via the Air Transport Network. *PLoS ONE*, 8(8).

Garg, S., Chakravarti, A., Singh, R., Masthi, N. R., Goyal, R. C., et al. (2017). Dengue serotype-specific seroprevalence among 5- to 10-year-old children in India: a community-based cross-sectional study. *International Journal of Infectious Diseases*, 54:25–30.

George, L., Lenhart, A., Toledo, J., Lazaro, A., Han, W. W., et al. (2015). Community-Effectiveness of Temephos for Dengue Vector Control: A Systematic Literature Review. *PLoS Neglected Tropical Diseases*, 9(9).

Gibbons, R. V., Kalanarooj, S., Jarman, R. G., Nisalak, A., Vaughn, D. W., et al. (2007). Analysis of repeat hospital admissions for dengue to estimate the frequency of third or fourth dengue infections resulting in admissions and dengue hemorrhagic fever, and serotype sequences. *The American journal of tropical medicine and hygiene*, 77(5):910–3.

Gjenero-Margan, I., Aleraj, B., Krajcar, D., Lesnikar, V., Klobučar, A., et al. (2011). Autochthonous dengue fever in Croatia, August-September 2010. *Euro surveillance : bulletin Europeen sur les maladies transmissibles = European communicable disease bulletin*, 16(9):1–4.

Goindin, D., Delannay, C., Ramdini, C., Gustave, J., and Fouque, F. (2015). Parity and longevity of aedes aegypti according to temperatures in controlled conditions and consequences on dengue transmission risks. *PLoS ONE*, 10(8):1–21.

Gratz, N. G. (2004). Critical review of the vector status of Aedes albopictus. *Medical and Veterinary Entomology*, 18(3):215–227.

Grenfell, B. T., Pybus, O. G., Gog, J. R., Wood, J. L. N., Daly, J. M., et al. (2004). Unifying the epidemiological and evolutionary dynamics of pathogens. *Science (New York, N.Y.)*, 303(5656):327–32.

Griebel, M. and Zaspel, P. (2010). A multi-GPU accelerated solver for the three-dimensional two-phase incompressible Navier-Stokes equations. *Computer Science - Research and Development*, 25(1-2):65–73.

Gubler, D. J. (1998). Dengue and Dengue Hemorrhagic Fever. *Clinical Microbiology Reviews*, 11(3):480–496.

Gubler, D. J. (2002). Epidemic dengue/dengue hemorrhagic fever as a public health, social and economic problem in the 21st century. *Trends in microbiology*, 10(2):100–3.

Gubler, D. J. (2006). Dengue/dengue haemorrhagic fever: history and current status. *Novartis Foundation symposium*, 277:3–16; discussion 16–22, 71–3, 251–3.

Gubler, D. J. (2009). Vector-borne diseases. *Revue scientifique et technique (International Office of Epizootics)*, 28(2):583–8.

Gubler, D. J. (2011). Dengue, Urbanization and Globalization: The Unholy Trinity of the 21st Century. *Tropical Medicine and Health*, 39(4SUPPLEMENT):S3–S11.

Gubler, D. J. and Clark, G. G. (1996). Community involvement in the control of Aedes aegypti. *Acta Tropica*, 61(2):169–179.

Guha-Sapir, D., Schimmer, B., Guzman, M., Kouri, G., Valdes, L., et al. (2005). Dengue fever: new paradigms for a changing epidemiology. *Emerging Themes in Epidemiology*, 2(1):1.

Guzman, M. G. and Harris, E. (2015). Dengue. *The Lancet*, 385(9966):453–465.

Guzmán, M. G. and Kourí, G. (2004). Dengue diagnosis, advances and challenges. *International Journal of Infectious Diseases*, 8(2):69–80.

Guzman, M. G., Alvarez, A., Vazquez, S., Alvarez, M., Rosario, D., et al. (2012). Epidemiological studies on dengue virus type 3 in Playa municipality, Havana, Cuba, 2001-2002. *International Journal of Infectious Diseases*, 16(3):e199–e204.

Haby, M. M., Pinart, M., Elias, V., and Reveiz, L. (2018). Prevalence of asymptomatic Zika virus infection: a systematic review. *Bulletin of the World Health Organization*, 96 (6):402–413D.

Halstead, S. B. (1980). Dengue haemorrhagic fever–a public health problem and a field for research. *Bulletin of the World Health Organization*, 58(1):1–21.

Halstead, S. B. (2003). Neutralization and Antibody-Dependent Enhancement of Dengue Viruses. *Advances in Virus Research*, 60:421–467.

Hancock, P. A. and Godfray, H. C. J. (2007). Application of the lumped age-class technique to studying the dynamics of malaria-mosquito-human interactions. *Malaria Journal*, 6.

Harrington, L. C., Edman, J. D., and Scott, T. W. (2001). Why do female Aedes aegypti (Diptera: Culicidae) feed preferentially and frequently on human blood? *Journal of Medical Entomology*, 38(3):411–422.

Harrington, L. C., Scott, T. W., Lerdthusnee, K., Coleman, R. C., Costero, A., et al. (2005). Dispersal of the dengue vector Aedes aegypti within and between rural communities. *The American journal of tropical medicine and hygiene*, 72(2):209–20.

Harrington, L. C., Jones, J. J., Kitthawee, S., Sithiprasasna, R., Edman, J. D., and Scott, T. W. (2008). Age-Dependent Survival of the Dengue Vector Aedes aegypti (Diptera: Culicidae) Demonstrated by Simultaneous Release-Recapture of Different Age Cohorts. *Journal of Medical Entomology*, 45(2):307–313.

Harris, M. J., Caldwell, J. M., and Mordecai, E. A. (2018). Climate drives spatial variation in Zika epidemics in Latin America. *bioRxiv*.

Hartfield, M. and Alizon, S. (2013). Introducing the Outbreak Threshold in Epidemiology. *PLoS Pathogens*, 9(6):9–12.

Hastings, W. K. (1970). Monte Carlo Sampling Methods Using Markov Chains and Their Applications. *Biometrika*, 57(1):97.

Heesterbeek, J. A. P. and Dietz, K. (1996). The concept of R0 in epidemic theory. *Statistica Neerlandica*, 50(1):89–110.

Hernández Pérez, F. E., Mukhadiyev, N., Xu, X., Sow, A., Lee, B. J., et al. (2018). Direct numerical simulations of reacting flows with detailed chemistry using many-core/GPU acceleration. *Computers and Fluids*, 173:73–79.

Hidayat, R., Spataro, D., Giorgio, E. D., Spataro, W., and DAmbrosio, D. Multi-agent System with Multiple Group Modelling for Bird Flocking on GPU. In *2016 24th Euromicro International Conference on Parallel, Distributed, and Network-Based Processing (PDP)*, 680–685. IEEE, (2016).

Hladish, T. J., Pearson, C. A. B., Chao, D. L., Rojas, D. P., Recchia, G. L., et al. (2016). Projected Impact of Dengue Vaccination in Yucatán, Mexico. *PLoS Neglected Tropical Diseases*, 10(5):1–19.

Holmes, E. C. (2003). Error thresholds and the constraints to RNA virus evolution. *Trends in Microbiology*, 11(12):543–546.

Hu, W., Tong, S., Mengersen, K., and Oldenburg, B. (2006). Rainfall, mosquito density and the transmission of Ross River virus: A time-series forecasting model. *Ecological Modelling*, 196(3-4):505–514.

Huber, J. H., Johnston, G. L., Greenhouse, B., Smith, D. L., and Perkins, T. A. (2016). Quantitative, model-based estimates of variability in the generation and serial intervals of Plasmodium falciparum malaria. *Malaria Journal*, 15(1):490.

Huber, J. H., Childs, M. L., Caldwell, J. M., and Mordecai, E. A. (2018). Seasonal temperature variation influences climate suitability for dengue, chikungunya, and Zika transmission. *PLoS Neglected Tropical Diseases*, 12(5):1–20.

Hugo, L. E., Jeffery, J. A., Trewin, B. J., Wockner, L. F., Thi Yen, N., et al. (2014). Adult Survivorship of the Dengue Mosquito Aedes aegypti Varies Seasonally in Central Vietnam. *PLoS Neglected Tropical Diseases*, 8(2).

