Mathematical modeling of the transmission dynamics of hepatitis B using phylogenetics

Master's thesis

Author:
Lieke van Schaijk
3241092

Supervisors:
Ir. R.J.F. Ypma (RIVM)
Dr. J. Wallinga (RIVM)
Dr. J. van de Kassteele (RIVM)
Dr. M.C.J. Bootsma (UU/UMCU)

October 31, 2013

Rijksinstituut voor Volksgezondheid
en Milieu
Ministerie van Volksgezondheid,
Welzijn en Sport

Universiteit Utrecht

Preface

This thesis is the result of my nine months research internship at the department Epidemiology & Surveillance of the National Institute of Public Health and the Environment (RIVM), which was the final project to complete my master study Stochastics and Financial Mathematics at Utrecht University. I was supervised by dr. M.C.J. Bootsma from the Mathematical Institute of Utrecht University and ir. R.J.F. Ypma and dr. J. Wallinga from the RIVM. During the last one and a half month I was also supervised by dr. J. van de Kassteele from the RIVM. During this period ir. R.J.F. Ypma, my first supervisor from the RIVM, was not physically present because of a move to Cambridge (UK). Despite his absence he helped me by email to finalize my research. The second reader of my thesis from the Mathematical Institute of Utrecht University was prof. dr. O. Diekmann.

Contents

1 Introduction

To gain insight in the spread of an infectious disease, it is important to know who infected whom. A transmission tree can be used to describe these transmission events between hosts, see Figure 1 for an example. For most diseases the complete transmission tree cannot be observed because of missing data. However, due to its great value for understanding the disease dynamics, epidemiological data, like the time and source of infection, are often used to reconstruct the transmission tree. Another valuable method, which has become more popular in recent years, is to estimate the transmission tree based on genetic data of the pathogen, i.e., the micro-organism causing the disease. The genetic differences between the pathogens obtained from different patients and/or at different time points are used to relate these micro-organisms evolutionarily; the larger the difference, the further away the pathogens are in the tree. While in a transmission tree infected hosts are related, the phylogenetic tree relates the pathogens within these hosts based on their genetic distance. The field of study that relates organisms evolutionarily is called phylogenetics.

In this thesis we look at the spread of hepatitis B among men who have sex with men (MSM) in the Netherlands. The aim is to investigate what role acutely and chronically infected individuals play in the spread of the disease. Are most infections caused by acutely infected hosts which are often sexually active but infected for a short period in time? Or is the disease mostly spread by chronically infected hosts which are infectious for a long time? For the effectiveness of the current vaccination program these are important questions. It is difficult to answer these questions by standard epidemiologic methods because the infection is often asymptomatic and most transmission events are unobserved. We use genetic data of the pathogen obtained from both acutely and chronically infected individuals to construct a phylogenetic tree and to infer the corresponding transmission dynamics.

In Section 2 we will start with some background information about genetics and we will introduce some basic concepts. Although hepatitis B is a virus, most concepts will hold for pathogens in general. We therefore make no distinction between them. An overview of hepatitis B and the data used is given in Section 3. In Section 4 we discuss the model used to construct the phylogenetic tree from our data. In Section 5 two models are introduced for quantifying the transmission dynamics from the phylogenetic tree. The phylogenetic tree generated from our data and the corresponding results using the two models for quantifying the transmission dynamics are discussed in Section 6. Finally, in Section 7, we discuss the models and give recommendations for future work.

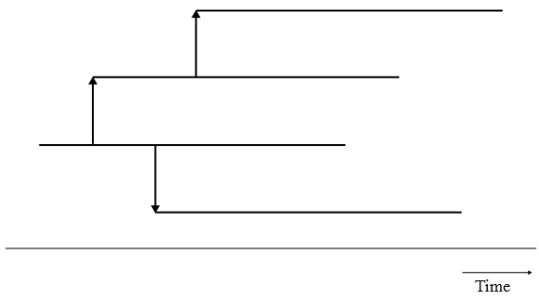

Figure 1: Transmission tree with time going to the right. Each branch corresponds to one infectious individual. During the infectious period, an individual can infect others (denoted by a vertical arrow). The duration of the infectious period equals the length of the branch; at the end of a branch the individual recovers or dies. In this tree one index case infects two individuals and one of these infects another individual. Note that we assume that an infected individual is immediately infectious after infection.

2 Genetics

DNA, or deoxyribonucleic acid, is a molecule that encodes the hereditary (or genetic) information of most living organisms. It usually consists of two complementary strands twisted around each other to form a double helix. Each strand is a sequence of four different nucleotides: adenine (A), guanine (G), cytosine (C) and thymine (T). The nucleotides in one strand pair with the nucleotides in the other strand in the combinations adenine-thymine (A-T) and cytosine-guanine (C-G) via two and three hydrogen bonds respectively. Therefore the A-T bond separates more easily.

The complete package of all hereditary information, coded by a sequence of nucleotides, is called the genome of an organism. Part of this string are genes, combinations of nucleotides coding for a protein, which are situated on the chromosomes in the nucleus of our cells. The other and biggest part of the genome has no known function. The genes are transcribed into ribonucleic acid (RNA) for translation into proteins. RNA is also a molecule encoding the genetic information. Some differences with DNA are that RNA is a single-stranded molecule and that the nucleotide uracil (U) replaces thymine (T). Pathogens use DNA as a carrier of their genetic information, except for many viruses that use RNA to encode their hereditary information [1]. Nevertheless, hepatitis B is a DNA virus and it has the rare property that it is partially double stranded. The full-length genome consists of 3020 to 3320 nucleotides [2].

DNA sequences are subject to evolutionary processes like resampling, recombination, mutation, selection and migration. During resampling, or reproduction, genetic information is passed on from one generation to the next. This process depends on the type of organism. For haploid organisms like bacteria, there is only one copy of the genetic material in the chromosomes. This means that at resampling there is only one parent and a copy of its genetic information is passed on to its offspring. For diploid organisms like humans, each chromosome has two copies of its genetic material. In this case, during resampling both parents pass on one of each of their pairs of chromosomes to their offspring. It is possible for a parent to pass on a combination of both of its chromosomes; the genetic material between the two copies of a chromosome is exchanged. This is called recombination. Since recombination is very complicated and it doesn't seem to occur in our data, we ignore this in our study.

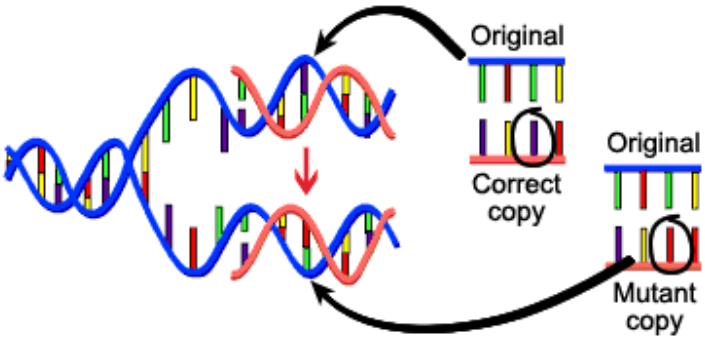

Figure 2: Replication of a DNA strand with a mutation in one of its replicated strands [3].

Another evolutionary process is mutation. A mutation corresponds to a change in the order or content of the nucleotides in a DNA sequence. For example, during replication of the DNA sequence in which the two strands are pulled apart and along both strands a new complementary chain is formed, a wrong nucleotide can be inserted. These substitutions can be between two nucleotides of the same type (A $\leftrightarrow$ G, both purines or C $\leftrightarrow$ T, both pyrimidines) which are called transitions. Transversions are all other possible nucleotide substitutions and occur less frequently. An illustration can be found in Figure 2. It is also possible during replication that one or more nucleotides are added to or omitted from the replicating strand. These types of mutations happen spontaneously. However, it is also possible that these mutations are caused by environ-

mental exposure to certain chemicals, ultraviolet radiation, or other external factors [4]. Mutations play an important role in this study, we will discuss this in the next section.

When different gene types or alleles have different predispositions for resampling, this is called selection. For example, one type could have a higher rate of reproduction. In this study we assume that there is no selection. This implies neutral mutations; mutations in the pathogen sequence don't change its rate of reproduction and its ability to survive.

Finally it is also possible that genetic material is spread from one population to another due to migration. We will see some kind of migration in the models used for estimating the transmission dynamics in Section 5.

2.1 Phylogenetic tree

To get an idea of the transmission tree of an infectious disease a phylogenetic tree can be constructed. While a transmission tree represents the transmission of an infectious disease between hosts, a phylogenetic tree relates the pathogen sequences within these hosts. Once a pathogen enters a host, replication can take place. During this process of replication, mutations can occur resulting in new pathogens with different genetic sequences. These mutated pathogens can be passed on to other hosts when new infections take place.

Like mentioned before we assume there is no selection, i.e., we have neutral mutations. Therefore we can assume that the mutation rate, the rate at which a nucleotide in a single pathogen mutates, equals the substitution rate. This is the rate at which a mutation is fixed in the total within-host pathogen population. The substitution rate is also assumed to be constant over time and among different pathogens, this is called the molecular clock hypothesis [5]. In case of selection we can't assume that the mutation and substitution rate are equal, since it is possible that the mutated pathogen goes extinct because of another within-host pathogen with better predispositions for resampling. In this case a mutation may result in no substitution.

Due to mutations, genetic sequences of pathogens sampled from different hosts differ. The less differences the strands have, the more likely it is that one of the hosts (indirectly) infected the other. How much the genetic sequences differ is measured by a possibly complicated function of the two genetic sequences, see Section 4. The genetic distance is indicative for how far the sequences are apart in the phylogenetic tree.

An example of a phylogenetic tree is shown in Figure 3. In this figure time goes to the right. The ends of the tree, the leaf nodes (taxa), represent the genetic sequences of the pathogens sampled from different hosts and the times at which they are sampled. When the sequences are sampled at the same time, we speak of homochronous sampling. When the sampling times aren't equal, this is called heterochronous sampling. The samples are related by their ancestral lineages, based on the genetic distances. A bifurcation event denotes a replication event; the pathogen corresponding to the branch before the bifurcation replicates and gives birth to a new pathogen initiating one line of descent and continues in the second 'descending' lineage itself. It is unknown to which of these two pathogens each of the two descending lineages belongs. These bifurcation events therefore denote the common ancestors of two or more of the sampled pathogens. The left-most node of the tree, the root, equals the Most Recent Common Ancestor (MRCA) of all genetic sequences. Note that a bifurcation event in the phylogenetic tree represents a replication within one host. However, infection of a new host has to take place (shortly) after the bifurcation because the sampled pathogens are from different hosts. In a transmission tree a branching event denotes an infection between different hosts.

The phylogenetic tree only depicts the ancestral relationships between sequences which are sampled. Actually, there will be hosts infected with the pathogen which aren't sampled. Therefore it could have been that a bifurcation took place at one of the branches despite it isn't visible in the phylogenetic tree. For example, suppose that there was a bifurcation at branch a in Figure 3, marked by the dotted line. There are two possible situations. The first one is that the replicating pathogen initiates the line of descent which

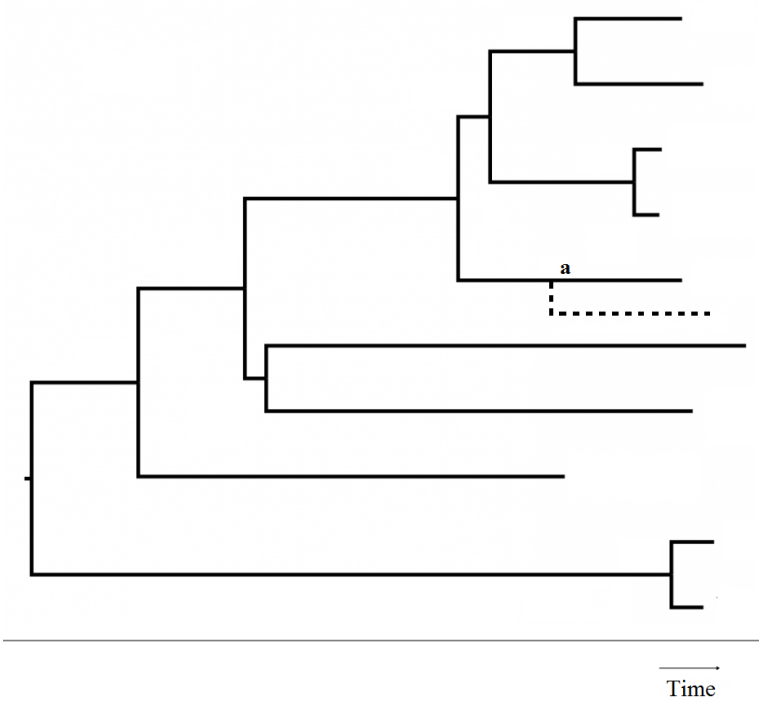

Figure 3: A phylogenetic tree with time going to the right. Genetic sequences of the pathogens sampled from different hosts are represented by the leaves of the tree and are related by their ancestral lineages. The root of the tree denotes their Most Recent Common Ancestor (MRCA). Each bifurcation event denotes a replication of a pathogen within a host. The dashed line descending from branch a belongs to an unsampled infection, also called an invisible infection.

is eventually sampled (black solid line) and its descendent initiates the dotted line which isn't sampled and has no sampled descendants. In the other case the replicating pathogen initiates the line which isn't sampled (dotted line) and its descendent initiates the line which is eventually sampled (black solid line).

The transmission and phylogenetic tree can differ in the times of the internal nodes and topology. This difference is the largest in case a large fraction of the total number of infected hosts has been sampled. The times of the internal nodes for both trees are almost the same in case this sample fraction is low [6]. For our study the sampling fraction is low, therefore we expect the phylogenetic tree to be indicative for the transmission tree. However, for the models described in Section 5 we need that the transmission times equal the bifurcation times in the phylogenetic tree and that each infected host corresponds to a single lineage, i.e., one branch of the phylogenetic tree. Therefore we will assume that each host has only one pathogen, which can mutate during the time within this host, and equal rates for replication of the pathogen and transmission between hosts. An infection in the phylogenetic tree will then always be between hosts.

3 Data

3.1 Hepatitis B

Hepatitis B is an inflammation of the liver which is caused by the hepatitis B virus (HBV). The virus is spread through contact with infected body fluids, for example by unsafe sexual contact, blood transfusion, from mother to child during birth, by puncture wounds or by shared use of needles [7]. When being deposited on a surface, the virus can stay infectious for about a week [8].

In developing countries with a high number of HBV infections, for example in sub-Saharan Africa and East Asia, the most common routes of transmission are from mother to child during birth or from person to person during childhood. In countries with a high standard of living, for example in Western Europe or North America, infection is mostly spread due to unsafe sexual contacts and by injecting drug users [9]. There are also differences in the genotype, the genetic information, of the virus between regions. These genotypes are labeled from A to H. In the Netherlands the most common genotypes are A and D, the first is associated with MSM and injecting drug users, the second with the Moroccan population in Amsterdam [10].

An infection with hepatitis B has several clinical stages. The time between infection and symptoms onset, or incubation time, depends on the initial viral dose and is about 6 to 26 weeks. When one gets infected with HBV, there are two different types of infection: acute hepatitis B with symptoms like tiredness, fever, joint complaints and jaundice (yellowing of the skin and eyes) or an acute infection with mild or even no symptoms [11]. About two-thirds of patients infected with HBV will have an asymptomatic acute infection. The average time of an acute HBV infection is 3-4 months [12] and afterwards most people recover. These recovered people obtain life-long immunity. About 1% of the acutely infected patients develop fulminant hepatitis, a sudden but intense infection, often with death as a result [13].

However, sometimes the infection is still present after 6 months and the acute infection will progress to a chronic infection. This is thought to happen in 5% of the cases [13], although a recent study [14] suggests this to be 23% and 28% for MSM and drug users respectively. Most of them will become an asymptomatic carrier of the virus, some others will suffer complaints like impairment of the liver (hepatic impairment), high blood pressure of the portal vein and chronic fatigue. 25-35% of the chronically infected individuals develop cirrhosis or liver cancer, resulting in an early death for 15-25% of them [11].

The course of the infection depends on the viral replication of HBV in the liver while the severity of the course depends on the immune response of the host. For example, impairment of the liver isn't indicative of the virulence of the virus but is caused by a strong cellular immune response of the host. The probability that an acute infection proceeds to chronic carriage differs between patients, depending on their age, immune status and gender. Also, an asymptomatic acute infection is more probable to develop to a chronic infection. 70-90% of the infected newborns (mostly in South East Asia) become chronic carriers. For children under age five this is 25-50% and for older children and adults just 5-10%. Furthermore the probability of developing chronic carriage is six times higher for men than for women. For acutely infected patients with less immunity or with an HIV infection the virus replication continues and chronic carriage is a result. In these cases the virus replication is higher but the severity of the infection is less [13].

In the Netherlands the number of new infections per year, the incidence, is relatively stable since 1990. Before 1981 there was a strong increase in the incidence followed by a strong decrease from 1981 until 1990. This strong decline can be partially explained by a change in sexual behavior due to the AIDS epidemic and is also partially due to the introduction of an effective vaccine in 1982. In 2012 in 61% of the acute HBV cases the route of transmission was unsafe sexual contact [11, 15].

To prevent the spread of hepatitis B the Netherlands uses screening of pregnant women, vaccination of newborns and vaccination of risk groups. In 1989 the screening of pregnant women is introduced nationally

in order to prevent HBV transmission from mother to child during the birth. Most of these infections won't be noted while in most cases the child will become chronic carrier of the virus with a realistic chance to die early. When HBV is found in a pregnant women the newborn will receive antibodies and a first vaccination against the virus. Nationwide vaccination of risk groups has been started in 2002. This program is directed to MSM, hard drugs users (both injecting and swallowing) and prostitutes (men and women). These risk groups have an enhanced probability to get infected with HBV due to their behavior. Since 2012 hard drugs users are excluded from this vaccination program because they weren't considered to be a risk group anymore [16]. In Amsterdam a vaccination program targeted against HBV in MSM already started in 1998 [12]. Since 2000 all medical staff are legally entitled to a vaccination. After the first of August of 2011 all newborns are vaccinated against HBV. Since 2003 the vaccination is already part of the National Immunisation Programme (NIP) for children who have an enhanced probability to get infected with the disease. These are children from mothers who are carrier of HBV and children from which at least one of the parents originates from a region with a high presence, prevalence, of HBV. Also children with the Down syndrome have a higher probability to get infected. These children are also included in the NIP since 2008. The vaccination protects for at least 25 years, even life-long immunity is expected. It is expected that the selective vaccination of risk groups and the recent introduction of vaccinating all newborns, will have an effect on the number of reported HBV cases. The effect on the prevalence will only be visible in the long-run [11, 17].

3.2 Data collection

In this study we want to investigate what role acutely and chronically infected individuals play in the spread of hepatitis B among MSM in the Netherlands. To answer this question we use DNA sequences of the virus collected from both acutely and chronically infected individuals. The data used in this study consists of 57 full-length genetic sequences of the hepatitis B virus collected from individuals with an acute HBV infection and 27 full-length genetic sequences collected from individuals with a chronic HBV infection, all associated with genotype A and 3221 nucleotides long. These genetic sequences are collected from blood samples taken from acutely or chronically infected MSM. 30 blood samples, from which 26 chronic, are obtained from the Amsterdam Cohort Studies, the other 54 blood samples are collected from the Municipal Health Services (GGD) in different parts of the Netherlands. Corresponding to these genetic sequences we also have epidemiological data like the sampling date (the date at which the blood sample has been taken from the patient) and place, personal information like birth date, age and postal code, the number of partners in the last 6 months, the most likely route of transmission and more. However, most of the information is only available for a part of our blood samples. The sampling dates for the chronically infected individuals range between 1985 and 2005; more than a half of these are sampled in 1985. For the acutely infected individuals the sampling dates range between 1986 and 2011, mostly after 2000.

4 A model to construct the phylogenetic tree from genetic data

In a phylogenetic tree pathogen sequences from infected hosts are evolutionarily related. The higher the genetic distance between two DNA sequences, the more they are apart from each other in the tree. A simple way of measuring this genetic distance is by counting the number of different nucleotides, i.e., the number of mutations. However, this measure only takes those mutations into account which are observable, not those who disappeared through time because of a second mutation on the same site. Another lack of this measure is that it is assumes that all mutations occur with the same frequency [18]. A more reasonable model describing the genetic distance is the Hasegawa-Kishino-Yano (HKY) model [19].

The Hasegawa-Kishino-Yano (HKY) model accounts for the fact that transitions, replacements of a purine by the other purine (A $\leftrightarrow$ G) or of a pyrimidine by the other pyrimidine (C $\leftrightarrow$ T), occur more often than transversions, which are replacements of a purine by a pyrimidine or vice versa. Furthermore this model corrects for different base frequencies, i.e., the nucleotides A, T, C and G can occur at different frequencies in the sequence. Changes in the nucleotides of a pathogen sequence evolve according to a Markov chain process. The corresponding transition matrix P, where $P_{i,j}$ denotes the probability that a base i is substituted by a base j in one small unit time, equals

$$P = \begin{array}{c} \\ A \\ T \\ C \\ G \end{array} \begin{array}{c} \begin{array}{cccc} A & T & C & G \end{array} \\ \begin{bmatrix} 1-(a\pi_G+b\pi_T+b\pi_C) & b\pi_T & b\pi_C & a\pi_G \\ b\pi_A & 1-(a\pi_C+b\pi_A+b\pi_G) & a\pi_C & b\pi_G \\ b\pi_A & a\pi_T & 1-(a\pi_T+b\pi_A+b\pi_G) & b\pi_G \\ a\pi_A & b\pi_T & b\pi_C & 1-(a\pi_A+b\pi_T+b\pi_C) \end{bmatrix} \end{array},$$

so the probability that at time $t+\Delta t$ a base equals i (with $i \in \{A,T,C,G\}$), denoted by p_i, equals

$$(p_A, p_T, p_C, p_G)(t+\Delta t) = (p_A, p_T, p_C, p_G)(t)P.$$

Here a and b denote the probabilities of transitions and transversions respectively and π_i equals the stationary frequency of nucleotide i ($i \in \{A,T,C,G\}$) in the genetic sequence so $\pi_A + \pi_T + \pi_C + \pi_G = 1$. Furthermore, $2b + a < 4$ in order to prevent that the probabilities to change from a base to itself become zero. The transition probabilities are independent of the state before the time of the transition and because of the molecular clock hypothesis our transition probabilities are also independent of the time of the transition. With these assumptions the substitutions in this HKY model indeed evolve like a Markov chain process. The substitution rates at different sites of the genetic sequence, i.e. positions, are assumed to be equal.

With the transition matrix P the expected genetic distance between two sequences, the average number of substitutions per site, can be calculated. Since the branch lengths in a phylogenetic tree are in units of the expected substitutions per site, the genetic distances between all sampled sequences can be used to construct the phylogenetic tree. Because of the molecular clock hypothesis that assumes a constant substitution rate, this distance can be translated into a distance in time.

In this study a phylogenetic tree is obtained by the use of BEAST [20], TreeAnnotator which is part of the BEAST package [20], R [21] and FigTree [22]. For the technical details see Appendix A.

5 Estimation of the transmission dynamics given the phylogenetic tree

There are several models to estimate the transmission dynamics from a phylogenetic tree. The two models used for this study are based on a multi-type birth-death branching model introduced by Tanja Stadler et al. [23] and a coalescent model introduced by Erik Volz [24]. The main idea is to calculate the likelihood of the model parameters given the phylogenetic tree and find the parameter set which maximizes the likelihood. The optimization methods we use for finding the maximum likelihood estimators of the parameters are discussed at the end of this section.

When we mention the host or infected population we consider only the part of the total population which is infected with the disease. The total population can denote a certain risk group, in our specific case MSM in the Netherlands.

5.1 Multi-type birth-death branching model

We will start with introducing the multi-type birth-death branching model for m different types. Afterwards we will focus on the two-type birth-death branching model that represents the transmission dynamics of hepatitis B.

5.1.1 Multi-type birth-death branching model with m different types

The multi-type birth-death branching (MTBD-m) model describes the transmission dynamics of a host population with m different types (transmission groups), also called states. All individuals are characterized by their type and secondary infections take place with type-dependent birth (transmission) rates. At the beginning of the process there is only one individual with an initial type. The rate at which an individual of type i gives birth to an individual of type j equals $\lambda_{i,j}$. Individuals of type i die with rate d_i. Directly after death an individual of type i may be sampled with probability s_i. In epidemiological context, death is defined as becoming-non-infectious due to host death, recovery, a change in behavior or successful treatment. Sampling an individual means that the pathogen sequence of this infected host is used for deducing the phylogenetic tree. At time t_0 the process stops. Furthermore, we assume that an individual can change from type i to type j with rate $\gamma_{i,j}$ ($\gamma_{i,i} = 0$ for $i \in \{1, \ldots, m\}$), due to migration between geographical locations or a change in infection state.

To calculate the likelihood of a parameter set given the phylogenetic tree two definitions are needed. Before we state these, note that time is going backwards. Time at present equals zero and time is increasing going into the past. For example, if a birth event occurs at time t this means that the time between this event and the present (time 0) equals t.

