11-7-2006

Phylogenetic analysis of within-host serially-sampled viral data

Patricia Buendia
Florida International University

FLORIDA INTERNATIONAL UNIVERSITY

Miami, Florida

PHYLOGENETIC ANALYSIS OF WITHIN-HOST SERIALLY-SAMPLED VIRAL DATA

A dissertation submitted in partial fulfillment of the

requirements for the degree of

DOCTOR OF PHILOSOPHY

in

COMPUTER SCIENCE

by

Patricia Buendia

2006

To: Dean Vish Prasad
College of Engineering and Computing

This dissertation, written by Patricia Buendia, and entitled Phylogenetic Analysis of Within-Host Serially-Sampled Viral Data, having been approved in respect to style and intellectual content, is referred to you for judgment.

We have read this dissertation and recommend that it be approved.

Timothy Collins

Geoffrey Smith

Ana Pasztor

Giri Narasimhan, Major Professor

Date of Defense: November 7, 2006

The dissertation of Patricia Buendia is approved.

Dean Vish Prasad
College of Engineering and Computing

Dean George Walker
University Graduate School

Florida International University, 2006

DEDICATION

I dedicate this dissertation to my husband and my daughter. Without their patience, understanding, support, and most of all love, the completion of this work would not have been possible. I dedicate this also to all the wonderful people I've met in Miami whom I'm proud to call friends. You have become my extended family, lifting my spirits through difficult times and lending a hand with the care of my daughter Melanie while I was away or busy with work. Thank you ever so much, without your support, I would not have made it this far.

ACKNOWLEDGMENTS

I am very grateful for having an exceptional doctoral committee and wish to thank them for their continual support and encouragement. Special thanks go to my advisor Dr. Narasimhan, who has invested much time and effort in helping me produce this document and has supported me every step of the way with his guidance, scientific advice, and patience. I am extremely grateful for the assistance and advice I received from Dr. Smith throughout the dissertation process. I owe a special note of gratitude to Dr. Collins for assisting me with his biological and phylogenetic knowledge in the completion of projects related to this dissertation. I thank Dr. Ana Pasztor for providing valuable assistance in my preparation for the defense presentation.

ABSTRACT OF THE DISSERTATION

PHYLOGENETIC ANALYSIS OF WITHIN-HOST SERIALLY-SAMPLED VIRAL DATA

by

Patricia Buendia

Florida International University, 2006

Miami, Florida

Professor Giri Narasimhan, Major Professor

The primary goal of this dissertation is the study of patterns of viral evolution inferred from serially-sampled sequence data, i.e., sequence data obtained from strains isolated at consecutive time points from a single patient or host. RNA viral populations have an extremely high genetic variability, largely due to their astronomical population sizes within host systems, high replication rate, and short generation time. It is this aspect of their evolution that demands special attention and a different approach when studying the evolutionary relationships of serially-sampled sequence data. New methods that analyze serially-sampled data were developed shortly after a groundbreaking HIV-1 study of several patients from which viruses were isolated at recurring intervals over a period of 10 or more years. These methods assume a tree-like evolutionary model, while many RNA viruses have the capacity to exchange genetic material with one another using a process called recombination.

A genealogy involving recombination is best described by a network structure. A more general approach was implemented in a new computational tool, Sliding MinPD, one that is mindful of the sampling times of the input sequences and that reconstructs the

viral evolutionary relationships in the form of a network structure with implicit representations of recombination events. The underlying network organization reveals unique patterns of viral evolution and could help explain the emergence of disease-associated mutants and drug-resistant strains, with implications for patient prognosis and treatment strategies. In order to comprehensively test the developed methods and to carry out comparison studies with other methods, synthetic data sets are critical. Therefore, appropriate sequence generators were also developed to simulate the evolution of serially-sampled recombinant viruses, new and more through evaluation criteria for recombination detection methods were established, and three major comparison studies were performed. The newly developed tools were also applied to "real" HIV-1 sequence data and it was shown that the results represented within an evolutionary network structure can be interpreted in biologically meaningful ways.

TABLE OF CONTENTS

LIST OF TABLES

TABLE PAGE

LIST OF FIGURES

FIGURE PAGE

1 Introduction

RNA viral populations have an extremely high genetic variability, largely due to their astronomical population sizes within host systems, high replication rate, and short generation time. Thus the rapidly evolving RNA viruses are an ideal system for testing phylogenetic methods with the goal of studying the viral evolutionary process, evolutionary pressures in its environment, speciations, gene transfers, host infection and survival, and general host-virus interactions. In 1999 a groundbreaking study on HIV evolution was published by Shankarappa et al. (Shankarappa *et al.* 1999). It is, to date, the most comprehensive study of HIV-1 evolution *in vivo*. This study performed phylogenetic and statistical analysis on sequence data (viral DNA and plasma RNA) from viral samples that were collected at recurring time intervals from nine patients over a span of 8 to 12 years. The study provided insight into within-host viral evolution and helped find patterns that may explain the emergence of harmful mutants associated with disease progression. The data was made available at GenBank and was used in numerous studies each of which sought to examine a different aspect of viral sequence evolution (Guindon *et al.* 2004; Meintjes and Rodrigo 2005; Rodrigo *et al.* 2003; Shriner *et al.* 2004; Williamson 2003).

Shortly thereafter, two phylogenetic computational methods were developed specifically to analyze serially-sampled data. These include TipDate (Rambaut 2000), a Maximum Likelihood method, and sUPGMA, a distance-based method inspired by UPGMA and adapted to handle serial samples (Drummond and Rodrigo 2000). Additional new methods that analyze serially-sampled data were developed in the

following years, several of which were included in a comprehensive comparison study that evaluated phylogenetic tree inference and estimation of evolutionary relationships (Buendia *et al.* 2006a). Among the methods to analyze serially-sampled data, MinPD and Sliding MinPD, both of which are described in this dissertation, are the only methods that take genetic recombination into account (Buendia and Narasimhan 2004). Many RNA viruses have the capacity to exchange genetic material with one another in a process called recombination. Recombination is known to be an integral part of the HIV life cycle, with the recombination rate in HIV-1 being one of the highest among all known organisms (Rambaut *et al.* 2004). However, traditional phylogenetic tools perform poorly on data sets containing recombination (Posada and Crandall 2002).

The *primary goal* of this dissertation is *the study of patterns of viral evolution inferred from serially-sampled sequence data*, i.e., sequence data obtained from strains isolated at consecutive time points from a single patient or host. Investigating within-host viral evolution over an extended period of time provides a comprehensive and direct approach to the understanding of evolutionary mechanisms (such as mutational changes that occur during the replication of a genome over many generations), evolutionary pressures due to interactions with the host environment, and corresponding viral adaptations. From the viewpoint of clinical and biomedical research, investigating the intra-host viral evolution through serial sampling of the viral strains over a period of time may lead to a better understanding of the progression of a disease in that patient, or assist in the evaluation of drug therapies or vaccines for the disease.

The primary goal of this dissertation is described in terms of several subgoals and is summarized as follows:

Goal 1: Existing tools to study the phylogeny of serially-sampled data are often adaptations of traditional phylogenetic tools. However, these tools are limited in their abilities and are often either inefficient and/or inaccurate. Thus the first goal of this dissertation is *to develop efficient computational tools to find ancestor-descendant relationships between serially-sampled sequence data and to construct an evolutionary network for them.*

Goal 2: Existing phylogenetic tools perform poorly when recombination is involved (Posada and Crandall 2002). The second goal of this dissertation is *to develop efficient computational tools to automatically detect recombination in serially-sampled sequence data and display the recombination events within an evolutionary network structure.* It is important to ensure that the designed algorithms also estimate the statistical significance of the predicted relationships.

Goal 3: There are only a limited number of serially-sampled viral data sets available from public databases. In order to comprehensively test all the developed methods and to carry out the comparison studies, synthetic data sets are critical. Therefore, the next goal of this dissertation is *to build tools to generate realistic synthetic data sets,* i.e., appropriate sequence generators to simulate the evolution of serially-sampled recombinant viruses.

Goal 4: All the methods developed here needed to be evaluated and compared to existing and competing tools. Due to the complexity of recombination detection approaches,

special evaluation procedures were needed. Previous studies did not compute specificity and sensitivity values, and more importantly, did not assess the number of breakpoints and donors that were correctly identified (Posada and Crandall 2001a; Wiuf *et al.* 2001). Thus, the next goal of this dissertation is *to establish the criteria and mechanisms to perform comparison studies on recombination detection tools and phylogenetic tools for serially-sampled recombinant data.*

Goal 5: In the last six years, several methods have been devised to study serially-sampled data, many of which are variants of phylogenetic methods for contemporaneous taxa (Drummond and Rambaut 2003; Drummond and Rodrigo 2000; Ogishima *et al.* 2001; Rambaut 2000; Ren *et al.* 2001). *The performance of these methods needs to be compared with the performance of our newly implemented tools.*

Goal 6: The sixth goal is *to evaluate methods to study serially-sampled recombinant sequence data through extensive computer simulation studies*. Note that this evaluation procedure is different to the one in goal 5 as it evaluates recombination detection methods. The evaluation procedure will be performed using the evaluation mechanisms and criteria devised as part of Goal 4 of this dissertation.

Goal 7: An important goal of this dissertation is to apply the newly developed tools and techniques to "real" data sets and to interpret the results in biologically meaningful ways. This step is critical in order to investigate the biological and evolutionary aspects of real viral sequences sampled serially from the same patient. The tools developed in this dissertation were intended to help study the evolution of viral species and to respond to a myriad of questions that may shed light on its effect on the host. For example, it is

important to understand HIV infection vis-à-vis the progression of the AIDS disease in a patient. This raises questions such as: (1) Which initial viral strains did the harmful mutants (such as the mutants with the X4 phenotype) originate from? (2) Which of the initial strains became extinct and when did this happen? (3) Which strains showed positive selection, proliferating with descendants surviving over extended periods of time? (4) When did most recombinant strains appear and how do these times correlate to any treatments that may have been administered? Demonstrating the viability of our tools when applied to real viral data sets will help translate our results to the clinical arena.

In the course of this dissertation we will discuss the biological, theoretical and computational factors that influenced or contributed to the development of this project.

As a start, in **Chapter 2**, the biological background of our project, RNA viral evolution, will be explained in detail. This chapter also includes a survey of the different phylogenetic methods that study serially-sampled data and the different tools that can be used to detect recombination.

Next, in **Chapter 3**, we will present a new simulation program, Serial NetEvolve, devised to simulate the sampling of serially-sampled nucleotide sequences. Serial NetEvolve is a sequence generator for serial samples that was developed to better emulate recombining RNA virus sequence evolution. In **Chapter 4**, we will present MinPD, the first version of our method to study serially-sampled viral sequence data. Results of a comprehensive comparison study of MinPD with competing methods will be presented in **Chapter 5**. The evaluation techniques implemented in the comparison studies are also detailed in that same chapter. We make a digression in **Chapter 6** and

discuss a tool called RecIdenitfy, which will prove valuable in evaluating recombination detection tools for synthetically generated data sets. In **Chapter 7,** Sliding MinPD, our final and complete version of the method to address the main goal of this dissertation, will be presented. The motivation for the algorithm and the computational details behind the design will also be discussed. Sliding MinPD carries out an automated recombination detection process and replaces the visual output of standard recombination detection methods with an output of the list of the inferred recombinant and non-recombinant sequences along with statistical significance values associated with the inference. Sliding MinPD uses this output to construct a network that describes the evolutionary relationship of the data. The evaluation tool described in chapter 6 will be used to evaluate the results of Sliding MinPD and the results will be discussed in detail. **Chapter 8** presents the results of applying Sliding MinPD to a data set of HIV sequences sampled from several patients over a period of 10 years. We conclude with a summary of newly gained biological or computational insights in **Chapter 9**.

2 Background and Related Research

This chapter introduces the relevant background and presents previous work on areas related to this dissertation.

2.1 Evolution and Phylogeny

A DNA sequence provides the instructions for how each process is carried out within a cell of an organism. In evolutionary studies, DNA sequences or gene sequences are also recognized as an invaluable document of the past history of the organism. The goal of phylogenetic analysis is to recover and display the evolutionary information that is written into the DNA sequence. The use of phylogenetic tools is standard practice in the analysis of many aspects of viral pathogenesis and viral evolution. The evolutionary history of the primate lentiviruses SIV, HIV-1 and HIV-2 has been reconstructed in great detail by inferring phylogenetic trees (Rambaut et al. 2004). Viral evolutionary parameters, such as mutation rate are often estimated from previously inferred phylogenetic trees. The analysis of timing of events, demographic processes, or natural selection, is now possible through the analysis of phylogenetic relationships.

2.1.1 Phylogenetic Trees

Traditional phylogenetic methods were conceived for the purpose of inferring the history of a set of contemporaneous taxa through the reconstruction of a phylogenetic tree. In such trees the taxa being analyzed appear at the leaves of the tree. The ancestral sequences are usually unknown and are represented by the internal nodes of the tree. The branch lengths in a phylogenetic tree represent the number of substitutions per site. When

the molecular clock model of evolution is assumed, branch lengths do also represent "time elapsed" since the "most common recent ancestor," i.e., the internal node at which the branches originate. Figure 1 shows two phylogenetic trees, one in which the branch lengths represent genetic distances but not time (Fig 1a), and the other in which the molecular clock model is assumed (Fig 1b).

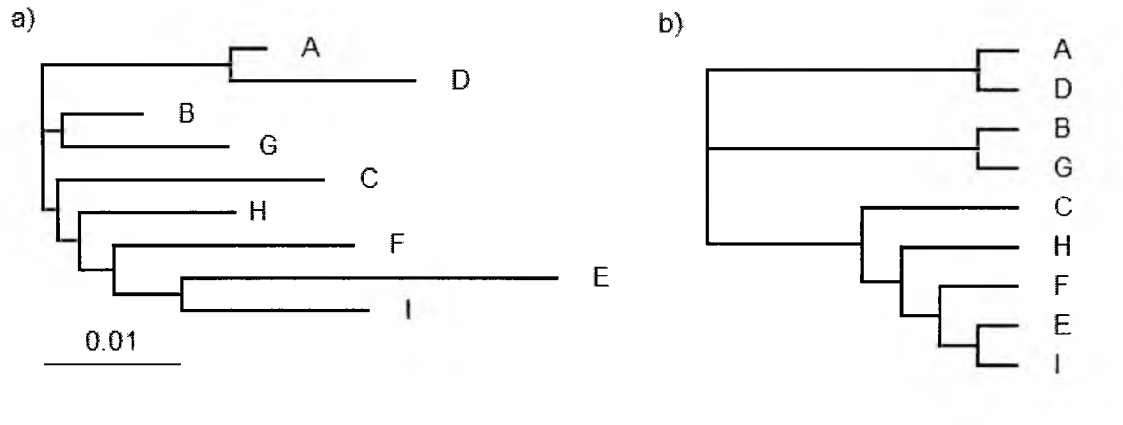


Figure 1 (a) Phylogenetic tree with different branch lengths (b) Phylogenetic tree assuming a clock model of evolution

The closer (in terms of substitutions per site and in terms of time, if the clock model is enforced) a sequence is to the direct ancestor, the shorter is the branch. A phylogenetic tree does not offer a direct way to represent ancestral sequences (sequences at the internal nodes) as such trees were designed to represent the genealogy of sequences sampled contemporaneously. However, a sequence that represents (and is therefore identical to) the direct ancestor at the internal node, might be represented by a leaf with a branch of length zero. While the branch lengths of a tree are often used in the estimation of evolutionary parameters, such as the time to the most recent ancestor (TMRA) or the mutation rate, the tree topology is the focal point in the study of cladistics. Each subtree

originating from a particular node that contains one or more taxa is called a *clade*. Questions related to determining which taxa cluster together in a subtree and the evolutionary meaning of a given clade is the topic of lengthy discussions and analyses among cladists and taxonomists.