Innis, B. L., Nisalak, A., Nimmannitya, S., Kusalerdchariya, S., Chongswasdi, V., et al. (1989). An enzyme-linked immunosorbent assay to characterize dengue infections where dengue and Japanese encephalitis co-circulate. *The American journal of tropical medicine and hygiene*, 40(4):418–27.

Instituto Brasileiro de Geografia e Estatistica. Estimativas de população publicadas no DOU, (2018). https://www.ibge.gov.br/estatisticas-novoportal/sociais/populacao/9103-estimativas-de-populacao.html?=&t=resultados. [Accessed on: 2019-03-22].

Irvine, M. A. and Hollingsworth, T. D. (2018). Kernel-density estimation and approximate Bayesian computation for flexible epidemiological model fitting in Python. *Epidemics*, 25(February 2018):80–88.

Jaimes-Dueñez, J., Arboleda, S., Triana-Chávez, O., and Gómez-Palacio, A. (2015). Spatio-Temporal Distribution of Aedes aegypti (Diptera: Culicidae) Mitochondrial Lineages in Cities with Distinct Dengue Incidence Rates Suggests Complex Population Dynamics of the Dengue Vector in Colombia. *PLoS Neglected Tropical Diseases*, 9(4):1–21.

Johansson, M. A., Dominici, F., and Glass, G. E. (2009). Local and global effects of climate on dengue transmission in Puerto Rico. *PLoS Neglected Tropical Diseases*, 3(2):1–5.

Johansson, M. A., Arana-Vizcarrondo, N., Biggerstaff, B. J., and Staples, J. E. (2010). Incubation periods of yellow fever virus. *American Journal of Tropical Medicine and Hygiene*, 83(1):183–188.

Jupp, P. G., Kemp, A., Grobbelaar, A., Leman, P., Burt, F. J., et al. (2002). The 2000 epidemic of Rift Valley fever in Saudi Arabia: Mosquito vector studies. *Medical and Veterinary Entomology*, 16(3):245–252.

Kalayanarooj, S. (1999). Standardized Clinical Management: Evidence of Reduction of Dengue Haemorrhagic Fever Case-Fatality Rate in Thailand. *Dengue Bulletin*, 23:10–17.

Kang, C., Ma, X., Tong, D., and Liu, Y. (2012). Intra-urban human mobility patterns: An urban morphology perspective. *Physica A: Statistical Mechanics and its Applications*, 391(4):1702–1717.

Karl, S., Halder, N., Kelso, J. K., Ritchie, S. A., and Milne, G. J. (2014). A spatial simulation model for dengue virus infection in urban areas. *BMC Infectious Diseases*, 14 (1):447.

Karyanti, M. R., Uiterwaal, C. S. P. M., Kusriastuti, R., Hadinegoro, S. R., Rovers, M. M., et al. (2014). The changing incidence of Dengue Haemorrhagic Fever in Indonesia: a 45-year registry-based analysis. *BMC Infectious Diseases*, 14(1):412.

Katzelnick, L. C., Montoya, M., Gresh, L., Balmaseda, A., and Harris, E. (2016). Neutralizing antibody titers against dengue virus correlate with protection from symptomatic infection in a longitudinal cohort. *Proceedings of the National Academy of Sciences*, 113 (3):728–733.

Katzelnick, L. C., Coloma, J., and Harris, E. (2017a). Dengue: knowledge gaps, unmet needs, and research priorities. *The Lancet Infectious Diseases*, 3099(16):1–13.

Katzelnick, L. C., Gresh, L., Halloran, M. E., Mercado, J. C., Kuan, G., et al. (2017b). Antibody-dependent enhancement of severe dengue disease in humans. *Science*, 358 (6365):929–932.

Kliks, S. C., Nisalak, A., Brandt, W. E., Wahl, L., and Burke, D. S. (1989). Antibody-dependent enhancement of dengue virus growth in human monocytes as a risk factor for dengue hemorrhagic fever. *The American journal of tropical medicine and hygiene*, 40 (4):444–451.

Ko, Y. C. (1989). Epidemiology of dengue fever in Taiwan. *The Kaohsiung journal of medical sciences*, 5(1):1–11.

Koenraadt, C. J. M. and Harrington, L. C. (2008). Flushing effect of rain on container-inhabiting mosquitoes Aedes aegypti and Culex pipiens (Diptera: Culicidae). *Journal of medical entomology*, 45(1):28–35.

Kong, L., Xu, C., Mu, P., Li, J., Qiu, S., and Wu, H. (2018). Risk factors spatial-temporal detection for dengue fever in Guangzhou. *Epidemiology and Infection*, 1–9.

Kraemer, M. U. G., Perkins, T. A., Cummings, D. A. T., Zakar, R., Hay, S. I., et al. (2015). Big city, small world: density, contact rates, and transmission of dengue across Pakistan. *Journal of the Royal Society, Interface / the Royal Society*, 12(111):20150468.

Krow-Lucal, E. R., Biggerstaff, B. J., and Staples, J. E. (2017). Estimated Incubation Period for Zika Virus Disease. *Emerging Infectious Diseases*, 23(5):841–845.

Kucharski, A. J., Funk, S., Eggo, R. M., Mallet, H.-P., Edmunds, W. J., and Nilles, E. J. (2016). Transmission Dynamics of Zika Virus in Island Populations: A Modelling Analysis of the 2013-14 French Polynesia Outbreak. *PLoS neglected tropical diseases*, 10 (5):e0004726.

Kucharski, A. J., Kama, M., Watson, C. H., Aubry, M., Funk, S., et al. (2018). Using paired serology and surveillance data to quantify dengue transmission and control during a large outbreak in Fiji. *eLife*, 7:1–26.

Kuno, G. (1995). Review of the factors modulating dengue transmission. *Epidemiologic reviews*, 17(2):321–335.

Kutter, O., Shams, R., and Navab, N. (2009). Visualization and GPU-accelerated simulation of medical ultrasound from CT images. *Computer Methods and Programs in Biomedicine*, 94(3):250–266.

Kyle, J. L. and Harris, E. (2008). Global spread and persistence of dengue. *Annual review of microbiology*, 62:71–92.

Ladhani, S. N., O'Connor, C., Kirkbride, H., Brooks, T., and Morgan, D. (2016). Outbreak of Zika virus disease in the Americas and the association with microcephaly, congenital malformations and Guillain–Barré syndrome. *Archives of Disease in Childhood*, 101(7): 600–602.

Ladner, J., Rodrigues, M., Davis, B., Besson, M. H., Audureau, E., and Saba, J. (2017). Societal impact of dengue outbreaks: Stakeholder perceptions and related implications. A qualitative study in Brazil, 2015. *PLoS Neglected Tropical Diseases*, 11(3):1–16.

Lai, S., Huang, Z., Zhou, H., Anders, K. L., Perkins, T. A., et al. (2015). The changing epidemiology of dengue in China, 1990-2014: a descriptive analysis of 25 years of nationwide surveillance data. *BMC medicine*, 13(1):100.

Lai, S., Johansson, M. A., Yin, W., Wardrop, N. A., van Panhuis, W. G., et al. (2018). Seasonal and interannual risks of dengue introduction from South-East Asia into China, 2005-2015. *PLOS Neglected Tropical Diseases*, 12(11):e0006743.

Lambert, B., Sikulu-Lord, M. T., Mayagaya, V. S., Devine, G., Dowell, F., and Churcher, T. S. (2018). Monitoring the Age of Mosquito Populations Using Near-Infrared Spectroscopy. *Scientific Reports*, 8(1):1–9.

Lambrechts, L., Paaijmans, K. P., Fansiri, T., Carrington, L. B., Kramer, L. D., et al. (2011). Impact of daily temperature fl uctuations on dengue virus transmission by Aedes aegypti. *Proceedings of the National Academy of Sciences of the United States of America*, 108(18):1–6.

Lanciotti, R. S., Calisher, C. H., Gubler, D. J., Chang, G.-J., and Vorndamt, A. V. (1992). Rapid Detection and Typing of Dengue Viruses from Clinical Samples by Using Reverse Transcriptase-Polymerase Chain Reaction. *Journal of Clinical Microbiology*, 30 (3):545–551.