Definition 1. *For an individual represented by a branch N at time t in state i ($i \in \{1, \ldots, m\}$), $D_{Ni}(t)$ equals the probability density that this individual gave rise to the phylogenetic tree as given between time t and the present (time 0).*

Definition 2. *$E_i(t)$ equals the probability that an individual in state i ($i \in \{1, \ldots, m\}$) is not sampled and has no sampled descendants between time t and the present (time 0).*

Backwards in time, starting from the leaf nodes, $D_{Ni}(t)$ can be derived along the branches of the phylogenetic tree ignoring all future events with time greater than t. Suppose that a leaf node in state j has been sampled at time τ in the past. Then, the probability density that an individual at time τ in state i produces the phylogenetic tree as observed equals

$$D_{Ni}(\tau) = \begin{cases} d_i s_i, & \text{if } j = i \\ 0, & \text{if } j \neq i \end{cases}. \tag{1}$$

The individual has to be sampled at time τ and has to be in state j. Furthermore, sampling only occurs immediately after becoming-non-infectious. For $t > \tau$, i.e., time t is further back in time than time τ, we can set up the differential equations for $D_{Ni}(t)$. Below we derive an equation for $D_{Ni}(t + \Delta t)$ based on $D_{Ni}(t)$ where Δt denotes a small positive time step. To derive this equation we use that along an arbitrary branch N and during a small time interval Δt five different 'events' can occur, each corresponding to one of the five terms in the equation below. The first term represents that no birth or death event happens and that the individual doesn't change state. The second possible event is the birth of an individual in state j whose lineage produces no samples in time t. The third term represents the birth of an individual in state j while the transmitting individual i produces no samples in time t. The fourth possible event represents the transition of an individual in state i to state j. The possibility of more than one event during the interval Δt, for example multiple births, is included in the last term.

$$\begin{aligned} D_{Ni}(t+\Delta t) &= \left(1 - \left(\sum_{j=1}^{m}(\lambda_{i,j} + \gamma_{i,j}) + d_i\right)\Delta t\right) D_{Ni}(t) + \sum_{j=1}^{m} \lambda_{i,j}\Delta t E_j(t) D_{Ni}(t) \\ &\quad + \sum_{j=1}^{m} \lambda_{i,j}\Delta t E_i(t) D_{Nj}(t) + \sum_{j=1}^{m} \gamma_{i,j}\Delta t D_{Nj}(t) + \mathcal{O}(\Delta t^2) \end{aligned}$$

For $\Delta t \to 0$ the differential equation for $D_{Ni}(t)$ becomes

$$\begin{aligned} \frac{d}{dt}D_{Ni}(t) &= -\left(\sum_{j=1}^{m}(\lambda_{i,j} + \gamma_{i,j}) + d_i\right) D_{Ni}(t) + \sum_{j=1}^{m} \lambda_{i,j} E_j(t) D_{Ni}(t) + \\ &\quad \sum_{j=1}^{m} \lambda_{i,j} E_i(t) D_{Nj}(t) + \sum_{j=1}^{m} \gamma_{i,j} D_{Nj}(t). \end{aligned} \tag{2}$$

This differential equation includes $E_i(t)$, the probability that an individual in state i is not sampled and has no sampled descendants between time t and the present (time 0). In an analogous way as for $D_{Ni}(t)$, we can find a differential equation for $E_i(t)$. As an initial condition, so at time 0, $E_i(t)$ equals 1 for all possible types. The individual at the present cannot be sampled since then he/she would have become non-infectious and therefore would have been removed. However, he/she is still present now and possibly in the future. So,

$$E_i(0) = 1 \text{ for all } i \in \{1, \ldots, m\}. \tag{3}$$

For time t and given $E_i(t)$ the formula for $E_i(t + \Delta t)$ becomes

$$\begin{aligned} E_i(t+\Delta t) &= (1 - s_i) d_i \Delta t + \left(1 - \left(\sum_{j=1}^{m}(\lambda_{i,j} + \gamma_{i,j}) + d_i\right)\Delta t\right) E_i(t) + \sum_{j=1}^{m} \lambda_{i,j}\Delta t E_i(t) E_j(t) + \\ &\quad \sum_{j=1}^{m} \gamma_{i,j}\Delta t E_j(t) + \mathcal{O}(\Delta t^2). \end{aligned}$$

In this equation the first term represents death without sampling. The second term equals the probability of no birth, death or state change in time Δt times the probability that the lineage does not produce samples in time t. The third term is the probability of a birth of an individual in state j where both lineages don't produce samples in time t. The fourth term represents the probability of a change in state from i to j times the probability that the lineage does not produce samples in time t. The fifth term equals the probability for more than one event during time Δt. When we let $\Delta t \to 0$, we obtain as a differential equation for $E_i(t)$

$$\frac{d}{dt}E_i(t) = (1-s_i)d_i - \left(\sum_{j=1}^{m}(\lambda_{i,j}+\gamma_{i,j})+d_i\right)E_i(t) + \sum_{j=1}^{m}\lambda_{i,j}E_i(t)E_j(t) + \sum_{j=1}^{m}\gamma_{i,j}E_j(t). \quad (4)$$

Now we have the differential equations for both $D_{Ni}(t)$ and $E_i(t)$ backwards in time, we can calculate these probabilities along the branches of the phylogenetic tree starting from the leaf nodes. At a bifurcation event A in state i and at time t we want to calculate the probability $D_{Ai}(t)$ of obtaining the two subtrees descending from this event. Suppose that the two branches descending from the bifurcation event are K and M. Then there are two possibilities. Either the individual that initiates branch M has given birth to the individual initiating lineage K, or the individual that initiates branch K has given birth to the individual initiating lineage M. This yields

$$D_{Ai}(t) = \sum_{j=1}^{m}\left(\lambda_{i,j}D_{M,i}(t)D_{Kj}(t) + \lambda_{i,j}D_{M,j}(t)D_{Ki}(t)\right). \quad (5)$$

This probability is subsequently used as the initial value for the branch starting from the bifurcation event. Along this branch the change in the probability can then be calculated by using the differential equations for both $D_{Ni}(t)$ and $E_i(t)$ again. This process repeats itself for all bifurcation events and along all branches of the phylogenetic tree until the root of the tree is reached. For the change in $E_i(t)$ we can use its differential equation since the probability that an individual in state i is not sampled and has no sampled descendants between time t and the present (time 0) doesn't depend on how the phylogenetic tree evolves. Actually, it is only depending on the time until present, state specific parameters and $E_i(t)$ for all possible states.

Using Equations (2), (4) and (5) together with the initial conditions (1) and (3) and going backwards in time until the origin of the tree is reached, we can yield the probability density of the whole phylogenetic tree given the root is in state i, $D_{Oi}(t_0)$, for $i \in \{1,\ldots,m\}$. The overall probability density of the phylogenetic tree equals

$$p\left(\mathcal{T}|\lambda,d,s,t_0\right) = \sum_{i=1}^{m} f_i \frac{D_{Oi}(t_0)}{1-E_i(t_0)}, \quad (6)$$

where f_i equals the fraction of individuals at time t_0 being in state i. The probability density $D_{Oi}(t_0)$ is in this expression divided by the probability that the individual at time t_0 gives rise to at least one sampled individual. Conditioning on this gives more accurate rate estimates [25]. This makes sense since analyzing non-sampled transmission chains is not of interest.

In this study we assume f_i to be equal to the equilibrium frequencies for all $i \in \{1,\ldots,m\}$.

5.1.2 Hepatitis B specific two-type birth-death branching model

As discussed in Section 3 a person infected with hepatitis B first becomes acutely infected. After roughly three to four months the infection is either cleared or the infection progresses to a chronic infection. Since an infection can be in two states, the acute or the chronic state, we can translate our transmission dynamics into a two-type birth-death branching model with the parameters given in Table 1. In this table state 1 and state 2 denote the acute and chronic state respectively.

Note that in this table the parameters $\lambda_{1,2}$, $\lambda_{2,2}$ and $\gamma_{2,1}$, which denote the rates at which acutely and chronically infected individuals give birth (transmission) to a new chronically infected individual and the rate at which a chronically infected individual progresses to the acute state respectively, aren't given. Since

$\lambda_{1,1}$	the rate at which an acutely infected individual gives birth (transmission) to a new acutely infected individual
$\lambda_{2,1}$	the rate at which a chronically infected individual gives birth (transmission) to a new acutely infected individual
d_1	the rate at which an acutely infected individual dies (becoming-non-infectious due to host death, recovery, behavior change or successful treatment)
d_2	the rate at which a chronically infected individual dies (becoming-non-infectious due to host death, recovery, behavior change or successful treatment)
$\gamma_{1,2}$	the rate at which an acutely infected individual progresses to the chronic state.

Table 1: The parameters for the two-type birth-death branching model that describes the transmission dynamics of hepatitis B. The acute and chronic state are denoted by 1 and 2 respectively.

an infected person always becomes acutely infected with hepatitis B first and it isn't possible to move from the chronic to the acute state, these parameters equal zero and are, therefore, irrelevant to the model. Besides these parameters we also have the parameters s_1 and s_2 which denote respectively the probability to sample either an acutely or a chronically infected individual directly after death.

For the two-type birth-death branching model the reproduction number R_0, the expected number of secondary infections caused by a single infected individual in a completely susceptible population, equals the expected number of infections caused by an individual in the acute state plus the probability that an acutely infected individual progresses to the chronic state multiplied with the expected number of infections caused by a chronically infected individual. The number of infections caused by an individual in state i ($i \in \{1, 2\}$) has a geometric distribution modeling the number of trials (infections) before a success (leaving state i). The success probabilities equal

$$\frac{d_1 + \gamma_{1,2}}{d_1 + \gamma_{1,2} + \lambda_{1,1}} \text{ and } \frac{d_2}{d_2 + \lambda_{2,1}}$$

for acutely and chronically infected individuals respectively. In these formulas the rates of leaving the acute or chronic state due to death or a change in state are divided by the total rate at which an acutely or chronically infected individual causes a birth, death or state change event. The average number of infections caused by an infected individual in the acute or chronic state therefore equal

$$\frac{\lambda_{1,1}}{d_1 + \gamma_{1,2}} \text{ and } \frac{\lambda_{2,1}}{d_2}$$

respectively. The reproduction number R_0 then becomes

$$R_0 = \frac{\lambda_{1,1}}{d_1 + \gamma_{1,2}} + \frac{\gamma_{1,2}}{d_1 + \gamma_{1,2}} \frac{\lambda_{2,1}}{d_2}.$$

This reproduction number doesn't depend on the fractions of the individuals in the acute and chronic state at time t_0 because the infectious period of an infected individual always starts with an acute period. However, these fractions are needed for calculating the likelihood in Equation (6) with $m = 2$. In case of equilibrium frequencies, f_1 and f_2 equal

$$f_1 = \frac{c + \Lambda}{c + \Lambda + 2\gamma_{1,2}}, \qquad f_2 = 1 - f_1 = \frac{c - \Lambda}{c - \Lambda + 2\lambda_{2,1}},$$

where $\Lambda = \lambda_{1,1} - (d_1 + \gamma_{1,2}) + d_2$ and $c = \sqrt{\Lambda^2 + 4\gamma_{1,2}\lambda_{2,1}}$. For the expressions corresponding to the general two-type birth-death branching model, see [23].

For this two-type birth-death model calculation of the likelihood in Equation (6) of a certain parameter set given the phylogenetic tree is performed by the bdtypes.stt.lik-function in the R package TreePar [26], adapted to the possibility for an individual to change state. The calculations were performed in R 3.0.1 [21]. The codes are given in Appendix C.2.

Variable sampling fraction over time

The multi-type birth-death branching model assumes a constant sampling fraction over time which may differ for acutely and chronically infected individuals. A problem with this method is the unknown sampling fraction. Furthermore, the data used for this study isn't collected at a constant rate over time; the chronic samples are mainly collected between 1985 and 1990 while the acute samples are mainly collected after 2000.

In the two-type birth-death branching model sampling is only possible immediately after an individual has become non-infectious. Therefore, at each time, the number of sampled infected individuals in state i equals the number of individuals in state i who became non-infectious times the fraction of infected individuals in state i sampled. With the extra assumption that we have a constant population of acutely and chronically infected individuals over time, the number of infected individuals in state i becoming-non-infectious is also constant. This implies that the number of sampled individuals is directly proportional to the sampling fraction. The constant sampling fraction for an individual in state i can therefore be replaced by

$$s_i = c_i \cdot (\text{number of sampled individuals from state i}),$$

which varies over time due to a possibly different number of sampled individuals for each time interval. In this formula c_i is a constant. The number of sampled individuals during each time interval is known from our data. Note that c_i equals one divided by the number of infected individuals in state i that become non-infectious each time interval. For the adapted two-type birth-death branching model, with extra parameters c_1 and c_2, we use different sampling fractions per calendar year.

We calculated the likelihood in Equation (6) for the two-type birth-death branching model with variable sampling fractions over time by the use of R 3.0.1 [21]. Codes can be found in Appendix C.3.

5.2 Coalescent model

The coalescent model used in this study assumes the population dynamics to behave like an SIR model. After an introduction of the SIR model, we will describe the coalescent model for m different types. We will end the section with the coalescent model that represents the transmission dynamics of hepatitis B.

5.2.1 SIR model

In the SIR model the total population N is divided into three compartments: susceptible (S), infected (I) and recovered (R) individuals. When an individual is susceptible this means that he/she hasn't been infected with the disease yet and is susceptible for the disease. Individuals in the infected category are those who are infected with the disease and are still able to spread the disease to those in the susceptible compartment. In the recovered compartment are those individuals who recovered and developed immunity to the disease or those who died as a result of the disease. It is not possible for recovered individuals to infect others or become infected again. By assumption the average time an individual resides in each of these compartments is exponentially distributed. In Figure 4 the typical course of an infectious disease for the SIR model is represented if the initial number of infected and recovered individuals is small compared to the population size and R_0, the reproduction number, is bigger than 1.

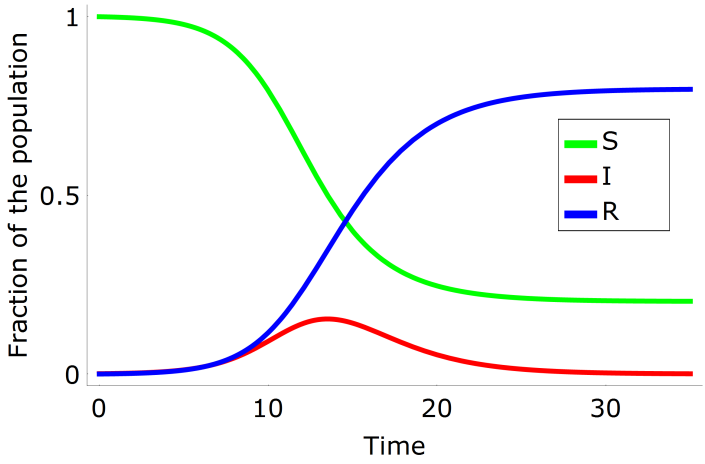

Figure 4: Typical dynamics of the population according to the SIR model [27]

At the beginning of an epidemic the number of susceptibles is high since no one has been infected yet. Once a small fraction of the population becomes infected, the infectious disease is spreading exponentially fast since the possibility of infecting a susceptible individual is high. This results in a fast decrease in the number of susceptibles. Shortly after the first individuals become infected, also the number of recovered individuals increases exponentially. At a certain point in time a lot of individuals are already infected and/or recovered so the number of susceptible individuals is declining. The number of infected individuals decreases, resulting in a slower increase in the number of recovered individuals and a slower decrease in the number of susceptibles. At the end of the epidemic all individuals who were infected are recovered while there are still some individuals who weren't infected by the disease at all.

The SIR model introduced by Kermack and McKendrick [28] can be represented by a system of ordinary differential equations (7). The assumptions made for this model are a constant closed population over time (no births or natural deaths), all susceptibles are equally susceptible, complete immunity is conferred by a single infection and infected individuals can infect others during the time they have the disease, excluding the time at which they get infected. $S(t)$, $I(t)$ and $R(t)$ denote the numbers of susceptible, infected and

recovered individuals at time t. We also need these numbers to be sufficiently large in order to approximate them by a system of ordinary differential equation.

$$\begin{aligned}
\frac{d}{dt}S(t) &= -\beta\frac{S(t)}{N}I(t) \\
\frac{d}{dt}I(t) &= \beta\frac{S(t)}{N}I(t) - \gamma I(t) \\
\frac{d}{dt}R(t) &= \gamma I(t)
\end{aligned} \tag{7}$$

Here β denotes the rate at which an infected individual infects new individuals at the beginning of the epidemic when almost all individuals are susceptible. γ denotes the rate at which an infected individual recovers or dies from the disease.

5.2.2 Coalescent model with m different types

The coalescent model used in this study is a continuous time birth-death process with varying rates and can be used to describe the transmission dynamics of a host population with m different types. Forwards in time, time is denoted by t and an individual in state i infects an individual in state j with rate $\beta_{i,j}$ in case almost the total population is susceptible. An individual in state i recovers with rate γ_i and state transition, a change in state during infection, from state i to state j happens with rate $\gamma_{i,j}$ ($\gamma_{i,i}$=0 for $i \in \{1,\ldots,m\}$). Time into the past is denoted by s. If two branches merge into one backwards in time this is called a coalescent event; two lineages coalesce. In this model the likelihood of a set of parameters given the phylogenetic tree equals the probability of observing the coalescence events in the phylogenetic tree. The likelihood therefore depends on the rate that two lineages coalesce and on the probabilities that lineages haven't coalesced prior to the coalescent event. These rate and probabilities are calculated backwards in time by conditioning on the sampling times. It is assumed that at the moment of sampling an infected individual is also recovered.

The SIR model that describes the population dynamics of this coalescent model can be represented by a set of $m+2$ differential equations, see Formula (8). For the moment we assume no births or natural deaths, later on we will add these events to the model (see Section 5.2.3).

$$\begin{aligned}
\frac{d}{dt}S(t) &= -\frac{S(t)}{N}\left(\sum_{i=1}^{m}\sum_{j=1}^{m}\beta_{i,j}I_i(t)\right) \\
\frac{d}{dt}I_1(t) &= \frac{S(t)}{N}\left(\sum_{i=1}^{m}\beta_{i,1}I_i(t)\right) + \sum_{i=1}^{m}\gamma_{i,1}I_i(t) - \sum_{i=1}^{m}\gamma_{1,i}I_1(t) - \gamma_1 I_1(t) \\
\frac{d}{dt}I_2(t) &= \frac{S(t)}{N}\left(\sum_{i=1}^{m}\beta_{i,2}I_i(t)\right) + \sum_{i=1}^{m}\gamma_{i,2}I_i(t) - \sum_{i=1}^{m}\gamma_{2,i}I_2(t) - \gamma_2 I_2(t) \\
&\vdots \\
\frac{d}{dt}I_m(t) &= \frac{S(t)}{N}\left(\sum_{i=1}^{m}\beta_{i,m}I_i(t)\right) + \sum_{i=1}^{m}\gamma_{i,m}I_i(t) - \sum_{i=1}^{m}\gamma_{m,i}I_m(t) - \gamma_m I_m(t) \\
\frac{d}{dt}R(t) &= \sum_{i=1}^{m}\gamma_i I_i(t)
\end{aligned} \tag{8}$$

In this model we have m possible states of infection, individuals can change state (migration) and the rates for changing state, infection and recovery are all time and state dependent.

It is assumed that infection and migration rates, with migration defined as movement between states that are independent of reproduction, are deterministic and time-dependent. The infection rates can be specified in a matrix $F(t)$ with $f_{i,j}(t)$ denoting the rate at which the entire infected population in state i infects a new individual in state j (note that f here isn't the same as the f used to denote the equilibrium frequencies in the birth-death branching model). In the same way the migration rates can be specified in a matrix $G(t)$ where $g_{i,j}(t)$ denotes the rate at which the entire infected population in state i causes a migration to state j. For the coalescent model with m different states, $F(t)$ and $G(t)$ equal (9) and (10) respectively. The diagonal of $G(t)$ consists of zeros since $\gamma_{i,i}$=0 for $i \in \{1, \ldots, m\}$.

$$F(t) = \begin{pmatrix} \beta_{1,1} I_1(t) \frac{S(t)}{N} & \beta_{1,2} I_1(t) \frac{S(t)}{N} & \ldots & \beta_{1,m} I_1(t) \frac{S(t)}{N} \\ \beta_{2,1} I_2(t) \frac{S(t)}{N} & \beta_{2,2} I_2(t) \frac{S(t)}{N} & \ldots & \beta_{2,m} I_2(t) \frac{S(t)}{N} \\ \vdots & \vdots & \ddots & \vdots \\ \beta_{m,1} I_m(t) \frac{S(t)}{N} & \beta_{m,2} I_m(t) \frac{S(t)}{N} & \ldots & \beta_{m,m} I_m(t) \frac{S(t)}{N} \end{pmatrix} \tag{9}$$

$$G(t) = \begin{pmatrix} 0 & \gamma_{1,2} I_1(t) & \ldots & \gamma_{1,m} I_1(t) \\ \gamma_{2,1} I_1(t) & 0 & \ldots & \gamma_{2,m} I_1(t) \\ \vdots & \vdots & \ddots & \vdots \\ \gamma_{m,1} I_1(t) & \gamma_{m,2} I_1(t) & \ldots & 0 \end{pmatrix} \tag{10}$$

For completeness we can also specify the exogenous birth and death rates for an individual in state i, denoted by $\eta_i(t)$ and $\mu_i(t)$. However, as we will see later, these rates will not have a direct effect on the coalescence rates.

$$\eta(t) = \begin{pmatrix} 0, & 0, & \ldots, & 0 \end{pmatrix} \tag{11}$$

$$\mu(t) = \begin{pmatrix} \gamma_1(t) I_1(t) \\ \gamma_2(t) I_2(t) \\ \vdots \\ \gamma_m(t) I_m(t) \end{pmatrix} \tag{12}$$

Remember that the likelihood of a set of parameters given the phylogenetic tree equals the probability of observing the coalescence events in the phylogenetic tree. In order to calculate this we need the coalescence rates of each pair of branches that coalesce. A coalescence event backwards in time equals a transmission from one state to another forwards in time. Therefore the coalescence rates depend on both the transmission rates from one state to another and the probabilities of each of the descending lineages being in each state. These rates and probabilities can be calculated by starting from the leaf nodes and moving over the branches backward in time. Since the leaf nodes can have different states and times of sampling, the coalescence rates can differ for each branch and between every pair of branches.

To give an idea how the probability of a branch being in a certain state changes backwards in time, we look at the two-state SIR model in Equation (13). Note that the disease dynamics in this model are similar to those of the branching model in Section 5.1.2.

$$
\begin{aligned}
\frac{d}{dt}S(t) &= -\frac{S(t)}{N}\left(\beta_{1,1}I_1(t) + \beta_{2,1}I_2(t)\right) \\
\frac{d}{dt}I_1(t) &= \frac{S(t)}{N}\left(\beta_{1,1}I_1(t) + \beta_{2,1}I_2(t)\right) - \gamma_{1,2}I_1(t) - \gamma_1 I_1(t) \\
\frac{d}{dt}I_2(t) &= \gamma_{1,2}I_1(t) - \gamma_2 I_2(t) \\
\frac{d}{dt}R(t) &= \gamma_1 I_1(t) + \gamma_2 I_2(t)
\end{aligned}
\tag{13}
$$

In this two-state SIR model individuals always enter state 1 upon infection. From this state an individual can progress to a state-2 infection with rate $\gamma_{1,2}$ or can recover or die from the disease with rate γ_1. During its average time of $1/(\gamma_1 + \gamma_{1,2})$ time units in state 1 it can transmit the disease with infection rate $\beta_{1,1}\frac{S}{N}$. An individual in state 2 infects new individuals with rate $\beta_{2,1}\frac{S}{N}$ and recovers or dies from the disease with rate γ_2.

For this model the infection, migration and exogenous birth and death rates are given in Equation (14).

$$
F(t) = \begin{pmatrix} \beta_{1,1}I_1(t)\frac{S(t)}{N} & 0 \\ \beta_{2,1}I_2(t)\frac{S(t)}{N} & 0 \end{pmatrix} \quad G(t) = \begin{pmatrix} 0 & \gamma_{1,2}I_1(t) \\ 0 & 0 \end{pmatrix} \quad \eta(t) = \begin{pmatrix} 0, & 0 \end{pmatrix} \quad \mu(t) = \begin{pmatrix} \gamma_1(t)I_1(t) \\ \gamma_2(t)I_2(t) \end{pmatrix} \tag{14}
$$

A possible phylogenetic tree generated by this two-state epidemic model can be seen in Figure 5. Red branches represent state-1 infected hosts and blue branches represent state-2 infected hosts.

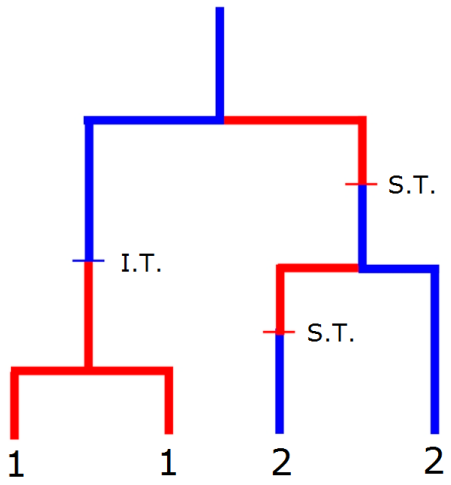

Figure 5: An example of a phylogenetic tree generated by the model in Equation (13). Red and blue branches correspond to state-1 and state-2 infected hosts respectively. [24]

When we move over the phylogenetic tree backwards in time, so upwards in Figure 5, four events can happen. The first possible event is a coalescence of two red branches, which can be seen in the bottom left part of the figure. This event represents a transmission by a state-1 infected individual forwards in time. The second possible event is a coalescence of a blue and a red branch, seen on the bottom right part of the figure, representing a transmission by a state-2 infected individual forwards in time. The third possible event is a state transition from a blue to a red branch, denoted by S.T. in the right part of the figure. This event corresponds to a change in state from a state-1 to a state-2 infection forwards in time. The last and fourth possible event is an invisible transition from a red to a blue branch, denoted by I.T. in

the left part of the figure. This event is the least obvious one. Forwards in time, it represents a transmission by a state-2 infection which is not ancestral to the sample itself. Normally we would have expected a coalescence denoting this transmission event. However, in this case the infecting host transmits and initiates a line of descent that is eventually sampled, but has no other line of descent that is (eventually) sampled.

We didn't include invisible events where the state isn't changed over a branch, for example an invisible transmission by a state-2 infected host which is ancestral to the sample itself but the host it infects isn't. These are not important since we are only interested in the events that change the probability that a lineage is in a certain state, including all possible coalescent events. Note that a coalescence of two blue branches isn't possible since a state-2 infected individual cannot transmit to a state-2 infected host; an infected individual always becomes state-1 infected first.

Now we have illustrated the possible events moving over a phylogenetic tree, representing the two-state model in Equation (13), backwards in time, we can generalize this for the coalescent model with m different states and underlying SIR model (8). We denote by $A(s)$ the number of lineages at time s in the past and by $A_i(s)$ the number of these lineages in state i. A(s) denotes the set of all lineages at time s in the past. The indexes k and l are used to denote lineages from the phylogenetic tree. Furthermore, $p_{ki}(s)$ equals the probability that a branch k is in state i at time s in the past, so $A_i(s) = \sum_{k=1}^{A(s)} p_{ki}(s)$. For simplicity the variable s is dropped from future expressions. However, keep in mind that all state variables and rates are time dependent.