An important focus of this dissertation is a variant of the phylogenetic reconstruction problems, which arises when some of the sequences at the internal nodes are available, such as with serially-sampled viral sequences. For this variant, traditional phylogenetic methods interpret all the inputs as contemporaneous taxa. We will discuss this issue later.

2.1.2 Models of Sequence Evolution

Many phylogenetic tree reconstruction methods assume a model of sequence evolution, i.e., a DNA substitution model, which makes certain assumptions about the nature of the molecular evolution process. For example, some models allow for variation in nucleotide frequencies; more complex models may allow different kinds of substitutions to occur with different probabilities, while others may take into account variations in the rate of substitution between sites (Page and Holmes 1998b). Figure 2 shows the possible nucleotide substitutions where *transitions* involve changes between A and G, or between C and T, while *transversions* involve changes between A and C, or between G and T.

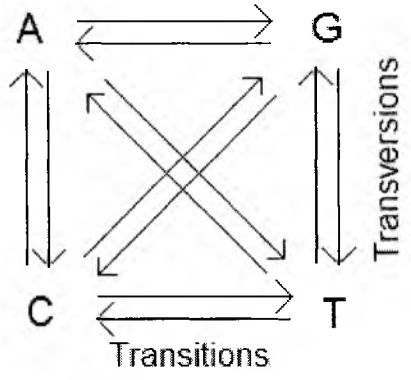


Figure 2. The possible substitutions among the four nucleotides.

The first model proposed was the Jukes Cantor model (JC69), which is perhaps the simplest model of sequence evolution. It assumes that all 4 bases have the same frequencies and that all substitutions are equally likely (Jukes and Cantor 1969). The Kimura 2 parameterized model (K2P) incorporates the observation that the rate of transitions (α) may differ from the rate of transversions (β) (Kimura 1980). Typically, transitions are more frequent than transversions, especially in mitochondrial DNA. The Hasegawa, Kishino, and Yano (HKY85) model allows transitions (α) and transversions (β) to occur at different rates and allows base frequencies (π_A, π_C, π_G, π_T) to vary as well (Hasegawa *et al.* 1985). The K2P and HKY85 transition probability matrices are given in Figure 3.

K2P

$$P_t = \begin{bmatrix} \cdot & \beta & \alpha & \beta \\ \beta & \cdot & \beta & \alpha \\ \alpha & \beta & \cdot & \beta \\ \beta & \alpha & \beta & \cdot \end{bmatrix}$$

HKY85

$$P_t = \begin{bmatrix} \cdot & \pi_C\beta & \pi_G\alpha & \pi_T\beta \\ \pi_A\beta & \cdot & \pi_G\beta & \pi_T\alpha \\ \pi_A\alpha & \pi_C\beta & \cdot & \pi_T\beta \\ \pi_A\beta & \pi_C\alpha & \pi_G\beta & \cdot \end{bmatrix}$$

Figure 3. K2P and HKY85 transition probability matrices

2.1.3 The Molecular Clock

One of the simplifying assumptions made in some of the early models of molecular evolution was that mutations occur at a uniform rate over time, giving rise to the "molecular clock hypothesis" (Page and Holmes 1998c). While this hypothesis has since been highly debated and widely disputed, it is an important assumption that enables the evolutionary history to be placed within a timeframe. Phylogenetic methods that assume a molecular clock infer trees in which the branch distances correlate to the evolutionary time elapsed (Figure 1b). Methods of phylogenetic reconstruction are more likely to be accurate if genes evolve at a constant rate. However, it has been shown in numerous studies that is not the case. A recent study showed that many viral data sets were better explained by different models of evolution and that the molecular clock hypothesis was rejected in most of the data sets analyzed (Posada and Crandall 2001b).

2.1.4 Phylogenetic Methods

The main phylogenetic methods that infer trees from aligned sequences can be divided into one of four categories: Parsimony, Maximum Likelihood, Bayesian, and distance-based methods.

Maximum Parsimony

Maximum parsimony is based on the principle of *Occam's Razor*, according to which the best tree is the one that describes the data set with the fewest changes in states of characters. The advantages of parsimony heuristic searches include computational simplicity and high efficiency, especially with large data sets. On the negative side, with small and highly divergent data sets, parsimony will greatly underestimate the

substitutions, especially for long branches in a tree (Hillis 1999). *DNAPARS* (Felsenstein 2004) carries out the traditional and popular maximum parsimony method using stepwise addition and local and global branch rearrangements. It is part of the PHYLIP software package (Version 3.6) and was included in a comparison study that we performed to assess how traditional methods compare to methods designed specifically to analyze serially-sampled data.

Consider the following 4 sequences (1) ATATT (2) ATCGT (3) GCAGT (4) GCCGT, and consider the first nucleotide of each sequence (i.e., first site). Figure 4 shows two rows of trees corresponding to two of the trees considered by the parsimony method for the first site. For each tree, the topology is shown on the leftmost column along with two possible reconstructions showing two (of the 16) possible labels on the two internal nodes. Every adjacent pair of nodes with distinct nucleotide labels corresponds to a mutational change. Thus, tree 1 has 2 reconstructions for site 1, of which the one with total change of 1 is the most parsimonious. Tree 2 has 2 reconstructions, both requiring 2 changes. Therefore for site 1, Tree 1 would be chosen as the most parsimonious.

The total number of evolutionary changes on a tree (the length of the tree) is the sum of the number of changes at each site (Page and Holmes 1998a):

$$L = \sum_{i=1}^{k} c_i$$

Tree 1 in Figure 4 has a total length of 1+1+2+1+0=5 changes. The other two trees have total lengths of 6 and 7 changes. Therefore, tree 1 is the most parsimonious.

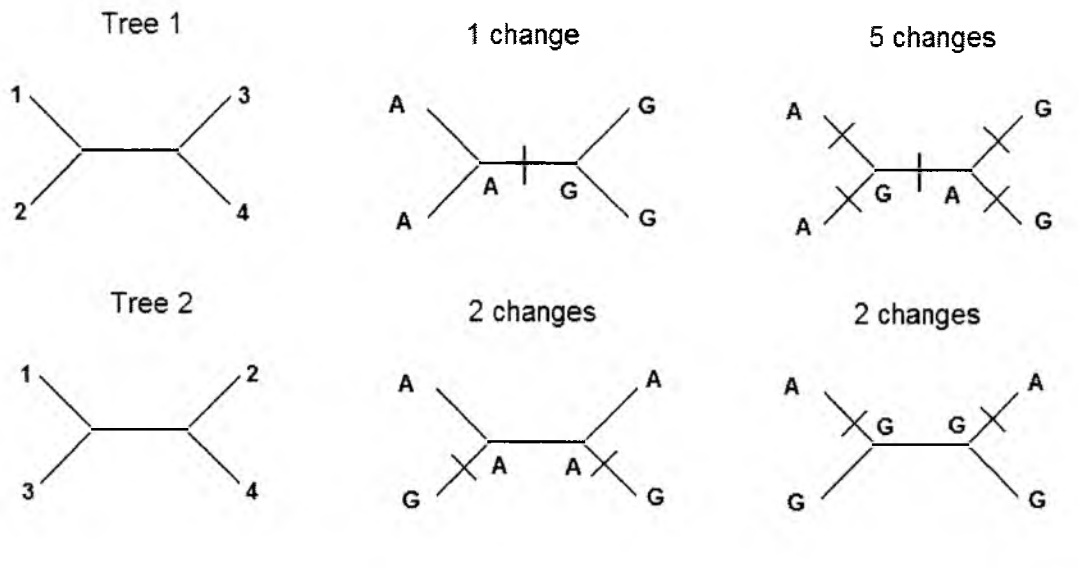


Figure 4. Two unrooted trees for four sequences 1-4 for the first site (column) of the sequences

Distance-based Methods

In distance-based analyses, a measure of dissimilarity between every pair of sequences is calculated and stored in a distance matrix. The distances can be a simple ratio of the number of site differences (i.e., p-distance or Hamming distance) to the length of the sequence, or a "corrected" estimate that uses a model of evolution to calculate the total number of sites that have changed between the two sequences (because some sites may have changed more than once). A corrected distance uses a model of sequence evolution such as the K2P or HKY85 models described above. Consequently a tree with respective branch lengths can be estimated from the distance matrix. In the K2P model for example, the corrected distance is given by

$$d = -\frac{1}{2}\ln(1-2P-Q) - \frac{1}{4}\ln(1-2Q),$$

where P and Q are the observed proportional differences between the two sequences due to transitions and transversions, whose numbers are P and Q, respectively (Page and Holmes 1998b).

Examples of methods that construct a tree from distance data are the Neighbor Joining (NJ) and the UPGMA methods. These methods are computationally fast but rely on a single point estimate of the tree rather than searching for an optimal solution (Hillis 1999). The *Neighbor-Joining* method of (Saitou and Nei 1987) is one of the most popular. It constructs a tree by successive clustering of lineages, setting branch lengths as the lineages join. The tree is not rearranged thereafter. The program is very fast which makes it particularly effective for large studies or for bootstrap or jackknife resampling studies, which require runs on multiple data sets.

Maximum Likelihood

For a given (explicit) model of evolution, a likelihood score is calculated for each possible tree as the probability of observing the data if that tree is the correct phylogenetic tree. The model describes the probability of mutational changes. Mutations at distinct sites are assumed to be independent events and the likelihood of a given tree is the product of the likelihoods for all the characters (sites), as shown below:

$$L = \prod_{i=1}^{k} L_i .$$

Since the likelihood values are very small numbers, the logarithm of the likelihood is normally used (with the log likelihood of the tree being the sum of the log likelihoods for the characters) (Page and Holmes 1998a), as shown here:

$$\ln L = \sum_{i=1}^{k} L_i \ .$$

The maximum likelihood solution is the tree with the highest log-likelihood score. The likelihood method outperforms parsimony and distance methods when it comes down to accurately inferring a phylogeny. One disadvantage is that the outcome hinges on the choice of a model of evolution, which has to be proposed *a priori*. The biggest disadvantage is its computational complexity, which until recently has impeded its use with complex models or large data sets (Hillis 1999). Recent efficient implementations have improved its ability to handle larger data sets. *FastDNAml* is an efficient implementation of the maximum likelihood (ML) method and was chosen as part of a comparison study that included methods for serial samples to examine how traditional phylogenetic methods perform with non-contemporaneous data (Felsenstein 1981; Olsen *et al.* 1994).

Bayesian Methods

Bayesian inference of phylogeny is based upon a quantity called the posterior probability distribution of trees, which is the probability of a tree conditioned on the observations. The conditioning is accomplished using Bayes Theorem. The posterior probability distribution of trees is impossible to calculate analytically; instead, a simulation technique called Markov Chain Monte Carlo (or MCMC) is used to approximate the posterior probabilities of trees. In each new step a new tree is either accepted or rejected based on a value that is calculated from the likelihood, the prior used, and the proposal ratio. A model of evolution may be incorporated into a Bayesian

method with a prior distribution on the model parameters. The proportion of the time any single tree is visited during the course of the chain is a valid approximation of the posterior probability (Huelsenbeck and Ronquist 2001).

2.1.5 Statistical Tests

Non-Parametric Bootstrapping of Phylogeny is a statistical method that is applied when distributions (such as that of phylogenetic trees) are difficult to calculate exactly. The distributions can be estimated by the repeated creation and analysis of artificial datasets. In the non-parametric bootstrap method, these datasets are generated by resampling the original data, whereas in the parametric bootstrap, the data are simulated according to the hypothesis being tested (Page and Holmes 1998a). Resampling methods are popular in the study of phylogenetics and are used to empirically determine the variability of the estimation. Non-parametric bootstrapping is a popular way of assessing the robustness of a tree. The bootstrap method involves resampling of sites/columns (with replacement) from a set of columns in a multiple sequence alignment. Each resampling is called a pseudoreplicate. Example: A run of a bootstrap procedure on two sequences

(1) AGCTG

(2) AGTAC

could randomly pick site 1, then 1 again, then 4, then 2, and 4 again. The resampled sequences would have sites 11244 of the original sequences. When maintaining the order of columns the pseudoreplicates would be composed of the following nucleotides:

(1) AAGTT

(2) AAGAA

The pseudoreplicates are used to reconstruct new phylogenies. The inferred bootstrap replicate trees are then analyzed by computing the proportion of the replicates in which each branch from the inferred tree occurs, thus providing a robustness measure for every edge of the inferred tree.

Likelihood Ratio Tests Comparisons of two competing models are also possible, using Likelihood Ratio Tests (LRT), a powerful statistical test in which competing hypotheses (H0 and H1) are compared using a statistic based on the ratio of the maximum likelihoods under each hypothesis;

$$\Delta = \log L_1 - \log L_0 .$$

Results can be expressed in terms of P-values, the probability of the statistic being at least as extreme as observed when H is true: low P-values (e.g., <0.05) suggest rejection of H0 in favor of H1 (Page and Holmes 1998a).

2.2 RNA Viruses

A virus cannot grow or reproduce on its own and in order to replicate it must infect the cells of a living organism, the host. RNA viruses are notorious for their tremendous evolutionary potential. Compared to human genes, RNA viral genes acquire multiple mutations in a relatively short time. Their great adaptability is thought to be a consequence of a high per generation genomic mutation rate caused by the error-prone RNA polymerases and by the very large population sizes often reached during infections (Drake and Holland 1999). Notable human pathogenic RNA viruses include SARS, Influenza, HIV-1 and Hepatitis C viruses.

RNA viruses living within a host have genetically diverse populations and are referred to as *quasispecies*. Fast evolving RNA viruses provide a quick and efficient way to explore the applicability of the classical theories of population genetics and evolution and the development of new theoretical frameworks. The evolution of human viruses, such as HIV-1 and Hepatitis C virus, are also influenced by host conditions, such as a weak immune system or the intake of antiretroviral therapy. Understanding these interactions may facilitate the prediction and prevention of emerging new viral pathogens and resistance-breaking variants.

2.2.1 Differences between RNA and DNA viruses

The structure of the RNA viruses is basically the same as that of DNA viruses - a core of genetic material, usually contained within a protective capsid of protein, and in many cases, a lipid envelope as well. The life cycle of all viruses follow the same general strategy to ensure their survival: An infectious cycle involves attachment to the host cell, penetration, genome translation, genome replication, creation and assembly of particles containing the genome, emergence from the cell (Flint *et al.* 2000a).