Lauer, S. A., Sakrejda, K., Ray, E. L., Keegan, L. T., Bi, Q., et al. (2018). Prospective forecasts of annual dengue hemorrhagic fever incidence in Thailand, 2010–2014. *Proceedings of the National Academy of Sciences*, 115(10):201714457.

Lazaro, A., Han, W. W., Manrique-Saide, P., George, L., Velayudhan, R., et al. (2015). Community effectiveness of copepods for dengue vector control: Systematic review. *Tropical Medicine and International Health*, 20(6):685–706.

Le Grand, S., Götz, A. W., and Walker, R. C. (2013). SPFP: Speed without compromise - A mixed precision model for GPU accelerated molecular dynamics simulations. *Computer Physics Communications*, 184(2):374–380.

Lee, H., Kim, J. E., Lee, S., and Lee, C. H. (2018). Potential effects of climate change on dengue transmission dynamics in Korea. *PLoS ONE*, 13(6):1–23.

Lemey, P., Rambaut, A., Bedford, T., Faria, N., Bielejec, F., et al. (2014). Unifying Viral Genetics and Human Transportation Data to Predict the Global Transmission Dynamics of Human Influenza H3N2. *PLoS Pathogens*, 10(2).

Li, C. F., Lim, T. W., Han, L. L., and Fang, R. (1985). Rainfall, abundance of Aedes aegypti and dengue infection in Selangor, Malaysia. *The Southeast Asian Journal of Tropical Medicine and Public Health*, 16(4):560–8.

Li, H., Kolpas, A., Petzold, L., and Moehlis, J. (2009). Parallel simulation for a fish schooling

model on a general-purpose graphics processing unit. *Concurrency and Computation: Practice and Experience*, 21(6):725–737.

Li, M., Dushoff, J., and Bolker, B. M. (2018). Fitting mechanistic epidemic models to data: A comparison of simple Markov chain Monte Carlo approaches. *Statistical Methods in Medical Research*, 27(7):1956–1967.

Li, P., Luo, Y., Zhang, N., and Cao, Y. (2015). HeteroSpark: A heterogeneous CPU/GPU Spark platform for machine learning algorithms. *Proceedings of the 2015 IEEE International Conference on Networking, Architecture and Storage, NAS 2015*, 347–348.

Li, R., Xu, L., Bjørnstad, O. N., Liu, K., Song, T., et al. (2019). Climate-driven variation in mosquito density predicts the spatiotemporal dynamics of dengue. *Proceedings of the National Academy of Sciences*, 116(9):3624–3629.

Lin, C. H. and Wen, T. H. (2011). Using geographically weighted regression (GWR) to explorespatial varying relationships of immature mosquitoes andhuman densities with the incidence of dengue. *International Journal of Environmental Research and Public Health*, 8(7):2798–2815.

Lin, H., Liu, T., Song, T., Lin, L., Xiao, J., et al. (2016). Community Involvement in Dengue Outbreak Control: An Integrated Rigorous Intervention Strategy. *PLoS Neglected Tropical Diseases*, 10(8):1–10.

Lindenbach, B. D. and Rice, C. M. (2003). Molecular biology of flaviviruses. *Advances in virus research*, 59:23–61.

Lipsitch, M., Cohen, T., Cooper, B., Robins, J. M., Ma, S., et al. (2003). Transmission dynamics and control of severe acute respiratory syndrome. *Science (New York, N.Y.)*, 300(5627):1966–70.

Liu, Y., San Liang, X., and Weisberg, R. H. (2007). Rectification of the Bias in the Wavelet Power Spectrum. *Journal of Atmospheric and Oceanic Technology*, 24(12):2093–2102.

Liu, Z., Zhang, Z., Lai, Z., Zhou, T., Jia, Z., et al. (2017). Temperature Increase Enhances Aedes albopictus Competence to Transmit Dengue Virus. *Frontiers in Microbiology*, 8 (December):1–7.

Liu-Helmersson, J., Quam, M., Wilder-Smith, A., Stenlund, H., Ebi, K., et al. (2016). Climate Change and Aedes Vectors: 21st Century Projections for Dengue Transmission in Europe. *EBioMedicine*, 7:267–277.

Lloyd, A. L., Kitron, U., Perkins, A., Vazquez Prokopec, G., Waller, L., et al. (2017). The Basic Reproductive Number for Disease Systems with Multiple Coupled Heterogeneities. *bioRxiv*.

Lloyd, L. S., Winch, P., Ortega-Canto, J., and Kendall, C. (1992). Results of a community-based Aedes aegypti control program in Merida, Yucatan, Mexico. *The American journal of tropical medicine and hygiene*, 46(6):635–42.

Lourenço, J. and Recker, M. (2013). Natural, Persistent Oscillations in a Spatial Multi-Strain Disease System with Application to Dengue. *PLoS Computational Biology*, 9 (10).

Lourenço, J. and Recker, M. (2014). The 2012 Madeira dengue outbreak: epidemiological determinants and future epidemic potential. *PLoS neglected tropical diseases*, 8(8):e3083.

Lourenço, J. and Recker, M. (2016). Dengue serotype immune-interactions and their consequences for vaccine impact predictions. *Epidemics*, 16:40–48.

Lourenço, J., de Lima, M. M., Faria, N. R., Walker, A., Kraemer, M. U. G., et al. (2017). Epidemiological and ecological determinants of Zika virus transmission in an urban setting. *eLife*.

Lourenço, J., Monteiro, M., Tomás, T., Monteiro Rodrigues, J., Pybus, O., and Rodrigues Faria, N. (2018a). Epidemiology of the Zika Virus Outbreak in the Cabo Verde Islands, West Africa. *PLoS Currents*, 85.

Lourenço, J., Tennant, W., Faria, N. R., Walker, A., Gupta, S., and Recker, M. (2018b). Challenges in dengue research: A computational perspective. *Evolutionary Applications*, 11(4):516–533.

Lowe, R., Chirombo, J., and Tompkins, A. M. (2013). Relative importance of climatic, geographic and socio-economic determinants of malaria in Malawi. *Malaria Journal*, 12 (1).

Lowe, R., Barcellos, C., Coelho, C. A., Bailey, T. C., Coelho, G. E., et al. (2014). Dengue outlook for the World Cup in Brazil: An early warning model framework driven by real-time seasonal climate forecasts. *The Lancet Infectious Diseases*, 14(7):619–626.

Lowe, R., Cazelles, B., Paul, R., and Rodó, X. (2016). Quantifying the added value of climate information in a spatio-temporal dengue model. *Stochastic Environmental Research and Risk Assessment*, 30(8):2067–2078.

Lowe, R., Stewart-Ibarra, A. M., Petrova, D., García-Díez, M., Borbor-Cordova, M. J., et al. (2017). Climate services for health: predicting the evolution of the 2016 dengue season in Machala, Ecuador. *The Lancet Planetary Health*, 1(4):e142–e151.

Luh, D. L., Liu, C. C., Luo, Y. R., and Chen, S. C. (2018). Economic cost and burden of dengue during epidemics and non-epidemic years in Taiwan. *Journal of Infection and Public Health*, 11(2):215–223.

Lyons, C. L., Coetzee, M., and Chown, S. L. (2013). Stable and fluctuating temperature effects on the development rate and survival of two malaria vectors, Anopheles arabiensis and Anopheles funestus. *Parasites and Vectors*, 6(1):1–9.

MacDonald, G. (1952). The analysis of the sporozoite rate. *Tropical diseases bulletin*, 49 (6):569–86.

Maciel-de Freitas, R., Codeço, C. T., and Lourenço-de Oliveira, R. (2007). Daily survival rates and dispersal of Aedes aegypti females in Rio de Janeiro, Brazil. *The American journal of tropical medicine and hygiene*, 76(4):659–65.

Manore, C. A., Hickmann, K. S., Hyman, J. M., Foppa, I. M., Davis, J. K., et al. (2014). A network-patch methodology for adapting agent-based models for directly transmitted disease to mosquito-borne disease. *Journal of Biological Dynamics*, 9(1):52–72.