Starting with the probability that a lineage is in a certain state, this changes backwards in time due to possible state transitions (migrations) and invisible transmission events. Over a branch k, the probability that the lineage is in state i can change backwards in time due to:

1. A migration from state i to j at rate $g_{i,j}$ causing the state of branch k to change from j to i with probability p_{kj}/I_j: the probability that branch k is in state j times the probability that from all lineages in state j branch k is the one that changes state.

2. A migration from state j to i at rate $g_{j,i}$ causing the state of branch k to change from i to j with probability p_{ki}/I_i: the probability that branch k is in state i times the probability that from all lineages in state i branch k is the one that changes state.

3. A transmission from state i to j at rate $f_{i,j}$ causing the state of branch k to change from j to i with probability $(p_{kj}/I_j)((I_i - A_i)/(I_i))$: the probability that lineage k changes state from j to i times the probability that the transmitting host is not among the A_i ancestral lineages.

4. A transmission from state j to i at rate $f_{j,i}$ causing the state of branch k to change from i to j with probability $(p_{ki}/I_i)((I_j - A_j)/(I_j))$: the probability that lineage k changes state from i to j times the probability that the transmitting host is not among the A_j ancestral lineages.

These events can be summarized in Equation (15) which represents the change in the probability that a lineage k is in state i backwards in time.

$$\frac{d}{ds}p_{ki} = \sum_{j=1}^{m}\left(\frac{p_{kj}}{I_j}g_{i,j} - \frac{p_{ki}}{I_i}g_{j,i} + \frac{p_{kj}}{I_j}\frac{I_i - A_i}{I_i}f_{i,j} - \frac{p_{ki}}{I_i}\frac{I_j - A_j}{I_j}f_{j,i}\right) \tag{15}$$

With this equation the probability of being in each state over a certain branch, starting from the leaf nodes of the phylogenetic tree and moving backwards in time until a coalescent event is reached, can be calculated. Directly after a coalescent event the probability that the new branch is in a certain state depends on the rate that its two daughter lineages are coalescing. Suppose for example that lineages k and l are coalescing. The coalescence rate corresponding to this event then equals the rate of a transmission from state i to j ($f_{i,j}$) times the probability that lineage k and l coalesce given this transmission event, summed over all possible

states i and j. This last probability equals the probability that lineage k is in state i and lineage l is in state j plus the probability that lineage k is in state j and lineage l is in state i, times the probability that lineages k and l are both coalescing. In formula the total coalescence rate for lineages k and l equals

$$\tilde{\lambda}_{kl} = \sum_{i=1}^{m}\sum_{j=1}^{m}\frac{f_{i,j}}{I_i I_j}\left(p_{ki}p_{lj} + p_{kj}p_{li}\right). \tag{16}$$

Following a coalescent event, the probability that a new branch α is in state i then equals the rate of lineage k infecting l or lineage l infecting k, summed over all possible states of the newly infected individual and divided by the total rate that lineages k and l coalesce. In other words, this probability $p_{\alpha i}$ equals the fraction of the total coalescence rate between lineages k and l for which the infecting individual is in state i. $p_{\alpha i}$ is given in Equation (17).

$$p_{\alpha i} = \frac{1}{\tilde{\lambda}_{kl}}\sum_{j=1}^{m}\frac{f_{i,j}}{I_i I_j}\left(p_{ki}p_{lj} + p_{kj}p_{li}\right) \tag{17}$$

With Equations (15), (16) and (17) we can calculate the probability that a branch at time s in the past is in state i where $i \in \{1, \ldots, m\}$. Here we use that for each leaf node of the phylogenetic tree the initial probability that the lineage is in a certain state equals 1 for its known state and zero otherwise. These probabilities are necessary for calculating the rates of all coalescence events which are in turn needed for calculating the probability of observing all these coalescence events.

Before we can give an equation for the probability of observing the coalescence events in a given phylogenetic tree, we need to calculate the rate that each two lineages coalesce and the probability that lineages haven't coalesced prior to the real time of coalescence. We denote by $\mathcal{C}$ the set of all coalescence events (k, l, s_α). Here k and l denote the lineages that coalesce and s_α equals the time in the past at which the coalescence event took place. The probability that no coalescent events occurred during an internode interval $[s_0, s]$ in the phylogenetic tree, will be denoted by $\theta(s)$ and equals

$$\theta(s) = e^{-\int_{s'=s_0}^{s}\tilde{\Lambda}(s')ds'}. \tag{18}$$

Here $\tilde{\Lambda}(s) = \sum_{k,l\in\mathcal{A}(s),k\neq l}\tilde{\lambda}_{kl}(s)/2$ equals the total rate at which the lineages at time s coalesce. Note that we divide by 2 since $\tilde{\lambda}_{kl} = \tilde{\lambda}_{lk}$: each pair of lineages is counted twice in the summand. The duration of an internode interval is exponentially distributed with parameter $\tilde{\Lambda}(s)$ and therefore has density $\tilde{\Lambda}(s)\theta(s)$.

The probability that a coalescence event between lineages k and l occurs after the internode interval $[s_0, s]$ equals the probability that two lineages coalesce after the internode interval $[s_0, s]$ times the probability that the coalescence happens between lineages k and l. This last probability equals $\tilde{\lambda}_{kl}(s)/\tilde{\Lambda}(s)$. The probability that two lineages coalesce after s can be seen as the density of the duration of the internode interval $[s_0, s]$. The probability that lineages k and l coalesce after internode interval $[s_0, s]$ therefore becomes

$$q_{kl}(s) = \tilde{\Lambda}(s)\theta(s)\frac{\tilde{\lambda}_{kl}(s)}{\tilde{\Lambda}(s)} = \tilde{\lambda}_{kl}(s)\theta(s). \tag{19}$$

The likelihood of the parameters given the phylogenetic tree equals the probability of observing all coalescence events (k, l, s_α) in $\mathcal{C}$ and can now be written down by

$$\mathcal{L}(\mathcal{C}) = \prod_{(k,l,s_\alpha)\in\mathcal{C}} q_{kl}(s_\alpha) = \prod_{(k,l,s_\alpha)\in\mathcal{C}} \tilde{\lambda}_{kl}(s_\alpha)\theta(s_\alpha). \tag{20}$$

5.2.3 Hepatitis B specific coalescent model

The transmission dynamics of hepatitis B can be translated into the coalescent model where the number of types equals 2 (an infection can be acute or chronic). The corresponding parameters are given in Table 2 where 1 denotes the acute state and 2 the chronic state.

$\beta_{1,1}$	the rate at which an acutely infected individual transmits to a new acutely infected individual at the beginning of an epidemic when almost all individuals are susceptible.
$\beta_{2,1}$	the rate at which a chronically infected individual transmits to a new acutely infected individual at the beginning of an epidemic when almost all individuals are susceptible.
γ_1	the rate at which an acutely infected individual recovers or dies from the disease
γ_2	the rate at which a chronically infected individual recovers or dies from the disease
$\gamma_{1,2}$	the rate at which an acutely infected individual progresses to the chronic state.

Table 2: The parameters for the coalescent model describing hepatitis B. 1 and 2 represent the acute and chronic state respectively.

The parameters $\beta_{1,2}$, $\beta_{2,2}$ and $\gamma_{2,1}$ aren't included in this table since it isn't possible to transmit from an acute or chronic state to an individual in the chronic state; an infected individual always becomes acutely infected first. Furthermore, an individual cannot change from the chronic to the acute state of infection.

Translating the population dynamics of hepatitis B into a standard SIR model we get Equation 13. Upon infection an individual always becomes acutely infected. From the acute state an individual can transmit with rate $\beta_{1,1}\frac{S}{N}$, recover or die from the infection with rate γ_1 and progress to the chronic state with rate $\gamma_{1,2}$. From the chronic state an individual transmits with rate $\beta_{2,1}\frac{S}{N}$ and recovers or dies from the disease with rate γ_2.

However, the population dynamics of hepatitis B among MSM in the Netherlands doesn't behave like an epidemic but the number of infected individuals is more or less constant over time. In order for the SIR model to represent a constant population over time the system should be in equilibrium: the inflow and outflow of the MSM population in the Netherlands should be equal. Therefore a new parameter ψ, which denotes the rate at which an individual has a change in behavior or dies a natural death, is added to the SIR model. Because we expect that transmission mostly takes place among active MSM, N denotes the total active MSM population in the Netherlands. The adapted SIR model is given in Equation (21). For a schematic representation see Figure 6.

$$\begin{aligned}
\frac{d}{dt}S(t) &= -\frac{S(t)}{N}\left(\beta_{1,1}I_1(t) + \beta_{2,1}I_2(t)\right) + \psi N - \psi S \\
\frac{d}{dt}I_1(t) &= \frac{S(t)}{N}\left(\beta_{1,1}I_1(t) + \beta_{2,1}I_2(t)\right) - \gamma_{1,2}I_1(t) - \gamma_1 I_1(t) - \psi I_1 \\
\frac{d}{dt}I_2(t) &= \gamma_{1,2}I_1(t) - \gamma_2 I_2(t) - \psi I_2 \\
\frac{d}{dt}R(t) &= \gamma_1 I_1(t) + \gamma_2 I_2(t) - \psi R.
\end{aligned} \tag{21}$$

Note that the total inflow into the population equals the total outflow ($\psi N = \psi(S + I_1 + I_2 + R)$). Intuitively the total inflow represents the number of new men changing their behavior to the active MSM population. By assuming only an inflow into the susceptible population, we assume that these new men haven't had the infection before. This is a reasonable assumption since probably most infections are spread among active MSM. Furthermore, the outflow from the recovered compartment also includes men who died from the disease. Intuitively this doesn't make sense. Nevertheless, we don't want these individuals to be be

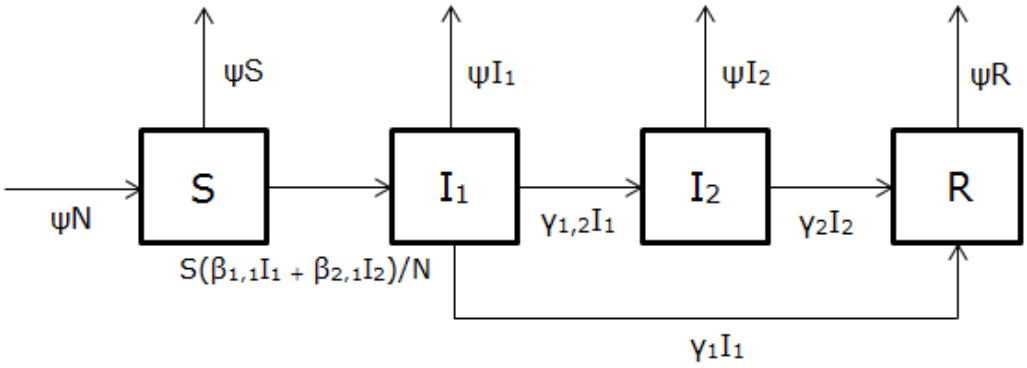

Figure 6: A schematic representation of the SIR model in equilibrium, representing the population dynamics of hepatitis B among MSM in the Netherlands.

in the system forever and therefore treat them as being recovered from the disease. We furthermore expect chronically infected individuals to leave the active population before they die from the disease, so the rate for this type of death will be small.

For this two-type coalescent model the reproduction number R_0, the expected number of secondary infections caused by a single infected individual in a completely susceptible population, is given in Equation (22). During the time an infected individual resides in the acute state, this individual spreads the disease to a new acutely infected individual with rate $\beta_{1,1}$. On average the infected individual will therefore cause $\beta_{1,1}$ times the average time of being in the acute state new infections during its acute period, denoted by the first term in the equation. When an infected individual leaves the acute state it is possible that the individual becomes chronically infected with the disease. If this happens the infected individual spreads the disease to a new acutely infected individual at rate $\beta_{2,1}$, during the time the individual resides in the chronic state. The average number of infections caused by the infected individual during its chronic period therefore equals the probability of moving from the acute to the chronic state of infection multiplied with $\beta_{2,1}$ and the average time in the chronic state. This average number of infections is denoted by the second term of the equation.

$$R_0 = \frac{\beta_{1,1}}{\gamma_1 + \gamma_{1,2} + \psi} + \frac{\gamma_{1,2}}{\gamma_1 + \gamma_{1,2} + \psi} \frac{\beta_{2,1}}{\gamma_2 + \psi}. \tag{22}$$

Together with the assumption that the SIR model is in equilibrium, the number of susceptible, acute and chronically infected and recovered individuals should be constant over time. These numbers can be found by setting the differential equations in Equation (21) to zero, and solve for S, I_1, I_2 and R. This yields

$$\begin{aligned} S &= \frac{1}{R_0} N \\ I_1 &= \frac{\psi N (1 - \frac{1}{R_0})}{\gamma_1 + \gamma_{1,2} + \psi} \\ I_2 &= \frac{\psi \gamma_{1,2} N (1 - \frac{1}{R_0})}{(\gamma_2 + \psi)(\gamma_1 + \gamma_{1,2} + \psi)} \\ R &= \frac{N(1 - \frac{1}{R_0})(\gamma_1(\gamma_2 + \psi) + \gamma_2 \gamma_{1,2})}{(\gamma_2 + \psi)(\gamma_1 + \gamma_{1,2} + \psi)}. \end{aligned} \tag{23}$$

Notice that this isn't the only possible solution, another possible solution is $S = N$ and $I_1 = I_2 = R = 0$. However, this 'disease-free' solution is not relevant if we consider a phylogenetic tree in which there is at least

one infected individual. The number of recovered individuals R isn't needed for calculating the likelihood in Equation (20) but we mention it here for completeness.

The calculation of the likelihood in Equation (20) is performed by the use of R 3.0.1 [21]. The codes can be found in Appendix C.4.

5.3 Estimation of the model parameters

The multi-type birth-death branching model and coalescent model introduced above can be used to calculate the likelihood of a certain parameter set given the phylogenetic tree estimated from the data. In order to find the model parameters corresponding to the maximum likelihood, the Maximum Likelihood Estimators (MLEs), we use two different optimization methods.

Both optimization methods are performed by the use of R 3.0.1 [21]. Codes can be found in Appendix C.5.

5.3.1 Optimization method based on a pre-specified grid

The first method is based on finding the maximum likelihood, i.e., minimum negative log likelihood, by calculating the negative log likelihood for the parameter sets in a pre-specified grid. From the parameter set corresponding to the minimum negative log likelihood calculated for this grid, we then proceed in two different ways. One possibility is to repeat the previous step by taking a finer grid which includes the MLE of the first grid. The second possibility is to start from the MLEs and for each parameter calculate the negative log likelihood in case only this parameter changes with a certain step size (both in positive and negative direction) and the other parameters remain the same. Subsequently the new parameter set is chosen for which the negative log likelihood is minimal. From this new parameter set we proceed in the same way until the minimum is reached, so we walk over an imaginary grid. In case we use the first possibility, we also use the second possibility afterwards in order to find the parameter set corresponding to the minimum negative log likelihood. We can repeat these steps as often as we want, with different step sizes for each parameter. This method can be used as we expect that the likelihood is continuous in the parameters.

5.3.2 Monte Carlo Markov Chain (MCMC)

A Monte Carlo Markov Chain, the Metropolis-Hastings algorithm, is used to generate a sequence of sampled parameter values from a desired probability distribution π of this parameter. The general idea of the method is that a new value is generated from a proposal distribution Q based on the present value of the parameter, and subsequently this new parameter value is accepted with an acceptance probability equal to the ratio of the likelihood of both parameters. In case the new parameter is accepted this new value becomes the parameter value on which a new proposal distribution is based. In case it is not accepted the present parameter value and corresponding proposal distribution is used again to generate a new value. This process is run for a long time. After a 'burn in' period, the time needed for the MCMC method to find a good starting point, the distribution of the parameter value will approximate the distribution π. What we only need for this method to work is that the distribution Q should satisfy the condition that the probability of sampling a new parameter value x given that the present parameter value is y, equals the probability of sampling the new parameter value y given the present parameter value x: $Q(x|y) = Q(y|x)$. If this isn't the case we can correct for this by adapting the acceptance ratio accordingly. This is the essential idea of the Hastings part in the Metropolis-Hastings algorithm. We choose the proposal distribution Q to equal the folded normal distribution which satisfies the condition $Q(x|y) = Q(y|x)$ [29].

For our model parameters we will use the following algorithm given that our initial parameter set equals x.

1. For each parameter x_i in the parameter set x generate a new value x_i' from the folded normal distribution with mean x_i and a parameter specific standard deviation.

2. Calculate the acceptance ratio $L(x')/L(x)$ where L denotes the likelihood of the parameter set given the phylogenetic tree and for the method (multi-type birth-death branching model or coalescent model) used.

3. In case the acceptance ratio is greater than or equal to one accept the new parameter set x', take it as the next parameter set x and start the process again. In case the acceptance ratio is smaller than one, choose the new parameter set x' with probability equal to $L(x')/L(x)$. If the new parameter set isn't accepted the present parameter set x is used again in the next iteration.

We will repeat this process $10,000$ times and after each step we will save the value of the parameter set. For each parameter we then expect the distribution of the parameter value to converge to the desired distribution. Note that we took the folded normal distribution instead of the normal distribution because our model parameters are rates or probabilities which cannot become negative.

From the results of the Metropolis-Hastings algorithm we can also easily derive the credibility intervals by taking the parameter values between which 95% of the parameter values are situated. For these credibility intervals we ignore the 'burn in' periods.

6 Results

We use the models introduced in Sections 4 and 5 to estimate the transmission dynamics of hepatitis B among MSM in the Netherlands. Based on our data we build a phylogenetic tree, which is presented and evaluated first. Afterwards we present and interpret for both the two-type birth death branching model and the coalescent model the estimated parameter values corresponding to the maximum likelihood for a phylogenetic tree of our data.

6.1 A phylogenetic tree generated from the data

For our data set we constructed a phylogenetic tree by using the method described in Section 4. This phylogenetic tree is presented in Figure 7.

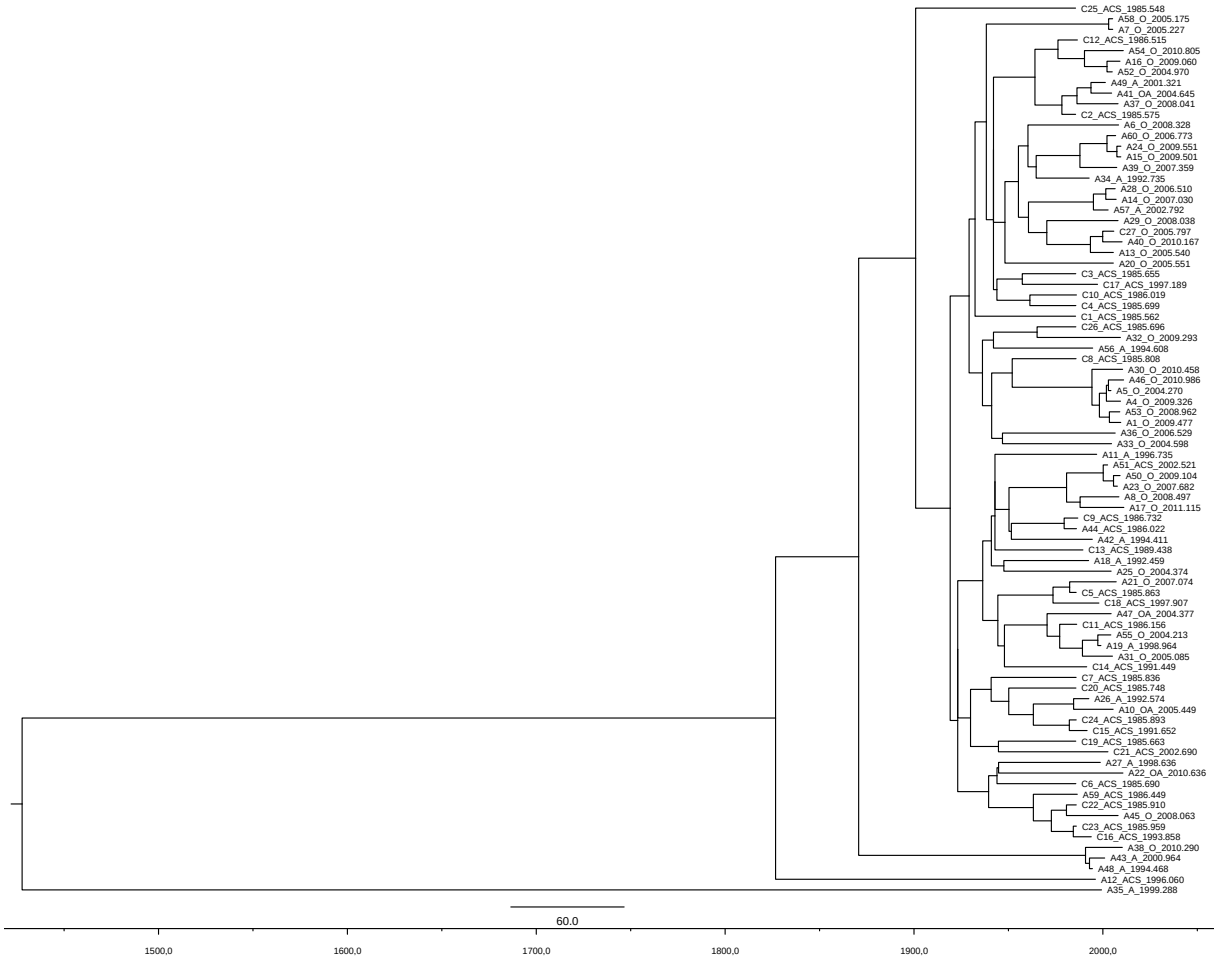

Figure 7: A phylogenetic tree corresponding to our hepatitis B data by using the method in Section 4. The labeled leaf nodes denote the sampled genetic sequences. The first letter of the label denotes the infection state of the individual at the time of sampling the genetic sequence, A for acute and C for chronic, and it is followed by an index number. The part between underscores denotes the origin of the sample: an 'O' or 'A' for a sample collected from one of the Municipal Health Services in the Netherlands (where the A is specific for the Municipal Health Services in Amsterdam) and 'ACS' for samples collected from the Amsterdam Cohort Studies. The number after the last underscore denotes the sampling date in years. Time is on the horizontal axis.

Our genetic sequences, denoted by the leaf nodes, are related by their ancestral lineages. While the second and third sequence from above are closely related because they coalesce shortly back in time, the bottom sequence is in distance far away from all other sampled genetic sequences. Its ancestor, which is also the most recent common ancestor (MRCA) of all genetic sequences in the tree, is situated before the year 1500. This suggests that the bottom sequence could be a misclassified genotype G sequence. Indeed, hepatitis B viruses of genotype G are frequently found as a co-infection with the genotype A hepatitis B virus [30].

We illustrate two different ways of removing this genetic sequence from our phylogenetic tree. The first way is by deleting this node and its branches to the rest of the tree by the use of the drop.tip function of the R package ape [31], yielding the phylogenetic tree in Figure 8. The code we used for this can be found in Appendix C.1.

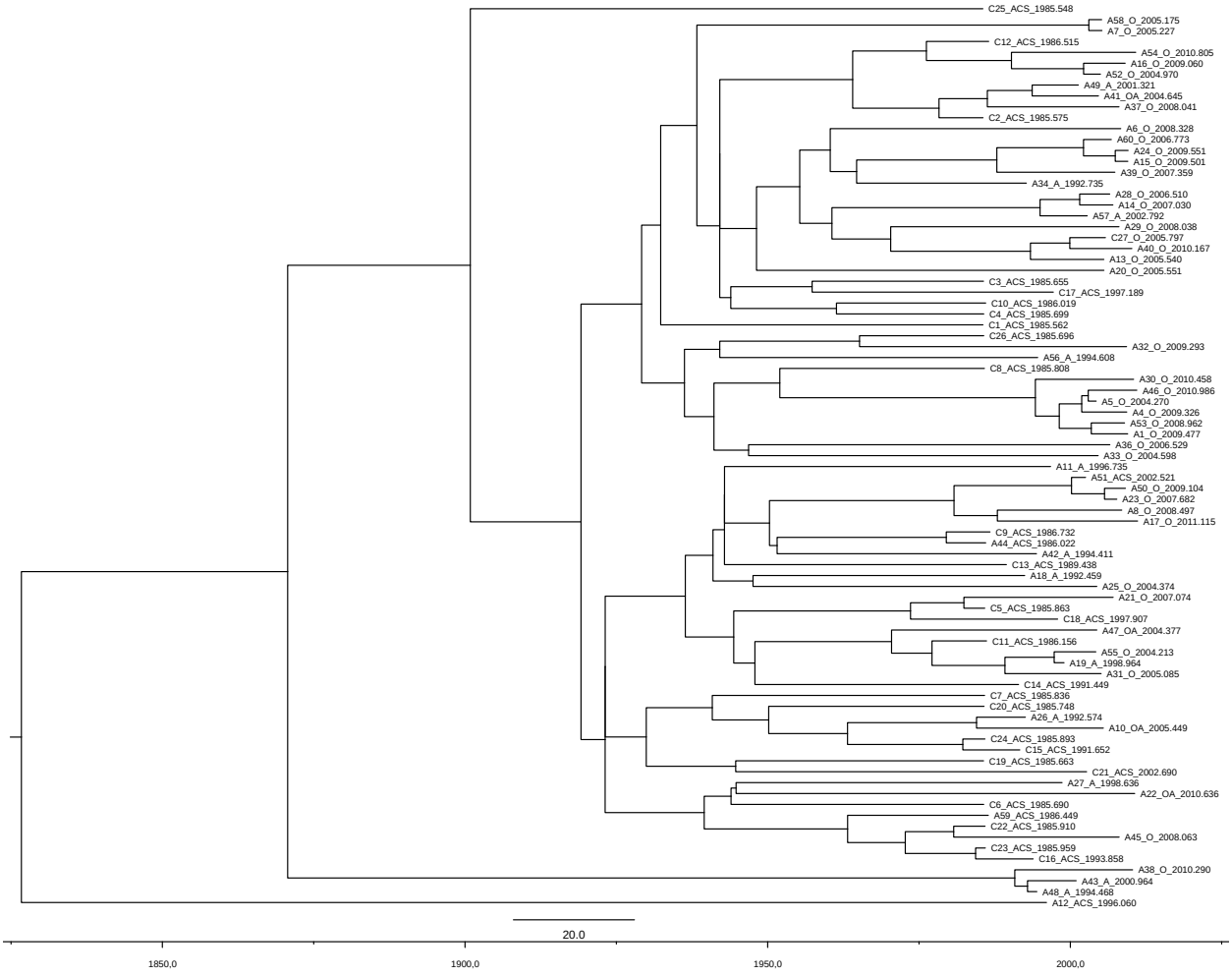

Figure 8: A phylogenetic tree for our hepatitis B data without the HBV sequence which is assumed to be of genotype G. This tree is obtained from the phylogenetic tree in Figure 7 by the use of the drop.tip function of the R package ape [31]. Time is on the horizontal axis.