The major differences between DNA and RNA viruses arise from the fact that the RNA viruses genetic information is stored, as their name suggests, in RNA, not DNA. This has important consequences in the life cycle of the virus - and gives it the potential to outwit the immune system. There are two other major differences between DNA and RNA viruses that explain the high diversity of RNA viral populations and the properties that allow RNA viruses to escape the immune system (Flint *et al.* 2000b):

1. Error-prone replication process: In contrast to DNA-directed DNA polymerases, RNA-dependant RNA polymerases, which are responsible for RNA genome replication, do not possess proof-reading capabilities to correct replication mistakes. The result is that the error frequencies in RNA replication can be as high as one error per 10^3 to 10^4 nucleotides polymerized, whereas the errors in DNA replication are about a 1000-fold lower. These mutations make it more difficult for any organism to develop any kind of lasting immunity to the virus.

2. RNA Recombination: In this process nucleotide sequences are exchanged among different genomic RNA molecules. Recombination is also prevalent in DNA viruses, RNA viruses, however, accomplish it in at least two different ways. First, there is segment reassortment, which is the exchange of the entire RNA molecules between genetically related viruses with segmented genomes. Influenza virus is an example of a virus with eight genome segments that are packaged into each virus particle. Second, in copy-choice replication, the viral polymerase switches between different RNA templates during transcription giving rise to mosaic genomes. Recombination is a feature of many RNA viruses, it is an important mechanism for producing new genomes with selective growth advantages. Some viruses with very high recombination rates include the polio virus, influenza virus and the HIV virus.

2.2.2 Examples of RNA Viruses

Hepatitis C Virus

Hepatitis C virus (HCV), the unique Hepacivirus of the Flaviviridae family, is an enveloped, positive single-stranded RNA virus. HCV infects humans, but the virus has been transmitted to chimpanzees in controlled experiments (World Health Organization 2002). Acute infection with HCV very often progresses to chronic infection, which may eventually lead to cirrhosis and hepatocellular carcinoma (HCC).

One of the most important features of HCV is that its genome exhibits significant genetic variability associated primarily with the error-prone nature of its RNA-dependent RNA polymerase. HCV probably follows the replication strategy of other positive-strand RNA viruses. The virus enters the cell and is uncoated in the cytoplasm. The viral RNA exerts two kinds of functions in the cytoplasm: initially it operates as a messenger and initiates translation at a ribosome, then, after viral RNA polymerase has been produced, it becames a template for replication (World Health Organization 2002).

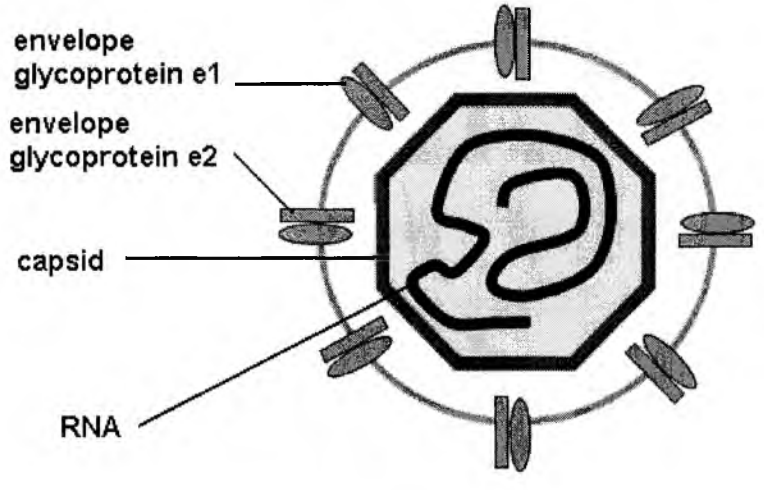


Figure 5. Structure of the Hepatitis Virus C. (http://www.edc.gsph.pitt.edu/virahepc)

A study was recently conducted to determine if immunologically-driven sequence variation occurs with persistence of Hepatitis C virus (HCV) infection. The study used serially-sampled data and consisted in analyzing HCV sequence evolution and the $CD8^+$ T-cell responses in subjects who were followed prospectively from before infection, at initial viremia to beyond the first year at multiple time points in acute HCV infection (Cox *et al.* 2005). The findings of the study reveal two distinct mechanisms of sequence evolution involved in HCV persistence: viral escape from CD8 T-cell responses and optimization of replicative capacity. Recombination was not recognized as a likely mechanism for creating diversity of HCV until a recent finding of a spontaneous HCV recombinant (RF1_2k/1b) in St Petersburg, which was further analyzed to reveal the possible mechanism of recombination in the virus (Kalinina *et al.* 2004). The study suggests that RF1_2k/1b has emerged by homologous recombination during minus-strand synthesis via template switching because of constraints imposed by the HS1 hairpin of the 3'-parental genome. However, the consensus at the moment is that homologous recombination does not play an extensive role in HCV evolution. HCV and HIV have similar and high rates of mutation (~$2x10^{-3}$ errors per site per year) (Allain *et al.* 2000; Rambaut et al. 2004), but HCV has a production rate of 10^{12} new virions a day, which exceeds comparable estimates of the production of HIV by more than an order of magnitude (Cox et al. 2005).

HIV-1 Virus

HIV belongs to a special class of viruses called retroviruses. Within this class, HIV is placed in the subgroup of lentiviruses. HIV-1 was the first strain of HIV that was discovered by Luc Montagnier at the Pasteur Institute in Paris in 1983. HIV-2 was discovered in 1986 as a distinct strain prevalent in certain regions of West Africa. The two major groups of HIV-1 isolates are classified as: group M, which includes most HIV-1 isolates and group O, which represents rare "outliers" (Flint *et al.* 2000c). Figure 6 shows the structure of an HIV-1 virus.

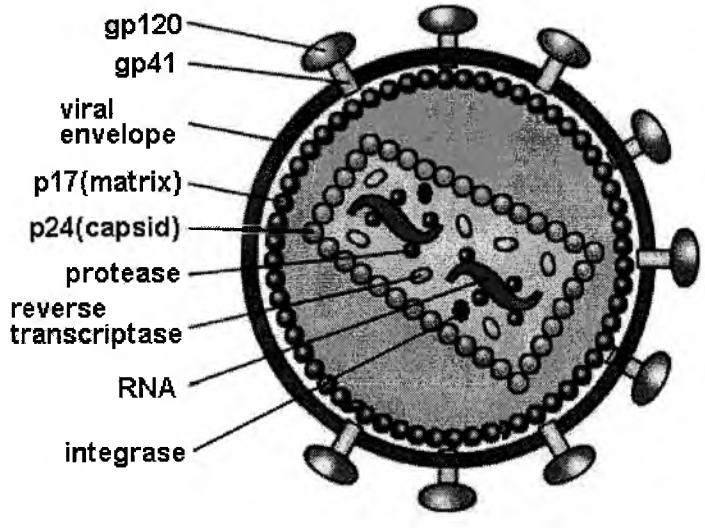


Figure 6. Structure of the HIV-1 virus (http://www.aids-india.org/hivbasics2.htm)

HIV has two RNA genomes in its capsid. The first step in the life cycle of HIV consists of the virus attaching itself to a cell by binding of the surface protein gp120 and the transmembrane protein gp41 to a specific receptor on the surface of the host's cell. The viral contents of the capsid are deposited into the cytoplasm. The viral RNA genome is reverse-transcribed into double stranded DNA by the reverse transcriptase. In the next step the DNA and integrase enter the nucleus of the infected cell and the DNA is

integrated into the host genome. This step is characteristic to all (and only) the retroviruses, other viruses do not integrate into the host genome. Integrated viral DNA is called a provirus. Transcription of the provirus produces mRNA some of which are later translated into viral proteins (p24, protease, integrase, gp120, gp41, and other virion components) in the cytoplasm and others which became viral progeny genomes. Virion components assemble at budding sites on the plasma membrane. Finally, the nascent virions bud from the surface of the cell (Flint *et al.* 2000d).

Acquired immunodeficiency syndrom (AIDS) is the name given to the disease caused by the HIV virus. AIDS has become one of the major causes of death and the number of new infections is rising rapidly in parts of sub-Saharan Africa. HIV has become the most intensely studied infectious agent (Flint et al. 2000c).

2.2.3 HIV Within-Host Evolution

One of the earliest and most striking observations made about HIV is the extensive genetic variation that the virus has within individual hosts, particularly in the hypervariable regions of the envelope (env) gene that code for the gp120 and gp41 proteins. This variation makes HIV one of the fastest evolving of all organisms. HIV has a high mutation rate of 0.2 errors per genome per cycle and a viral generation time of 2.5 days (Rambaut et al. 2004). When considering within-host genetic variation, one may want to ask how much of this diversity is shaped by immune selection, and what is the relationship between genetic diversity and clinical outcomes? One answer is that HIV successively fixes mutations that allow it to evade immune responses, especially in the *env* gene. The remarkable strength of immune selection was revealed in a recent analysis

of evolutionary dynamics in 50 HIV-1 patients (Williamson 2003). Most fixed *env* amino-acid changes in these patients confer a selective advantage, with an average of one adaptive fixation event every ~2.5 months. Under these criteria, HIV shows stronger positive selection than any other organism studied so far.

The analysis of phylogenetic trees from serially-sampled data has revealed that within-host evolution and inter-host evolution are radically different processes, with positive selection dominating in the former, but not in the latter. Intra-host HIV phylogenies have a strong temporal structure, reflecting the successive fixation of advantageous mutations and the extinction of unfavorable lineages. In contrast, trees that track viral evolution among hosts show little evidence of continual positive selection (Rambaut et al. 2004).

2.2.4 Emergence of HIV X4 Mutations and Drug Resistant Mutations

The emergence of X4 viruses has been associated with disease progression to AIDS(Shankarappa et al. 1999). X4 strains have a syncytium-inducing (SI) phenotype. Syncytium is a large cell-like structure formed by the joining together of two or more cells forming a single cell with many nuclei. The majority of these SI viruses are capable of using both CCR5 and CXCR4 as co-receptors for viral entry. A recent study showed that X4 variants emerged around the time of the early to intermediate phase transition during infection and then achieved peak representation and began a decline around the time of failure of T-Cell homeostasis and decline of $CD4^+$ T-cells (Shankarappa et al. 1999). The results of a study of subtype C virus isolates from 28 patients from Harare, Zimbabwe, 20 of whom were receiving antiretroviral treatment, suggest that CXCR4-

tropic viruses are present within the quasispecies of patients infected with subtype C virus and that antiretroviral treatment may create an environment for the emergence of CXCR4 tropism (Johnston *et al.* 2004). Indeed, one of the major factors limiting the efficacy of virus-specific therapies against many retroviruses and RNA viruses is the development of resistance to antiviral drugs. Drug resistant mutations have been observed in Hepatitis B and C virus and are the primary cause of treatment failure of the first approved hepatitis B virus-specific drug (Lin *et al.* 2004). The emergence of drug-resistant HIV strains constitutes the largest setback in the treatment of AIDS. Highly active antiretroviral therapy (HAART), involving combinations of drugs that act against different aspects of the viral life cycle, was believed to rid the body of virus altogether. But two key aspects of HIV biology allowed the virus to persist long after the initiation of therapy: viral reservoirs and evolution. These reservoirs can serve to replenish the main pool of replicating virus and are now known from a variety of cell types housed in various tissues throughout the body (Rambaut et al. 2004). Using phylogenetic analysis and a criteria that categorizes reservoir virus and contemporary virus as serially-sampled data, a study concluded that viral divergence from a calculated most recent common ancestor is a strong predictor of viral reservoirs (Nickle *et al.* 2003).

2.2.5 Recombination in Viruses

Genetic recombination is an integral part of an RNA virus lifecycle, in which nucleotide sequences are exchanged among different RNA molecules. In HIV, recombination occurs between two co-encapsidated RNA genomes during reverse transcription. During DNA synthesis, the reverse transcriptase, which is prone to errors,

may switch from one strand to the other, either during the first (-) strand DNA synthesis, or during the second (+) strand DNA synthesis, as part of the mechanism of strand displacement assimilation (Flint *et al.* 2000e). The recombination rate of HIV is one of the highest of all organisms, with an estimated three recombination events occurring per genome per replication cycle (Rambaut et al. 2004). Within individual hosts, recombination is more difficult to detect, but it interacts with selection and drift to produce complex population dynamics, and perhaps provides an efficient mechanism for the virus to escape from the accumulation of deleterious mutations or to jump between adaptive peaks. A recent study of serial HIV samples illustrates the unusual and important patterns of viral adaptation through recombination that can accelerate the progression to AIDS. Although only a single patient was included in the study, it was shown that the mechanism of recombination between divergent viruses, with its ability to create chimeras with increased fitness, correlated well with the disease progression of the patient (Liu et al. 2002). Recombination allows the virus to accumulate and exchange drug-resistant mutations in a nonlinear fashion, leading also to rapid evolution of drug-resistant mutants, even between different viral reservoirs. Recombination also complicates the analysis of within-host sequence variation and virus-host interaction in HIV confounding the distinction between total genetic variation and selectively advantageous changes. In particular, many evolutionary inferences about HIV are made after the reconstruction of phylogenies, which can be greatly affected by recombination. Because recombination is so frequent, it cannot be factored out by simply identifying recombinants and excluding those from the analysis (as was often the practice in early analysis (Korber *et al.* 2000)). HIV should, therefore, be studied with methods that are robust to the occurrence of

recombination events, or that explicitly take recombination into account (Rambaut et al. 2004).

2.3 Recombination Detection

Phylogenetic methods that do not account for recombination can make incorrect inferences in the presence of recombination (Posada and Crandall 2002; Schierup and Hein 2000a; Schierup and Hein 2000b). It is therefore critical to detect recombination as the assumption of a single underlying phylogeny can be severely biased by the presence of recombination. In recent years the development of new tools to model and test for recombination have led to several studies that compare the different methods and assess their accuracy (Posada and Crandall 2001a; Wiuf et al. 2001). Some methods only determine the presence or absence of recombination, without trying to infer recombination breakpoints (Worobey 2001). Other more sophisticated methods attempt to detect recombinant signals and parental/donor sequences for one or more "query" sequences (Lole *et al.* 1999; Martin *et al.* 2005b; Strimmer *et al.* 2003). Here, we will present and discuss some of the methods that detect breakpoints and parental/donor sequences.

2.3.1 Effect of Recombination on Traditional Phylogenetic Analyses

Conventional phylogenetic programs are constrained to produce simple branching trees and can lead to gross misinterpretations if the data set contained recombinants. One main assumption of most phylogenetic methods is that there is only one phylogeny underlying the evolution of the taxa under study. Recombination violates this assumption

by generating mosaic genes, where different regions have different phylogenetic histories. By ignoring the presence of recombination, phylogenetic analyses may be severely compromised. Ignoring recombination can have dramatic effects on the topology of the resulting phylogenetic tree and on phylogeny-based analyses (such as the estimation of the time to the most recent common ancestor). Ignoring recombination leads to larger genetic distances and an overestimation of the inferred time to the most recent common ancestor (MRCA), whereas variation in rates among nucleotide sites (rate heterogeneity) may be inferred (by maximum-likelihood methods) to be much higher than it actually is. The molecular clock hypothesis may also be falsely rejected when recombination signals in the data are not taken into account (Posada and Crandall 2002; Schierup and Hein 2000a; Schierup and Hein 2000b).