Marinho, R. A., Beserra, E. B., Bezerra-Gusmão, M. A., Porto, V. d. S., Olinda, R. A., and Dos Santos, C. A. C. (2016). Effects of temperature on the life cycle, expansion, and dispersion of Aedes aegypti (Diptera: Culicidae) in three cities in Paraiba, Brazil. *Journal of vector ecology : journal of the Society for Vector Ecology*, 41(1):1–10.

Martínez-Bello, D. A., López-Quílez, A., and Torres-Prieto, A. (2017). Bayesian dynamic modeling of time series of dengue disease case counts. *PLoS Neglected Tropical Diseases*, 11(7):1–19.

Martínez-Bello, D. A., López-Quílez, A., and Torres Prieto, A. (2018). Spatio-temporal modeling of zika and dengue infections within Colombia. *International Journal of Environmental Research and Public Health*, 15(7).

Massad, E., Coutinho, F. a., Burattini, M. N., and Lopez, L. F. (1996). The risk of yellow fever in a dengue-infested area. *Transactions of the Royal Society of Tropical Medicine and Hygiene*, 95(4):370–374.

Massad, E., Ma, S., Chen, M., Struchiner, C. J., Stollenwerk, N., and Aguiar, M. (2008). Scale-free network of a dengue epidemic. *Applied Mathematics and Computation*, 195(2): 376–381.

Massad, E., Rocklov, J., and Wilder-Smith, A. (2013). Dengue infections in non-immune travellers to Thailand. *Epidemiology and infection*, 141(2):412–7.

Mbaika, S., Lutomiah, J., Chepkorir, E., Mulwa, F., Khayeka-Wandabwa, C., et al. (2016). Vector competence of Aedes aegypti in transmitting Chikungunya virus: Effects and implications of extrinsic incubation temperature on dissemination and infection rates. *Virology Journal*, 13(1):1–9.

Mendes Luz, P., Torres Codeço, C., Massad, E., and Struchiner, C. J. (2003). Uncertainties Regarding Dengue Modeling in Rio de Janeiro, Brazil. *Memorias do Instituto Oswaldo Cruz*, 98(7):871–878.

Messina, J. P., Kraemer, M. U., Brady, O. J., Pigott, D. M., Shearer, F. M., et al. (2016). Mapping global environmental suitability for Zika virus. *eLife*, 5:1–19.

Metropolis, N., Rosenbluth, A. W., Rosenbluth, M. N., Teller, A. H., and Teller, E. (1953). Equation of State Calculations by Fast Computing Machines. *The Journal of Chemical Physics*, 21(6):1087–1092.

Meyers, L. A., Pourbohloul, B., Newman, M. E., Skowronski, D. M., and Brunham, R. C. (2005). Network theory and SARS: Predicting outbreak diversity. *Journal of Theoretical Biology*, 232(1):71–81.

Michalakes, J. and Vachharajani, M. (2008). GPU acceleration of numerical weather prediction. *Parallel Processing Letters*, 18(04):531–548.

Ministério da Saúde. Ministério da Saúde alerta para aumento de 149% dos casos de dengue no país, (2019). http://portalms.saude.gov.br/noticias/agencia-saude/45257-

ministerio-da-saude-alerta-para-aumento-de-149-dos-casos-de-dengue-no-pais. [Accessed on: 2019-03-22].

Moghadas, S. M., Shoukat, A., Espindola, A. L., Pereira, R. S., Abdirizak, F., et al. (2017). Asymptomatic Transmission and the Dynamics of Zika Infection. *Scientific Reports*, 7 (1):5829.

Mohamed, M. S., Hany, a. K., and Emad, I. K. (2013). Life table characteristics of Aedes aegypti (Diptera:Culicidae) from Saudi Arabia. *Tropical biomedicine*, 30(2):301–14.

Montibeler, E. E. and de Oliveira, D. R. (2018). Dengue endemic and its impact on the gross national product of Brazilian economy. *Acta Tropica*, 178(November 2017): 318–326.

Montoya, M., Gresh, L., Mercado, J. C., Williams, K. L., Vargas, M. J., et al. (2013). Symptomatic Versus Inapparent Outcome in Repeat Dengue Virus Infections Is Influenced by the Time Interval between Infections and Study Year. *PLoS Neglected Tropical Diseases*, 7(8):1–10.

Mordecai, E. A., Cohen, J. M., Evans, M. V., Gudapati, P., Johnson, L. R., et al. (2017). Detecting the impact of temperature on transmission of Zika, dengue, and chikungunya using mechanistic models. *PLOS Neglected Tropical Diseases*, 11(4):e0005568.

Morrison, A. C., Getis, A., Santiago, M., Rigau-Perez, J. G., Reiter, P., et al. (1998). Exploratory space-time analysis of reported dengue cases during an outbreak in Florida, Puerto Rico, 1991-1992. *American Journal of Tropical Medicine and Hygiene*, 58(3): 287–298.

Morrison, A. C., Zielinski-Gutierrez, E., Scott, T. W., and Rosenberg, R. (2008). Defining Challenges and Proposing Solutions for Control of the Virus Vector Aedes aegypti. *PLoS Medicine*, 5(3):e68.

Mosegaard, J. and Sorensen, T. GPU Accelerated Surgical Simulators for Complex Morphology. In *IEEE Virtual Reality Conference 2005 (VR'05)*, volume 2005, 147–154,. IEEE, (2006).

Muir, L. E. and Kay, B. H. (1998). Aedes Aegypti Survival and Dispersal Estimated By Mark-Release-Recapture in Northern Australia. *Am. J. Trop. Med. Hyg*, 58(3):277–282.

Muller, D. A., Depelsenaire, A. C., and Young, P. R. (2017). Clinical and laboratory diagnosis of dengue virus infection. *Journal of Infectious Diseases*, 215(Suppl 2):S89–S95.

Murray, L. (2012). GPU acceleration of Runge-Kutta integrators. *IEEE Transactions on Parallel and Distributed Systems*, 23(1):94–101.

Murray, N. E. A., Quam, M. B., and Wilder-Smith, A. (2013). Epidemiology of dengue: Past, present and future prospects. *Clinical Epidemiology*, 5(1):299–309.

Nagao, Y. and Koelle, K. (2008). Decreases in dengue transmission may act to increase the incidence of dengue hemorrhagic fever. *Proceedings of the National Academy of Sciences of the United States of America*, 105(6):2238–43.

Naish, S., Dale, P., Mackenzie, J. S., McBride, J., Mengersen, K., and Tong, S. (2014). Climate change and dengue: a critical and systematic review of quantitative modelling approaches. *BMC Infectious Diseases*, 14(1):167.

Nam, V. S., Yen, N. T., Holynska, M., Reid, J. W., and Kay, B. H. (2000). National progress in dengue vector control in Vietnam: survey for Mesocyclops (Copepoda), Micronecta (Corixidae), and fish as biological control agents. *The American journal of tropical medicine and hygiene*, 62(1):5–10.

Netto, E. M., Moreira-Soto, A., Pedroso, C., Höser, C., Funk, S., et al. (2017). High Zika Virus Seroprevalence in Salvador, Northeastern Brazil Limits the Potential for Further Outbreaks. *mBio*, 8(6):1–14.

Newman, M. E. (2002). Spread of epidemic disease on networks. *Physical Review E - Statistical Physics, Plasmas, Fluids, and Related Interdisciplinary Topics*, 66(1):1–11.

Newman, M. E. J. and Girvan, M. (2004). Finding and evaluating community structure in networks. *Physical Review E*, 69(2):026113.

Nisalak, A. (2015). Laboratory diagnosis of dengue virus infections. *The Southeast Asian journal of tropical medicine and public health*, 46 Suppl 1:55–76.

Nisalak, A., Endy, T. P., Nimmannitya, S., Kalayanarooj, S., Thisayakorn, U., et al. (2003). Serotype-specific dengue virus circulation and dengue disease in Bangkok, Thailand from 1973 to 1999. *The American journal of tropical medicine and hygiene*, 68(2):191–202.

Nunes, M. R., Palacios, G., Faria, N. R., Sousa, E. C., Pantoja, J. A., et al. (2014). Air Travel Is Associated with Intracontinental Spread of Dengue Virus Serotypes 1-3 in Brazil. *PLoS Neglected Tropical Diseases*, 8(4).