The second way is to construct a phylogenetic tree by the use of the model described in Section 4, with the genetic sequence assumed to be of genotype G not part of the data (see Figure 9). In case the constructed phylogenetic tree in Figure 7 is a good representation of the real gene genealogy describing the ancestral relationships between the pathogens, we expect these two different methods to generate a similar tree.

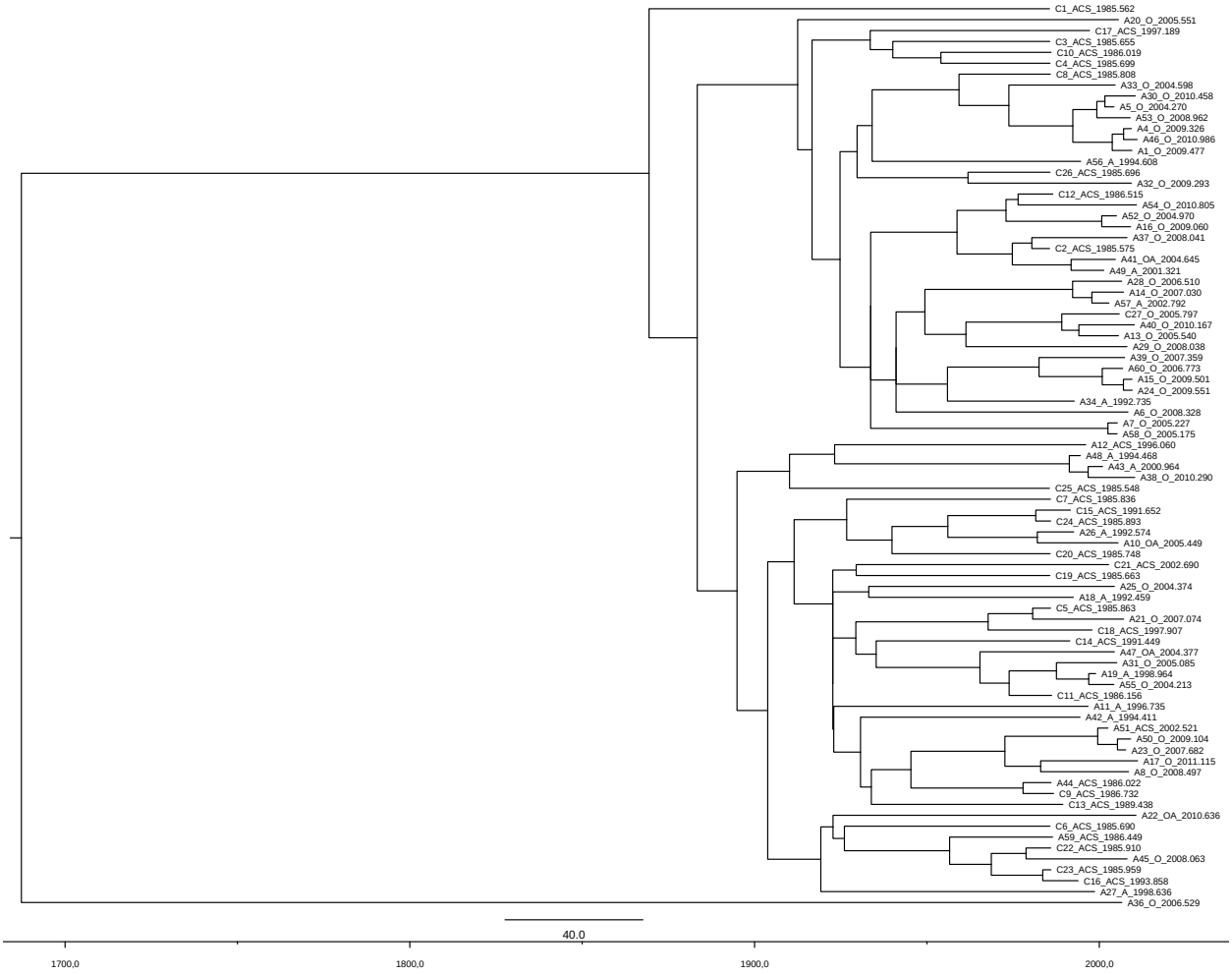

Figure 9: The phylogenetic tree corresponding to our hepatitis B data without the HBV sequence assumed to be of genotype G. This phylogenetic tree is created by using the method described in Section 4. Time is on the horizontal axis.

When we compare the phylogenetic trees in Figure 8 and 9 we see that they don't look quite the same. For example, the genetic sequence with label $A12\_ACS\_1996.060$, the bottom leaf node in Figure 8, is far away from all other sequences in this tree while in Figure 9 it is in the middle of all genetic sequences; it is more related to the other sequences.

6.2 Parameter inference for the two-type birth-death branching model

We apply the two-type birth-death branching model to the phylogenetic tree in Figure 8 and search for the values of the parameters in Table 1 corresponding to the minimum negative log likelihood. Instead of calculating the negative log likelihood over the whole phylogenetic tree, we only calculate it until about 82 years back in time, corresponding to one of the internal nodes, and sum the negative log likelihoods of the separate trees. By doing this we don't loose much information while we prevent problems like infinite negative log likelihoods because of a very small probability density of the whole tree. Indeed, how longer the branches become, the less information they contain; all kind of events could have occurred over the branch. Furthermore, we don't use the restriction $R_0 > 1$ (the reproduction number can also be between zero and one) and $\lambda_{1,2} = \lambda_{2,2} = \gamma_{2,1} = 0$ are fixed parameters. The details of the optimization method used for finding the parameter estimators are given in Appendix B.1.

6.2.1 Two-type birth-death branching model with constant sampling fractions

For the two-type birth-death branching model with constant sampling fractions we assume a sampling fraction of 0.05 per year, for both acutely and chronically infected individuals. This means that 5% of the acutely infected individuals that become non-infectious each year are subsequently sampled. The same holds for chronically infected individuals.

The parameter values in rates per year, found by optimizing the negative log likelihood for the phylogenetic tree in Figure 8, are presented in Table 3. The corresponding reproduction number and negative log likelihood are also given.

Parameter	MLE
$\lambda_{1,1}$	1.193
$\lambda_{2,1}$	0.008
d_1	0.0001
d_2	0.0001
$\gamma_{1,2}$	1.19
R_0	81.00
Negative Log Likelihood	-276.40

Table 3: Two-type birth-death branching model parameters for the hepatitis B data using the phylogenetic tree in Figure 8 and a constant sampling fraction of 5% for both acutely and chronically infected individuals. $\lambda_{1,1}$, $\lambda_{2,1}$, d_1, d_2 and $\gamma_{1,2}$ are the rates per year.

We can interpret these results by looking at the probability that an infected individual in the acute or chronic state infects a new acutely infected individual and at the probability that an acutely infected individual moves to the chronic state before leaving the acute state. We present these probabilities in the following matrix which we denote by P. Note that the probability from a chronic to a chronic infection equals 0 since a newly infected individual always becomes acutely infected first and a state transition from a state to itself isn't possible.

$$P = \begin{matrix} \\ A \\ C \end{matrix} \begin{matrix} \begin{matrix} A & \quad C \end{matrix} \\ \begin{pmatrix} \frac{\lambda_{1,1}}{\lambda_{1,1}+d_1+\gamma_{1,2}} & \frac{\gamma_{1,2}}{d_1+\gamma_{1,2}} \\ \frac{\lambda_{2,1}}{\lambda_{2,1}+d_2} & 0 \end{pmatrix} \end{matrix} = \begin{matrix} \\ A \\ C \end{matrix} \begin{matrix} \begin{matrix} A & \quad C \end{matrix} \\ \begin{pmatrix} 0.501 & 1.000 \\ 0.988 & 0 \end{pmatrix} \end{matrix}$$

The probability that an acutely infected individual gives birth to a new acutely infected individual equals this birth rate per year divided by the rate per year that a birth, death or state transition takes place. For a chronically infected individual the probability of giving birth to an acutely infected individual equals the

birth rate per year divided by the rate per year that a birth or death event occurs (a state change isn't possible). The probability that an acutely infected individual progresses to the chronic state before leaving the acute state equals this state transition rate per year divided by the total rate per year of a death or state change.

We can interpret these results as follows. Once an individual from the MSM population becomes infected with HBV its average time in the acute state equals $1/(d_1+\gamma_{1,2}) = 0.840$ years, i.e. over 10 months. During this time the acutely infected individual infects a new individual with probability 0.501 and the average number of infections caused equals $\lambda_{1,1}/(d_1+\gamma_{1,2}) = 1.002$. At the end of its acute period the infected individual progresses to the chronic state with probability close to 1. An infected individual in the chronic state resides here for about $1/d_2 = 10,000$ years. During this time it infects a new individual with probability 0.988 and the average number of infections caused equals $\lambda_{2,1}/d_2 = 80$. Because an infected individual almost always goes through both the acute and chronic state of infection and the total number of infections caused during these periods is very high, the reproduction number R_0 given in Table 3 is also high. Despite of this high reproduction number, the transmission dynamics are very slow because of the long acute and chronic period. Therefore the infected population will grow slowly over time.

These results are unrealistic and contrast with our current knowledge of the dynamics of an HBV infection (Section 3); the average times in the acute and chronic period are estimated too high, just like the rate for an acutely infected individual to progress to the chronic state of infection.

6.2.2 Two-type birth-death branching model with variable sampling fractions over time

For the two-type birth-death branching model with variable sampling fractions over time, where the infected population is assumed to be constant over time (see Section 5.1.2), the sampling fraction per year equals a state dependent constant multiplied with the number of acutely or chronically sampled individuals that year. While these last numbers are known from our data, the state dependent constants become parameters of our model. We can estimate these two parameters together with the other parameters of the model given in Table 1 by finding the minimum negative log likelihood. The results are presented in Table 4.

Parameter	MLE
$\lambda_{1,1}$	1.423
$\lambda_{2,1}$	0.001
d_1	0.0001
d_2	0.0001
$\gamma_{1,2}$	1.271
c_1	0.00001
c_2	0.0001
R_0	11.12
Negative Log Likelihood	-623.80

Table 4: Two-type birth-death branching model parameters for the hepatitis B data using the phylogenetic tree in Figure 8 and a variable sampling fraction. $\lambda_{1,1}$, $\lambda_{2,1}$, d_1, d_2 and $\gamma_{1,2}$ are the rates per year.

The matrix P that denotes the probabilities of an infection or state transition becomes

$$P = \begin{array}{c} \\ A \\ C \end{array} \begin{array}{c} \begin{array}{cc} A & C \end{array} \\ \begin{pmatrix} 0.528 & 1.000 \\ 0.909 & 0 \end{pmatrix} \end{array}.$$

The results imply that when an individual becomes infected with hepatitis B it remains for about $1/(d_1+\gamma_{1,2}) = 0.787$ years, i.e. over 9 months, in the acute state. During this acute period the infected individual in-

fects a new acutely infected individual with probability 0.528 and will cause on average $\lambda_{1,1}/(d_1+\gamma_{1,2}) = 1.120$ new acute infections. With probability almost one the acutely infected individual will progress to the chronic state. The average time that an infected individual resides in the chronic state equals $1/d_2 = 10,000$ years. During this time the chronically infected individual gives birth to a new acutely infected individual with probability 0.909 and on average he causes $\lambda_{2,1}/d_2 = 10$ new infections. Although the reproduction number R_0 is much lower than in the case of constant sampling fractions, the results still imply slow transmission dynamics.

Again the average times of an acute and chronic HBV infection are overestimated; over 9 months and 10,000 years respectively. Also the probability that an acute infection progresses to the chronic state is far too high. The very small values for c_1 and c_2 imply that we have a very large population of acutely and chronically infected MSM that became non-infectious per year; in total over 110,000. In comparison with the total number of MSM in the Netherlands, which is estimated to be in the range 278,000-392,000 [32], this is far too high.

We correct for this by adding prior information to the model, namely by assuming that the rate per year to leave the acute state of infection is between three and four ($d_1 + \gamma_{1,2} \in [3:4]$). This implies that an acute infections lasts on average three to four months. This yields the results given in Table 5.

Parameter	MLE
$\lambda_{1,1}$	2.3
$\lambda_{2,1}$	0.06
d_1	1.1
d_2	0.087
$\gamma_{1,2}$	1.99
c_1	0.001
c_2	0.001
R_0	1.19
Negative Log Likelihood	-18.392

Table 5: Two-type birth-death branching model parameters for the hepatitis B data using the phylogenetic tree in Figure 8, a variable sampling fraction and as prior information that an acute infection lasts for about 3 to 4 months. $\lambda_{1,1}$, $\lambda_{2,1}$, d_1, d_2 and $\gamma_{1,2}$ are the rates per year.

The corresponding probabilities of infection and the probability to progress from the acute to the chronic state are denoted in the matrix P below.

$$P = \begin{array}{c} \\ A \\ C \end{array} \begin{array}{c} \begin{array}{cc} A & C \end{array} \\ \begin{pmatrix} 0.427 & 0.644 \\ 0.408 & 0 \end{pmatrix} \end{array}$$

With these results an infected individual is on average $1/(d_1 + \gamma_{1,2}) = 0.324$ years, i.e. almost 17 weeks, in the acute state from which it infects a new acutely infected individual with probability 0.427. On average an infected individual will cause $\lambda_{1,1}/(d_1 + \gamma_{1,2}) = 0.744$ new acute infections during its acute period. An acutely infected individual will progress to the chronic state in about 64% of the cases. An infected individual will reside on average $1/d_2 = 11.494$ years in the chronic state. During this period a chronically infected individual will give birth to a new acutely infected individual with probability 0.408 and will cause on average $\lambda_{2,1}/d_2 = 0.690$ new infections. The estimated reproduction number R_0 is slightly above one, so the model expects the infected population to grow slowly over time.

In comparison with the results for the previous two models, these results seem more reliable. However, the probability for an acute infection to progress to a chronic infection remains high (64%). Furthermore

we assumed a constant infected population size while the R_0 implies exponential growth of the infected population. We also still have small values for c_1 and c_2. As mentioned in Appendix B.1 a decrease in these numbers even leads to a lower negative log likelihood and therefore to an improvement of the parameter set. This implies that there is no sampling, while from our data it is known that there are actually infected individuals sampled. Therefore the infected population size equals infinity. This shows that the two-type birth-death model, with or without constant sampling fractions and prior information, is not sufficient for our data.

6.3 Parameter inference for the coalescent model

In this section we apply the coalescent model to the phylogenetic tree in Figure 8 and calculate the maximum likelihood estimators, in our case minimum negative log likelihood estimators, for the parameters in Table 2. Instead of cutting the phylogenetic tree at a certain time in the past, we calculate the negative log likelihood over the whole phylogenetic tree. Furthermore we assume $R_0 \geq 1$ since otherwise the equilibrium values for the number of susceptible, infected and recovered individuals become negative. The total population size is assumed to equal $N = 24000$, which is estimated to be the part of the Dutch MSM population in the three highest risk groups [33]. We only look at this part of the Dutch MSM population because mostly these men will contribute to the transmission of the disease since they are sexually active. For the rate at which an individual changes in behavior or dies a natural death we use $\psi = 0.1$. This means that an individual resides on average 10 years in the active MSM population.

We will also show the results for applying the MCMC method described in Section 5. Finally, we will compare the parameter values calculated for the phylogenetic tree in Figure 8 with the parameter values found when we apply the coalescent model to the phylogenetic tree in Figure 9. The detailed optimization steps used to estimate the parameter values of the coalescent model for both phylogenetic trees can be found in Appendix B.2.

Coalescent model applied to the phylogenetic tree in Figure 8

We estimate the parameter values of the coalescent model for the phylogenetic tree in Figure 8. The parameter values in rates per year, the corresponding reproduction number and negative log likelihood, and the number of susceptibles and acutely and chronically infected individuals at each moment of time are presented in Table 6. Note that we don't present the number of recovered individuals since these do not affect the minus log likelihood calculated and therefore aren't relevant to the system.

Parameter	MLE
$\beta_{1,1}$	5.501
$\beta_{2,1}$	0.001
γ_1	4.116
γ_2	0.007
$\gamma_{1,2}$	0.105
R_0	1.273
S	18848.48
I_1	119.2205
I_2	116.9921
Negative Log Likelihood	307.099

Table 6: Coalescent model parameters for the hepatitis B data using the phylogenetic tree in Figure 8, $N = 24000$ and $\psi = 0.1$. $\beta_{1,1}$, $\beta_{2,1}$, γ_1, γ_2 and $\gamma_{1,2}$ are the rates per year.

We can present the results in the probability matrix P, denoting the probabilities at which acutely and

chronically infected individuals transmit to a new acutely infected individual at the beginning of the epidemic when almost all individuals are susceptible, and the probability of moving from the acute to the chronic state of infection.

$$P = \begin{array}{c} \\ A \\ C \end{array} \begin{array}{c} \begin{array}{cc} A & C \end{array} \\ \begin{pmatrix} \frac{\beta_{1,1}}{\beta_{1,1}+\gamma_1+\gamma_{1,2}+\psi} & \frac{\gamma_{1,2}}{\gamma_1+\gamma_{1,2}+\psi} \\ \frac{\beta_{2,1}}{\beta_{2,1}+\gamma_2+\psi} & 0 \end{pmatrix} \end{array} = \begin{array}{c} \\ A \\ C \end{array} \begin{array}{c} \begin{array}{cc} A & C \end{array} \\ \begin{pmatrix} 0.560 & 0.024 \\ 0.009 & 0 \end{pmatrix} \end{array}$$

These results imply that an infected individual resides on average $1/(\gamma_1 + \gamma_{1,2} + \psi) = 0.231$ years, i.e., over 12 weeks, in the acute state. At the beginning of the epidemic, an acutely infected individual will infect a new acutely infected individual with probability 0.560 and during his acute period the individual causes on average $\beta_{1,1}/(\gamma_1 + \gamma_{1,2} + \psi) = 1.273$ new acute infections. About $2,4\%$ of the acutely infected individuals is estimated to progress to the chronic state and the average duration of a chronic period equals $1/(\gamma_2 + \psi) = 9.346$ years. A chronically infected individual will infect new acutely infected individuals with probability 0.009 and during the chronic period the individual will cause on average $\beta_{2,1}/(\gamma_2 + \psi) = 0.009$ new infections, both at the beginning of the epidemic when almost all individuals are susceptible. The estimated reproduction number equals 1.273 so an infected individual will on average cause 1.273 secondary infections in a totally susceptible population. In our situation only a fraction S/N of our total population is susceptible and therefore the effective reproduction number equals $R_0 \cdot \frac{S}{N} = 1$. This satisfies our assumption that the infected population size is constant over time.

The equilibrium value for the number of acutely infected MSM equals 119.2205, so at each moment of time there are 119.2205 acutely infected MSM in the population. With the average duration of an acute period of infection we can calculate the expected number of new acute infections per year. This therefore equals $119.2205 \cdot (1/0.231) = 515.15$ new acute infections per year, which seems a realistic value. The equilibrium value for the number of chronically infected MSM at each moment of time equals 116.9921. We therefore expect $116.9921 \cdot (1/9.346) = 12.518$ new chronic infections per year. This number seems very low; the number of chronically infected MSM in the Netherlands is said to be around 1% of the total MSM population that ranges from $278,000$ to $392,000$ [32]. A possible problem could be that we assumed ψ, the rate per year at which an individual has a change in behavior or dies a natural death, to be equal for all different compartments of the SIR model. For example, it could be that a chronic infection occurs mostly at later ages and therefore the infected individual leaves the active MSM population earlier than when an individual is acutely infected. Another explanation for the low value of new chronically infected individuals each year could be that a lot of the chronically infected MSM are not active and therefore not in our population N.

From above results we can conclude that the acutely infected individuals are mainly responsible for the new infections among MSM in the Netherlands. While a chronically infected individual only infects on average 0.009 new individuals, an acutely infected individual causes on average 1.273 new infections.

In Figure 10 we see how the probability of being in a certain state changes over a lineage backwards in time, with the lineages starting from a leaf node in the acute or chronic state. When we look at Figure 10a and 10b we see that for a lineage starting from a leaf node in the acute state the probability of being in the acute state changes slowly backward in time, even in case the branch is very long. For a lineage starting from a leaf node in the chronic state, Figure 10c and 10d, the probability of the lineage being in the chronic state decreases very fast over time. For a long branch the probability even tends to zero, implying that a progression from the acute to the chronic state and an infection by a chronically infected individual is rare.

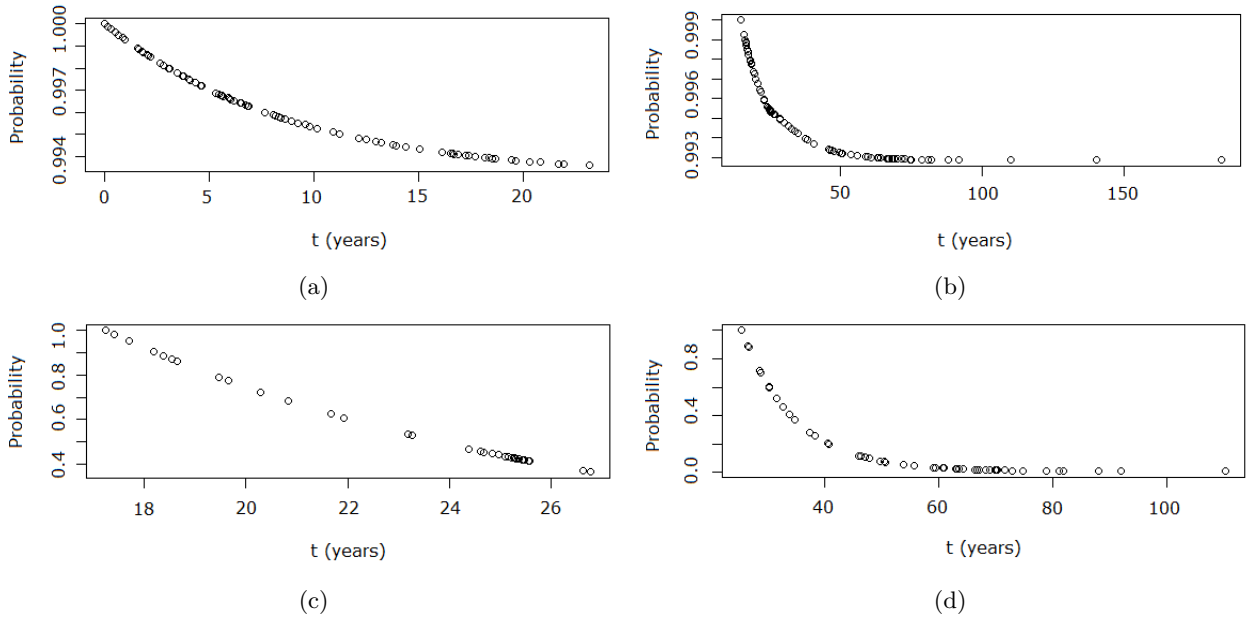

Figure 10: The course of the probability that a lineage is in a certain state for four different lineages. In Figure (a) and (b) we look at the course of the probability that the lineage is in the acute state for a short and long branch respectively. Both lineages start from a leaf node in the acute state. Figures (c) and (d) show the course of the probability that the lineage is in the chronic state for a short and long branch respectively. In this case both lineages start from a leaf node in the chronic state. The horizontal axis denotes the number of years until the present (time 0).

These are the probabilities over a branch until a coalescent event occurs. However, we are also interested in the probability that the lineage after a coalescence event is in the acute state because this tells us whether mostly acutely or chronically infected individuals cause new infections. In Figure 11 this probability is given for all lineages following a coalescence event.

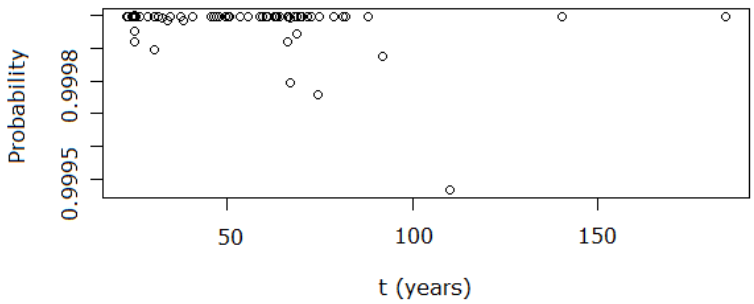

Figure 11: The probability that at each time of coalescence the lineage following this event is in the acute state. The horizontal axis denotes the number of years between the time of the coalescence event and the present (time 0).

Both the results in Figure 10 and in Figure 11 confirm that the acutely infected individuals cause most new infections; the lineage on the left of a coalescence event is with very high probability in the acute state and over a branch an individual is almost always with high probability in the acute state.

The results we found for the coalescent model using the phylogenetic tree in Figure 8 seem reasonable. Therefore we will use the MCMC method described in Section 5 to re-estimate these parameters and their corresponding credibility intervals. We used as the initial values for the parameters $(\beta_{1,1}, \beta_{2,1}, \gamma_1, \gamma_2, \gamma_{1,2}) = (5.5, 0.1, 4, 0.12, 0.1)$ and we tried multiple different standard deviations for each of these parameters. Note that this initial parameter set is one of the interim values found when searching for the parameter values corresponding to the minimum negative log likelihood, see Appendix B.2. In Figure 12 we present the results for the parameter values using $10,000$ iterations and standard deviations $(1.1, 0.02, 0.5, 0.015, 0.02)$ for the parameters $\beta_{1,1}$, $\beta_{2,1}$, γ_1, γ_2 and $\gamma_{1,2}$ respectively.