When trying to understand the interaction between host and virus, one has to search for evidence that rates of non-synonymous (dN) substitution vary according to disease status. Here again, recombination complicates the analysis, as it might produce false-positive evidence for natural selection in measures of dN/dS (Rambaut et al. 2004).

The development of new methods that take into account recombination is necessary to make reliable inferences on many aspects of evolution. A simple way to start might be through the use of network approaches for phylogenetic inference, in which individual sequences are allowed to have many ancestors, and which provide a good alternative to traditional trees (Rambaut et al. 2004).

2.3.2 Recombination Detection Methods

Modeling and detection of recombination is receiving increased attention with an ever-growing number of recombination detection tools being published in recent years (Fan and Robertson 2006). A great majority of these tools use pairwise sequence comparisons, such as the online tool RIP (Siepel and Korber 1995). Another large group of methods uses phylogenetic trees, like the popular bootscanning method (Lole et al. 1999; Salminen *et al.* 1995). The goal of these methods is to identify the putative recombinant ancestor sequences for a query sequence from a set of aligned sequences. Two examples of recombination detection methods will be presented next.

Recombination Identification Program (RIP)

A sliding window is moved along the aligned input sequences, and at every position the query sequence is compared to each of the background representatives, similarity being quantified as the percentage of identical base pairs. After the window has traversed the alignment from left to right, the program displays the output revealing which background representative the query sequence most resembles at each position (Siepel and Korber 1995). Figure 7 shows the output of the detection process with the online tool RIP. The query sequence is 006.R and is compared to 8 background/reference sequences. By looking at the highest matching percentage to the left and right of a putative breakpoint, one may surmise that of the 8 sequences, 004.15 is the left recombinant donor and 005.93 the right recombinant donor. An alternative output that displays an alignment of the query sequence to the background sequences (data not shown here) indicates that the breakpoint site is around position 268. Since the analysis

was performed on a simulated data set, the correct answer is known. The donors and breakpoint (269 is the true breakpoint site) would be correctly identified with the above analysis.

In a second step RIP identifies the so-called "best matches," if they are significant according to a statistical test that it applies.

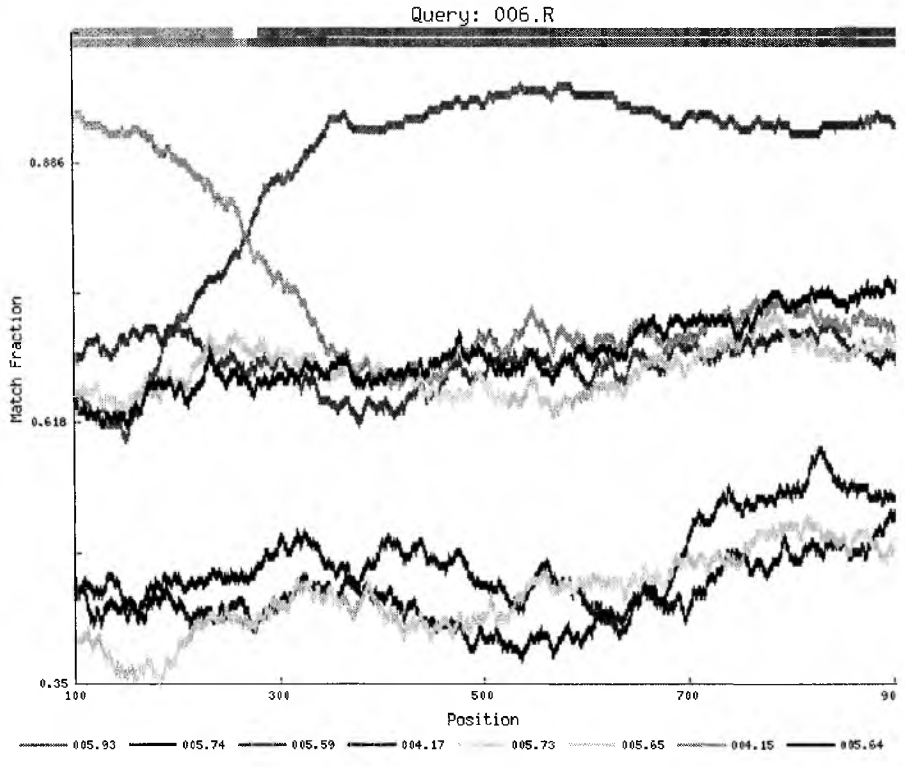


Figure 7: RIP Graph

The standard Bootscanning method (SB)

Bootscanning relies on the alignment of a suspected recombinant sequence (the query sequence) with a set of potential donor reference sequences. Bootstrapped phylogenetic trees are built for each segment and finally the bootstrap value for placing the unknown with each of the reference sequences or sequence groups is tabulated and

plotted along the genome (Lole et al. 1999; Martin et al. 2005b; Salminen et al. 1995). Recombination breakpoints can be found as the coordinate of the intersection of the plotted bootscan plots. The process requires a minimum of 4 sequences. Figure 8 shows the output of the RDP2 Bootscanning method for the same data set used with the RIP method and discussed above. The recombination signal is clear for this artificially generated data set. This sort of clear signal is, however, typically observed when the evolutionary history of the data set contains only one recombination event and the reference sequences are not too closely related to each other.

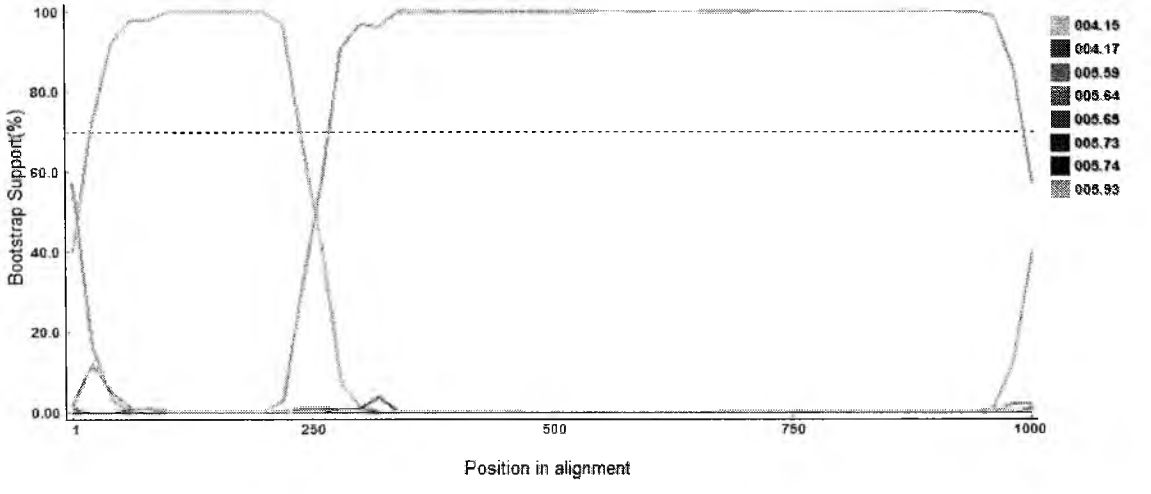


Figure 8: Bootscan Graph

2.4 Analysis of Serially-Sampled Data

2.4.1 Methods that Analyze Serially-Sampled Data

A viral sequence data set that has been sampled from the same host at successive time points is said to be *serially-sampled.* In the last few years, several methods have been devised to study serially-sampled data, many of which are variants of phylogenetic

methods for contemporaneous taxa. Drummond and Rodrigo developed *sUPGMA* by modifying the conventional UPGMA method (Drummond and Rodrigo 2000; Posada and Crandall 2002). Ren and Ogishima published a sequential linking algorithm (its implementation will be referred to as *SeqLink*) based on the Holmes evolutionary framework (Ren *et al.* 2001). *MinPD,* a distance based method (Buendia and Narasimhan 2004), is a program that was also inspired by the evolutionary framework, and is described in Chapter 4. Rambaut's *Tipdate* program is a variant of the traditional Maximum Likelihood (ML) method (Rambaut 2000). Finally, Drummond and Rambaut designed *BEAST*, which uses Bayesian MCMC (Drummond *et al.* 2002; Drummond and Rambaut 2003). Three of the above methods – sUPGMA, TipDate, and BEAST – assume a molecular clock, i.e., assume a stochastically constant or ultrametric rate of evolution. They are further constrained by the traditional tree style of handling contemporaneous data where the leaves (tips) correspond to the input taxa. The methods mentioned above cover most of the major approaches used for traditional phylogenetic analysis.

Table 1 gives a summary description of the relevant methods, while the next section provides a more detailed description of each. The method *MinPD* will be discussed in full detail in Chapter 4 of this dissertation, and the method *Sliding MinPD* will be discussed in Chapter 7. All of the above programs and two traditional phylogenetic methods mentioned in the previous section were included in a comparison study (discussed in Chapter 5) that sought to assess the performance of the different methods when provided with serially-sampled DNA data as input.

Method	Version	Description	Imp.	Clock	Ref
SeqLink	-	A distance-based method inspired by the *Holmes92* Framework. Links all sequences from one time period with one sequence from immediate previous period with the minimum overall distance.	C	No	(Ogishima *et al.* 2001; Ren et al. 2001; Ren *et al.* 2003)
MinPD	1.0	*MinPD* includes a recombination detection feature. Uses minimum distances to link portions of a sequence to portions of any other sequence from any previous time period.	C	No	(Buendia and Narasimhan 2004)
Sliding MinPD	1.0	*Additional Features: 3 automated sliding window recombination detection options. Outputs the bootstrap support for inferred relationships.*	C	No	
TipDate	*1.2*	Obtains Maximum Likelihood estimates of the mutation rate from an input of non-contemporaneous sequences and a known tree topology. http://evolve.zoo.ox.ac.uk/software.html	C	Yes	(Rambaut 2000)
BEAST	1.1.2	A Bayesian MCMC program. Unlike TipDate, tree topology is not required. http://evolve.zoo.ox.ac.uk/software.html	Java	Yes	(Drummond and Rambaut 2003)
sUPGMA	Pal 1.4	A distance-based program modified from the UPGMA method. A command line version of the JAVA PAL 1.4 package was used for the experiments. http://bioweb.pasteur.fr/docs/PAL/. http://www.cebl.auckland.ac.nz/pages/CeblRun.html	Java	Yes	(Drummond and Rodrigo 2000)

Table 1: Description of algorithms developed to study serially-sampled data.

Sequential-linking algorithm (SeqLink)

Two versions of a sequential-linking algorithm have been designed, one based on the traditional neighbor-joining (NJ) method and one based on the ML method (Ogishima et al. 2001; Ren et al. 2001; Ren et al. 2003). We implemented the NJ version of the algorithm (as described in (Ogishima et al. 2001; Ren et al. 2003)). We refer to our implementation (written in the C programming language) by the name *SeqLink*. The algorithm is based on an evolutionary framework (Holmes et al. 1992). *SeqLink* is based on two assumptions:

- The sequence from time point n with the minimum distance to some sequence in sampling period n+1 is the ancestor of **all** the sequences from sampling period n+1.
- The ancestor of a sequence was sampled at the previous time period.

Ties are broken by using additional criteria involving NJ branch lengths. Distances were measured using the JC69 distance, as other distance measures decreased the accuracy of the algorithm. A fatal flaw of this algorithm is that assumption (2) above need not be true, since there is no guarantee that the ancestral sequence was sampled in the previous time period.

TipDate

TipDate (Version 1.2) was designed to compute maximum likelihood (ML) estimates of the mutation rate from a set of non-contemporaneous input sequences (dated tips) assuming a molecular clock and a known tree topology (Rambaut 2000). The input sequences are called *dated tips*, as they are placed at the tips and not at the internal nodes of both the input (known topology) and output trees. According to the authors, *TipDate* was not intended as a method for constructing phylogenies, but rather to test evolutionary hypotheses and for estimating rates of molecular evolution.

Incorporating the sequence dates into the ML tree reconstruction method allows the sequences to be fitted to a constant rate model, enabling the inference of relative times of lineage splitting, and a rate of evolution that calibrates the tree into absolute time. The Tipdate tree has two scales: the time scale measured in time units and the branch-lengths scale measured in expected number of substitutions per site. A single rate of evolution is assumed for the entire tree (clock model), which gives a linear relationship between the two scales. TipDate recomputes the tree branch lengths to fit the molecular clock assumption by obtaining the maximum likelihood of the tree for different estimates of branch lengths.

BEAST

BEAST (Bayesian evolutionary analysis sampling trees), version 1.1.2, is a software package for evolutionary inference from molecular sequences (Drummond and Rambaut 2003). *BEAST* infers information about the unknown true ancestral coalescent tree, population size, and the overall mutation rate from temporally spaced data, that is, from nucleotide sequences gathered at different times, from different individuals, in an evolving haploid population. *BEAST* is a Bayesian MCMC program that uses a complex and powerful input format (specified in XML) to describe the evolutionary model. The program assumes a molecular clock by default and accepts either contemporaneous data or non-contemporaneous data as input. The approach for non-contemporaneous data (also called dated tips here) is equivalent to the *TipDate* approach; BEAST differs from *TipDate* however in that it does not require a user-specified input tree topology. Instead the tree topology and the branch lengths are estimated jointly at each step during the MCMC run. The prior in BEAST is a coalescent prior on the tree shape with one of two population models: constant size model or exponential growth model. These models basically act as priors on the ages of nodes in the tree but the parameters (population size and growth rate) can be sampled and estimated.

sUPGMA

This is another distance-based program modified from the *UPGMA* method, which by definition assumes a uniform rate of evolution (Drummond and Rodrigo 2000). The program was implemented as a command line script using the *JAVA PAL 1.4* package available from the URL: http://bioweb.pasteur.fr/docs/PAL/. While the online

version only requires a distance matrix and the sampling time information, the *PAL* version also requires a mutation rate as additional input. Our experiments (data not shown) showed that the rate greatly affects the accuracy of the program. The online version does not require a rate input and is available from the URL: http://www.cebl.auckland.ac.nz/pages/CeblRun.html. The tree output is similar to the other two methods in that the branch lengths stop at vertical points that correspond to the sampling time.

With *UPGMA*, all tips on the tree terminate at the same time (i.e., the tree is ultrametric). *sUPGMA* allows the tips to terminate at different times but constrains tips sampled at the same time to terminate at identical distances from the root. The method consists of four sequential steps:

1. Step one involves estimating the expected number of substitutions per site accumulated between two sampling periods $\delta_{early \rightarrow late}$. The expected distance between a pair of sequences, one from a later time point and the other from an earlier time point, is

$$E[dist(S_{early}, S_{late})] = E[dist(S(1)_{early}, S(2)_{early})] + \delta_{early \rightarrow late}.$$

The first term on the right-hand side is simply the expected average distance between any two sequences from the earlier time point. To obtain an estimate of δ, we substitute the average pairwise distance between early and late sequences calculated from our sample for the term on the left and the average pairwise distance between pairs of early sequences for the first term on the right and solve. The transitive condition $\delta_{AC}=\delta_{AB}+\delta_{BC}$ may not be valid for

three (or more) consecutive time points A, B, and C, in which case a general regression approach is adopted.