OhAinle, M., Balmaseda, A., Macalalad, A. R., Tellez, Y., Zody, M. C., et al. (2011). Dynamics of Dengue Disease Severity Determined by the Interplay Between Viral Genetics and Serotype-Specific Immunity. *Science Translational Medicine*, 3(114): 114ra128–114ra128.

Ooi, E. E. (2015). The re-emergence of dengue in China. *BMC Medicine*, 13(1):1–3.

Ooi, E.-E. and Gubler, D. J. (2009). Dengue in Southeast Asia: epidemiological characteristics and strategic challenges in disease prevention. *Cadernos de Saúde Pública*, 25 (suppl 1):S115–S124.

O'Reilly, K. M., Lowe, R., Edmunds, W. J., Mayaud, P., Kucharski, A., et al. (2018). Projecting the end of the Zika virus epidemic in Latin America: A modelling analysis. *BMC Medicine*, 16(1):1–13.

Otero, M., Solari, H. G., and Schweigmann, N. (2006). A stochastic population dynamics model for Aedes aegypti: Formulation and application to a city with temperate climate. *Bulletin of Mathematical Biology*, 68(8):1945–1974.

Paaijmans, K. P., Wandago, M. O., Githeko, A. K., and Takken, W. (2007). Unexpected high losses of Anopheles gambiae larvae due to rainfall. *PLoS ONE*, 2(11).

Padmanabha, H., Durham, D., Correa, F., Diuk-Wasser, M., and Galvani, A. (2012). The Interactive Roles of Aedes aegypti Super-Production and Human Density in Dengue Transmission. *PLoS Neglected Tropical Diseases*, 6(8).

Paixão, E. S., Costa, M. d. C. N., Rodrigues, L. C., Rasella, D., Cardim, L. L., et al. (2015). Trends and factors associated with dengue mortality and fatality in Brazil. *Revista da Sociedade Brasileira de Medicina Tropical*, 48(4):399–405.

Pandey, A., Mubayi, A., and Medlock, J. (2013). Comparing vector-host and SIR models for dengue transmission. *Mathematical Biosciences*, 246(2):252–259.

Pang, T., Mak, T. K., and Gubler, D. J. (2017). Prevention and control of dengue—the light at the end of the tunnel. *The Lancet Infectious Diseases*, 17(3):e79–e87.

Pavía-Ruz, N., Balam-May, A., Longini, I. M., Manrique-Saide, P., Halloran, M. E., et al. (2018). Seroprevalence of Dengue Antibodies in Three Urban Settings in Yucatan, Mexico. *The American Journal of Tropical Medicine and Hygiene*, 98(4):1202–1208.

Percival, D. B. and Walden, A. T. Introduction to Wavelets. In *Wavelet Methods for Time Series Analysis*, 1–19. Cambridge University Press, Cambridge, (2000). ISBN 1714821838.

Perkins, T. A., Scott, T. W., Le Menach, A., and Smith, D. L. (2013). Heterogeneity, Mixing, and the Spatial Scales of Mosquito-Borne Pathogen Transmission. *PLoS Computational Biology*, 9(12).

Perkins, T. A., Garcia, A. J., Paz-Soldan, V. A., Stoddard, S. T., Reiner, R. C., et al. (2014). Theory and data for simulating fine-scale human movement in an urban environment. *Journal of The Royal Society Interface*, 11(99):20140642–20140642.

Perkins, T. A., Paz-Soldan, V. A., Stoddard, S. T., Morrison, A. C. A. A. C., Forshey, B. B. M., et al. (2016). Calling in sick: impacts of fever on intra-urban human mobility. *Proc. R. Soc. B*, 283(1834):36–51.

Perkins, T. A., Rodriguez-Barraquer, I., Manore, C., Siraj, A. S., Espana, G., et al. (2018). Heterogeneous local dynamics revealed by classification analysis of spatially disaggregated time series data. *bioRxiv*.

Pitt, M. K., Silva, S., Giordani, P., and Kohn, R. (2012). On some properties of Markov chain Monte Carlo simulation methods based on the particle filter. *Journal of Econometrics*, 171(2):134–151.

Polwiang, S. (2016). Estimation of dengue infection for travelers in Thailand. *Travel Medicine and Infectious Disease*, 14(4):398–406.

Ponlawat, A. and Harrington, L. C. (2005). Blood feeding patterns of Aedes aegypti and Aedes albopictus in Thailand. *Journal of medical entomology*, 42(5):844–9.

Powell, J. R., Gloria-Soria, A., and Kotsakiozi, P. (2018). Recent History of Aedes aegypti : Vector Genomics and Epidemiology Records. *BioScience*, 68(11):854–860.

Premaratna, R., Pathmeswaran, A., Amarasekara, N. D. D. M., Motha, M. B. C., Perera, K. V. H. K. K., and de Silva, H. J. (2009). A clinical guide for early detection of dengue

fever and timing of investigations to detect patients likely to develop complications. *Transactions of the Royal Society of Tropical Medicine and Hygiene*, 103(2):127–131.

Proakis, J. G. and Manolakis, D. G. *Digital Signal Processing: Principles, Algorithms, and Applications*. Prentice-Hall, Inc., Upper Saddle River, New Jersey, 3 edition, (1996).

Qi, X., Wang, Y., Li, Y., Meng, Y., Chen, Q., et al. (2015). The Effects of Socioeconomic and Environmental Factors on the Incidence of Dengue Fever in the Pearl River Delta, China, 2013. *PLOS Neglected Tropical Diseases*, 9(10):e0004159.

Quam, M. B., Sessions, O., Kamaraj, U. S., Rocklöv, J., and Wilder-Smith, A. (2016). Dissecting Japan's dengue outbreak in 2014. *American Journal of Tropical Medicine and Hygiene*, 94(2):409–412.

Raghwani, J., Rambaut, A., Holmes, E. C., Hang, V. T., Hien, T. T., et al. (2011). Endemic dengue associated with the co-circulation of multiple viral lineages and localized density-dependent transmission. *PLoS Pathogens*, 7(6):1–10.

Recker, M., Blyuss, K. B., Simmons, C. P., Hien, T. T., Wills, B., et al. (2009). Immunological serotype interactions and their effect on the epidemiological pattern of dengue. *Proceedings. Biological sciences / The Royal Society*, 276(1667):2541–8.

Reich, N. G., Shrestha, S., King, A. A., Rohani, P., Lessler, J., et al. (2013). Interactions between serotypes of dengue highlight epidemiological impact of cross-immunity. *Journal of The Royal Society Interface*, 10(86):20130414–20130414.

Reiner, R. C., Perkins, T. A., Barker, C. M., Niu, T., Chaves, L. F., et al. (2013). A systematic review of mathematical models of mosquito-borne pathogen transmission: 1970-2010. *Journal of the Royal Society, Interface*, 10(81):20120921.

Reiner, R. C., Stoddard, S. T., and Scott, T. W. (2014). Socially structured human movement shapes dengue transmission despite the diffusive effect of mosquito dispersal. *Epidemics*, 6:30–36.

Reiner, R. C., Le Menach, A., Kunene, S., Ntshalintshali, N., Hsiang, M. S., et al. (2015). Mapping residual transmission for malaria elimination. *eLife*, 4(DECEMBER2015): 1955–1969.

Rezza, G. (2014). Dengue and chikungunya: long-distance spread and outbreaks in naïve areas. *Pathogens and Global Health*, 108(8):349–355.

Ribeiro, R. M., Qin, L., Chavez, L. L., Li, D., Self, S. G., and Perelson, A. S. (2010). Estimation of the initial viral growth rate and basic reproductive number during acute HIV-1 infection. *Journal of virology*, 84(12):6096–6102.

Rigau-Pérez, J. G., Clark, G. G., Gubler, D. J., Reiter, P., Sanders, E. J., and Vorndam, A. V. (1998). Dengue and dengue haemorrhagic fever. *Lancet*, 352(9132):971–977.

Riley, S. (2007). Models of Infectious Disease. *Science*, 316(5829):1298–1301.

Riley, S. and Ferguson, N. M. (2006). Smallpox transmission and control: Spatial dynamics in Great Britain. *Proceedings of the National Academy of Sciences*, 103(33):12637–12642.