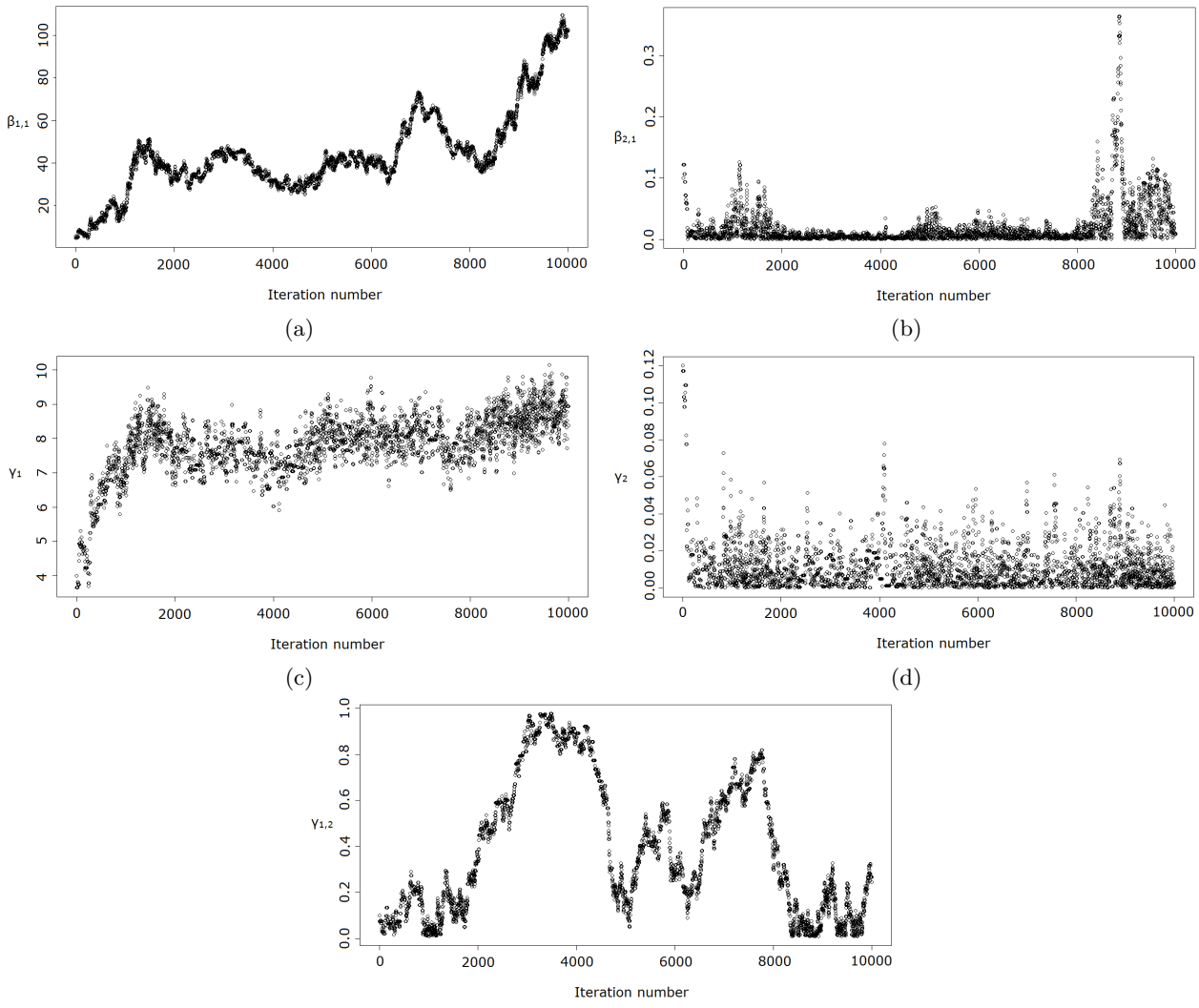

Figure 12: Estimation of the parameters of the coalescent model by the use of the MCMC method described in Section 5 with 10,000 iterations and the initial parameter set $(\beta_{1,1}, \beta_{2,1}, \gamma_1, \gamma_2, \gamma_{1,2}) = (5.5, 0.1, 4, 0.12, 0.1)$. The corresponding standard deviations used are equal to $(1.1, 0.02, 0.5, 0.015, 0.02)$ respectively. The parameter values are the rates per year.

We clearly see from the figures that the distributions of the parameter values didn't reach convergence yet, especially for the distributions of the parameters $\beta_{1,1}$, $\beta_{2,1}$ and $\gamma_{1,2}$. Two possible explanations could be the use of too low standard deviations or the use of a too low number of iterations. Increasing the number of iterations would lead to a computational cost; for $10,000$ iterations it already took several days. Therefore this method doesn't seem to be efficient for this model. However, these pictures seem to imply that the values for $\beta_{1,1}$ and γ_1 are far higher than we expect. The average value of $\beta_{1,1}$ in Figure 12a equals 44.227 and the average number of γ_1 in Figure 12c equals 7.760. For example, using the parameter values $(\beta_{1,1}, \beta_{2,1}, \gamma_1, \gamma_2, \gamma_{1,2}) = (44, 0.001, 7.86, 0.002, 0.77)$ gives us indeed a lower negative log likelihood (307.053) than for our results in Table 6 (307.0988). Although the difference is very small, the MCMC method is going to find the parameters corresponding to the minimum negative log likelihood, so probably with very high values for $\beta_{1,1}$ and γ_1. In the example given this would mean that an acutely infected individual will infect on average 5.04 individuals during its acute period which lasts on average 6 weeks. This is not realistic.

Using prior information about the parameter values will therefore be necessary in order for the coalescent model to find reliable results. The way in which we calculated the parameters in Table 6 already made use of some priors, namely by specifying grids in which we are searching for the minimum negative log likelihood (see Appendix B.2). However, what also can be seen is that some of the parameter values found are on the boundaries of our grids. This could imply that we excluded some parameter sets corresponding to an even lower negative log likelihood. For example for values of $\beta_{2,1}$, γ_2 and $\gamma_{1,2}$ tending to zero or for values of $\beta_{1,1}$ between 5.5 and 6. By expanding our grid we found for instance that $(\beta_{1,1}, \beta_{2,1}, \gamma_1, \gamma_2, \gamma_{1,2}) = (5.46, 0.002, 4.05, 0, 0.183)$ gives a lower negative log likelihood (306.401). This negative log likelihood is even smaller than the smallest value calculated with the MCMC method (306.450). This suggests that it is indeed reasonable to set some priors.

Although we didn't find the parameter set corresponding to the real minimum negative log likelihood, our results in Table 6 seem to be close to the real optimal values. They also seem to reflect the transmission dynamics well. Regardless of whether we look at these results or the two better parameter sets discussed above, in all these three cases the acutely infected individuals cause almost all new infections and the contribution of the chronic infections is of minor importance.

Coalescent model applied to the phylogenetic tree in Figure 9

In order to check whether the model presented in Section 4 constructs a good phylogenetic tree, we compare the results from the previous section with the parameter values estimated for the phylogenetic tree in Figure 9. The results are presented in Table 7. Note that we also included the results from Table 6 to make it easy to compare.

The results for the phylogenetic tree in Figure 9 can be presented in the probability matrix P, denoting the probabilities of infection at the beginning of an epidemic when almost all individuals are susceptible and the probability for a transition from the acute to the chronic state of infection.

$$P = \begin{array}{c} \\ A \\ C \end{array} \begin{array}{c} \begin{array}{cc} A & C \end{array} \\ \begin{pmatrix} 0.566 & 0.025 \\ 0.001 & 0 \end{pmatrix} \end{array}.$$

We can interpret these results as follows. When an individual becomes infected, it stays on average $1/(\gamma_1 + \gamma_{1,2} + \psi) = 0.261$ years, i.e., over 13 weeks, in the acute state. At the beginning of the epidemic, when almost all individuals are susceptible, an acutely infected individual infects a new acutely infected individual with probability 0.566 and during his acute period the individual infects on average $\beta_{1,1}/(\gamma_1 + \gamma_{1,2} + \psi) = 1.302$ new acute individuals. About $2,5\%$ of the acutely infected individuals is estimated to progress to the chronic state. The chronic period last on average $1/(\gamma_2 + \psi) = 10$ years. A

Parameters	MLE based on	
	Figure 9	Figure 8
$\beta_{1,1}$	4.99	5.501
$\beta_{2,1}$	0.0001	0.001
γ_1	3.636	4.116
γ_2	0	0.007
$\gamma_{1,2}$	0.096	0.105
R_0	1.302217	1.273
S	18430.11	18848.48
I_1	145.3521	119.2205
I_2	139.5380	116.9921
Negative Log Likelihood	331.2661	307.099

Table 7: Coalescent model parameters for the hepatitis B data using the phylogenetic tree in Figure 9, $N = 24000$ and $\psi = 0.1$. $\beta_{1,1}$, $\beta_{2,1}$,γ_1, γ_2 and $\gamma_{1,2}$ are the rates per year. For convenience the results from Table 6 are presented in the third column.

chronically infected individual will infect new acutely infected individuals with probability 0.001 and during his chronic period the individual causes $\beta_{2,1}/(\gamma_2 + \psi) = 0.001$ new infections, both at the beginning of an epidemic. The estimated reproduction number equals 1.302, so an infected individual will on average cause 1.302 new infections in a totally susceptible population. The effective reproduction number equals $R_0 \cdot \frac{S}{N} = 1$, which corresponds to our assumption that the infected population is constant over time.

The total number of new acute infections per year equals on average $145.3521 \cdot (1/0.261) = 556.989$ and the expected number of new chronic infections per year equals $139.5380 \cdot (1/10) = 13.954$.

When we compare these results with the results for the phylogenetic tree in Figure 8 we don't see large differences. We therefore conclude that despite the great visual differences between the phylogenetic trees in Figures 8 and 9, parameter inference gives us comparable results. Therefore the model used to generate a phylogenetic tree seems to perform well. Again the acutely infected individuals are almost totally responsible for the new infections.

7 Discussion

We estimated the transmission dynamics of hepatitis B among MSM in the Netherlands by fitting two different transmission models to a phylogenetic tree generated from the data. The first was a two-type birth-death branching model with constant birth, death and state transition rates for each type. We found results implying an infinite population size of infected individuals, even in case we added prior information to the model. The second model used for parameter inference was a coalescent model with type-specific birth, death and state transition rates depending on the number of susceptible and infected individuals in the population. For this model we found results that are consistent with observed prevalence of infection, indicating that acutely infected individuals are mostly responsible for the new infections among MSM in the Netherlands.

An important difference between the two models is the stochastically varying infected population size in the two-type birth-death branching model and the deterministic population size in the coalescent model. Where in the coalescent model the infected population size is determined by the dynamics of the SIR model and is included in the birth, death and state transition rates, the infected population size in the two-type birth-death branching model isn't explicitly stated. For each time interval it can be deduced from the number of sampled individuals, the sampling fraction and the death rate since the number of infected individuals of a certain type equals the number of these infected individuals becoming-non-infectious, i.e. the number of sampled individuals of this type divided by its sampling fraction, multiplied with the type-specific death rate. A second difference is that in the two-type birth-death branching model the sampling times are part of the data while in the coalescent model we condition on these times. In the two-type birth-death branching model the (constant) sampling fractions imply random sampling. However, for the coalescent model it isn't known how the samples are taken. We suspect that in this model random sampling is ensured by the underlying SIR model. Each susceptible individual is equally likely to become infected and in each state of infection all individuals have equal infection rates, which implies a homogeneous population. In a homogeneous population sampling will be random because all individuals are of the same type.

For this study we assumed that an infected individual is immediately infectious. In reality it takes some time before the infected individual is infectious itself. This assumption made it possible to assume that at each time of the phylogenetic tree an infection could occur. A susceptible-exposed-infected-recovered (SEIR) model could be used to account for this possibility. Just like for the compartments of the SIR model, the duration of the exposed period is assumed to be exponentially distributed. Although the distributions of these periods seem more Gaussian distributed, the SIR and SEIR model seem fair approximations of the real population dynamics.

To estimate the transmission parameters for both models we needed the time of a bifurcation event in the phylogenetic tree to equal the time of transmission. Furthermore each infected host should correspond to a single lineage in the phylogenetic tree. These requirements were equivalent to assuming a single pathogen sequence per infected host, which can mutate during the time within this host, and equal rates for replication of the pathogen and transmission between hosts. Because for our data the number of sequenced pathogens is relatively small compared to the total number of infected individuals each year, the difference in time between bifurcation and transmission events will be small [6]. Furthermore superinfection, multiple infections of different HBV genotypes, is rare [34]. Therefore our requirements seem to be fair approximations. The assumption that each infected host has only one pathogen sequence also implies that the mutation rate equals the substitution rate, which is needed for generating a phylogenetic tree.

Another assumption made for both models is that at the moment of sampling an individual is also recovered. If this wasn't the case an individual could have transmitted to one of the other sampled individuals after it has been sampled, but then the bifurcation event in the phylogenetic tree relating these two sampled individuals cannot be seen as a transmission event and we couldn't use the models for parameter inference. However, where this assumption is suitable for an infectious disease like HIV where sampled individuals receive a

treatment reducing their infectivity, it doesn't hold for individuals with an HBV infection; while chronically infected individuals are often treated by reducing the viral replication, acutely infected individuals almost never receive treatment because they will often cure fast and on their own. In case of the previous example where a sampled individual infected another sampled individual after it was sampled, the branch lengths in the phylogenetic tree from these two individuals to their common ancestor could be overestimated. We especially expect this to be a problem for chronically infected individuals because they still can be infectious for a long time after they got sampled. It is less probable that acutely infected individuals will cause infections after they got sampled because they are only shortly infectious. However, after being sampled acutely infected individuals can progress to the chronic state from which new infections can be caused. We therefore expect both an underestimation of the number of infections caused by chronically infected individuals and an underestimation of the rate at which acutely infected individuals progress to the chronic state. This would result in less infections caused by acutely infected individuals.

In the two-type birth-death branching model constant rates for birth, death and state transition are assumed. In case these rates correspond to a reproduction number $R_0 > 1$ the total infected population will grow exponentially over time. This is a shortcoming of the model because eventually the exponential growth of the infected population needs to be slowed down due to saturation effects; transmission decreases due to depletion of the group of susceptibles. From our results we saw that in case a constant infected population over time is assumed or expected from the data, the number of infected individuals became infinite in order to satisfy the exponential growth of the infected population size. A possible solution would therefore be variable birth, death and state transition rates over time, directly depending on the infected population size by using a saturation effect.

For the coalescent model also a constant infected population size over time is assumed. By using prior information about the intervals in which the parameters are situated, we defined grids over which we minimized the negative log likelihood. This resulted in good parameter values and estimates for the number of susceptibles and acutely infected individuals. The number of chronically infected individuals seemed too low, although this might be explained if infected individuals in the chronic state are mostly inactive and therefore do not belong to our total population, or if a chronic infection occurs at a later age, which implies an incorrect assumption that the rate ψ of leaving the population due to a natural death or change in behavior is equal for each state of infection. An improvement of the method would be to use the real prevalence of hepatitis B under MSM in the Netherlands. In this case we also do not need to use a constant ψ anymore. Furthermore, the MCMC chain applied to estimate the parameter values and their credibility intervals showed that the distributions of the parameters didn't reach convergence yet. A first reasonable explanation could be that the number of iterations used for the MCMC chain might be too small. Increasing this number would lead to a great computational cost; it took us already several days to perform $10,000$ iterations. Another possible explanation could be that the credibility intervals are very wide due to a lack of data. Therefore our results might not be very precise, although we don't expect this to change our main finding that the acutely infected individuals are most important in the spread of hepatitis B among MSM much. It could be valuable to add data and check whether this gives us more precise results. For this study there are still some HBV sequences available from Dutch MSM, also including partners. However, these sequences are not full length but they only cover the C or S gene from the virus.

Whether both models can be used for estimating the transmission dynamics of hepatitis B among MSM in the Netherlands can be tested by simulating our epidemic. Using different parameter sets we can simulate our epidemic by using each of the transmission models. By randomly choosing the sampled sequences we can re-estimate our parameters by applying the same transmission model. In case we get the same parameter values as used for simulating the epidemic, the models are valid. This is an important step for future research.

To generate the phylogenetic tree we used the HKY substitution model with equal rates at which different sites evolve. In case of the coalescent model we checked the results for two different phylogenetic trees for the

same data. They gave similar results and therefore we expect the model used to generate phylogenetic trees to perform well. However, a better way for estimating the parameters for each model would be by averaging the optimal parameter values estimated for a large number of phylogenetic trees generated from BEAST, weighted according to their posterior probabilities. Ideally the phylogenetic tree and the transmission rates will be inferred at the same time instead of subsequently generating the phylogenetic tree from the data and fitting a transmission model to the tree. In this case it is possible to apply different substitution rates for each different type and time; the estimated state of the individual at a certain branch and time can be used to generate the phylogenetic tree. This relaxes the molecular clock hypothesis assumed in this study. We recommend this simultaneous estimation together with using different substitution rates for future research.

This study shows the great importance of using the right transmission model for phylogenetic inference. The assumptions made are restrictive and may not be well-defined for our situation and can therefore lead to wrong parameter estimates. The two-type birth-death branching model with constant rates doesn't fit our data and can be improved by using variable birth, death and state transition rates over time. In recent literature a new transmission model building on the birth-death branching model but using saturation effects has been published [35]. In this model the population dynamics behave like a stochastic susceptible-infected-susceptible (SIS) model. Although it can be extended to the SIR model, this would lead to a large computational cost. Furthermore, the transmission model is only suitable for infectious diseases with one state of infection. An extension has to be made in order to let it be appropriate for the spread of hepatitis B.

The results found by the coalescent model confirm our current knowledge of the HBV infection. We estimated that an acute infection lasts about 12 weeks, an acute infection progresses in about 2.4% of the cases to the chronic state and the number of new acute infections each year is about 500. The results also show us that new infections are almost all caused by acutely infected individuals, which wasn't known before. Further research should confirm whether we could rely on these results.

Appendices

A Construction of the phylogenetic tree: technical details

To construct a phylogenetic tree from our sequence data we follow the next steps.

1. Aligned genetic sequences, so with a maximum number of matching nucleotides at each site, and their sampling times are provided to BEAST [20]. We use the Hasegawa-Kishino-Yano (HKY) substitution model with equal rates at which sites evolve. As a tree prior we use the linear Extended Bayesian Skyline plot, calculating the effective population size through time, i.e., the population size of interest in an idealized model. This Extended Bayesian Skyline plot is used when minimal assumptions about the population growth are desirable. The population grows or declines linearly between two points of change in the effective population size.

2. We let BEAST produce 1001 posterior phylogenetic trees by using a Monte Carlo Markov Chain analysis.

3. These 1001 trees are summarized into one phylogenetic tree by using TreeAnnotator which is part of the BEAST package [20]. This single tree is the maximum clade credibility tree; for each posterior tree the posterior clade probabilities, the probability that a clade (a collection of an ancestor and all of its descendants) is present in all posterior trees, are multiplied and the tree corresponding to the highest product is chosen. The node heights are based on the median heights for the clades in the maximum clade credibility tree.

4. In case the maximum clade credibility tree consists of branches with negative-lengths, so one of the descendent nodes is older than its direct ancestor, we correct for these by using R [21] by placing the descendent node one-tenth year after its direct ancestor. These negative branches can occur when the times of two adjacent nodes are derived from different sets of posterior trees and therefore may not have any direct ancestor-descendent relationship. This can happen when one of the clades of the maximum clade credibility tree is at low frequency in the total posterior sample of trees and tends not to occur in those trees that have the maximum clade credibility tree's ancestral clade [20].

5. Assign the states to each of the leaf nodes by using R [21].

6. The resulting tree is visualized by FigTree [22], using a reversed time scale.

We use the generated tree as the phylogenetic tree for our data. The code for adapting the phylogenetic tree by the use of R [21] can be found in Appendix C.1.

B Determination of the optimal parameters using a pre-specified grid

B.1 Two-type birth-death branching model

Parameter estimation in case of constant sampling fractions over time

To determine the maximum likelihood estimators of the parameters of the two-type birth-death branching model with constant sampling fractions over time, we applied the optimization method using the grids summarized from left to right in Table 8. After applying these grids we applied the path methods denoted in Table 9. Note that for the used parameter optimization method we determine the MLEs by finding the parameters corresponding to the minimum negative log likelihood.

Parameters	Grid 1			Grid 2		
	Interval	Step size	MLE	Interval	Step size	MLE
$\lambda_{1,1}$	[1:6]	1	1	[0.5:3]	0.5	0.5
$\lambda_{2,1}$	[0.5:2.5]	0.5	0.5	[0.3:1.5]	0.3	0.3
d_1	[0.5:5]	0.5	4.5	[3:5]	0.5	3
d_2	[0.02:0.1]	0.02	0.02	[0.01:0.1]	0.01	0.01
$\gamma_{1,2}$	[0.4:2]	0.4	0.4	[0.1:0.5]	0.1	0.5

Table 8: Determination of the parameters corresponding to the hepatitis B specific two-type birth-death branching model with constant sampling fractions using different grids. For each applied grid the interval and step size for each parameter are given and we denote the value of the parameter corresponding to the minimum negative log likelihood (MLE). The order in which the grids are applied is from left to right; the MLEs found from the previous grid are included in the intervals used for the next grid.

Parameters	Path 1		Path 2		Path 3	
	Step size	MLE	Step size	MLE	Step size	MLE
$\lambda_{1,1}$	0.01	0.5	0.01	1.08	0.001	1.193
$\lambda_{2,1}$	0.01	0.08	0.01	0.02	0.001	0.008
d_1	0.01	0.01	0.001	0.001	0.0001	0.0001
d_2	0.01	0.01	0.001	0.001	0.0001	0.0001
$\gamma_{1,2}$	0.01	1.86	0.01	1.19	0.001	1.19

Table 9: Determination of the parameters corresponding to the hepatitis B specific two-type birth-death branching model with constant sampling fractions using different paths after applying the grids in Table 8. For each applied path and each parameter the step sizes and the MLE, the parameter corresponding to the minimum negative log likelihood, are given. The order in which the paths are applied is from left to right. The MLEs from the previous path are used as the initial values for the next path.

As can be seen the MLEs found for the second grid in Table 8 are all on the boundaries of the grid. This suggests that the optimal parameter values are outside this grid. This can also be seen from our results in Table 9.

Parameter estimation in case of variable sampling fractions over time

For the two-type birth-death branching model with variable sampling fractions over time we determined the MLE by applying the steps described in Table 10. Again we find the maximum likelihood estimator by taking the parameter value corresponding to the minimum negative log likelihood. Note that we have two extra parameters in this model which we need to estimate: c_1 and c_2.

Parameters	Grid 1			Path 1		Path 2		Path 3	
	Interval	Step size	MLE	Step size	MLE	Step size	MLE	Step size	MLE
$\lambda_{1,1}$	[1:5]	1	1	0.1	1.1	0.01	1.42	0.001	1.423
$\lambda_{2,1}$	[0.25:1]	0.25	0.25	0.01	0.03	0.001	0.001	0.0001	0.001
d_1	[1:4]	1	2	0.1	0.1	0.001	0.001	0.0001	0.0001
d_2	[0.025:0.1]	0.025	0.1	0.001	0.001	0.0001	0.0001	0.00001	0.0001
$\gamma_{1,2}$	[0.25:1]	0.25	1	0.01	1.27	0.001	1.271	0.001	1.271
c_1	[0.01:0.04]	0.01	0.01	0.001	0.001	0.0001	0.0001	0.00001	0.00001
c_2	[0.01:0.04]	0.01	0.01	0.001	0.001	0.0001	0.0001	0.00001	0.00001

Table 10: Determination of the parameters corresponding to the hepatitis B specific two-type birth-death branching model with variable sampling fractions using a grid and multiple paths. For the applied grid the interval and step size for each parameter is given and we denote the value of the parameter corresponding to the minimum negative log likelihood by the MLE. For each applied path and each parameter the step sizes and the MLE, the parameter corresponding to the minimum negative log likelihood, are given. The order in which the steps are applied is from left to right; the MLEs found from the grid are used as initial values for the first path and so on.

We saw that in case d_1, d_2, c_1 and c_2 tend to zero, the negative log likelihood even became smaller. Because this means that we could have proceed further infinitely long by taking smaller step sizes, we decided to stop after path 3. Note that the MLEs found for the grid, except the MLE for d_1, are on the boundaries of the grid. This implies our optimal parameter values are outside the grid.

Parameter estimation in case of variable sampling fractions over time and using prior information

In case we use the prior information that $d_1 + \gamma_{1,2} \in [3:4]$ together with the 2-type birth-death branching model with variable sampling fractions over time, the optimal model parameters are estimated by the steps summarized in Table 11.

Parameters	Grid 1			Path 1	
	Interval	Step size	MLE	Step size	MLE
$\lambda_{1,1}$	[1:5]	1	1	0.1	2.3
$\lambda_{2,1}$	[0.25:1]	0.25	0.25	0.01	0.06
d_1	[1:4]	1	2	0.1	1.1
d_2	[0.025:0.1]	0.025	0.1	0.001	0.087
$\gamma_{1,2}$	[0.25:1]	0.25	1	0.01	1.99
c_1	[0.01:0.04]	0.01	0.01	0.001	0.001
c_2	[0.01:0.04]	0.01	0.01	0.001	0.001

Table 11: Determination of the parameters corresponding to the hepatitis B specific two-type birth-death branching model with variable sampling fractions using one grid and path. We furthermore use the prior information that $d_1 + \gamma_{1,2} \in [3:4]$. For the applied grid the interval and step size for each parameter is given and we denote the value of the parameter corresponding to the minimum negative log likelihood (MLE). For the applied path the step sizes and the MLE, corresponding to the minimum negative log likelihood, for each parameter are given. The order in which the methods are applied is from left to right; the MLEs found from the grid are used as initial values for the path.

A lower negative log likelihood can be obtained in case c_1 and c_2 become smaller and tend to zero. However, we decided to stop here. Note that the MLEs for all parameters except d_1 are on the boundaries of the grid we optimized over. This suggests our optimal parameter values are outside the grid, which is also suggested by the final results in Table 11.