2. Pairwise distances are transformed to corrected distances.
3. UPGMA is applied to the corrected distances.
4. Tree branches are trimmed back with tips terminating in the appropriate order of sampling (Clock model).

2.4.2 Holmes Evolutionary Framework

Phylogenetic methods are typically applied to contemporaneous taxa, and result in the taxa being placed at the tips or leaves of the tree. In a serial sampling scenario an evolutionary framework may offer a more meaningful representation of the evolutionary relationships, in which the rise, persistence, branching, and extinction of different viral lineages is readily observable. Holmes et al. investigated the evolution of the V3 region of the HIV envelope gene (Holmes *et al.* 1992). They analyzed sequences of plasma viral RNA donated over a seven-year period by a single patient. Holmes et al. created an *evolutionary framework* (Figure 9a) to express the inferred ancestor-descendent relationships by placing ancestral sequences at the internal nodes of the framework. Holmes et al. stated that *"unlike most molecular phylogenies, real ancestors may be present in the data and the framework expresses the postulated ancestor-descendent relationships."* This statement provides the basis of some of the work presented in this dissertation. Figure 9b shows the phylogenetic tree inferred from the Holmes data set with the prefix of each sequence ID indicating the sampling time point.

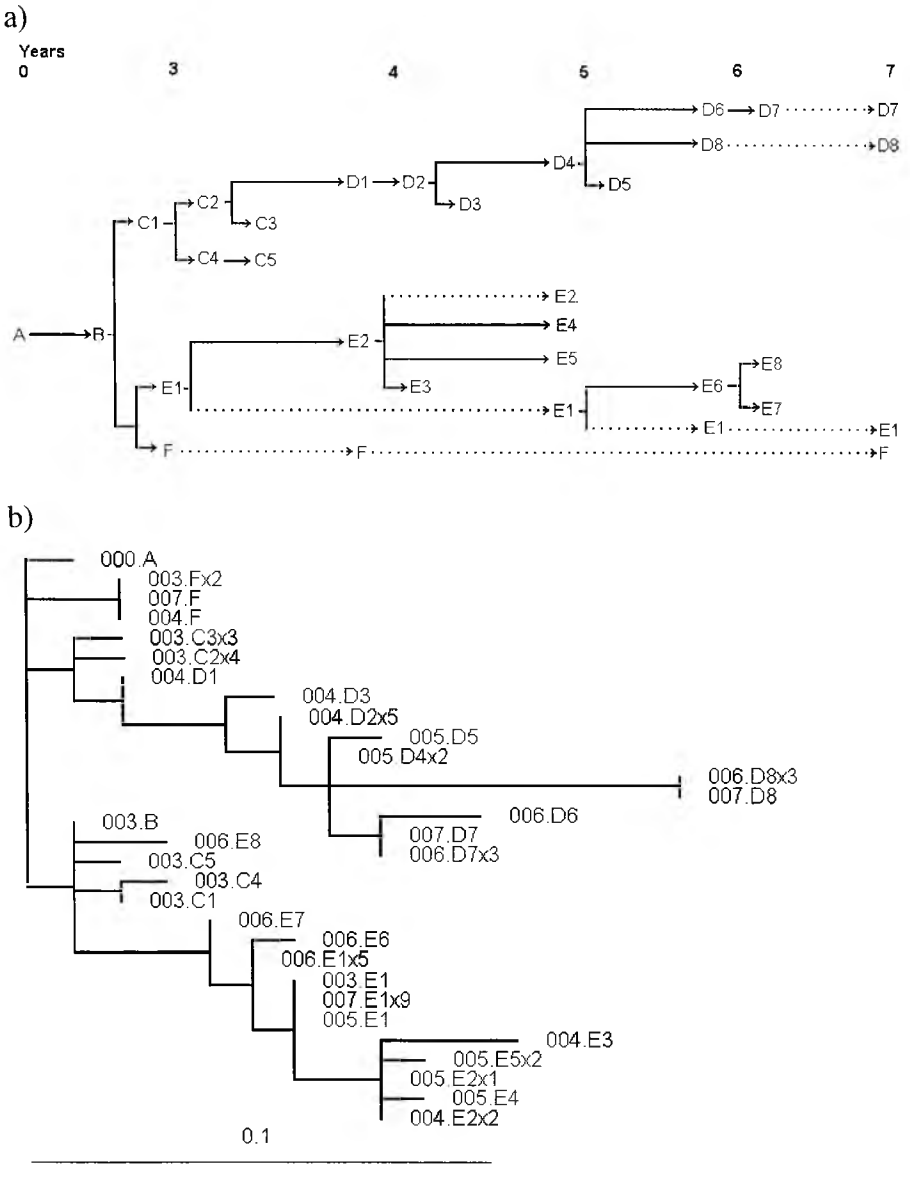


Figure 9. Comparison of (a) an evolutionary framework and (b) the equivalent phylogenetic tree

2.5 Simulating DNA sequence evolution

A common approach to evaluating the performance of phylogenetic methods is to first simulate the evolution of nucleotide sequences along a phylogenetic tree with a pre-specified topology, and to subsequently use the output of the simulation program as input to the phylogeny programs. The results of the phylogenetic programs are then compared with the "true" phylogeny. This involves either comparing the tree topologies or comparing inferred population or evolutionary parameters with the parameter settings used in generating the simulated tree.

2.5.1 Seq-Gen

Seq-Gen is an application for the Monte Carlo simulation of DNA sequence evolution along phylogenetic trees using standard models of substitution. A range of models of molecular evolution are available in Seq-Gen. Nucleotide frequencies and other parameters of the model may be given and site-specific rate heterogeneity may also be incorporated in a number of ways. The program requires a phylogenetic tree topology (a template genealogy) as input. The DNA sequences are then evolved along this tree. Any number of trees may be read in and the program will produce any number of data sets for each tree. Thus large sets of replicate simulations can be easily created. It has been designed to be a general purpose simulator that incorporates most of the commonly used (and computationally tractable) models of DNA sequence evolution (Rambaut and Grassly 1997).

Seq-Gen reads in a tree that must include branch lengths. Each branch length is assumed to denote the mean number of nucleotide substitution per site that will be simulated along that branch. The program evolves a root sequence along each branch of the input tree using a transition probability matrix (see section on models of evolution) until it reaches the tips of the tree. The tip sequences are finally output to a file. The process is repeated for each replicate requested by the user.

2.5.2 Treevolve and Other Coalescent Simulators

Coalescent theory is a model of population genetics applied to the study of gene sequences and is used to describe the characteristics of the joining of lineages back in time to a common ancestor. This lineage joining is referred to as coalescence. The coalescent process operates *backwards* in time and is terminated when the number of lineages in the sample equals one. The coalescent model of evolution provides a way to computationally model the genealogy of a sample drawn from a larger population under a variety of different dynamic parameters (Kingman 1982).

The coalescent method was also formulated for samples of gene sequences collected at different times. The ability to fit serial samples into the coalescent framework also translates into an ability to simulate the evolutionary genetics of the population over time. The coalescent model for serial samples was implemented in the Serial Coalescent Simulator (Drummond and Strimmer 2001) and the recently published Serial SIMCOAL (Anderson *et al.* 2005). The option to generate a recombinant network is not featured in either of the two programs.

Like its predecessors, SIMCOAL and Serial Coalescent Simulator, Treevolve is a program that simulates the evolution of DNA sequences under a coalescent model. A special feature of Treevolve, that is not present in its predecessors, is that it generates recombinant sequences evolved along a phylogenetic network (Grassly *et al.* 1999). Treevolve offers an extensive set of population parameters to adjust the evolutionary model. These include the exponential growth rate, the migration rate, the population size, and a rate of recombination. The coalescent model implemented in Treevolve extends the work of (Hudson 1983). Gene genealogies are generated under the chosen population dynamic model(s) with the specified recombination rate r. When r=0 the genealogy is a simple bifurcating tree, whereas with recombination a network is produced.

Treevolve does not require a tree or network topology as input but generates the topology using the coalescent process. A recombination event resulting in the production of two parents can occur anywhere along the sequence, and in the model implemented in Treevolve according to a uniform distribution. Coalescent events result in a reduction of the number of lineages by one, recombination events in the duplication of lineages. Once a tree or network has been generated the process of evolving the sequences along the structure is carried out using a process similar to that described in Seq-Gen. However, Treevolve has a few shortcomings. It does not simulate the evolution of serially-sampled data, and it does not output the tree or network structure it generates. It assumes a molecular clock of evolution and it does not output the sequences at the internal nodes. In the next chapter we will discuss a new sequence simulator program that was developed based on the coalescent theory and the Treevolve method discussed here.

3 Simulation of Serially Sampled Data

Whether it is for understanding the limitations of existing methods or to compare different methods, it is imperative that appropriate data sets be used. From the point of view of this dissertation, there are only a limited number of serially-sampled viral data sets available from public databases. It is therefore clear that good quality synthetic data sets with a number of adjustable parameters be used in order to comprehensively test all the developed methods and to carry out the comparison studies. This chapter presents the software package, Serial NetEvolve, which is a modification of the program Treevolve, and which addresses the goal of generating realistic simulations of serially-sampled sequence data (Goal 3). Other modified methods were also used in some of the earlier experiments. These include a modified version of the Treevolve program that incorporates different "sampling strategies" into the generation of simulated sequences. We also describe a modified version of the Seq-Gen method to simulate serially-sampled data without recombination. A version of the work related to Serial NetEvolve has been published (Buendia and Narasimhan 2006).

3.1 Introduction

Simulated sequence data may be used to compare the performance of different phylogenetic tree reconstruction methods and recombination detection methods. The simulation program that fits our needs must be able to mimic the serial sampling of recombinant viral nucleotide sequences from an infected host. As no such program has been developed, the program Treevolve (Grassly *et al.* 1999), which uses the coalescent

method to generate sequences along a recombinant network, was modified to simulate the evolution of recombinant serially-sampled data.

3.2 Modification of existing methods

The option to generate a recombinant network is not featured in the recently published Serial SIMCOAL (Anderson *et al.* 2005) or in the earlier Serial Coalescent Simulator (Drummond and Strimmer 2001), both of which simulate the evolution of serially-sampled data. In our experiments we used modified versions of Treevolve (Grassly et al. 1999), and Seq-Gen (Rambaut and Grassly 1997), both of which simulate recombination, but not serially-sampled DNA sequences.

3.2.1 Seq-Gen with internal nodes output

In most simulation studies the application of choice is Seq-Gen, a Monte Carlo DNA sequence generator that simulates the evolution of DNA sequences along a phylogenetic tree. Seq-Gen is the most often-cited synthetic data "sequence generator." The program was not conceived to simulate serially-sampled data, although it has an option to output the sequences for each of the internal nodes in the tree, which we observed to be equivalent to sampling the sequences at different times. This option was employed in the early simulation studies that were used to evaluate the performance of MinPD without recombination (see Chapter 4). Seq-Gen 1.2.5 was used in the study and enhanced by the twister randomization function of Seq-Gen 1.2.7. The input to Seq-Gen is a tree along which the evolution of sequences will be simulated. Optionally, an ancestral sequence can be added as input parameter. The output of the program is a

sequence alignment. An interesting effect occurs when more than one tree is provided as input for different partitions in the datasets; it simulates recombination of the sequences. However, this technique only simulates simple recombination histories. Another critical point to note is that when the recombination option mentioned above is selected the ancestral sequences are not output.

3.2.2 Treevolve with modified sampling strategies

For a specified set of parameters, *Treevolve* (Grassly et al. 1999) generates a coalescent tree (Kingman 1982) or recombinant network (Hudson 1983) and evolves a set of sequences along that structure. *Treevolve* does not require a user specified topology; however, it does not output the topology it generates.

Determining ancestor-descendant relationships from empirical data is difficult since only a small fraction of the total number of sequences representing a lineage are sampled in practice. This also complicates evaluating the success of the various algorithms. We used *Treevolve* to develop a simulation strategy that incorporates a random sampling step to reflect this incomplete sampling. A large tree is randomly generated and a small (randomly chosen) fraction of the sequences from the leaves and nodes of this tree is identified as constituting a sample. The program *Treevolve v1.3.2* was modified to return the sampled sequences from the internal nodes and the genealogy of only the sampled sequences (Grassly et al. 1999). The twister randomization function of *SeqGen 1.2.7* was also added. The randomization function was used to randomly sample a small number of sequences from the leaves and nodes of the full tree. Treevolve was used to generate sequences of lengths similar to those found in the serially-sampled

data sets in GenBank. The simulated data sets were generated through the following four steps:

1. Generate the random tree with the specified tree generation parameters.
2. Assign all sequences (at nodes and leaves) to sampling periods.
3. Randomly sample sequences from sampling periods using the specified sampling parameters.
4. Build and output a smaller tree containing only the sampled sequences and the linking nodes.

Data sets with both constant and variable rate of evolution were simulated. To simulate data sets with a constant rate of evolution, sequences roughly equidistant from the root were assigned to the same sampling period. The data sets with variable rates of evolution were designed to obtain more realistic simulated data sets and to better mimic the sampling of viral DNA or RNA sequences from an infected patient over an extended period of time; the variable rate was achieved by controlling how sequences were assigned to sampling periods.

The two modified tools described above were not adequate for experiments that required comparisons of tree topologies, or assessing the impact of sampling from the internal nodes. We therefore implemented a method we called Serial NetEvolve, which is also a modification of the Treevolve program, although its features are quite different to the "sampling strategies" approach that we discussed above. The results of applying different sampling strategies were of experimental interest to us and we carried out comparisons studies using that option (see chapter 5), but it is also necessary to remark that the coalescent model already provides a way to model the genealogy of a sample

drawn from a larger population (Rodrigo and Felsenstein 1999), which is another reason why we left that option out in the method Serial NetEvolve that we will discuss in the next section.

3.2.3 Serial NetEvolve

Serial NetEvolve is a flexible simulation program that generates DNA sequences evolved along a tree or recombinant network. It offers a user-friendly Windows graphical interface and a simulator with a diverse selection of parameters to control the evolutionary model. *Serial NetEvolve* is a modification of the *Treevolve* program with the following additional features: simulation of serially-sampled data, the choice of either a clock-like or a variable-rate model of sequence evolution, sampling from the internal nodes and the output of the randomly generated tree or network in our newly proposed *NeTwick* format.

Features

The new features of *Serial NetEvolve,* which are additions or modifications of the original *Treevolve* program are described in detail below.

Serially-sampled sequences

A theoretical framework for generating serial samples in the coalescent was developed (Rodrigo and Felsenstein 1999) and implemented in Java in *Serial Coalescent Simulator* (Drummond and Strimmer 2001). In *Serial NetEvolve*, we implemented the technique by modifying *Treevolve* to have sequences assigned to different time points instead of assigning them to the zero-time baseline.