Roberts, G. O. and Rosenthal, J. S. (2009). Examples of adaptive MCMC. *Journal of Computational and Graphical Statistics*, 18(2):349–367.

Rock, K. S., Wood, D. A., and Keeling, M. J. (2015). Age- and bite-structured models for vector-borne diseases. *Epidemics*, 12:20–29.

Rodrigues, N. C. P., Lino, V. T. S., Daumas, R. P., Andrade, M. K. d. N., O'Dwyer, G., et al. (2016). Temporal and Spatial Evolution of Dengue Incidence in Brazil, 2001-2012. *PLoS ONE*, 11(11):e0165945.

Rodríguez-Barraquer, I., Buathong, R., Iamsirithaworn, S., Nisalak, A., Lessler, J., et al. (2014). Revisiting rayong: Shifting seroprofiles of dengue in thailand and their implications for transmission and control. *American Journal of Epidemiology*, 179(3): 353–360.

Romeo-Aznar, V., Paul, R., Telle, O., and Pascual, M. (2018). Mosquito-borne transmission in urban landscapes: The missing link between vector abundance and human density. *Proceedings of the Royal Society B: Biological Sciences*, 285(1884).

Ross, R. *The Preventation of malaria*. John Murray, London, (1911).

Sabin, A. B. (1952). Research on dengue during World War II. *The American journal of tropical medicine and hygiene*, 1(1):30–50.

Salathé, M. and Jones, J. H. (2010). Dynamics and Control of Diseases in Networks with Community Structure. *PLoS Computational Biology*, 6(4):e1000736.

Salathé, M., Bengtsson, L., Bodnar, T. J., Brewer, D. D., Brownstein, J. S., et al. (2012). Digital Epidemiology. *PLoS Computational Biology*, 8(7):e1002616.

Salazar, M. I., Richardson, J. H., Sánchez-Vargas, I., Olson, K. E., and Beaty, B. J. (2007). Dengue virus type 2: replication and tropisms in orally infected Aedes aegypti mosquitoes. *BMC microbiology*, 7:9.

Sang, S., Yin, W., Bi, P., Zhang, H., Wang, C., et al. (2014). Predicting local dengue transmission in Guangzhou, China, through the influence of imported cases, mosquito density and climate variability. *PLoS ONE*, 9(7):1–10.

Satria, M. T., Huang, B., Hsieh, T.-J., Chang, Y.-L., and Liang, W.-Y. (2012). GPU Acceleration of Tsunami Propagation Model. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 5(3):1014–1023.

Schmidt, W. P., Suzuki, M., Thiem, V., White, R. G., Tsuzuki, A., et al. (2011). Population density, water supply, and the risk of dengue fever in vietnam: Cohort study and spatial analysis. *PLoS Medicine*, 8(8).

Schwartz, E., Mileguir, F., Grossman, Z., and Mendelson, E. (2000). Evaluation of ELISA-based sero-diagnosis of dengue fever in travelers. *Journal of Clinical Virology*, 19(3): 169–173.

Schwartz, I. B., Shaw, L. B., Cummings, D. A., Billings, L., Mccrary, M., and Burke, D. S. (2005). Chaotic desynchronization of multistrain diseases. *Physical Review E - Statistical, Nonlinear, and Soft Matter Physics*, 72(6):1–6.

Scott, T. W., Chow, E., Strickman, D., Kittayapong, P., Wirtz, R. A., et al. (1993a). Blood-Feeding Patterns of Aedes aegypti (Diptera: Culicidae) Collected in a Rural Thai Village. *Journal of Medical Entomology*, 30(5):922–927.

Scott, T. W., Clark, G. G., Lorenz, L. H., Amerasinghe, P. H., Reiter, P., and Edman, J. D. (1993b). Detection of Multiple Blood Feeding in Aedes aegypti (Diptera: Culicidae) During a Single Gonotrophic Cycle Using a Histologic Technique. *Journal of Medical Entomology*, 30(1):94–99.

Scott, T. W., Amerasinghe, P. H., Morrison, A. C., Lorenz, L. H., Clark, G. G., et al. (2000a). Longitudinal studies of Aedes aegypti (Diptera: Culicidae) in Thailand and Puerto Rico: blood feeding frequency. *J Med Entomol*, 37(1):89–101.

Scott, T. W., Morrison, A. C., Lorenz, L. H., Clark, G. G., Strickman, D., et al. (2000b). Longitudinal Studies of Aedes aegypti (Diptera : Culicidae) in Thailand and Puerto

Rico : Population Dynamics Longitudinal Studies of Aedes aegypti (Diptera : Culicidae) in Thailand and Puerto Rico : Population Dynamics. *J Med Entomol*, 37(1):77–88.

Shang, C. S., Fang, C. T., Liu, C. M., Wen, T. H., Tsai, K. H., and King, C. C. (2010). The role of imported cases and favorable meteorological conditions in the onset of dengue epidemics. *PLoS Neglected Tropical Diseases*, 4(8).

Sharma, R. S., Kaul, S. M., and Sokhay, J. (2005). Seasonal fluctuations of dengue fever vector, Aedes aegypti (Diptera: Culicidae) in Delhi, India. *The Southeast Asian journal of tropical medicine and public health*, 36(1):186–90.

Sharpe, P. J. and DeMichele, D. W. (1977). Reaction kinetics of poikilotherm development. *Journal of theoretical biology*, 64(4):649–70.

Sherlock, C., Thiery, A. H., Roberts, G. O., and Rosenthal, J. S. (2015). On the efficiency of pseudo-marginal random walk metropolis algorithms. *Annals of Statistics*, 43(1): 238–275.

Shimokawabe, T., Aoki, T., Muroi, C., Ishida, J., Kawano, K., et al. An 80-Fold Speedup, 15.0 TFlops Full GPU Acceleration of Non-Hydrostatic Weather Model ASUCA Production Code. In *2010 ACM/IEEE International Conference for High Performance Computing, Networking, Storage and Analysis*, 1–11. IEEE, (2010).

Shutt, D. P., Manore, C. A., Pankavich, S., Porter, A. T., and Del Valle, S. Y. (2017). Estimating the reproductive number, total outbreak size, and reporting rates for Zika epidemics in South and Central America. *Epidemics*, 21:63–79.

Silva, S. L., Ferreira, J. A., and Martins, M. L. (2007). Epidemic spreading in a scale-free network of regular lattices. *Physica A: Statistical Mechanics and its Applications*, 377 (2):689–697.

Sinka, M. E., Bangs, M. J., Manguin, S., Rubio-Palis, Y., Chareonviriyaphap, T., et al. (2012). A global map of dominant malaria vectors. *Parasites & Vectors*, 5(1):69.

Siqueira, J. B., Martelli, C. M. T., Maciel, I. J., Oliveira, R. M., Ribeiro, M. G., et al. (2004). Household survey of dengue infection in central Brazil: spatial point pattern analysis and risk factors assessment. *The American journal of tropical medicine and hygiene*, 71(5):646–51.

Siraj, A. S., Oidtman, R. J., Huber, J. H., Kraemer, M. U. G., Brady, J., et al. (2017). Temperature modulates dengue virus epidemic growth rates through its effects on reproduction numbers and generation intervals. *PLoS Neglected Tropical Diseases*, 1–19.

Sirisena, P., Noordeen, F., Kurukulasuriya, H., Romesh, T. A., and Fernando, L. K. (2017). Effect of climatic factors and population density on the distribution of dengue in Sri Lanka: A GIS based evaluation for prediction of outbreaks. *PLoS ONE*, 12(1).

Smith, D. L., Dushoff, J., and McKenzie, F. E. (2004). The risk of a mosquito-borne infectionin a heterogeneous environment. *PLoS Biology*, 2(11).

Smith, D. L., McKenzie, F. E., Snow, R. W., and Hay, S. I. (2007). Revisiting the basic reproductive number for malaria and its implications for malaria control. *PLoS Biology*, 5(3):0531–0542.

Smith, D. L., Perkins, T. A., Reiner, R. C., Barker, C. M., Niu, T., et al. (2014). Recasting the theory of mosquito-borne pathogen transmission dynamics and control. *Transactions of the Royal Society of Tropical Medicine and Hygiene*, 108(4):185–97.