B.2 Coalescent model

Parameter estimation for the phylogenetic tree in Figure 8

We use that our data is represented by the phylogenetic tree in Figure 8. Estimation of the parameter values corresponding to the minimum negative log likelihood when using the coalescent model is then performed by applying the optimization method summarized in Table 12 and 13.

Parameters	Grid 1			Grid 2		
	Interval	Step size	MLE	Interval	Step size	MLE
$\beta_{1,1}$	[1:6]	1	5	[2:5.5]	0.5	5.5
$\beta_{2,1}$	[0.5:2.5]	0.5	0.5	[0.1:1]	0.1	0.1
γ_1	[0.5:5]	0.5	5	[3.2:5.2]	0.4	4
γ_2	[0.02:0.1]	0.02	0.1	[0.04:0.16]	0.04	0.12
$\gamma_{1,2}$	[0.4:2]	0.4	0.4	[0.1:1]	0.1	0.1

Table 12: Determination of the parameters corresponding to the hepatitis B specific coalescent model using different grids. The phylogenetic tree used is in Figure 8. For each applied grid the interval and step size for each parameter are given and we denote the value of the parameter corresponding to the minimum negative log likelihood (MLE). The order in which the grids are applied is from left to right; the MLEs found from the previous grid are included in the intervals used for the next grid.

Parameters	Path 1		Path 2		Path 3	
	Step size	MLE	Step size	MLE	Step size	MLE
$\beta_{1,1}$	0.01	5.5	0.01	5.5	0.001	5.501
$\beta_{2,1}$	0.001	0.013	0.001	0.001	0.0001	0.001
γ_1	0.01	4	0.01	4.12	0.001	4.116
γ_2	0.01	0.01	0.001	0.007	0.001	0.007
$\gamma_{1,2}$	0.001	0.1	0.001	0.104	0.001	0.105

Table 13: Determination of the parameters corresponding to the hepatitis B specific coalescent model using different paths after applying the grids in Table 12. The phylogenetic tree used is in Figure 8. For each applied path and each parameter the step sizes and the MLE, the parameter corresponding to the minimum negative log likelihood, are given. The order in which the paths are applied is from left to right. The MLEs from the previous path are used as the initial values for the next path.

Note that the MLEs found by optimizing over the second grid are partially on the boundaries of this grid. This suggests these parameter values might be outside the pre-defined ranges. The results from Table 13 also show this.

Parameter estimation for the phylogenetic tree in Figure 9

We use the phylogenetic tree in Figure 9 to represent our data. To estimate the parameter values corresponding to the minimum negative log likelihood when using the coalescent model we apply the optimization method summarized in Table 14 and 15.

Parameters	Grid 1			Grid 2		
	Interval	Step size	MLE	Interval	Step size	MLE
$\beta_{1,1}$	[1:6]	1	4	[2:5.5]	0.5	5
$\beta_{2,1}$	[0.5:2.5]	0.5	0.5	[0.1:1]	0.1	0.1
γ_1	[0.5:5]	0.5	4	[3.2:5.2]	0.4	3.6
γ_2	[0.02:0.1]	0.02	0.1	[0.04:0.16]	0.04	0.12
$\gamma_{1,2}$	[0.4:2]	0.4	0.4	[0.1:1]	0.1	0.1

Table 14: Determination of the parameters corresponding to the hepatitis B specific coalescent model using different grids. The phylogenetic tree used is in Figure 9. For each applied grid the interval and step size for each parameter are given and we denote the value of the parameter corresponding to the minimum negative log likelihood (MLE). The order in which the grids are applied is from left to right; the MLEs found from the previous grid are included in the intervals used for the next grid.

Parameters	Path 1		Path 2		Path 3	
	Step size	MLE	Step size	MLE	Step size	MLE
$\beta_{1,1}$	0.01	5	0.01	4.99	0.001	4.99
$\beta_{2,1}$	0.001	0.012	0.001	0.001	0.0001	0.0001
γ_1	0.01	3.6	0.01	3.64	0.001	3.636
γ_2	0.01	0.01	0.001	0	0.001	0
$\gamma_{1,2}$	0.001	0.1	0.001	0.098	0.001	0.096

Table 15: Determination of the parameters corresponding to the hepatitis B specific coalescent model using different paths after applying the grids in Table 14. The phylogenetic tree used is in Figure 9. For each applied path and each parameter the step sizes and the MLE, the parameter corresponding to the minimum negative log likelihood, are given. The order in which the paths are applied is from left to right. The MLEs from the previous path are used as the initial values for the next path.

Note that the MLEs shown for the second grid in Table 14 are partially on the boundaries of the grid. We therefore expect some of these parameter values to lie outside the grid. We also see this from Table 15.

C Codes

C.1 Correction of the phylogenetic tree generated by TreeAnnotator [20]

```
# Set the working directory and load the packages we need
setwd("N:/Data_hepatitis_b")
library(ape)

# Load the genetic sequences from the data
x <- read.dna("acuutenchronisch_goed2.fsta", format='fasta')
N <- dim(x)[1]

# Read the tree generated by Tree Annotator
treebeast <- read.nexus("BEAST/outputTreeAnnotator_acuutenchronisch_goed.txt")

# In case of negative branch lengths, place its descending note 0.01 year
    after the ancestor
# Also replace the edge lengths.
for (i in 1:(2*N-2)){
        if (treebeast$edge.length[i] < 0 & treebeast$edge[i,2] > N){# for
            branches that connect two internal nodes
                treebeast$edge.length[treebeast$edge[,1]==treebeast$edge[i
                    ,2]][1] <- treebeast$edge.length[treebeast$edge[,1]==
                    treebeast$edge[i,2]][1] + treebeast$edge.length[i] - 0.01
                treebeast$edge.length[treebeast$edge[,1]==treebeast$edge[i
                    ,2]][2] <- treebeast$edge.length[treebeast$edge[,1]==
                    treebeast$edge[i,2]][2] + treebeast$edge.length[i] - 0.01
                treebeast$edge.length[i] <- 0.01
        }
        else{# for branches corresponding to a leaf node
                if (treebeast$edge.length[i] < 0 & treebeast$edge[i,2] <= N){
                        treebeast$edge.length[treebeast$edge[,2]==treebeast$
                            edge[i,1]] <- treebeast$edge.length[treebeast$edge
                            [,2]==treebeast$edge[i,1]] + treebeast$edge.length
                            [i] - 0.01
                        j <- which(treebeast$edge[,1]==treebeast$edge[i,1] &
                            treebeast$edge[,2]!=treebeast$edge[i,2])
                        treebeast$edge.length[j] <- treebeast$edge.length[j] -
                             treebeast$edge.length[i] + 0.01
                        treebeast$edge.length[i] <- 0.01
                }
        }
}

# Save the corrected tree
write.nexus(treebeast, file = "BEAST/outputTreeAnnotator_acuutenchronisch_goed
    _modified.txt", translate = FALSE)

# Remove tip "A35_A_1999.288" from the phylogenetic tree
tree <- treebeast
tree <- drop.tip(treebeast,c("A35_A_1999.288"))
```

```
# Denote the states to the leaf nodes (check tree$tip.label to check for the
    order of the leaf nodes)
tree$states <- rep(c(1,2), times=c(56,27))

# Save the tree, note that the first N nodes (sorted) correspond to the leaf
    nodes, node N+1 equals the root of the tree and N+2 t/m 2*N-1 are the
    other internal nodes in descending order
save(tree, file="R/tree2_without_A35")
```

C.2 Two-type birth-death branching model with constant sampling fractions

```
# Function for calculating the likelihood of a certain parameter set par given
    a phylogenetic tree
# brpoint=0 means we calculate the likelihood for the whole phylogenetic tree,
    else brpoint equals the number of years back in time from the present at
    which we cut the tree. It should at least be greater than the time (number
    of years from the present) of the left-most leaf node but smaller than
    the time (number of years from the present) of the root of the tree.
# par must be (lambda11, lambda12, lambda21, lambda22, death1, death2,
    gamma12, gamma21)
# fix determines which parameters are constraint when optimizing is performed.
    First row specifies the parameters being constraint (1 for lambda11, 2
    for lambda12 etc). Second row: (i) If entry (2,j) is non-negative, say x,
    then parameter (1,j) is fixed to x. (ii) If entry (2,j) is negative, say -
    m, then parameter (1,j) is fixed to parameter m times entry (3,j) (
    exception is m=0.4: then the parameter lambda22 is fixed to lambda21*
    lambda12/lambda11, used in Stadler et al [24] for superspreaderdynamics)
# survival=1 conditions the likelihood on sampling at least one tip (or one
    tip per root descendant if root=1)
# posR=1 constrains the parameters (when optimizing) on R0>1
# unknownStates=FALSE means that the states of the leaf nodes are known
# root=0 indicates that there is an edge above the root (mrca) in the tree
    phylo. root=1 indicates that there is no edge above the root.
# states[i] belongs to leaf i
# The sampling fractions are constant
# setwd("N:/Data hepatitis b/R")
# With sourceDirectory("Functies_Tanja", modifiedOnly=FALSE); we can load all
    functions at once
# Also load the phylogenetic tree

bdtypes.stt.lik.statechange <- function(brpoint,par,phylo,fix=rbind(c(0,0),c
    (0,0)),sampfrac,survival=0,posR=0,unknownStates=FALSE,root=0){
        prpar <- FALSE
        maxpar <- 100
        partemp <- vector()
        k <- 1
        for (i in 1:8){
                index <- which(i == fix[1,])
                if (length(index)>0){
                        if (fix[2,index]>=0){
                                partemp <- c(partemp,fix[2,index])
                        }
                        else{
                                temp <- - fix[2,index]
                                if (temp == 0.4){#make lambdas in same ratio
                                        partemp <- c(partemp,partemp[3]*
                                            partemp[2]/partemp[1])
                                }
                                else{
                                        partemp <- c(partemp,partemp[temp]*fix
```

```
                                        [3,index])
                                }
                        }
                }
                else {
                        partemp <- c(partemp,par[k])
                        k <- k+1
                }
}
#print(partemp)

death <- partemp[5:6]
l <- partemp[1:4]
gamma <- partemp[7:8]
psi <- death*sampfrac
m <- death*(1-sampfrac)

if (root==1){
        cut <- phylo$edge[1,1]
        for (i in 1:length(phylo$edge[,1])){
                if (phylo$edge[i,1] >= cut){phylo$edge[i,1] <- phylo$
                    edge[i,1]+1}
                if (phylo$edge[i,2] >= cut){phylo$edge[i,2] <- phylo$
                    edge[i,2]+1}
        }
        phylo$edge <- rbind(c(cut,phylo$edge[1,1]),phylo$edge)
        phylo$edge.length <- c(0,phylo$edge.length)
}

outmatrix <- vector() # needed for cutting of tree at t=brpoint
summary <- get.times2(phylo)
out <- 10^10
temp <- 1
R0temp <- try(R0types.statechange(l[1],l[2],l[3],l[4],death[1],death
    [2],gamma[1],gamma[2]))
if (posR==1 && class(R0temp)=="numeric" && R0temp<1){temp <- 0}
if (posR==1 && class(R0temp)=="try-error"){temp <- 0}
check <- ((length(which(partemp=="NaN"))>0)||(min(l,psi))<0 || m<0 ||
     max(l,m,psi)>maxpar || (temp==0))
if (check){out <- 10^10}
else{
        if (brpoint==0){
                lik <- try(BDSSnum.help.statechange(brpoint,phylo,1,l,
                    gamma,m,psi,summary,unknownStates))
                if (class(lik)!="try-error"){
                        LambMu <- l[1]-l[4]-(m[1]+psi[1]+gamma[1])+(m
                            [2]+psi[2]+gamma[2])
                        c <- sqrt(LambMu^2 + 4*(l[2]+gamma[1])*(l[3]+
                            gamma[2]))
                        f1 <- (c+LambMu)/(c+LambMu+2*(l[2]+gamma[1]))
                        out <- try(-log((lik[3]*f1)/(1-lik[1]))^(
```

```
                                survival) + (lik[4]*(1-f1))/(1-lik[2])^(
                                survival)))
                            if((class(out)!="numeric") || (out=="NaN") ||
                                (out=="Inf") ){out <- 10^10}
                        }
                        else{out <- 10^10}
                    }
                    else{
                        for (i in 1:length(phylo$edge[,1])){# calculate
                            negative log likelihood for all separated trees
                            if (round(summary[phylo$edge[i,1],1], digits
                                =6)>round(brpoint, digits=6) && round(
                                summary[phylo$edge[i,2],1], digits=6)<=
                                round(brpoint, digits=6)){
                                lik <- try(BDSSnum.help.statechange(
                                    brpoint,phylo,i,l,gamma,m,psi,
                                    summary,unknownStates))
                                if (class(lik)!="try-error"){
                                    LambMu <- l[1]-l[4]-(m[1]+psi
                                        [1]+gamma[1])+(m[2]+psi
                                        [2]+gamma[2])
                                    c <- sqrt(LambMu^2 + 4*(l[2]+
                                        gamma[1])*(l[3]+gamma[2]))
                                    f1 <- (c+LambMu)/(c+LambMu+2*(
                                        l[2]+gamma[1]))
                                    out <- try(-log((lik[3]*f1)/
                                        (1-lik[1])^(survival) + (
                                        lik[4]*(1-f1))/(1-lik[2])
                                        ^(survival)))
                                    if((class(out)!="numeric") ||
                                        (out=="NaN") || (out=="
                                        Inf") ){out <- 10^10}
                                }
                                else{out <- 10^10}
                                outmatrix <- rbind(outmatrix,c(out,i))
                            }
                        }
                        out <- sum(outmatrix[,1])# sum the negative log
                            likelihoods of all separated trees
                    }
                }

                if (out>=10^10){out <- 10^1000}
                if (prpar==TRUE){print(par)}
                out <- c(out,par,R0temp)
                out
}

# Function to generate a matrix with for each node of the phylogenetic tree
    its time (the number of years from the present) and type (0 for a leaf
    node, 1 for an internal node)
```

```
get.times2 <- function(tree){
        nodes <- sort(unique(c(tree$edge)))
        ttype <- (1:length(nodes))*0
        times <- ttype
        ttype[tree$edge[1,1]] <- 1
        for (j in (tree$edge[1,1]+1):length(nodes)){
                ttype[j] <- 1
                temp <- which(tree$edge[,2]==j)
                ancestor <- tree$edge[temp,1]
                times[j] <- times[ancestor]+tree$edge.length[temp]
        }
        for (j in 1:(tree$edge[1,1]-1)){
                temp <- which(tree$edge[,2]==j)
                ancestor <- tree$edge[temp,1]
                times[j] <- times[ancestor]+tree$edge.length[temp]
        }
        maxt <- max(times)
        times <- -times+maxt
        out <- cbind(times,ttype)
        out
}

# Function for calculating R_0

R0types.statechange <- function(l11,l12,l21,l22,death1,death2,gamma12,gamma21)
    {
        R0 <- (l11+l12)/(death1+gamma12)+(gamma12/(death1+gamma12))*((l22+l21)
            /(death2+gamma21))
        R0
}

# Function for calculating the vector (E_1(t_0), E_2(t_0),D_{O1}(t_0),D_{O2}(t
    _0)) for the phylogenetic tree

BDSSnum.help.statechange <- function(brpoint,phylo,rootedge,l,gamma,m,psi,
    summary,unknownStates) {
        newroot <- phylo$edge[rootedge,2]
        newtrees <- which(phylo$edge[,1]==newroot)
        tyoung <- summary[phylo$edge[rootedge,2]]

        if (brpoint>0){
                told <- min(brpoint,summary[phylo$edge[rootedge,1]])
        }
        else{
                told <- summary[phylo$edge[rootedge,1]]
        }

        if (length(newtrees)==0) {# if the edge corresponds to a leaf node
                if (unknownStates==FALSE && phylo$states[newroot]>0){
                        state <- phylo$states[newroot]
                        initpsi <- c(0,0)
                        initpsi[state] <- log(psi[state])
```

```
                }
                else{
                        initpsi <- c(log(psi[1]),log(psi[2]))
                }
                inity1 <- integrator2.statechange(c(1,1),l,gamma,m,psi,c(0,
                    tyoung))
                res <- integrator.statechange(init=c(inity1,initpsi),l,gamma,m
                    ,psi,c(tyoung,told))
        }
        else{# if the edge connects two internal nodes
                likleft <- BDSSnum.help.statechange(brpoint,phylo,newtrees[1],
                    l,gamma,m,psi,summary,unknownStates)
                likright <- BDSSnum.help.statechange(brpoint,phylo,newtrees
                    [2],l,gamma,m,psi,summary,unknownStates)
                res1 <- c(likleft[1],likleft[3]*likright[3]*l[1]*2+ likleft[3]
                    *likright[4]*l[2]+ likleft[4]*likright[3]*l[2])  # state 1
                     above joining at tyoung
                res2 <- c(likleft[2],likleft[4]*likright[4]*l[4]*2+ likleft[3]
                    *likright[4]*l[3]+ likleft[4]*likright[3]*l[3])  # state 2
                     above joining at tyoung
                res <- integrator.statechange(init=c(res1[1],res2[1],log(res1
                    [2]),log(res2[2])),l,gamma,m,psi,c(tyoung,told))
        }
        res
}

# Function for solving the differential equations for E_1 and E_2

integrator2.statechange <- function(init,l,gamma,m,psi,times){
        ode <- function(times,y,p){
                lambda11 <- p[1]
                lambda12 <- p[2]
                lambda21 <- p[3]
                lambda22 <- p[4]
                gamma12 <- p[5]
                gamma21 <- p[6]
                mu1 <- p[7]
                mu2 <- p[8]
                psi1 <- p[9]
                psi2 <- p[10]

                yd1 <- mu1-(lambda11+lambda12+gamma12+mu1+psi1)*y[1]+lambda11*
                    y[1]*y[1]+lambda12*y[1]*y[2]+gamma12*y[2]
                yd2 <- mu2-(lambda21+gamma21+lambda22+mu2+psi2)*y[2]+lambda21*
                    y[1]*y[2]+lambda22*y[2]*y[2]+gamma21*y[1]
                list(c(yd1,yd2))
        }
        p <- c(l,gamma,m,psi)
        out <- lsoda(init,times,ode,p)[2,2:3]
        out
}
```

```
# Function for solving the differential equations for E_1, E_2, log(D_1) and
    log(D_2)

integrator.statechange <- function(init,l,gamma,m,psi,times){
        ode <- function(times,y,p){
                lambda11 <- p[1]
                lambda12 <- p[2]
                lambda21 <- p[3]
                lambda22 <- p[4]
                gamma12 <- p[5]
                gamma21 <- p[6]
                mu1 <- p[7]
                mu2 <- p[8]
                psi1 <- p[9]
                psi2 <- p[10]

                yd1 <- mu1-(lambda11+lambda12+gamma12+mu1+psi1)*y[1]+lambda11*
                    y[1]*y[1]+lambda12*y[1]*y[2]+gamma12*y[2]
                yd2 <- mu2-(lambda21+gamma21+lambda22+mu2+psi2)*y[2]+lambda21*
                    y[1]*y[2]+lambda22*y[2]*y[2]+gamma21*y[1]
                yd3 <- -(lambda11+lambda12+gamma12+mu1+psi1) + 2*lambda11*y[1]
                     + lambda12*y[1]*exp(y[4]-y[3]) + lambda12*y[2] + gamma12*
                    exp(y[4]-y[3])
                yd4 <- -(lambda22+lambda21+gamma21+mu2+psi2) + 2*lambda22*y[2]
                     + lambda21*y[2]*exp(y[3]-y[4]) + lambda21*y[1] +gamma21*
                    exp(y[3]-y[4])
                list(c(yd1,yd2,yd3,yd4))
        }
        out <- lsoda(init,times,ode,c(l,gamma,m,psi))[2,2:5]
        out[3:4] <- exp(out[3:4]) # to get the values for D_1 and D_2 instead
            of log(D_1) and log(D_2)
        out
}
```