Molecular clock

Distance to the root and time are equivalent when enforcing a molecular clock. To simulate data sets with a constant rate of evolution (enforced molecular clock), sequences from the same sampling time were assigned to the same distance from the root, as implemented in *Serial Coalescent Simulator*. For the variable rates setting, all sequences (independent of the sampling time they belonged to) were assigned to randomly generated distances from the root. For consistency, distances of the nodes were constrained to decrease as one traverses toward the root.

Internal node sampling

Serial NetEvolve allows the user to set the probability of internal node sampling as a parameter. Sampling of internal nodes makes sense only if the effective population is small (Drummond and Rodrigo 2000), which in turn is more likely when the length of the sequences sampled is small. It is worth noting that fairly short sequences (approximately 700bp in (Shankarappa et al. 1999)) were used in many large studies of viral populations, and that effective population size for serially-sampled HIV-1 was estimated to be only 4232.2 and negatively correlated to the evolutionary rate (Seo *et al.* 2002).

Tree or network output

Serial NetEvolve was further enhanced to output the coalescent tree or network. When the recombination rate is zero, the tree is output to a file in Newick format, which represents trees using nested parentheses (Felsenstein 1999). If internal nodes are sampled, they are assigned to zero-length branches. In order to write a recombinant network to a file, we devised the "NeTwick" format, a variant of the Newick format,

incorporating additional information (breakpoint position, right and left parent) to represent recombinant nodes. Unlike tree nodes, recombinant nodes have more than one parental node. In *Serial NetEvolve*, we (arbitrarily) chose the left parental node of a recombinant sequence to appear twice in the NeTwick format to indicate the linking relationship. One of the copies of the left parental node appears followed by the symbol "#", along with the breakpoint position and it represents a link, not a taxon. If the left parent was not sampled, it also appears with a "~" prefix. Figure 10 shows a network with 9 taxa and its tree equivalent.

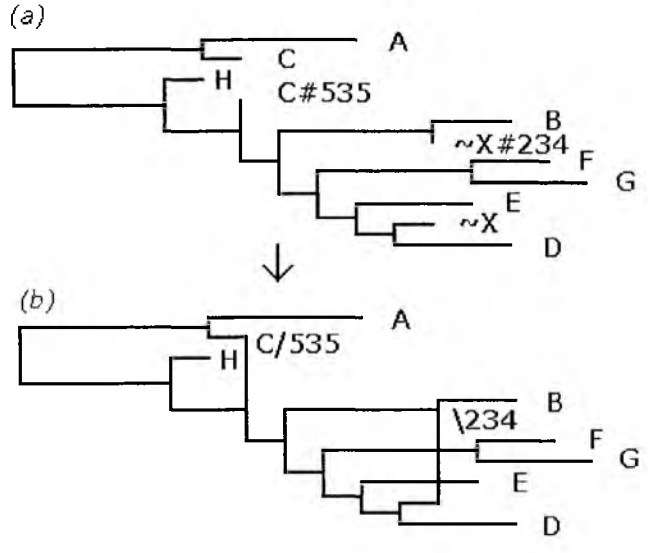


Figure 10. The (a) tree representation and the (b) equivalent network representation

The network is represented by the NeTwick code: ((A:4,C:1):5,((((B:2,~X#234:0):4,((E:3,(~X:1,D:3):1):1,(F:2,G:3):4):1):1,C#535:0):2,H:1):4). The left parent X was not sampled (indicated by the ~), but is present in the tree to indicate the linking relationship as shown in the network. In the proposed network representation a backward (forward, respectively) slash followed by the breakpoint

number indicates whether the left parent is below (above, respectively) in a horizontally drawn network. The advantage of making two copies of the left parental node of every recombinant node is that the network can then be represented by an equivalent tree. The tree can be viewed using any tree-viewing program; a network viewer is currently being developed. NeTwick supports tree and network polytomies, but is restricted to one child per recombinant parent.

3.3 Summary

Seq-Gen was used to simulate serially-sampled sequences without recombination. However, all ancestral sequences were internal node sequences, which is only realistic when carrying out experimental evolution studies in the lab as discussed in section 5.1. *Seq-Gen* can also be used to simulate recombination, however, its recombination feature is very limited in that it only offers the user option of providing a few different trees for different parts of the alignment. *Serial NetEvolve* offers a flexible set of population parameters (migration rate, population growth rate, among others) present in the original *Treevolve* and additional options for the generation of both trees and recombinant networks. The new features are: simulation of serially-sampled data, the choice of either a clock-like or a variable rate model of sequence evolution, sampling from the internal nodes and the output of the randomly generated tree or network in our newly proposed NeTwick format. The power of *Serial NetEvolve* lies in the ease with which it is possible to generate a collection of data sets using a wide range of parameters with the goal of comparing different programs that analyze any aspect of serially-sampled sequence data. Such experiments are fundamental to the evaluation strategies described in several of the

later chapters of this dissertation. With the help of *Serial NetEvolve*, it is possible, for example, to compare with relative ease the topologies of the output of several programs.

4 MinPD

One of the main goals of this dissertation is to develop new computational tools to find ancestor-descendant relationships between serially-sampled sequence data (see Goal 1). The MinPD algorithm presented in this chapter is a step in that direction. A preliminary version of this chapter appeared in the IEEE-CSB 2004 conference proceedings (Buendia and Narasimhan 2004).

In this chapter we will explore a new computational method to study within-host viral evolution to better understand the evolution and pathogenesis of viruses. Traditional phylogenetic tree methods are better suited to study relationships between contemporaneous species, which appear as leaves of a phylogenetic tree. However, viral sequences are often sampled serially from a single host. Consequently, data may be available at the leaves as well as the internal nodes of a phylogenetic tree. Recombination may further complicate the analysis. Such relationships are not easily inferred by traditional phylogenetic methods, nor expressed appropriately using phylogenetic trees. We have designed a new algorithm called MinPD, which is based on pairwise distances between taxa. Our algorithm uses multiple distance matrices and correlation rules to output a tree or network (Buendia and Narasimhan 2004).

4.1 Introduction

MinPD is a distance-based algorithm to infer evolutionary relationships (including recombination) in serially-sampled sequence data. An important feature of the algorithm is that it does not need the assumption of a molecular clock. Unlike methods

based on maximum likelihood (ML) or maximum parsimony (MP), the *MinPD* method is computationally efficient and can deal with a much larger number of input sequences. The method assigns ancestor relationships using minimum pairwise distances without resorting to a multiple alignment of the sequences. Ties are broken by resorting to divergence information. Recombinant strains are detected using sequence fragment matrices, correlations and distance rules. The *MinPD* algorithm was implemented in C. The accuracy of the method was assessed using extensive experimentation on both simulated data and on real HIV sequence data (obtained from the HIV database). The simulated data included sequences associated with randomly chosen leaves as well as internal nodes of a phylogenetic tree. A critical feature of the simulations is that it attempts to mimic the fact that, in reality, only a small random sample of all the viral strains that may be present in a patient is actually sampled. This is achieved by simulating a large number of sequences and discarding a large fraction of them.

Another contribution of this work is to show how to incorporate recombination into longitudinal phylogenetic trees without losing any of its essential features and advantages. The resulting phylogenetic networks make it convenient for a biologist to draw useful conclusions. This work is similar to the work of Ren *et al.*, with the significant added feature that it accounts for recombination. The *MinPD* method, unlike recombination detection methods, identifies recombinant strains, donors, and approximate breakpoint positions, and does not require the intervention of the user.

4.2 Method

The *MinPD* algorithm is based on the concept of *minimum pairwise distance*. It assumes that an ancestor of any given taxa must have been sampled at one of the previous time points and that the distance to the closest ancestor must be the minimum among all distances to taxa sampled during all prior time points. It utilizes the same criteria to find minimum distance fragments to all other sequences to identify possible recombinant strains. It also assumes that pairwise alignments give less distorted evolutionary distances than do multiple alignments.

4.2.1 The Multiple Alignment Problem

Phylogenetic analysis methods ranging from tree-building methods to recombination detection techniques such as bootscanning, employ (as an initial step) a multiple sequence alignment of all input sequences. Multiple alignments of sequences of different lengths must necessarily add gaps, which often lead to loss of information and gap scoring artifacts, which in turn distort the distance computations. In existing phylogenetic programs, all distances are computed using this multiple alignment, as is true, for example of the programs MEGA and PAUP (Kumar *et al.* 2004; Swofford 2000).

Figure 11 shows a multiple alignment of four sequences and also two pairwise alignments. The gap columns are ignored and do not count as mismatches. However, the pairwise distances in the two alignments are different. What is striking in the example is that the distances between two pairs of sequences exhibited a different order in the multiple alignment as compared to the corresponding distances in the pairwise

alignments. It is for the above reasons, that we use pairwise alignments that offer a more accurate distance measure.

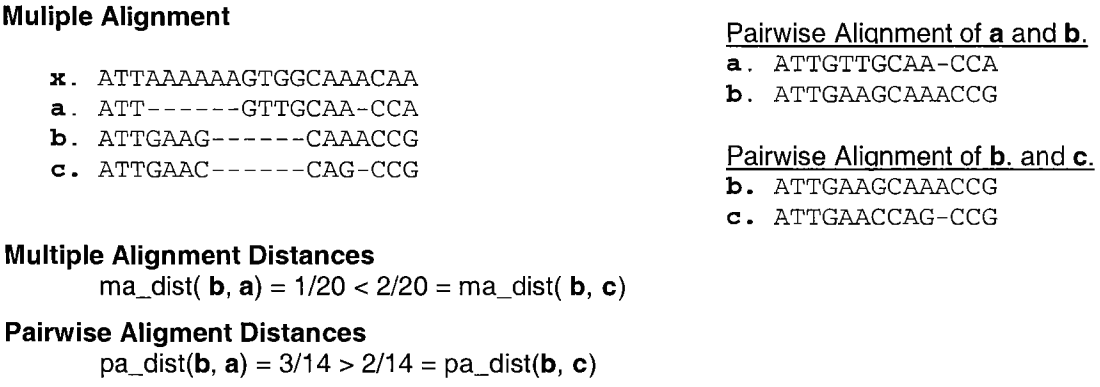


Figure 11. The problem with multiple alignments

Note, however, that multiple alignments can be manually improved by a judicious and critical examination by eye. This is a time-consuming process and, if one is confident of obtaining a good alignment, there is no need for the pairwise alignment discussed above.

4.2.2 The MinPD Algorithm

The inputs to the algorithm are: s, a set of sequences with associated time periods, k, the number of fragments, and t, the threshold for the Pearson Correlation Coefficient. We use the Needleman-Wunsch algorithm to compute an alignment between each pair of sequences. For computing pairwise distances we use the Tamura-Nei Model (TN93) of nucleotide substitution with Gamma-distances (Nei and Kumar 2000). Henceforth, whenever we refer to *distance* in this text, we mean the TN93 distance, and we calculate this distance from a pair of aligned sequences that did not undergo a multiple alignment operation. Finally, we also assume that if the distances indicate two possible candidates

for the closest ancestor, then ties are broken using divergence values. The *divergence* between two sequences denoted by Div(x,y) is given by:

$$Div(x,y) = distance(x,y) - [r(x) + r(y)]/(n-2),$$

where $r(x) = \Sigma_j\ distance(x,s_j)$ is the net total divergence of x to all other sequences, and n is the number of sequences being considered (Saitou and Nei 1987).

For recombination detection we will assume that there is at most one recombination or crossover point for any recombination between 2 sequences, limiting the number of donor strains to two. The *MinPD* algorithm is given below.

Algorithm MinPD

1. **For each** pair of sequences $\mathbf{s_i}$ and $\mathbf{s_j}$ **do**
   a. Pairwise align them and compute the distance $\mathbf{Dist(s_i,s_j)}$ between them.
   b. Partition $\mathbf{s_i}$ and $\mathbf{s_j}$ into **k** fragments and compute the distance vector $\mathbf{DistVec(s_i,s_j)}$ of the **k** distances between the **k** pairs of aligned fragments. Let its $\boldsymbol{\ell}^{\mathbf{th}}$ component be denoted by $\mathbf{Dist(s_i,s_j,\ell)}$, the distance between the $\boldsymbol{\ell}^{\mathbf{th}}$ fragments of $\mathbf{s_i}$ and $\mathbf{s_j}$.
2. **For each** sequence s_i **do**
   a. **if** ($\mathbf{s_i}$ passes the test described below for being a recombinant strain) **then** identify two donor strains and choose them as ancestors of $\mathbf{s_i}$.
   b. **else** choose as ancestor of $\mathbf{s_i}$ the sequence at minimum distance from it among sequences sampled at all previous time periods. Break ties using divergence values.
3. **For each** set of sequences with the same chosen ancestor, construct a NJ tree and connect the root of the NJ tree to the chosen ancestor.

Pairwise distances are computed in Step [1a]. However, distances are uninformative when recombination is involved because distance measures consider the entire sequence. In order to deal with the possibility of recombination, *MinPD* breaks up the sequences into fragments and finds distances between fragments. The fragment distances provide fine-grained distance information between two sequences. This information is used the recombination test descrbed below. In Step [3], to handle the case where many sequences may choose the same ancestor, we replace such a "star"-like configuration with a NJ tree. This is done mainly to provide detailed insight for such scenarios. Note that in Step [2a] above, any method could have been used to test for recombination or to identify the donor strains. In this preliminary version, we propose a simple uniform distance-based method to achieve the same goal. (In Chapter 7, we present an improved algorithm that uses more general recombination detection methods.)

MinPD Recombination Test for sequence s

1. **For each** of the **k** fragments of **s do**

   select the sequence $\mathbf{s_i}$ whose $\mathbf{i^{th}}$ fragment has minimum distance to the $\mathbf{i^{th}}$ fragment of **s**. Put all selected sequences in a list called *Candidates*. These sequences are candidates for being donors if s is a recombinant strain.

2. From the list *Candidates*, let *minSeq* be the sequence with minimum overall distance to **s**.

3. **For each** pair of sequences $\mathbf{s_i}$ and $\mathbf{s_j}$ from *Candidates* **do**

   if the Pearson Correlation Coefficient (PCC) between their distance vectors is above a distance threshold,

   then discard the sequence $\mathbf{s_i}$ or $\mathbf{s_j}$ with the higher overall distance.

4. **For all** sequences in Candidates **do**

   discard those that have a fragment with minimum distance in the middle of the sequence, and not at either end.

5. **For each** sequence $\mathbf{s_i} \neq minSeq$ **do**

   calculate $\mathbf{s_i\_dom} = \Sigma$ (**Dist**(*minSeq*,**s**,**i**) - **Dist**($\mathbf{s_i}$,**s**,**i**)) in all fragments **i** where $\mathbf{s_i}$ has the minimum distance to the corresponding fragment in **s**.

   If $\mathbf{s_i\_dom}$ is below a distance threshold, **then** discard $\mathbf{s_i}$.

6. **If** exactly two sequences are left undiscarded, **then** report **s** as being recombinant with the two sequences as potential donors.