Soda, K. J., Moore, S. M., Espana, G., Bloedow, J., Raybaud, B., et al. (2018). DTK-Dengue: A new agent-based model of dengue virus transmission dynamics. *bioRxiv*.

Soler, M. (1998). Laboratory diagnosis of dengue virus infections. *Acta cientifica venezolana*, 49 Suppl 1:25–32.

Stanaway, J. J. D., Shepard, D. S. D., Undurraga, E. E. A., Halasa, Y. Y. A., Coffeng, L. E. L., et al. (2016). The global burden of dengue: an analysis from the Global Burden of Disease Study 2013. *The Lancet Infectious Diseases*, 16(6):712–723.

Stewart-Ibarra, A. M., Muñoz, Á. G., Ryan, S. J., Ayala, E. B., Borbor-Cordova, M. J., et al. (2014). Spatiotemporal clustering, climate periodicity, and social-ecological risk factors for dengue during an outbreak in Machala, Ecuador, in 2010. *BMC Infectious Diseases*, 14(1):610.

Stoddard, S. T., Morrison, A. C., Vazquez-Prokopec, G. M., Soldan, V. P., Kochel, T. J., et al. (2009). The role of human movement in the transmission of vector-borne pathogens. *PLoS Neglected Tropical Diseases*, 3(7).

Stoddard, S. T., Forshey, B. M., Morrison, A. C., Paz-Soldan, V. a., Vazquez-Prokopec, G. M., et al. (2013). House-to-house human movement drives dengue virus transmission. *Proceedings of the National Academy of Sciences*, 110(3):994–999.

Strickman, D. and Kittayapong, P. (2002). Dengue and its vectors in Thailand: Introduction to the study and seasonal distribution of Aedes larvae. *American Journal of Tropical Medicine and Hygiene*, 67(3):247–259.

Styer, L. M., Carey, J. R., Wang, J. L., and Scott, T. W. (2007). Mosquitoes do senesce: Departure from the paradigm of constant mortality. *American Journal of Tropical Medicine and Hygiene*, 76(1):111–117.

Succo, T., Leparc-Goffart, I., Ferré, J.-B., Roiz, D., Broche, B., et al. (2016). Autochthonous dengue outbreak in Nîmes, South of France, July to September 2015. *Eurosurveillance*, 21(21):30240.

Sumi, A., Telan, E. F. O., Chagan-Yasutan, H., Piolo, M. B., Hattori, T., and Kobayashi, N. (2017). Effect of temperature, relative humidity and rainfall on dengue fever and leptospirosis infections in Manila, the Philippines. *Epidemiology and infection*, 145(1): 78–86.

Sunarso, A., Tsuji, T., and Chono, S. (2010). GPU-accelerated molecular dynamics simulation for study of liquid crystalline flows. *Journal of Computational Physics*, 229 (15):5486–5497.

Sweet, J., Richter, D. H., and Thain, D. (2018). GPU acceleration of Eulerian–Lagrangian particle-laden turbulent flow simulations. *International Journal of Multiphase Flow*, 99: 437–445.

ten Bosch, Q. A., Clapham, H. E., Lambrechts, L., Duong, V., Buchy, P., et al. (2018). Contributions from the silent majority dominate dengue virus transmission. *PLOS Pathogens*, 14(5):e1006965.

Tesla, B., Demakovsky, L. R., Mordecai, E. A., Ryan, S. J., Bonds, M. H., et al. (2018). Temperature drives Zika virus transmission: evidence from empirical and mathematical models. *Proceedings of the Royal Society B: Biological Sciences*, 285(1884):20180795.

Teurlai, M., Huy, R., Cazelles, B., Duboz, R., Baehr, C., and Vong, S. (2012). Can Human

Movements Explain Heterogeneous Propagation of Dengue Fever in Cambodia? *PLoS Neglected Tropical Diseases*, 6(12):3–10.

Thai, K. T. D., Nishiura, H., Hoang, P. L., Tran, N. T. T., Phan, G. T., et al. (2011). Age-specificity of clinical dengue during primary and secondary infections. *PLoS Neglected Tropical Diseases*, 5(6).

Tian, H., Sun, Z., Faria, N. R., Yang, J., Cazelles, B., et al. (2017). Increasing airline travel may facilitate co-circulation of multiple dengue virus serotypes in Asia. *PLOS Neglected Tropical Diseases*, 11(8):e0005694.

Tirado, S. M. C. and Yoon, K.-J. (2003). Antibody-Dependent Enhancement of Virus Infection and Disease. *Viral Immunology*, 16(1):69–86.

Towers, S., Brauer, F., Castillo-Chavez, C., Falconar, A. K., Mubayi, A., and Romero-Vivas, C. M. (2016). Estimate of the reproduction number of the 2015 Zika virus outbreak in Barranquilla, Colombia, and estimation of the relative role of sexual transmission. *Epidemics*, 17:50–55.

Tran, A., L'Ambert, G., Lacour, G., Benoît, R., Demarchi, M., et al. (2013). A rainfall- and temperature-driven abundance model for Aedes albopictus populations. *International Journal of Environmental Research and Public Health*, 10(5):1698–1719.

Tuncer, N., Marctheva, M., LaBarre, B., and Payoute, S. (2018). Structural and Practical Identifiability Analysis of Zika Epidemiological Models. *Bulletin of Mathematical Biology*, 80(8):2209–2241.

United Nations. World Urbanization Prospects: The 2014 Revision, (ST/ESA/SER.A/366). Technical report, Department of Economic and Social Affairs, Population Division, (2015).

Valdez, L. D., Sibona, G. J., Diaz, L. A., Contigiani, M. S., and Condat, C. A. (2017). Effects of rainfall on Culex mosquito population dynamics. *Journal of Theoretical Biology*, 421:28–38.

Vanlerberghe, V., Toledo, M. E., Rodríguez, M., Gomez, D., Baly, A., et al. (2009). Community involvement in dengue vector control: Cluster randomised trial. *BMJ (Online)*, 338(7709):1477–1479.

Vasilakis, N. and Weaver, S. C. *Chapter 1 The History and Evolution of Human Dengue Emergence*, volume 72. Elsevier Inc., (2008).

Vasilakis, N., Shell, E. J., Fokam, E. B., Mason, P. W., Hanley, K. A., et al. (2007). Potential of ancestral sylvatic dengue-2 viruses to re-emerge. *Virology*, 358(2):402–412.

Vasilakis, N., Cardosa, J., Hanley, K. A., Holmes, E. C., and Weaver, S. C. (2011). Fever from the forest: Prospects for the continued emergence of sylvatic dengue virus and its impact on public health. *Nature Reviews Microbiology*, 9(7):532–541.

Vázquez, S., Valdés, O., Pupo, M., Delgado, I., Álvarez, M., et al. (2003). MAC-ELISA and ELISA inhibition methods for detection of antibodies after yellow fever vaccination. *Journal of Virological Methods*, 110(2):179–184.

Velázquez-Castro, J., Anzo-Hernández, A., Bonilla-Capilla, B., Soto-Bajo, M., and Fraguela-Collar, A. (2018). Vector-borne disease risk indexes in spatially structured populations. *PLOS Neglected Tropical Diseases*, 12(2):e0006234.

Villabona-Arenas, C. J., de Oliveira, J. L., Capra, C. d. S., Balarini, K., Loureiro, M., et al. (2014). Detection Of Four Dengue Serotypes Suggests Rise In Hyperendemicity In Urban Centers Of Brazil. *PLoS Neglected Tropical Diseases*, 8(2):e2620.

Villela, D. A. M., Bastos, L. S., DE Carvalho, L. M., Cruz, O. G., Gomes, M. F. C., et al. (2017). Zika in Rio de Janeiro: Assessment of basic reproduction number and comparison with dengue outbreaks. *Epidemiology and infection*, 145(8):1649–1657.

Vincenti-Gonzalez, M. F., Grillet, M. E., Velasco-Salas, Z. I., Lizarazo, E. F., Amarista, M. A., et al. (2017). Spatial Analysis of Dengue Seroprevalence and Modeling of Transmission Risk Factors in a Dengue Hyperendemic City of Venezuela. *PLoS Neglected Tropical Diseases*, 11(1):1–21.