C.3 Two-type birth-death branching model with variable sampling fractions

```
# Function for calculating the likelihood of a certain parameter set par given
    a phylogenetic tree
# brpoint=0 means we calculate the likelihood for the whole phylogenetic tree,
    else brpoint equals the number of years back in time from the present at
    which we cut the tree. It should at least be greater than the time (number
    of years from the present) of the left-most leaf node but smaller than
    the time (number of years from the present) of the root of the tree.
# par must be  (lambda11, lambda12, lambda21, lambda22, death1, death2,
    gamma12, gamma21, c1, c2)
# fix determines which parameters are constraint when optimizing is performed.
    First row specifies the parameters being constraint (1 for lambda11, 2
    for lambda12 etc). Second row: (i) If entry (2,j) is non-negative, say x,
    then parameter (1,j) is fixed to x. (ii) If entry (2,j) is negative, say -
    m, then parameter (1,j) is fixed to parameter m times entry (3,j) (
    exception is m=0.4: then the parameter lambda22 is fixed to lambda21*
    lambda12/lambda11, used in Stadler et al [24] for superspreaderdynamics)
# survival=1 conditions the likelihood on sampling at least one tip (or one
    tip per root descendant if root=1)
# posR=1 constrains the parameters (when optimizing) on R0>1
# unknownStates=FALSE means that the states of the leaf nodes are known
# root=0 indicates that there is an edge above the root (mrca) in the tree
    phylo. root=1 indicates that there is no edge above the root.
# states[i] belongs to leaf i
# The sampling fractions differ per year
# setwd("N:/Data hepatitis b/R")
# With sourceDirectory("Functies_Tanja", modifiedOnly=FALSE); we can load all
    functions at once
# Also load the phylogenetic tree

bdtypes.stt.lik.statechange.s <- function(brpoint,par,phylo,fix=rbind(c(0,0),c
    (0,0)),survival=0,posR=0,unknownStates=FALSE,root=0){
        prpar <- FALSE
        maxpar <- 100
        partemp <- vector()
        k <- 1
        for (i in 1:10){
                index <- which(i == fix[1,])
                if (length(index)>0){
                        if (fix[2,index]>=0){
                                partemp <- c(partemp,fix[2,index])
                        }
                        else{
                                temp <- - fix[2,index]
                                if (temp == 0.4){# make lambdas in same ratio
                                        partemp <- c(partemp,partemp[3]*
                                            partemp[2]/partemp[1])
                                }
                                else{
                                        partemp <- c(partemp,partemp[temp]*fix
```

```
                                        [3,index])
                        }
                }
        }
        else{
                partemp <- c(partemp,par[k])
                k <- k+1
        }
}
#print(partemp)

death <- partemp[5:6]
l <- partemp[1:4]
gamma <- partemp[7:8]
c1 <- partemp[9]
c2 <- partemp[10]

# define the sampling fraction for each state (rows) and each year (
    columns)
# the first column denotes all years before 1985 (the year in which
    the left-most leaf node of the phylogenetic tree is sampled)
# the second column denotes year 1985, the third year 1986 etc. until
    the 28th column which denotes 2011 (the year in which the right-
    most leaf node of the phylogenetic tree is sampled)
# the vectors of numbers equal the number of sampled acutely and
    chronically infected individuals during each year
sf <- matrix(NA,2,28)
sf[1,] <- c(0,0,2,0,0,0,0,0,3,0,3,0,2,0,2,0,1,1,2,0,7,6,3,4,6,7,6,1)*
    c1
sf[2,] <- c(0,15,4,0,0,1,0,2,0,1,0,0,0,2,0,0,0,0,1,0,0,1,0,0,0,0,0,0)*
    c2

oneminussf <- matrix(NA,2,28)
for (i in 1:28){
        oneminussf[,i] <- 1-sf[,i]
}

psi <- matrix(NA,2,28)
psi[1,] <- death[1]*sf[1,]
psi[2,] <- death[2]*sf[2,]

m <- matrix(NA,2,28)
m[1,] <- death[1]*oneminussf[1,]
m[2,] <- death[2]*oneminussf[2,]

if (root==1){
        cut <- phylo$edge[1,1]
        for (i in 1:length(phylo$edge[,1])){
                if (phylo$edge[i,1]>=cut){
                        phylo$edge[i,1] <- phylo$edge[i,1]+1
                }
```

```
                if (phylo$edge[i,2]>=cut){
                        phylo$edge[i,2] <- phylo$edge[i,2]+1
                }
        }
        phylo$edge <- rbind(c(cut,phylo$edge[1,1]),phylo$edge)
        phylo$edge.length <- c(0,phylo$edge.length)
}

outmatrix <- vector() # needed for cutting of tree at t=brpoint
summary <- get.times2(phylo)
out <- 10^10
temp <- 1
R0temp <- try(R0types.statechange(l[1],l[2],l[3],l[4],death[1],death
    [2],gamma[1],gamma[2]))
if (posR==1 && class(R0temp)=="numeric" && R0temp<1){temp <- 0}
if (posR==1 && class(R0temp)=="try-error"){temp <- 0}

# In case we use prior information
if (death[1]+gamma[1]<3 || death[1]+gamma[1]>4){
        temp <- 0
}

check <- ((length(which(partemp=="NaN"))>0)||(min(l,psi))<0 || min(m)
    <0|| max(l,m,psi)>maxpar || (temp==0))
if (check){
        out <- 10^10
}
else{
        if (brpoint==0){
                lik <- try(BDSSnum.help.statechange.s(brpoint,phylo,1,
                    l,death,gamma,m,psi,summary,unknownStates))
                if (class(lik)!="try-error"){
                        LambMu <- l[1]-l[4]-(m[1]+psi[1]+gamma[1])+(m
                            [2]+psi[2]+gamma[2])
                        c <- sqrt(LambMu^2 + 4*(l[2]+gamma[1])*(l[3]+
                            gamma[2]))
                        f1 <- (c+LambMu)/(c+LambMu+2*(l[2]+gamma[1]))
                        out <- try(-log((lik[3]*f1)/(1-lik[1])^(
                            survival) + (lik[4]*(1-f1))/(1-lik[2])^(
                            survival)))
                        if((class(out)!="numeric") || (out=="NaN") ||
                            (out=="Inf") ){out <- 10^10}
                }
                else{
                        out <- 10^10
                }
        }
        else{
                for (i in 1:length(phylo$edge[,1])){# calculate
                    negative log likelihood for all separated trees
                        if (round(summary[phylo$edge[i,1],1], digits
```

```
                                        =6)>round(brpoint, digits=6) && round(
                                        summary[phylo$edge[i,2],1], digits=6)<=
                                        round(brpoint, digits=6)){
                                            lik <- try(BDSSnum.help.statechange.s(
                                                brpoint,phylo,i,l,death,gamma,m,
                                                psi,summary,unknownStates))
                                            if (class(lik)!="try-error"){
                                                LambMu <- l[1]-l[4]-(m[1]+psi
                                                    [1]+gamma[1])+(m[2]+psi
                                                    [2]+gamma[2])
                                                c <- sqrt(LambMu^2 + 4*(l[2]+
                                                    gamma[1])*(l[3]+gamma[2]))
                                                f1 <- (c+LambMu)/(c+LambMu+2*(
                                                    l[2]+gamma[1]))
                                                out <- try(-log((lik[3]*f1)/
                                                    (1-lik[1])^(survival) + (
                                                    lik[4]*(1-f1))/(1-lik[2])
                                                    ^(survival)))
                                                if((class(out)!="numeric") ||
                                                    (out=="NaN") || (out=="Inf
                                                    ")){out <- 10^10}
                                            }
                                            else{
                                                out <- 10^10
                                            }
                                            outmatrix <- rbind(outmatrix,c(out,i))
                                    }
                                }
                                out <- sum(outmatrix[,1]) # sum the negative log
                                    likelihoods of all separated trees
                            }
                    }

                    if (out>=10^10){out <- 10^1000}
                    if (prpar==TRUE){print(par)}
                    out <- c(out,par,R0temp)
                    out
}

# Function to generate a matrix with for each node of the phylogenetic tree
    its time (the number of years from the present) and type (0 for a leaf
    node, 1 for an internal node)

get.times2 <- function(tree){
        nodes <- sort(unique(c(tree$edge)))
        ttype <- (1:length(nodes))*0
        times <- ttype
        ttype[tree$edge[1,1]] <- 1
        for (j in (tree$edge[1,1]+1):length(nodes)){
                ttype[j] <- 1
                temp <- which(tree$edge[,2]==j)
                ancestor <- tree$edge[temp,1]
```

```
                times[j] <- times[ancestor]+tree$edge.length[temp]
        }
        for (j in 1:(tree$edge[1,1]-1)){
                temp <- which(tree$edge[,2]==j)
                ancestor <- tree$edge[temp,1]
                times[j] <- times[ancestor]+tree$edge.length[temp]
        }
        maxt <- max(times)
        times <- -times+maxt
        out <- cbind(times,ttype)
        out
}

# Function for calculating R_0

R0types.statechange <- function(l11,l12,l21,l22,death1,death2,gamma12,gamma21)
    {
        R0 <- (l11+l12)/(death1+gamma12)+(gamma12/(death1+gamma12))*((l22+l21)
            /(death2+gamma21))
        R0
}

# Function for calculating the vector (E_1(t_0), E_2(t_0),D_{O1}(t_0),D_{O2}(t
    _0)) for the phylogenetic tree

BDSSnum.help.statechange.s <- function(brpoint,phylo,rootedge,l,death,gamma,m,
    psi,summary,unknownStates){
        newroot <- phylo$edge[rootedge,2]
        newtrees <- which(phylo$edge[,1]==newroot)
        tyoung <- summary[phylo$edge[rootedge,2]]

        if (brpoint>0){
                told <- min(brpoint,summary[phylo$edge[rootedge,1]])}
        else{
                told <- summary[phylo$edge[rootedge,1]]
        }

        if (length(newtrees)==0){# if the edge corresponds to a leaf node
                # calculation of log(D_1) and log(D_2) at the leaf node
                if (unknownStates==FALSE && phylo$states[newroot]>0){
                        state <- phylo$states[newroot]
                        initpsi <- c(0,0)
                        initpsi[state] <- log(psi[state,max(floor(get.times2.s
                            (phylo)[newroot,1])-1983,1)]) #1983 equals the
                            sampling year of the left-most leaf node minus two
                }
                else{
                        initpsi <- c(log(psi[1,max(floor(get.times2.s(phylo)[
                            newroot,1])-1983,1)]),log(psi[2,max(floor(get.
                            times2.s(phylo)[newroot,1])-1983,1)]))
                }
```

```
        b <- max(get.times2.s(phylo)[,1])-floor(max(get.times2.s(phylo
            )[,1]))
        inity1 <- integrator2.statechange.s(c(1,1),l,death,gamma,m,psi
            ,c(0,b))

        # calculation of E_1 and E_2 at the leaf node
        if (tyoung-b>=1){
                for (i in 1:floor(tyoung-b)){
                        inity1 <- integrator2.statechange.s(inity1,l,
                            death,gamma,m,psi,c(b-1+i,b+i))
                }
                inity1 <- integrator2.statechange.s(inity1,l,death,
                    gamma,m,psi,c(b+i,tyoung))
        }
        else{
                inity1 <- integrator2.statechange.s(inity1,l,death,
                    gamma,m,psi,c(b,tyoung))
        }

        res <- c(inity1,initpsi)
        e <- (tyoung+(get.times2.s(phylo)[newroot,1]-floor(get.times2.
            s(phylo)[newroot,1])))
        res <- integrator.statechange.s(res,l,death,gamma,m,psi,c(
            tyoung,e)) # this gives us D_1 and D_2, not log(D_1) and
            log(D_2)

        # calculation of (E_1,E_2,D_1,D_2) at time told
        if (told-e>=1){
                for (i in 1:floor(told-e)){
                        res <- integrator.statechange.s(c(res[1],res
                            [2],log(res[3]),log(res[4])),l,death,gamma
                            ,m,psi,c(e-1+i,e+i))
                }
                res <- integrator.statechange.s(c(res[1],res[2],log(
                    res[3]),log(res[4])),l,death,gamma,m,psi,c(e+i,
                    told))
        }
        else{
                res <- integrator.statechange.s(c(res[1],res[2],log(
                    res[3]),log(res[4])),l,death,gamma,m,psi,c(e,told)
                    )
        }
}
else{# if the edge connects two internal nodes
        likleft <- BDSSnum.help.statechange.s(brpoint,phylo,newtrees
            [1],l,death,gamma,m,psi,summary,unknownStates)
        likright <- BDSSnum.help.statechange.s(brpoint,phylo,newtrees
            [2],l,death,gamma,m,psi,summary,unknownStates)
        res1 <- c(likleft[1],likleft[3]*likright[3]*l[1]*2+likleft[3]*
            likright[4]*l[2]+likleft[4]*likright[3]*l[2])# state 1
            above joining at tyoung
```

```
                res2 <- c(likleft[2], likleft[4]*likright[4]*l[4]*2+likleft[3]*
                    likright[4]*l[3]+likleft[4]*likright[3]*l[3])# state 2
                    above joining at tyoung

                res <- c(res1[1], res2[1], log(res1[2]), log(res2[2]))
                g <- (tyoung+(get.times2.s(phylo)[newroot,1]-floor(get.times2.
                    s(phylo)[newroot,1])))
                res <- integrator.statechange.s(res, l, death, gamma, m, psi, c(
                    tyoung, g))

                # calculation of (E_1,E_2,D_1,D_2) at time told
                if (told-g>=1){
                        for (i in 1:floor(told-g)){
                                res <- integrator.statechange.s(c(res[1], res
                                    [2], log(res[3]), log(res[4])), l, death, gamma
                                    , m, psi, c(g-1+i, g+i))
                        }
                        res <- integrator.statechange.s(c(res[1], res[2], log(
                            res[3]), log(res[4])), l, death, gamma, m, psi, c(g+i,
                            told))
                }
                else{
                        res <- integrator.statechange.s(c(res[1], res[2], log(
                            res[3]), log(res[4])), l, death, gamma, m, psi, c(g, told)
                            )
                }
        }
        res
}

# Function to generate a matrix with for each node of the phylogenetic tree
    its time (prospectively in years) and type (0 for a leaf node, 1 for an
    internal node)

get.times2.s <- function(tree){
        nodes <- sort(unique(c(tree$edge)))
        ttype <- (1:length(nodes))*0
        times <- ttype
        ttype[tree$edge[1,1]] <- 1
        for (j in (tree$edge[1,1]+1):length(nodes)){
                ttype[j] <- 1
                temp <- which(tree$edge[,2]==j)
                ancestor <- tree$edge[temp,1]
                times[j] <- times[ancestor]+tree$edge.length[temp]
        }
        for (j in 1:(tree$edge[1,1]-1)){
                temp <- which(tree$edge[,2]==j)
                ancestor <- tree$edge[temp,1]
                times[j] <- times[ancestor]+tree$edge.length[temp]
        }
        maxt <- max(times)
        times <- -times+maxt
```

```
        out <- cbind(2011.115-times,ttype) # 2011.115 denotes the sampling
            time of the right-most leaf node of the phylogenetic tree
        out
}

# Function for solving the differential equations for E_1 and E_2

integrator2.statechange.s <- function(init,l,death,gamma,m,psi,times){
        ode <- function(times,y,p){
                lambda11 <- p[1]
                lambda12 <- p[2]
                lambda21 <- p[3]
                lambda22 <- p[4]
                death1 <- p[5]
                death2 <- p[6]
                gamma12 <- p[7]
                gamma21 <- p[8]
                m1 <- p[9]
                m2 <- p[10]

                yd1 <- m1-(lambda11+lambda12+gamma12+death1)*y[1]+lambda11*y
                    [1]*y[1]+lambda12*y[1]*y[2]+gamma12*y[2]
                yd2 <- m2-(lambda21+gamma21+lambda22+death2)*y[2]+lambda21*y
                    [1]*y[2]+lambda22*y[2]*y[2]+gamma21*y[1]
                list(c(yd1,yd2))
        }
        index <- max(1,floor(2011.115-times[2])-1983) # the number of the
            column of the matrix m corresponding to the year over which we
            integrate
        # 2011.115 is the sampling time of the right-most leaf node of the
            phylogenetic tree used, 1983 equals the sampling year of the left-
            most leaf node minus two
        out <- lsoda(init,times,ode,c(l,death,gamma,m[1,index],m[2,index]))
            [2,2:3]
        out
}

# Function for solving the differential equations for E_1, E_2, log(D_1) and
    log(D_2)

integrator.statechange.s <- function(init,l,death,gamma,m,psi,times){
        ode <- function(times,y,p){
                lambda11 <- p[1]
                lambda12 <- p[2]
                lambda21 <- p[3]
                lambda22 <- p[4]
                death1 <- p[5]
                death2 <- p[6]
                gamma12 <- p[7]
                gamma21 <- p[8]
                m1 <- p[9]
                m2 <- p[10]
```

```
            yd1 <- m1-(lambda11+lambda12+gamma12+death1)*y[1]+lambda11*y
                [1]*y[1]+lambda12*y[1]*y[2]+gamma12*y[2]
            yd2 <- m2-(lambda21+gamma21+lambda22+death2)*y[2]+lambda21*y
                [1]*y[2]+lambda22*y[2]*y[2]+gamma21*y[1]
            yd3 <- -(lambda11+lambda12+gamma12+death1)+2*lambda11*y[1]+
                lambda12*y[1]*exp(y[4]-y[3])+lambda12*y[2]+gamma12*exp(y
                [4]-y[3])
            yd4 <- -(lambda22+lambda21+gamma21+death2)+2*lambda22*y[2]+
                lambda21*y[2]*exp(y[3]-y[4])+lambda21*y[1]+gamma21*exp(y
                [3]-y[4])

            list(c(yd1,yd2,yd3,yd4))
        }
        index <- max(1,floor(2011.115-times[2])-1983) # the number of the
            column of the matrix m corresponding to the year over which we
            integrate
        # 2011.115 is the sampling time of the right-most leaf node of the
            phylogenetic tree used, 1983 equals the sampling year of the left-
            most leaf node minus two
        out <- lsoda(init,times,ode,c(l,death,gamma,m[1,index],m[2,index]))
            [2,2:5]
        out[3:4] <- exp(out[3:4]) # to get the values for D_1 and D_2 instead
            of log(D_1) and log(D_2)
    out
}
```

C.4 Coalescent model

```
# Function for calculating the likelihood of a phylogenetic tree
# As input we need a phylogenetic tree with defined states for the leaf nodes
# State 1 and 2 denote the acute and chronic state respectively
# In this code Y denotes I from the SIR model
# setwd("N:/Data hepatitis b/R")
# With sourceDirectory("Functies_Erik", modifiedOnly=FALSE); we can load all
    functions at once
# Also load the phylogenetic tree

likelihoodphylo <- function(par, tree){
        beta1 <- par[1]
        beta2 <- par[2]
        gamma1 <- par[3]
        gamma2 <- par[4]
        gamma12 <- par[5]
        mu <- 0.1
        N <- 24000

        R0 <- (beta1*(mu+gamma2)+beta2*gamma12)/((mu+gamma2)*(mu+gamma1+
            gamma12))

        # Times (number of years from present) and types (0=leaf node, 1=
            interne node) for all nodes (sorted)
        nodes <- sort(unique(c(tree$edge)))
        timesandtype <- get.times.type(tree, nodes)

        # Determine times (number of years from present and sorted) from all
            nodes (tsorted) but also only those from internal nodes (ssorted)
        tsorted <- sort(timesandtype[,1])
        ssorted <- sort(timesandtype[timesandtype[,2]==1,1])

        # We assume a constant population over time, using the SIR dynamics
        outSIR <- matrix(,nrow=length(tsorted),ncol=4,dimnames=list(NULL,c("
            time","S","Y1","Y2")))
        outSIR[,1] <- tsorted
        outSIR[,2] <- rep(N/R0,length(tsorted))
        outSIR[,3] <- rep(mu*N*(1-(1/R0))/(mu+gamma1+gamma12),length(tsorted))
        outSIR[,4] <- rep(mu*N*gamma12*(1-(1/R0))/((mu+gamma2)*(mu+gamma1+
            gamma12)),length(tsorted))

        ## If we do not assume a constant population over time we use the
            differential equations for the SIR model and some initial values (
            these are guesses)
        #Si <- 21000
        #Y1i <- 300
        #Y2i <- 3350
        #outSIR <- diff.SIR(Si, Y1i, Y2i, beta1, beta2, gamma12, gamma1,
            gamma2, mu, N, tsorted, R0)
```

```
# Stop if R_0<1 and give -1 for the negative log likelihood as a
    result
if (R0 < 1){
        return(c(-1,outSIR[1,2:4],par,R0))
}

# Stop if one of the parameters is below 0 and return -2 for the
    negative log likelihood
if (par[1]<0 || par[2]<0 || par[3]<0 || par[4]<0 || par[5]<0){
        return(c(-2,outSIR[1,2:4],par,R0))
}

# Determine the initial chance for all leaf nodes to be in the acute
    or chronic state
p.nodes <- matrix(,length(nodes),2)
for (i in 1:(tree$edge[1,1]-1)){
        if (tree$states[i]==1){
                p.nodes[i, ] <- c(1, 0)
        }
        else{
                p.nodes[i, ] <- c(0, 1)
        }
}

# Initial values for A1 and A2
A1i <- ifelse(tree$states[which.min(timesandtype[,1])]==1, 1, 0)
A2i <- ifelse(tree$states[which.min(timesandtype[,1])]==2, 1, 0)
# Initial vector for A1 and A2
A1 <- c(A1i)
A2 <- c(A2i)
# Initial matrix for the lambdas that need to be saved for calculating
     the likelihood
lambda.needed <- matrix(, length(nodes), length(nodes))
# Initial vector for Lambda
Lambda <- c(0)

## Initial matrix to calculate the probability of an individual to be
    in the acute or chronic state immediately after a coalescence
    event
#p.nodes.alpha <- matrix(,length(nodes),2)

## Initial vector for the probabilities of an individual to be in the
    acute and chronic state over a branch starting at the leaf node
    and ending at a coalescence event
#poverbranch <- c(1,0) # in this case we choose a leaf node (nr. 8)
    which is collected from an acutely infected individual

# Initial value for stopping the process (0=proceed, 1=stop)
gestopt <- 0

# Calculate the change in p.nodes over the tree, the coalescence rates
```

```
    and Lambda
for (j in 2:length(tsorted)){# tsorted[1]=0 and for this leaf node we
    already know p.nodes
      id.branches <- vector()
      S <- outSIR[j-1,2]
      Y1 <- outSIR[j-1,3]
      Y2 <- outSIR[j-1,4]

      for (i in 1:length(tree$edge[,1])){# We use round because
          timesandtype[,1] and tsorted don't have the same number of
           digits
            if (round(timesandtype[tree$edge[i,1],1], digits=6)>=
                round(tsorted[j], digits=6) & round(timesandtype[
                tree$edge[i,2],1], digits=6)<round(tsorted[j],
                digits=6)){# For the internal nodes
                  id.branches <- c(id.branches,tree$edge[i,2])
                  outP <- diff.P(i, j, beta1, beta2, gamma12,
                      gamma1, gamma2, N, tree, p.nodes, tsorted,
                       S, Y1, Y2, A1, A2) # solving differential
                       equations for the probabilities of being
                       in each state
                  # Stop if the probabilities become negative or
                       bigger than 1 (or NaN)
                  if (outP[2,2] < 0 || outP[2,3] < 0 || outP
                      [2,2] > 1 || outP[2,3] > 1 || outP[2,2]=="
                      NaN" || outP[2,3]=="NaN"){
                        gestopt <- 1
                        break
                  }
                  else{
                        p.nodes[tree$edge[i,2],1] <- outP[2,2]
                            # Replace the probabilities
                        p.nodes[tree$edge[i,2],2] <- outP[2,3]
                  }
            }
            else{# For the leaf nodes
                  if (timesandtype[tree$edge[i,2],2]==0 & round(
                      timesandtype[tree$edge[i,2],1], digits=6)
                      ==round(tsorted[j], digits=6)){
                        id.branches <- c(id.branches,tree$edge
                            [i,2])
                  } # We do not need to replace the
                      probabilities cause they are already known
                       for the leaf nodes
                  else{id.branches <- id.branches}
            }
      }

      ## Save the probability for the branch corresponding to leaf
          node 8
      #poverbranch <- rbind(poverbranch,p.nodes[8,])
```

```
        # Stop process if gestopt=1
        if (gestopt == 1){
                break
        }

        # Save the vectors for A1 and A2 over tsorted (sum of p.nodes
            [,1] and p.nodes[,2] for all lineages at time tsorted[j])
        nA1 <- 0
        nA2 <- 0
        for (i in 1:length(id.branches)){
                nA1 <- nA1 + p.nodes[id.branches[i],1]
                nA2 <- nA2 + p.nodes[id.branches[i],2]
        }
        A1 <- c(A1,nA1)
        A2 <- c(A2,nA2)

        # Calculate the coalescence rates for all lineages at time
            tsorted[j]
        lambda <- coal.all(N, beta1, beta2, id.branches, S, Y1, Y2, p.
            nodes)

        # Determine which node corresponds to time tsorted[j]
        node.alpha <- which(round(timesandtype[,1], digits=6)==round(
            tsorted[j], digits=6))

        # Save the coalescence rate for the real coalescence event at
            time tsorted[j]
        lambda.needed <- coal(tree, node.alpha, lambda, timesandtype,
            lambda.needed, id.branches)

        # Save the probabilities of being in each state after a
            coalescence event
        p.nodes[node.alpha,] <- p.coal(N, beta1, beta2, node.alpha,
            lambda.needed, S, Y1, Y2, p.nodes, timesandtype, tree, id.
            branches)
        ## Save the probabilities of being in each state immediately
            immediately after a coalescence event
        #p.nodes.alpha[node.alpha,] <- p.coal(N, beta1, beta2, node.
            alpha, lambda.needed, S, Y1, Y2, p.nodes, timesandtype,
            tree, id.branches)


        # Save Lambda(s) for tsorted[j]
        Lambda <- c(Lambda, sum(lambda)/2)
}

# Create a vector theta(s)
theta <- get.theta(ssorted, tsorted, Lambda)

# Calculate the likelihood and negative log likelihood of the
```

```
        phylogenetic tree
    Lik <- get.lik(tree, timesandtype, ssorted, lambda.needed, theta)
    minusloglik <- -log(Lik)

    if ((class(minusloglik)!="numeric") || (minusloglik=="NaN") || (
        minusloglik=="Inf") || gestopt == 1){
            minusloglik <- 10^100
    }

    return(c(minusloglik,outSIR[1,2:4],par,R0))
}

get.times.type <- function(tree, nodes){
    ttype <- (1:length(nodes))*0
    times <- ttype
    ttype[tree$edge[1,1]] <- 1
    for (j in (tree$edge[1,1]+1):length(nodes)){
            ttype[j] <- 1
            ancestor <- tree$edge[tree$edge[,2]==j,1]
            times[j] <- times[ancestor]+tree$edge.length[tree$edge[,2]==j]
    }
    for (j in 1:(tree$edge[1,1]-1)) {
            ancestor <- tree$edge[tree$edge[,2]==j,1]
            times[j] <- times[ancestor]+tree$edge.length[tree$edge[,2]==j]
    }
    maxt <- max(times)
    times <- -times+maxt
    timesandtype <- cbind(times,ttype)

    return(timesandtype)
}

diff.SIR <- function(Si, Y1i, Y2i, beta1, beta2, gamma12, gamma1, gamma2, mu,
    N, tsorted, R0){
    initSIR <- c(S=Si, Y1=Y1i, Y2=Y2i)
    timesSIR <- tsorted
    parmsSIR <- c(beta1=beta1, beta2=beta2, gamma12=gamma12, gamma1=gamma1
        , gamma2=gamma2, mu=mu, N=N, R0=R0)

    odeSIR <- function(timesSIR,x,parmsSIR){
    with(as.list(c(parmsSIR, x)),{
            dS <- (S/N)*(beta1*Y1+beta2*Y2) - mu*N + mu*S
            dY1 <- (-S/N)*(beta1*Y1+beta2*Y2) + gamma12*Y1 + gamma1*Y1 +
                mu*Y1
            dY2 <- -gamma12*Y1 + gamma2*Y2 + mu*Y2
            #dR <- -gamma1*Y1 - gamma2*Y2 + mu*R
            list(c(dS,dY1,dY2))#,dR)) # we leave out dR because the value
                for R isn't needed to calculate the likelihood
    })
    }
    outSIR <- lsoda(initSIR,timesSIR,odeSIR,parmsSIR)
    return(outSIR)
```

```
}

diff.P <- function(i, j, beta1, beta2, gamma12, gamma1, gamma2, N, tree, p.
    nodes, tsorted, S, Y1, Y2, A1, A2){
        initP <- c(P1 = p.nodes[tree$edge[i,2],1],P2 = p.nodes[tree$edge[i
            ,2],2])
        timesP <- c(tsorted[j-1],tsorted[j])
        parmsP <- c(beta1=beta1,beta2=beta2,gamma12=gamma12,gamma1=gamma1,
            gamma2=gamma2,N=N)

        odeP <- function(timesP,x,parmsP){
        with(as.list(c(parmsP,x)),{
                dP1 <- (P2/Y2)*gamma12*Y1 - (P1/Y1)*((Y2-A2[j-1])/Y2)*(beta2/N
                    )*S*Y2
                dP2 <- -1*(P2/Y2)*gamma12*Y1 + (P1/Y1)*((Y2-A2[j-1])/Y2)*(
                    beta2/N)*S*Y2
                list(c(dP1,dP2))
        })
        }
        outP <- lsoda(initP, timesP, odeP, parmsP)
        return(outP)
}

coal.all <- function(N, beta1, beta2, id.branches, S, Y1, Y2, p.nodes){
        lambda <- matrix(,length(id.branches),length(id.branches))
        for (i in 1:length(id.branches)){
                for (k in 1:length(id.branches)){
                        if (i==k){
                                lambda[i,k] <- 0
                        }
                        else{
                        lambda[i,k] <- 2*((S/N)*beta1*Y1/(Y1*Y1))*(p.nodes[id.
                            branches[i],1]*p.nodes[id.branches[k],1])+((S/N)*
                            beta2*Y2/(Y1*Y2))*(p.nodes[id.branches[i],2]*p.
                            nodes[id.branches[k],1] + p.nodes[id.branches[i
                            ],1]*p.nodes[id.branches[k],2])
                        }
                }
        }
        return(lambda)
}

coal <- function(tree, node.alpha, lambda, timesandtype, lambda.needed, id.
    branches){
        if (timesandtype[node.alpha,2]==1){
                node.i <- tree$edge[tree$edge[,1]==node.alpha,2][1]
                node.j <- tree$edge[tree$edge[,1]==node.alpha,2][2]
                lambda.needed[node.i,node.j] <- lambda[which(id.branches[]==
                    node.i), which(id.branches[]==node.j)]
                lambda.needed[node.j,node.i] <- lambda[which(id.branches[]==
                    node.i), which(id.branches[]==node.j)]
        }
```

```
        return(lambda.needed)
}

p.coal <- function(N, beta1, beta2, node.alpha, lambda.needed, S, Y1, Y2, p.
    nodes, timesandtype, tree, id.branches){
        if (timesandtype[node.alpha,2]==1){
                node.i <- tree$edge[tree$edge[,1]==node.alpha,2][1]
                node.j <- tree$edge[tree$edge[,1]==node.alpha,2][2]
                p.nodes[node.alpha,] <- (1/lambda.needed[node.i,node.j])*c(2*
                    ((S/N)*beta1*Y1/(Y1*Y1))*(p.nodes[node.i,1]*p.nodes[node.j
                    ,1]),((S/N)*beta2*Y2/(Y1*Y2))*(p.nodes[node.i,2]*p.nodes[
                    node.j,1]+p.nodes[node.i,1]*p.nodes[node.j,2]))
        }
        return(p.nodes[node.alpha,])
}

get.theta <- function(ssorted, tsorted, Lambda){
        int <- 0
        t.at.in.s <- which(round(tsorted[], digits=6)==round(ssorted[1],
            digits=6))
        for (j in 1:(t.at.in.s-1)){
                int <- int + (tsorted[j+1]-tsorted[j])*(1/2)*abs(Lambda[j+1]-
                    Lambda[j])
        }
        theta <- c(exp(-1*int))

        for (i in 2:length(ssorted)){
                t.at.s <- which(round(tsorted[], digits=6)==round(ssorted[i],
                    digits=6))
                t.at.pre.s <- which(round(tsorted[], digits=6)==round(ssorted[
                    i-1], digits=6))
                int <- 0
                for (j in t.at.pre.s:(t.at.s-1)){
                        int <- int + (tsorted[j+1]-tsorted[j])*(1/2)*abs(
                            Lambda[j+1]-Lambda[j])
                }
                theta <- c(theta, exp(-1*int))
        }
        return(theta)
}

get.lik <- function(tree, timesandtype, ssorted, lambda.needed, theta){
        Lik <- 1
        for (i in 1:length(ssorted)){
                alpha <- which(round(timesandtype[,1], digits=6)==round(
                    ssorted[i], digits=6))
                alphai <- tree$edge[tree$edge[,1]==alpha,2][1]
                alphaj <- tree$edge[tree$edge[,1]==alpha,2][2]
                Lik <- Lik*(lambda.needed[alphai,alphaj]*theta[i])
        }
        return(Lik)
}
```