Steps [1] and [2] identify sequences that are at minimum distance in each of the ***k*** fragments. If two sequences have very similar distance vectors (from ***s***), then it is probably because they are very similar. In this case, one of them is discarded in Step [3]. Similarity of two distance vectors is computed using the *Person Correlation Coefficient*, which is high if the vectors are highly correlated (i.e., similar). Step [4] is justified because of the earlier assumption that there is at most one breakpoint; a sequence with minimum distance fragment in the middle of the sequence would suggest two breakpoints, a case that is being ignored for this preliminary version of the algorithms. (Note that the general algorithm presented in Chapter 7 does consider multiple breakpoints.) Step [5] discards cases where the minimum distance sequence is sufficiently far away. As per the assumption of at most one recombination event per sequence, in Step [6], all instances that do not show exactly two donors are discarded.

4.3 Results

The *MinPD* algorithm was extensively tested with a large number of simulated data sets, and was also applied to a set of HIV sequence data isolated from one patient over a period of ten years. The proposed visualization of the phylogenetic tree/network further enhances the benefits of our methods.

4.3.1 Experiments with Simulated Data

To test the *MinPD* algorithm, two synthetic data collections were generated, the first without recombination, and the second with. The first collection (without recombination) was generated using SeqGen1.2.5 (Rambaut and Grassly 1997), which was enhanced by the twister randomization function of SeqGen 1.2.7. Each of the 100 data sets in this collection contained 1023 sequences from the leaves and internal nodes of a template tree, out of which an average of 32 were randomly chosen (to simulate sampling from a population) and was input to the *MinPD* algorithm. The results (see Table 2) show that more than 90% of the time, *MinPD* chose the correct closest ancestor (referred to in the table as a *Match*). A *subtree relative* (a direct descendant of the correct closest ancestor) was chosen about 9% of the time. All other outcomes were counted as errors. The errors included cases where a *grand ancestor* (ancestor of actual closest ancestor) was picked, although multiple mutations on the same location during evolution can lead to "backward" substitutions resulting in a grand ancestor being genetically closer to the queried sequence. The overall error rate was less than 0.5%. Note that picking a subtree relative is not classified as an outright "error" because we consider it as a minor deviation from the correct relationship.

	Runs	Sequence Length	Match	Subtree Relative	Errors
	100	600n	90.9%	8.8%	0.37%
	100	1000n	90.9%	9.1%	0.06%
Total/Average	200		90.9%	8.95%	0.22%

Table 2. Experiments with non-recombinant data

The second data collection (with recombination) consisted of 100 data sets each containing about 500 sequences (or slightly more, depending on how many recombination events occur) generated using the software package Treevolve version 1.3. Treevolve was modified to include the Twister randomization function of SeqGen 1.2.7., and to output sequences at the internal nodes. Treevolve evolves a sequence using Hudson's coalescent method with recombination (Grassly et al. 1999). As before, to mimic the actual sampling from large populations, an average of 45 of the roughly 500 sequences were randomly chosen for input to the *MinPD* algorithm. The sets of data were simulated under the HKY model of evolution with the alpha parameter of the gamma distribution set to 0.5. A transition/transversion ratio of 4 was chosen and the base frequencies were set to A=0.22, C=0.18, G=0.40, and T=0.2. The results of our experiments on the simulated data are shown in the table below. In order to get realistic data, a mutation rate of 0.5×10^{-4} and a population growth rate of 0.75×10^{-4} were selected. The recombination rate of 0.1×10^{-7} was selected since for the given mutation rate, higher recombination rates gave enormously long lineages. This is because although coalescent events result in a reduction of the number of lineages by one, recombination events cause an increase.

# Frag	Thres holds	Runs	Len	Total Count	Non Rec Matches	Non Rec Subtree Relative	Non Rec Errors	Rec Count	Rec Detected	Rec Matches	Rec Subtree Relative	Rec Errors	FP
4	0.75	100	600	4540	74.4%	20.9%	0.7%	149	67%	49%	37%	14%	0.6%
8	0.67	100	600	4540	73.4%	20.6%	0.6%	149	63.3%	55.8%	26.3%	17.9%	1.9%
4	0.9	100	1000	4671	72.8%	21.1%	0.3%	212	67.9%	56.9%	27.1%	16.0%	0.8%
8	0.8	100	1000	4674	74.3%	19.9%	0.5%	199	61.3%	52.5%	35.3%	12.3%	0.8%
Total		400			73.7%	20.6%	0.5%	177	64.9%	53.6%	31.4%	15.0%	1.0%

Table 3: Experiments with recombinant sequences

Table 3 shows the results of these experiments. The labels on the columns are explained below. In the presence of recombination events, the ability of the *MinPD* algorithm to correctly establish phylogenetic relationship among the input sequences is adversely affected. Even on non-recombinant sequences, the percentage of correct predictions dropped from over 90% (Table 2) to under 75% (Non Rec Matches) in Table 3. In each data set, about 3-5% (Rec Count/Total Count) of the strains sampled were recombinant strains. Of these, about 65% (Rec Detected) were correctly detected as being recombinant. The donors were correctly identified in over 50% (Rec Matches) of those cases. In about 15% (Rec Errors) of the cases, the program identified the donors incorrectly. In the remaining 31% (Rec Subtree Relative) of the cases, a *subtree relative* was determined to be the donor. Of the over 4500 sequences (Total Count) in the 4-fragment (#Frag) run, only 39 non-recombinant sequences were reported as being recombinant sequences by the *MinPD* program, accounting for a 1% false positive rate (False Pos).

The results were somewhat weaker when the sequences were divided into 8 fragments instead of 4, which required additional fine-tuning of the thresholds chosen for

the program. Note that when the sequences were divided into 8 fragments, there were more potential candidates for the choice of donors, which complicates the selection of correct donors. The threshold for PCC was set to values between 0.67 to 0.9 and the threshold for $\mathbf{s_i\_dom}$, was set to the average of the distances from $\mathbf{s_i}$ scaled by the quantity **n/k**, where **k** is the number of fragments, and **n** the number of fragments where $\mathbf{s_i}$ has a minimum distance. We conjecture that more fine-tuning can further improve the program's sensitivity, and that sliding-window methods could improve the specificity.

It would be reasonable to conjecture that detecting recombination signals is harder with shorter sequences, although this was not observed in our experiments with sequences of length 600. It is possible that this is balanced out by the fact that there is a corresponding lower probability of recombination events in smaller sequences.

4.3.2 Experiments with Empirical Data

The final evaluation of the *MinPD* algorithm was performed by constructing the phylogenetic network for HIV sequence data from patients available from the Los Alamos HIV database. The viral strains were sampled and sequenced for a single patient (patient S) at month numbers 5, 12, 20, 30, 40, 51, 61, 68, 73, 80, 85, 91, 103, and 126. The resulting "longitudinal" phylogenetic network is shown in Figure 13. Each sequence is labeled with the month number and an identification number. There is no reasonable way to evaluate the correctness of the resulting network. Therefore, we focus our discussions on how well it correlates with the emergence of X4 strains (see below), and on how the resulting network makes it convenient to draw a variety of conclusions. It is

worthwhile to compare the difficulty of drawing similar conclusions from the ML tree generated for the same data, as shown in Figure 12 below.

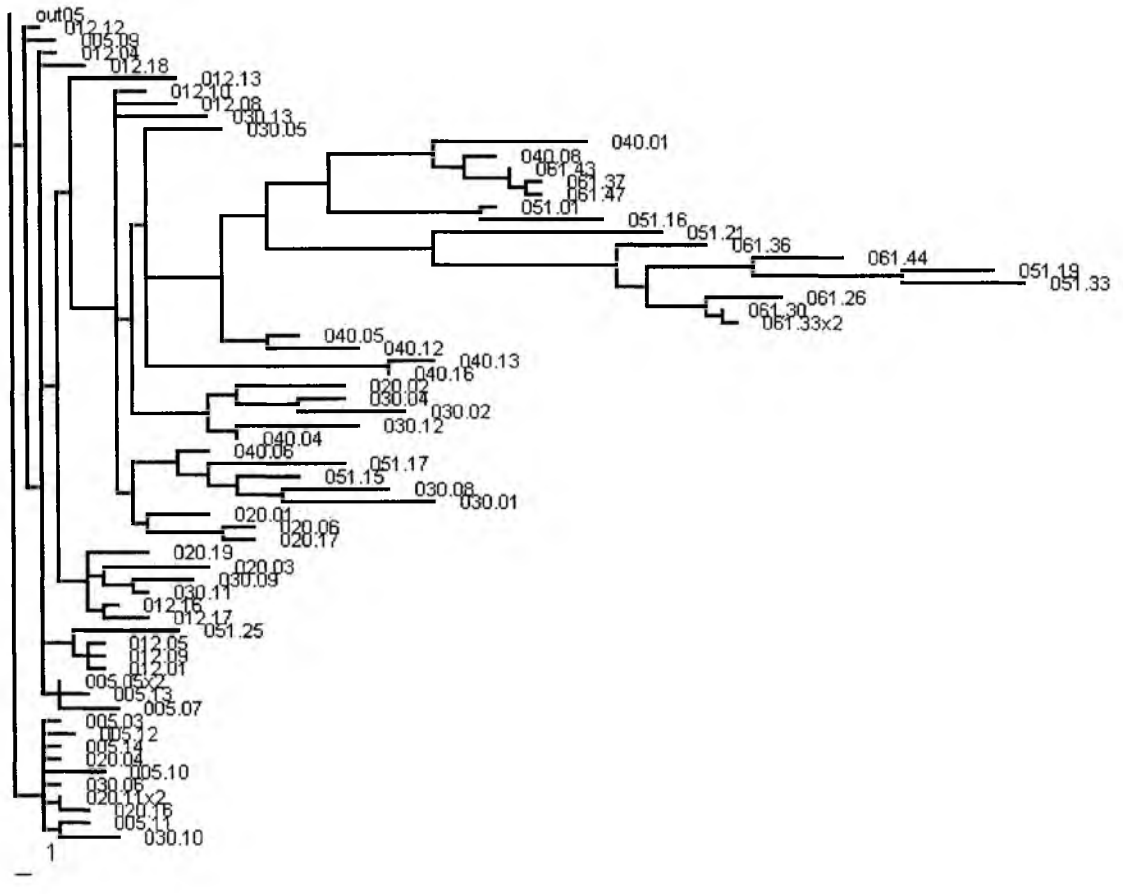


Figure 12. Maximum Likelihood (ML) tree of serially-sampled HIV sequence data from patient S.

The longitudinal network shown in Figure 13 is drawn from left to right and requires that sequences sampled at the same time be vertically aligned. This does not mean that all sequences undergo the same amount of evolution from the root sequence. On the contrary, every link between a parent and child node consists of straight-line segments, but are broken into vertical lines, horizontal thick lines and horizontal dashed lines. Horizontal thick lines are a measure of the amount of evolutionary changes that take place between the sequences. Horizontal dashed lines are added merely to achieve

the vertical alignment of the nodes corresponding to contemporaneous sequences. The only purpose of vertical lines is to ensure the correct connectivity.

All sequences marked with a red "x" have a lysine (K) or arginine (R) at position 320, a mutation that is predictive of the X4 phenotype. With the help of immunological data, it was shown by Shankarappa *et al.* that patient S's CD4+ and CD3+ T-cell numbers fell rapidly during the emergence of X4 genotypic strains (Shankarappa et al. 1999). The longitudinal network makes it convenient to understand how widespread the X4 genotype is in each sampling period.

Furthermore, it is interesting to note that patient S was prescribed antiretroviral drugs called zidovudine (ZDV) and stavudine (d4T) before the 103 months sampling period, and a few months later was prescribed lamivudine (3TC). The administering of this drug therapy coincides with a decrease in the X4 genotypic strains (Shankarappa et al. 1999), as is easily observed in the MinPD network in Figure 13.

The X4 genotypic strains made its first appearance in the samples from the 30 month period, and had proliferated by the 68 month period. Before its large-scale emergence (after 51 months), the MinPD network suggests that three groups of genetically similar quasispecies sequences were present in the population. One group became extinct at 51 months, while the other two groups each contributed a sequence, 051.19 and 051.16, that recombined to create strain 061.30, the possible closest ancestor of the large X4 quasispecies that dominated the sampled population in the ensuing years.

In the second half of the network corresponding to time periods 61 to 126 months, only two groups of quasispecies, linked by recombinant sequence 073.12, were identified by *MinPD*, one of the groups becoming extinct probably at the onset of antiretroviral

therapy with 091.19 as its last sampled sequence, and the other group formed by descendants from the recombinant sequence 061.30, giving rise to a mixed population of X4 and non-X4 genotypic strains. It is also interesting to note that the first X4 mutations that were sampled at 30 months had a relatively large genetic distance to its ancestor in comparison to the contemporaneous strains, suggesting a higher rate of mutations for those particular strains or during that time period. It should be noted that the above conclusions are made more convenient by the way the MinPD network is presented.

To make the comparisons more clear, the reader is encouraged to compare the result of *MinPD* from Figure 13 to the ML tree generated for the same data and shown in Figure 12. We aligned 65 sequences from the first 61 months using ClustalX. Subsequently we did a heuristic search for the ML tree using PAUP (version 4b10) (Swofford *et al.* 1996). If the horizontal axis is to be thought of as time, then the ML tree shown in Figure 12 exhibits several anomalies with strain 051.19 (sampled at 51 months) appearing after strain 061.31 (sampled at 61 months), and strain 030.01 appearing after strain 051.51.

Furthermore recombinant taxa cannot be identified in a traditional phylogenetic tree, but for the fact that they often have long branches leading into them. In the *MinPD* network, recombinant sequences are linked to their donor ancestors by blue lines and the breakpoint position is added left and next to the recombinant sequence. The recombination results output by *MinPD* were studied in detail using graph analysis, and only recombination relationships with the strongest signals were added to the network. Sequences with weaker recombinant signals were underlined in blue. A 2002 study of *in vitro* HIV-1 sequences and recombination site analysis suggested that the C2 env domain

breakpoint position is added left and next to the recombinant sequence. The recombination results output by *MinPD* were studied in detail using graph analysis, and only recombination relationships with the strongest signals were added to the network. Sequences with weaker recombinant signals were underlined in blue. A 2002 study of *in vitro* HIV-1 sequences and recombination site analysis suggested that the C2 env domain was a particularly "hot" region for recombination (Quinones-Mateu *et al.* 2002). The data of patient S does not contain the entire C2 region but includes the regions V3, C3, V4, C4, and V5, all of which were also found to have several recombination sites. Upon inspection of the *MinPD* network it can be observed that most recombinant sequences and signals were detected at the 61(2) and 68(2) months - these seems to correlate with the emergence of the X4 genotype. At 85(3) months there is another surge in recombination signals. The recombination signals are markedly stronger for the sequences with the X4 genotype than for those without, which corresponds with the higher genetic diversity of those time periods (Shankarappa et al. 1999).

1.12 Summary

In this chapter we have addressed Goal 1 by describing a new method called *MinPD* to study the phylogenetic relationship of serially-sampled quasispecies and to visualize the relationships between them. We presented results of extensive computer simulations in which we mimic random sampling of sequences. We also discussed how to interpret the results of our algorithm in the context of viral disease progression and showed how to incorporate the information in the visualization of the tree.