Vu, S. N., Nguyen, T. Y., Tran, V. P., Truong, U. N., Le, Q. M., et al. (2005). Elimination of dengue by community programs using Mesocyclops(Copepoda) against Aedes aegypti in central Vietnam. *The American journal of tropical medicine and hygiene*, 72(1):67–73.

Wang, E., Ni, H., Xu, R., Barrett, A. D. T., Watowich, S. J., et al. (2000). Evolutionary Relationships of Endemic/Epidemic and Sylvatic Dengue Viruses. *Journal of Virology*, 74(7):3227–3234.

Wang, T., Wang, M., Shu, B., qin Chen, X., Luo, L., et al. (2015). Evaluation of Inapparent Dengue Infections During an Outbreak in Southern China. *PLoS Neglected Tropical Diseases*, 9(3):1–11.

Waterman, S. H., Zielinski-Gutierrez, E., Anaya-Lopez, L., Brunkard, J. M., Smith, B., et al. (2008). Epidemic Dengue and Dengue Hemorrhagic Fever at the Texas–Mexico Border: Results of a Household-based Seroepidemiologic Survey, December 2005. *The American Journal of Tropical Medicine and Hygiene*, 78(3):364–369.

Wearing, H. J. and Rohani, P. (2006). Ecological and immunological determinants of dengue epidemics. *Proceedings of the National Academy of Sciences*, 103(31):11802–11807.

Weaver, S. C. and Vasilakis, N. (2009). Molecular evolution of dengue viruses: Contributions of phylogenetics to understanding the history and epidemiology of the preeminent arboviral disease. *Infection, Genetics and Evolution*, 9(4):523–540.

Webster, D. P., Farrar, J., and Rowland-Jones, S. (2009). Progress towards a dengue vaccine. *The Lancet Infectious Diseases*, 9(11):678–687.

Weeraratne, T. C., Perera, M. D. B., Mansoor, M. A., and Karunaratne, S. H. (2013). Prevalence and breeding habitats of the dengue vectors Aedes aegypti and Aedes albopictus (Diptera: Culicidae) in the semi-urban areas of two different climatic zones in Sri Lanka. *International Journal of Tropical Insect Science*, 33(4):216–226.

Wen, T. H., Lin, M. H., and Fang, C. T. (2012). Population Movement and Vector-Borne Disease Transmission: Differentiating Spatial-Temporal Diffusion Patterns of Commuting and Noncommuting Dengue Cases. *Annals of the Association of American Geographers*, 102(5):1026–1037.

Wesolowski, A., Qureshi, T., Boni, M. F., Sundsøy, P. R., Johansson, M. A., et al. (2015). Impact of human mobility on the emergence of dengue epidemics in Pakistan. *Proceedings of the National Academy of Sciences*, 112(38):11887–11892.

Whitehead, S. S., Blaney, J. E., Durbin, A. P., and Murphy, B. R. (2007). Prospects for a dengue virus vaccine. *Nature Reviews Microbiology*, 5(7):518–528.

Wilder-Smith, A. (2012). Dengue infections in travellers. *Paediatrics and International Child Health*, 32(sup1):28–32.

Wilder-Smith, A. and Byass, P. (2016). The elusive global burden of dengue. *The Lancet Infectious Diseases*, 16(6):629–631.

Wilder-Smith, A. and Gubler, D. J. (2008). Geographic Expansion of Dengue: The Impact of International Travel. *Medical Clinics of North America*, 92(6):1377–1390.

Wilder-Smith, A., Ooi, E. E., Vasudevan, S. G., and Gubler, D. J. (2010). Update on dengue: Epidemiology, virus evolution, antiviral drugs, and vaccine development. *Current Infectious Disease Reports*, 12(3):157–164.

Wilder-Smith, A., Quam, M., Sessions, O., Rocklov, J., Liu-Helmersson, J., et al. (2014). The 2012 dengue outbreak in Madeira: exploring the origins. *Eurosurveillance*, 19(8).

Wolfe, N. D., Kilbourn, A. M., Karesh, W. B., Rahman, H. A., Bosi, E. J., et al. (2001). Sylvatic transmission of arboviruses among Bornean orangutans. *The American journal of tropical medicine and hygiene*, 64(5-6):310–6.

Woolhouse, M., Ward, M., van Bunnik, B., and Farrar, J. (2015). Antimicrobial resistance in humans, livestock and the wider environment. *Philosophical Transactions of the Royal Society B: Biological Sciences*, 370(1670):20140083–20140083.

World Health Organization. Comprehensive Guidelines for Prevention and Control of Dengue and Dengue Haemorrhagic Fever. Technical report, World Health Organization, (2011).

World Health Organization. Questions and Answers on Dengue Vaccines, (2018a). https://www.who.int/immunization/research/development/dengue_q_and_a/en/. [Accessed on: 2019-03-18].

World Health Organization. Background Paper on Dengue Vaccines. Technical Report April, World Health Organization, (2018b). https://www.who.int/immunization/sage/meetings/2018/april/2_DengueBackgrPaper_SAGE_Apr2018.pdf

Wu, W. and Heng, P. A. (2004). A hybrid condensed finite element model with GPU acceleration for interactive 3D soft tissue cutting. *Computer Animation and Virtual Worlds*, 15(3-4):219–227.

Xiao, F. Z., Zhang, Y., Deng, Y. Q., He, S., Xie, H. G., et al. (2014). The effect of temperature on the extrinsic incubation period and infection rate of dengue virus serotype 2 infection in Aedes albopictus. *Archives of Virology*, 159(11):3053–3057.

Xu, Z., Bambrick, H., Yakob, L., Devine, G., Lu, J., et al. (2019). Spatiotemporal patterns and climatic drivers of severe dengue in Thailand. *Science of the Total Environment*, 656:889–901.

Yan, X.-Y., Zhao, C., Fan, Y., Di, Z., and Wang, W.-X. (2014). Universal predictability of mobility patterns in cities. *Journal of The Royal Society Interface*, 11(100):20140834–20140834.

Yang, C. F., Hou, J. N., Chen, T. H., and Chen, W. J. (2014). Discriminable roles of Aedes aegypti and Aedes albopictus in establishment of dengue outbreaks in Taiwan. *Acta Tropica*, 130(1):17–23.

Yang, H. M., Macoris, M. L. G., Galvani, K. C., Andrighetti, M. T. M., and Wanderley, D. M. V. (2009). Assessing the effects of temperature on the population of Aedes aegypti, the vector of dengue. *Epidemiology and Infection*, 137(8):1188–1202.

Yang, J., Wang, Y., and Chen, Y. (2007). GPU accelerated molecular dynamics simulation of thermal conductivities. *Journal of Computational Physics*, 221(2):799–804.

Young, K. I., Mundis, S., Widen, S. G., Wood, T. G., Tesh, R. B., et al. (2017). Abundance and distribution of sylvatic dengue virus vectors in three different land cover types in Sarawak, Malaysian Borneo. *Parasites and Vectors*, 10(1):1–14.

Yu, H. L., Yang, S. J., Yen, H. J., and Christakos, G. (2011). A spatio-temporal climate-based model of early dengue fever warning in southern Taiwan. *Stochastic Environmental Research and Risk Assessment*, 25(4):485–494.

Zambrana, J. V., Bustos Carrillo, F., Burger-Calderon, R., Collado, D., Sanchez, N., et al. (2018). Seroprevalence, risk factor, and spatial analyses of Zika virus infection after the 2016 epidemic in Managua, Nicaragua. *Proceedings of the National Academy of Sciences*, 115(37):9294–9299.

Zhang, N., Wang, J. L., and Chen, Y. S. (2010). Image parallel processing based on GPU. *Proceedings - 2nd IEEE International Conference on Advanced Computer Control, ICACC 2010*, 3:367–370.

Zhou, Y., Liepe, J., Sheng, X., Stumpf, M. P., and Barnes, C. (2011). GPU accelerated biochemical network simulation. *Bioinformatics*, 27(6):874–876.

Zhu, G., Liu, T., Xiao, J., Zhang, B., Song, T., et al. (2019). Effects of human mobility, temperature and mosquito control on the spatiotemporal transmission of dengue. *Science of The Total Environment*, 651:969–978.