C.5 Optimization methods

```
# Functions for the optimization method based on a pre-specified grid

# load the working directory and needed packages
setwd("N:/Data_hepatitis_b")
library(ape)
library(deSolve)
library(R.utils)

# load all functions
sourceDirectory("R/Functies_Tanja", modifiedOnly=FALSE);
sourceDirectory("R/Functies_Erik", modifiedOnly=FALSE);

# load the phylogenetic tree
load("R/tree2_without_A35")

# calculate the negative log likelihood over a certain grid for the two-type
    birth-death branching model with constant sampling fractions
loglikvecT <- c()
for (i in 1:6){
  for (j in 1:5){
        for (k in 6:10){
          for (l in 1:10){
                for (m in 1:5){
                  loglikvecT <- rbind(loglikvecT,bdtypes.stt.lik.statechange(
                      brpoint=get.times2(tree)[88],par=c(0.5*i,0.3*j,0.5*k
                      ,0.01*l,0.1*m),phylo=tree,fix=rbind(c(2,4,8),c(0,0,0),c
                      (1,1,1)),sampfrac=c(0.05,0.05),survival=1,posR=0,
                      unknownStates=FALSE,root=1))
                  save(list = ls(all = TRUE), file = "R/Functies_Tanja/
                      loglikvecT.RData")
                }
          }
        }
  }
}

# Find the minimum negative log likelihood for the two-type birth-death
    branching model with constant sampling fractions by walking over a path
padmatrixT <- c()
par <- c(0.5,0.3,3,0.01,0.5) # the parameter set corresponding to the minimum
    negative log likelihood of a grid

n <- 1
doorgaan <- 1

while(doorgaan==1){
        minloglik <- bdtypes.stt.lik.statechange(brpoint=get.times2(tree)[88],
            par=par,phylo=tree,fix=rbind(c(2,4,8),c(0,0,0),c(1,1,1)),sampfrac=
            c(0.05,0.05),survival=1,posR=0,unknownStates=FALSE,root=1)
```

```
# set up a matrix for the negative log likelihoods around par
parm <- matrix(,10,7)
parm[1,] <- bdtypes.stt.lik.statechange(brpoint=get.times2(tree)[88],
    par=par-c(0.01,0,0,0,0),phylo=tree,fix=rbind(c(2,4,8),c(0,0,0),c
    (1,1,1)),sampfrac=c(0.05,0.05),survival=1,posR=0,unknownStates=
    FALSE,root=1)
parm[2,] <- bdtypes.stt.lik.statechange(brpoint=get.times2(tree)[88],
    par=par+c(0.01,0,0,0,0),phylo=tree,fix=rbind(c(2,4,8),c(0,0,0),c
    (1,1,1)),sampfrac=c(0.05,0.05),survival=1,posR=0,unknownStates=
    FALSE,root=1)
parm[3,] <- bdtypes.stt.lik.statechange(brpoint=get.times2(tree)[88],
    par=par-c(0,0.01,0,0,0),phylo=tree,fix=rbind(c(2,4,8),c(0,0,0),c
    (1,1,1)),sampfrac=c(0.05,0.05),survival=1,posR=0,unknownStates=
    FALSE,root=1)
parm[4,] <- bdtypes.stt.lik.statechange(brpoint=get.times2(tree)[88],
    par=par+c(0,0.01,0,0,0),phylo=tree,fix=rbind(c(2,4,8),c(0,0,0),c
    (1,1,1)),sampfrac=c(0.05,0.05),survival=1,posR=0,unknownStates=
    FALSE,root=1)
parm[5,] <- bdtypes.stt.lik.statechange(brpoint=get.times2(tree)[88],
    par=par-c(0,0,0.01,0,0),phylo=tree,fix=rbind(c(2,4,8),c(0,0,0),c
    (1,1,1)),sampfrac=c(0.05,0.05),survival=1,posR=0,unknownStates=
    FALSE,root=1)
parm[6,] <- bdtypes.stt.lik.statechange(brpoint=get.times2(tree)[88],
    par=par+c(0,0,0.01,0,0),phylo=tree,fix=rbind(c(2,4,8),c(0,0,0),c
    (1,1,1)),sampfrac=c(0.05,0.05),survival=1,posR=0,unknownStates=
    FALSE,root=1)
parm[7,] <- bdtypes.stt.lik.statechange(brpoint=get.times2(tree)[88],
    par=par-c(0,0,0,0.01,0),phylo=tree,fix=rbind(c(2,4,8),c(0,0,0),c
    (1,1,1)),sampfrac=c(0.05,0.05),survival=1,posR=0,unknownStates=
    FALSE,root=1)
parm[8,] <- bdtypes.stt.lik.statechange(brpoint=get.times2(tree)[88],
    par=par+c(0,0,0,0.01,0),phylo=tree,fix=rbind(c(2,4,8),c(0,0,0),c
    (1,1,1)),sampfrac=c(0.05,0.05),survival=1,posR=0,unknownStates=
    FALSE,root=1)
parm[9,] <- bdtypes.stt.lik.statechange(brpoint=get.times2(tree)[88],
    par=par-c(0,0,0,0,0.01),phylo=tree,fix=rbind(c(2,4,8),c(0,0,0),c
    (1,1,1)),sampfrac=c(0.05,0.05),survival=1,posR=0,unknownStates=
    FALSE,root=1)
parm[10,] <- bdtypes.stt.lik.statechange(brpoint=get.times2(tree)[88],
    par=par+c(0,0,0,0,0.01),phylo=tree,fix=rbind(c(2,4,8),c(0,0,0),c
    (1,1,1)),sampfrac=c(0.05,0.05),survival=1,posR=0,unknownStates=
    FALSE,root=1)

a <- which.min(parm[,1])

if(parm[a,1] <= minloglik[1]){
        par <- parm[a,2:6]
}
else{
        doorgaan <- 0
```

```
        }

        padmatrixT <- rbind(padmatrixT, minloglik)

        # Stop when the path hops between the same parameter sets
        if (n>3){
                if (padmatrixT[n,2]==padmatrixT[n-2,2] & padmatrixT[n,3]==
                    padmatrixT[n-2,3] & padmatrixT[n,4]==padmatrixT[n-2,4] &
                    padmatrixT[n,5]==padmatrixT[n-2,5]
                        & padmatrixT[n,6]==padmatrixT[n-2,6]){
                doorgaan <- 0
                }
        }
        save(list = ls(all = TRUE), file = "R/Functies_Tanja/padmatrixT.RData"
            )
        n <- n+1
}


# calculate the negative log likelihood over a certain grid for the two-type
    birth-death branching model with variable sampling fractions
loglikvecT.s <- c()
for (i in 1:5){
  for (j in 1:4){
        for (k in 1:4){
          for (l in 1:4){
                for (m in 1:4){
                  for (o in 1:4){
                        for (p in 1:4){
                          loglikvecT.s <- rbind(loglikvecT.s,bdtypes.stt.lik.
                              statechange.s(brpoint=get.times2(tree)[88],par=c
                              (i,0.25*j,k,0.025*l,0.25*m,0.01*o,0.01*p),phylo=
                              tree,fix=rbind(c(2,4,8),c(0,0,0),c(1,1,1)),
                              survival=1,posR=0,unknownStates=FALSE,root=1))
                          save(list = ls(all = TRUE), file = "R/Functies_Tanja
                              /loglikvecT.s.RData")
                        }
                  }
                }
          }
        }
  }
}

# Find the minimum negative log likelihood for the two-type birth-death
    branching model with variable sampling fractions by walking over a path
padmatrixT.s <- c()
par <- c(1,0.25,2,0.1,1,0.01,0.01) # the parameter set corresponding to the
    minimum negative log likelihood of a grid

n <- 1
```

```
doorgaan <- 1 # 1=proceed, 0=stop

while(doorgaan==1){
        minloglik <- bdtypes.stt.lik.statechange.s(brpoint=get.times2(tree)
            [88],par=par,phylo=tree,fix=rbind(c(2,4,8),c(0,0,0),c(1,1,1)),
            survival=1,posR=0,unknownStates=FALSE,root=1)

        # set up a matrix for the negative log likelihoods around par
        parm <- matrix(,14,9)
        parm[1,] <- bdtypes.stt.lik.statechange.s(brpoint=get.times2(tree)
            [88],par=par-c(0.1,0,0,0,0,0,0,0),phylo=tree,fix=rbind(c(2,4,8),c
            (0,0,0),c(1,1,1)),survival=1,posR=0,unknownStates=FALSE,root=1)
        parm[2,] <- bdtypes.stt.lik.statechange.s(brpoint=get.times2(tree)
            [88],par=par+c(0.1,0,0,0,0,0,0,0),phylo=tree,fix=rbind(c(2,4,8),c
            (0,0,0),c(1,1,1)),survival=1,posR=0,unknownStates=FALSE,root=1)
        parm[3,] <- bdtypes.stt.lik.statechange.s(brpoint=get.times2(tree)
            [88],par=par-c(0,0.01,0,0,0,0,0,0),phylo=tree,fix=rbind(c(2,4,8),c
            (0,0,0),c(1,1,1)),     survival=1,posR=0,unknownStates=FALSE,root
            =1)
        parm[4,] <- bdtypes.stt.lik.statechange.s(brpoint=get.times2(tree)
            [88],par=par+c(0,0.01,0,0,0,0,0,0),phylo=tree,fix=rbind(c(2,4,8),c
            (0,0,0),c(1,1,1)),survival=1,posR=0,unknownStates=FALSE,root=1)
        parm[5,] <- bdtypes.stt.lik.statechange.s(brpoint=get.times2(tree)
            [88],par=par-c(0,0,0.1,0,0,0,0,0),phylo=tree,fix=rbind(c(2,4,8),c
            (0,0,0),c(1,1,1)),survival=1,posR=0,unknownStates=FALSE,root=1)
        parm[6,] <- bdtypes.stt.lik.statechange.s(brpoint=get.times2(tree)
            [88],par=par+c(0,0,0.1,0,0,0,0,0),phylo=tree,fix=rbind(c(2,4,8),c
            (0,0,0),c(1,1,1)),survival=1,posR=0,unknownStates=FALSE,root=1)
        parm[7,] <- bdtypes.stt.lik.statechange.s(brpoint=get.times2(tree)
            [88],par=par-c(0,0,0,0.001,0,0,0,0),phylo=tree,fix=rbind(c(2,4,8),c
            (0,0,0),c(1,1,1)),survival=1,posR=0,unknownStates=FALSE,root=1)
        parm[8,] <- bdtypes.stt.lik.statechange.s(brpoint=get.times2(tree)
            [88],par=par+c(0,0,0,0.001,0,0,0,0),phylo=tree,fix=rbind(c(2,4,8),c
            (0,0,0),c(1,1,1)),survival=1,posR=0,unknownStates=FALSE,root=1)
        parm[9,] <- bdtypes.stt.lik.statechange.s(brpoint=get.times2(tree)
            [88],par=par-c(0,0,0,0,0.01,0,0,0),phylo=tree,fix=rbind(c(2,4,8),c
            (0,0,0),c(1,1,1)),survival=1,posR=0,unknownStates=FALSE,root=1)
        parm[10,] <- bdtypes.stt.lik.statechange.s(brpoint=get.times2(tree)
            [88],par=par+c(0,0,0,0,0.01,0,0,0),phylo=tree,fix=rbind(c(2,4,8),c
            (0,0,0),c(1,1,1)),survival=1,posR=0,unknownStates=FALSE,root=1)
        parm[11,] <- bdtypes.stt.lik.statechange.s(brpoint=get.times2(tree)
            [88],par=par-c(0,0,0,0,0,0.001,0,0),phylo=tree,fix=rbind(c(2,4,8),c
            (0,0,0),c(1,1,1)),survival=1,posR=0,unknownStates=FALSE,root=1)
        parm[12,] <- bdtypes.stt.lik.statechange.s(brpoint=get.times2(tree)
            [88],par=par+c(0,0,0,0,0,0.001,0,0),phylo=tree,fix=rbind(c(2,4,8),c
            (0,0,0),c(1,1,1)),survival=1,posR=0,unknownStates=FALSE,root=1)
        parm[13,] <- bdtypes.stt.lik.statechange.s(brpoint=get.times2(tree)
            [88],par=par-c(0,0,0,0,0,0,0.001),phylo=tree,fix=rbind(c(2,4,8),c
            (0,0,0),c(1,1,1)),survival=1,posR=0,unknownStates=FALSE,root=1)
        parm[14,] <- bdtypes.stt.lik.statechange.s(brpoint=get.times2(tree)
            [88],par=par+c(0,0,0,0,0,0,0.001),phylo=tree,fix=rbind(c(2,4,8),c
```

```
        (0,0,0),c(1,1,1)),survival=1,posR=0,unknownStates=FALSE,root=1)

        a <- which.min(parm[,1])

        if(parm[a,1] <= minloglik[1]){
                par <- parm[a,2:8]
        }
        else{
        doorgaan <- 0
        }

        padmatrixT.s <- rbind(padmatrixT.s, minloglik)

        # Stop when the path hops between the same parameter sets
        if (n>3){
                if (padmatrixT.s[n,2]==padmatrixT.s[n-2,2] & padmatrixT.s[n
                    ,3]==padmatrixT.s[n-2,3] & padmatrixT.s[n,4]==padmatrixT.s
                    [n-2,4] & padmatrixT.s[n,5]==padmatrixT.s[n-2,5] &
                    padmatrixT.s[n,6]==padmatrixT.s[n-2,6] & padmatrixT.s[n
                    ,7]==padmatrixT.s[n-2,7] & padmatrixT.s[n,8]==padmatrixT.s
                    [n-2,8]){
                doorgaan <- 0
                }
        }
        save(list = ls(all = TRUE), file = "R/Functies_Tanja/padmatrixT.s.
            RData")
        n <- n+1
}

# calculate the negative log likelihood over a certain grid for the coalescent
    model
loglikvecE <- c()
for (i in 4:11){
  for (j in 1:10){
        for (k in 8:13){
          for (l in 1:4){
                for (m in 1:10){
                  loglikvecE <- rbind(loglikvecE,likelihoodphylo(c(0.5*i,0.1*j
                      ,0.4*k,0.04*l,0.1*m),tree))
                  save(list = ls(all = TRUE), file = "R/Functies_Erik/
                      loglikvecE.RData")
                }
          }
        }
  }
}

# Find the minimum negative log likelihood for the coalescent model  by
    walking over a path
padmatrixE <- c()
par <- c(5.5,0.1,4,0.12,0.1) # the parameter set corresponding to the minimum
```

```
    negative log likelihood of a grid

n <- 1
doorgaan <- 1 # 1=proceed, 0=stop

while(doorgaan==1){
        minloglik <- likelihoodphylo(par,tree)

        # set up a matrix for the negative log likelihoods around par
        parm <- matrix(,10,10)
        parm[1,] <- likelihoodphylo(par=par-c(0.01,0,0,0,0),tree)
        parm[2,] <- likelihoodphylo(par=par+c(0.01,0,0,0,0),tree)
        parm[3,] <- likelihoodphylo(par=par-c(0,0.001,0,0,0),tree)
        parm[4,] <- likelihoodphylo(par=par+c(0,0.001,0,0,0),tree)
        parm[5,] <- likelihoodphylo(par=par-c(0,0,0.01,0,0),tree)
        parm[6,] <- likelihoodphylo(par=par+c(0,0,0.01,0,0),tree)
        parm[7,] <- likelihoodphylo(par=par-c(0,0,0,0.01,0),tree)
        parm[8,] <- likelihoodphylo(par=par+c(0,0,0,0.01,0),tree)
        parm[9,] <- likelihoodphylo(par=par-c(0,0,0,0,0.001),tree)
        parm[10,] <- likelihoodphylo(par=par+c(0,0,0,0,0.001),tree)

        a <- which.min(parm[,1])
        # Remove the cases when the negative log likelihood equals -1 (if R_
            0<1) and -2 (if at least one of the parameters is negative)
        while(parm[a,1]==-2 || parm[a,1]==-1){
                parm <- parm[-a,]
                a <- which.min(parm[,1])
        }

        if(parm[a,1] <= minloglik[1]){
                par <- parm[a,5:9]}
        else{
                doorgaan <- 0
        }

        padmatrixE <- rbind(padmatrixE, minloglik)

        # Stop when the path hops between the same parameter sets
        if (n>3){
                if (padmatrixE[n,5]==padmatrixE[n-2,5] & padmatrixE[n,6]==
                    padmatrixE[n-2,6] & padmatrixE[n,7]==padmatrixE[n-2,7] &
                    padmatrixE[n,8]==padmatrixE[n-2,8] & padmatrixE[n,9]==
                    padmatrixE[n-2,9]){
                        doorgaan <- 0
                }
        }
        save(list = ls(all = TRUE), file = "R/Functies_Erik/padmatrixE.RData")
        n <- n+1

}

# Function for the MCMC method applied to the coalescent model
```

```
theta = c(5.5,0.1,0.1,4,0.12) # initial parameter set
NRuns = 10000
Savetheta <- matrix(nrow=NRuns, ncol=length(theta))
SaveL <- numeric()
AcceptedTheta = theta
AcceptedL = -likelihoodphylo(theta, tree)[1] # we use the log likelihood here
sd = c(0.7,0.025,0.025,0.55,0.015) # standard deviations used for each
    parameter
Accepted = 0

for(b in 1:NRuns){
        print("b")
        print(b)
        theta = abs(rnorm(length(theta),mean=AcceptedTheta,sd=sd))

        L = -likelihoodphylo(theta, tree)[1]
        print("L")
        print(L)

        RescaleL = max(L, AcceptedL)+5 # trick for avoiding the exponent in the
            next ratio to become zero
        AcceptYN = runif(1,min=0,max=1) <= exp(L-RescaleL)/exp(AcceptedL-
            RescaleL) & L!=-Inf # check whether the new candidate sampled from
            the proposal distribution is accepted
        if(AcceptYN){
                Accepted = Accepted+1
                AcceptedTheta = theta
                AcceptedL = L
        }

        # saving the parameter set for each iteration and the corresponding
            log likelihood
        Savetheta[b,] = AcceptedTheta
        SaveL[b] = AcceptedL
}
```

References

[1] R. Durrett. *Probability Models for DNA Sequence Evolution*. Springer, 2 edition, 2008.

[2] Virology Journal. The big picture book of viruses: Hepadnaviridae. http://www.virology.net/big_virology/BVDNAhepadna.html.

[3] The University of California Museum of Paleontology, Berkeley, and the Regents of the University of California. The causes of mutations, 2006. http://evolution.berkeley.edu/evosite/evo101/IIIC3Causes.shtl.

[4] Nature Education. Dna is constantly changing through the process of mutation, 2012. http://www.nature.com/scitable/topicpage/dna-is-constantly-changing-through-the-process-6524898.

[5] S. Ho. The molecular clock and estimating species divergence. *Nature Education*, 1(1), 2008.

[6] R.J.F. Ypma, W.M. van Ballegooijen, and J. Wallinga. Relating phylogenetic trees to transmission trees of infectious disease outbreaks. *Genetics*, 2013. [Epub ahead of print].

[7] National Institute for Public Health and the Environment (RIVM). Hepatitis b. http://www.rivm.nl/Onderwerpen/H/Hepatitis_B.

[8] World Health Organization. Global alert and response (gar): Hepatitis b, 2002. http://www.who.int/csr/disease/hepatitis/whocdscsrlyo20022/en/index3.html.

[9] World Health Organization. Fact sheet n^o 204: Hepatitis b, 2013. http://www.who.int/mediacentre/factsheets/fs204/en/.

[10] S.M. Bruisten, J.E. van Steenbergen, A.S. Pijl, H.G. Niesters, G.J. van Doornum, and R.A. Coutinho. Molecular epidemiology of hepatitis a virus in amsterdam, the netherlands. *Journal of Medical Virology*, 63(2):88–95, 2001.

[11] National Institute for Public Health and the Environment (RIVM). Nationaal kompas volksgezondheid: Hepatitis b, 2013. http://www.nationaalkompas.nl/gezondheid-en-ziekte/ziekten-en-aandoeningen/infectieziekten-en-parasitaire-ziekten/soa/hepatitis-b/.

[12] W.M. van Ballegooijen, R. van Houdt, S.M. Bruisten, H.J. Boot, R.A. Coutinho, and J. Wallinga. Molecular sequence data of hepatitis b virus and genetic diversity after vaccination. *American Journal of Epidemiology*, 170(12):1455–1463, 2009.

[13] M.J.W. van de Laar, R.J. Beuker, J. Rijlaarsdam, and Y.T.H.P. van Duynhoven. Hepatitis b. Technical Report 441500 011, National Institute for Public Health and the Environment (RIVM), 2000.

[14] R. van Houdt, S.M. Bruisten, A.G. Speksnijder, and M. Prins. Unexpectedly high proportion of drug users and men having sex with men who develop chronic hepatitis b infection. *Journal of Hepatology*, 57(3):529–533, 2012.

[15] L.C. Soetens, F.D.H. Koedijk, I.V.F. van den Broek, H.J. Vriend, E.L.M. Op de Coul, F. van Aar, A.I. van Sighem, I. Stirbu-Wagner, and B.H.B. van Benthem. Sexually transmitted infections, including hiv, in the netherlands in 2012. Technical Report report 150002003/2013, Centre for Infectious Disease Control - National Institute for Public Health and the Environment (RIVM), 2013.

[16] M. van Dam, I.M.S. van Ouwerkerk, J.H.T.C. van den Kerkhof, and A. Timen. Vaccinatieprogramma hepatitis b-risicogroepen: harddruggebruikers vanaf 2012 geen risicogroep meer, 2011. http://www.rivm.nl/Documenten_en_publicaties/Algemeen_Actueel/Uitgaven/Infectieziekten_Bulletin/Jaargang_22_2011/November_2011/Inhoud_IB_november_2011/Vaccinatieprogramma_hepatitis_B_risicogroepen_harddruggebruikers_vanaf_2012_geen_risicogroep_meer.

[17] National Institute for Public Health and the Environment (RIVM). Rijksvaccinatieprogramma hepatitis b. http://www.rivm.nl/Onderwerpen/R/Rijksvaccinatieprogramma/De_ziekten/Hepatitis_B.

[18] B. Golding, D. Morton, and W. Haerty. *Elementary Sequence Analysis*. Department of Biology, McMaster University, Hamilton, Ontario, 2013.

[19] M. Hasegawa, H. Kishino, and T. Yano. Dating of the human-ape splitting by a molecular clock of mitochondrial dna. *Journal of Molecular Evolution*, 22:160–174, 1985.

[20] A.J. Drummond, M.A. Suchard, D. Xie, and A. Rambaut. Bayesian phylogenetics with beauti and the beast 1.7. *Molecular Biology and Evolution*, 29(8):1969–1973, 2012.

[21] R Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria, 2013.

[22] A. Rambaut. Figtree: Tree figure drawing tool version 1.3.1, 2009. http://tree.bio.ed.ac.uk/software/figtree/.

[23] T. Stadler and S. Bonhoeffer. Uncovering epidemiological dynamics in heterogeneous host populations using phylogenetic methods. *Philosophical Transactions of the Royal Society B: Biological Sciences*, 368, 2013.

[24] E.M. Volz. Complex population dynamics and the coalescent under neutrality. *Genetics*, 190:187–201, 2012.

[25] T. Stadler. How can we improve accuracy of macroevolutionary rate estimates? *Systematic Biology*, 62(2):321–329, 2013.

[26] T. Stadler. Treepar in r - estimating diversification rates in phylogenies (version 2.3), 2011. http://cran.r-project.org/web/packages/TreePar/index.html.

[27] University of Colorado at Boulder. Mbw: Stochastic epidemic modeling, 2013. http://mathbio.colorado.edu/mediawiki/index.php/MBW:Stochastic_Epidemic_Modeling.

[28] W.O. Kermack and A.G. McKendrick. A contribution to the mathematical theory of epidemics. *Proceedings of the Royal Society Of London: Series A, Physical and Mathematical Sciences*, 115(772), 1927.

[29] J. Besag, P. Green, D. Higdon, and K. Mengersen. Bayesian computation and stochastic systems. *Statistical Science*, 10(1), 1995.

[30] H. Kato, E. Orito, R.G. Gish, N. Bzowej, F. Newsom, M. Sugachi, S. Suzuki, R. Ueda, Y. Miyakawa, and M. Mizokami. Hepatitis b e antigen in sera from individuals infected with hepatitis b virus of genotype g. *Hepatology*, 35(4), 2002.

[31] E. Paradis, J. Claude, and K. Strimmer. APE: analyses of phylogenetics and evolution in R language. *Bioinformatics*, 20:289–290, 2004.

[32] Rutgers Nisso Groep. Sexuele gezondheid in nederland. Technical report, Eburon, Delft.

[33] M. Xiridou, R. van Houdt, S. Hahné, R. Coutinho, J. van Steenbergen, and M. Kretzschmar. Hepatitis b vaccination of men who have sex with men in the netherlands: should we vaccinate more men, younger men or high-risk men? *Sexually transmitted infections*, 2013. [Epub ahead of print].

[34] J.H. Kao, P.J. Chen, M.Y. Lai, and Chen D.S. Acute exacerbations of chronic hepatitis b are rarely associated with superinfection of hepatitis b virus. *Journal of Hepatology*, 4 Pt 1(34):817–23, 2001.

[35] G.E. Leventhal, H.F. Günthard, S. Bonhoeffer, and T. Stadler. Using an epidemiological model for phylogenetic inference reveals density-dependence in hiv transmission. *Molecular Biology and Evolution*, 2013. [Epub ahead of print].