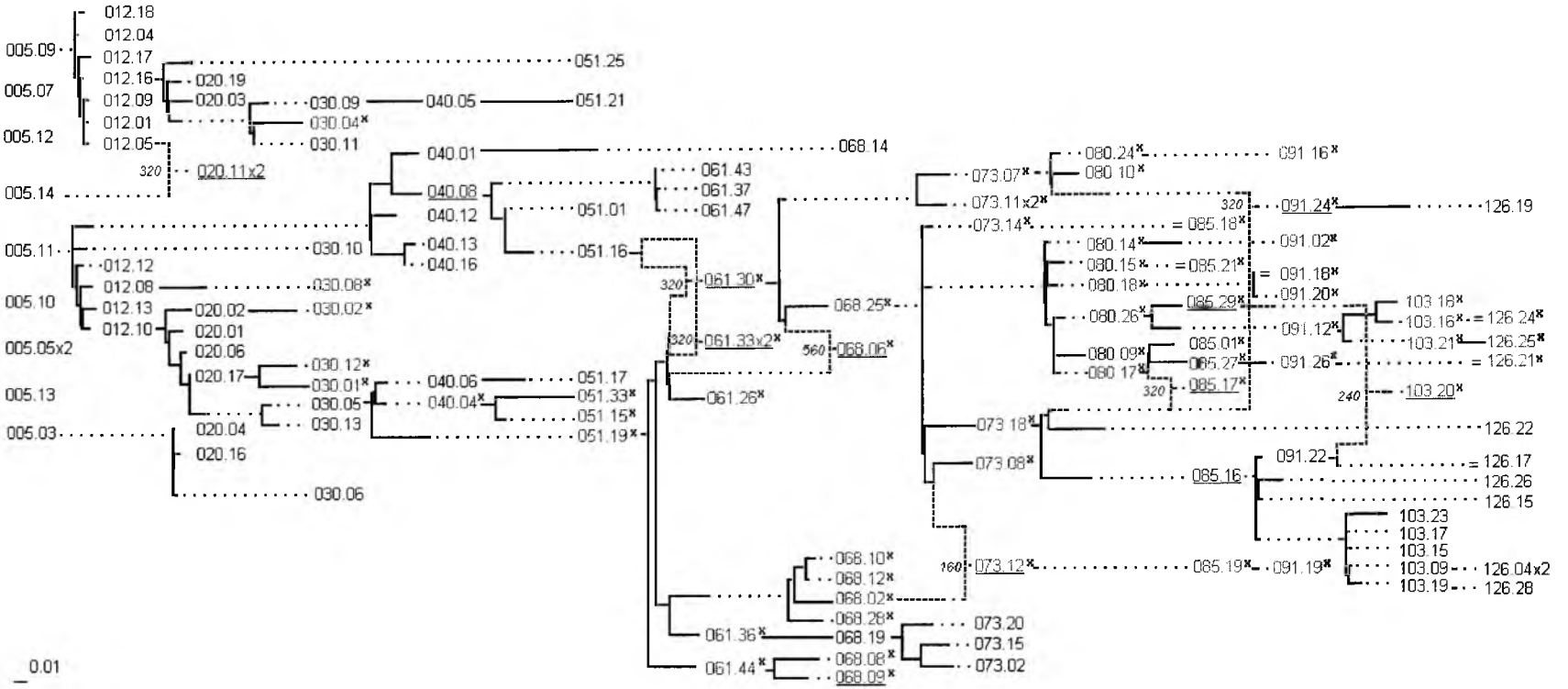


Figure 13. MinPD Network of Patient S.

Solid lines indicate distances, while dotted lines serve to extend the linking relationships. Each sequence is labeled with the month number and an identification number. Sequences with a mutation predictive of the X4 phenotype are written in red font and also marked with a red "x". Blue dashed lines are used to link recombinant sequences with their predicted donor sequences. The small numbers in blue next to branch points in the tree are the predicted (approximate) recombination breakpoint positions. Sequences with weaker recombinant signals are underlined in blue, but are not linked to their putative donor sequences. Note that the sequences were divided into 8 and 4 fragments for the recombination analysis.

We studied a method to detect recombination in serially-sampled data and presented the results of and presented the results of simulation experiments. Our method is especially helpful in selecting putative recombinant sequences among a large set of sequences. At this point the selected sequences may be analyzed using another tool to find exact breakpoints and detect more than one crossover, but in the cases where the sequence length is short the presence of more than one crossover is less likely and our method will return better results. Applying our method on simulated recombinant data returned a 65% success rate with a small number of false positives. In Chapter 7, we generalize the *MinDP* method to allow for (a) improved recombination detection methods, (b) for multiple recombination events within the same sequence, and (c) for computing statistical significance information for the predictions.

5 Comparison of Phylogenetic Methods for Analyzing Serially-Sampled Sequence Data

In the previous chapter, we described the MinPD algorithm, an algorithm to infer ancestor/descendant relationships from serially-sampled sequence data. The algorithm was evaluated on the basis of its performance on an empirical data set of HIV sequences serially-sampled from a single patient. Furthermore, it was also evaluated on the basis of its performance on a large number of synthetically generated data sets. However, in chapter 4, MinPD was not compared against other existing methods; this chapter addresses this shortcoming. The focus of this chapter is a comparison study of seven methods to assess how well they recover tree topology and ancestor/descendant relationships in lineages of serially-sampled sequence data, thus addressing Goal 5 of this dissertation. The assessment was performed using two experimental phylogenies (known molecular phylogenies that were produced in the laboratory) and a large collection of simulated data sets (using the coalescent method). Parts of this chapter have appeared in one publication (Buendia et al. 2006a) and a second one (Buendia *et al.* 2006b) is under review.

Two different simulation studies were carried out, and each will be discussed in a separate section:

1. **"Sampling Strategies and evolutionary parameters" study:** The objective of this study was to simulate the effect of sampling from a large population by focusing on sampling strategies (Buendia et al. 2006a). This study used an evaluation criterion

based on branch lengths and compared 5 different methods on 2 empirical data sets and on simulated data.

2. **"The Clock model and internal nodes sampling" study:** This study made use of the program Serial NetEvolve (Buendia et al. 2006b) to assess the effect of the molecular clock and sampling of internal nodes in the computation of topologies and branch distances. Two criteria were used in the evaluation of the results, one based on topology, the other on branch distances. The study compared 7 methods using input data from two experimental evolution studies and simulated data.

We extended our comparison study to include two traditional phylogenetic methods, one based on Maximum Parsimony (MP) and one on the Maximum Likelihood (ML) approach, to assess the effects that non-contemporaneous data has on these methods.

5.1 Introduction

Phylogenetic methods typically analyze contemporaneous taxa, and result in phylogenetic trees in which the taxa are placed solely at the tips or leaves of the tree. However an increasing amount of serially-sampled data of rapidly evolving organisms within a single host is now available in public databases. (For example, see URL: http://hiv-web.lanl.gov/content/hiv-db/mainpage.html.) Viral evolutionary history within a host can shed light on fundamental evolutionary mechanisms such as selective pressures and the evolution of drug-resistant strains, with implications for patient prognosis and treatment strategies.

Experimentally generated phylogenies, where the actual ancestors are known (Hillis et al. 1992), provide a rare, but valuable, source of DNA sequences for the

comparison of methods that analyze serially-sampled data. Hillis et al. pioneered the use of known molecular phylogenies, producing a known T7 phage phylogeny in the laboratory (Hillis et al. 1992). Cunningham et al. extended this work by serially propagating six bifurcating lineages of bacteriophage T7 according to the protocol of Hillis et al. resulting in a data set with known phylogeny (Cunningham et al. 1997). The rate of mutation of T7 phage was accelerated with the mutagen nitrosoguanidine, and each lineage was bottlenecked to a single individual from which the descendant lineages were further propagated.

In contrast to the T7 phylogeny, the experimental evolution of DNA sequences using a bifurcate series of nested polymerase chain reactions (PCR) constitutes a neutral evolution system for which the molecular clock hypothesis is justified (Sanson et al. 2002). Sanson et al. used MP, ML, and distance–based analysis of only the terminal sequences to reconstruct the topology and the branch lengths of the real phylogeny. As in the 1997 study by Cunningham et al., ancestral sequences were excluded from the tree reconstruction by Sanson et al., although they were used to assess ML predictions of ancestral states. In our study, the complete data set consisting of both ancestral as well as terminal sequences was taken into consideration. Henceforth, we will refer to the T7 phage sequences as the Cunningham97 data set and the PCR sequences as the Sanson02 data set.

In a 1992 study, Holmes et al. created an "evolutionary framework" to express the inferred ancestor-descendent relationships in HIV sequence data serially-sampled from a single patient (see section 2.4.2). Holmes et al. stated that "unlike most molecular phylogenies, real ancestors may be present in the data and the framework expresses the

postulated ancestor-descendent relationships" (Holmes et al. 1992). Figure 14 shows an evolutionary framework relating 24 different amino acid sequences found in the V3 loop, and redrawn in rectangular format from the paper by Holmes et al. (see Figure 2 (Holmes et al. 1992)). Time scale is given along the top. Dashed lines indicate identical sequences.

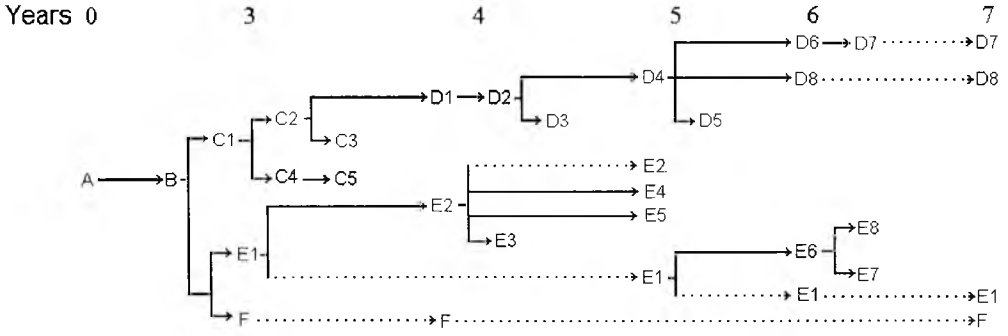


Figure 14. Evolutionary Framework of Holmes et al. HIV sequences

We used the Holmes viral data set in one of the empirical studies and will henceforth refer to the HIV data as the Holmes92 data set

It has been conjectured that different approaches and tools would be needed to analyze serially-sampled sequence data (Rodrigo and Steel 2004). More recently, several researchers have attempted to adapt existing phylogenetic methods to analyze serially-sampled data (see Methods section). The goal of our study is to compare different phylogenetic methods and assess how well they capture ancestor/descendant relationships in lineages of serially-sampled sequence data. The methods were tested on the three published phylogenies described above and on a large number of simulated data sets, simulated using the coalescent method. Two types of simulated data were generated:

- Data generated with different sampling strategies and population parameters: Since in practice only a small fraction of the total number of sequences representing a lineage is sampled, we incorporated a random sampling step into the simulation strategy.
- Data generated with Serial NetEvolve for the comparison of topologies and to study the effect of internal node sampling.

5.2 Methods and Settings

The goal of the current study was to compare methods that could be used to analyze serially-sampled sequence data and to assess their accuracy in recovering ancestor-descendant relationships. Towards this end, we compared methods that were specifically designed to analyze serially-sampled data, but we also included standard phylogenetic methods, such as Maximum Likelihood and Parsimony. Therefore, besides the methods reviewed in section 2.4, we also chose two traditional phylogenetic methods (see section 2.2.2) — fastDNAml, an efficient implementation of the ML method (Felsenstein 1981; Olsen et al. 1994), and DNAPARS, an MP-based program (Felsenstein 2004). The methods used in our study and the chosen settings are summarized below in Table 4.

Method	Version	Clock	Settings
SeqLink	-	No	JC69 distance matrix
MinPD	1	No	TN93 distances, no recombination detection.
TipDate	1.2	Yes	Dated Tips
sUPGMA	Pal1.4	Yes	Tipdate's rate and 3 fixed mutation rates: 0.0009, 0.004, 0.009
BEAST	1.1.2	Yes	HKY, constant growth, chain length: 1000000. Empirical base frequencies and equal among-site rates
DNAPARS	3.6	No	Default: Thorough search, 10000 trees to save
fastDNAml	1.2.2	No	HKY, Ts/Tv: 2. Empirical base frequencies and equal among-site rates

Table 4. Description of algorithms compared in this study.

5.3 Evaluation Techniques

We performed comprehensive experimentation to compare the seven methods mentioned above for their ability to correctly infer known or highly corroborated ancestor-descendant lineages from simulated serially-sampled or empirical sequence data. Given that the outputs of the programs have different formats, our main focus was not the comparison of frameworks and tree topologies, but mainly the assessment of how well serially-sampled data from later periods were linked to their ancestral lineages by the programs. We did carry out comparisons that involved calculating topology scores and comparisons that involved calculating a so-called ancestors-descendant score (A-D Score).

5.3.1 A-D Tree Performance Score

A scoring mechanism was devised to evaluate the correctness of the inferred relationships between serially-sampled data. Different methods produced different kinds of output, and a fair and flexible scoring mechanism was needed. For example, MinPD and SeqLink calculate ancestor-descendant relationships to construct the evolutionary framework, and was thus relatively straightforward to score. Other methods simply output a tree, and for these programs, the correctness of an inferred ancestor-descendant relationship was judged on the basis of branch lengths in the tree output. The resulting evaluation measure is referred to as the *A-D Score* and is based on the percentage of correctly inferred ancestor-descendant relationships. For a given taxon, the closest ancestral relative is defined as the closest sequence (i.e., with minimum path length if output is a tree with branch lengths) sampled at some previous sampling period, and

corresponding to either the sampled most recent common ancestor or to the closest sampled descendant of an unsampled most recent common ancestor. Every ancestor-descendant pair of sequences (at minimum distance from each other) that is found in the output of a program that matches the true relationship is counted positively towards the *A-D Score* of the program.

As some methods output a phylogenetic tree without explicitly inferring ancestral relationships, we devised a program, Nwk2Ances, which reads in a phylogenetic tree in Newick format as input, and returns for each sequence the "closest" sequence from any previous sampling period according to the definition given above. Given a phylogenetic tree with inferred lengths, Nwk2Ances uses an additive metric to search for the minimum-length path between a sampled sequence and a sampled ancestral sequence, where the path length is the sum of branch lengths along that path.

5.3.2 R-F Tree Score

A different scoring mechanism was used to evaluate the topologies predicted by the different methods. The topological distance score is based on the widely known *Symmetric Difference* measure devised by Robinson and Foulds and referred to as the RF distance (Robinson and Foulds 1981). It was calculated using TreeDist program in the PHYLIP software package (Version 3.5) (Felsenstein 2004). Distances were normalized by its largest possible value (twice the number of internal branches), which is $2n$-6 for two binary trees with n leaves. Lower values of the RF distance imply a better performance. Although two of the methods, MinPD and SeqLink, do not output a tree but a collection of trees rooted at their ancestral node (a framework), we implemented a

technique to merge these trees into one large binary tree: each tree rooted at an ancestral node was joined to the tree that contained the ancestral node leaf. This puts both programs at a disadvantage, but is necessary to perform a fair comparison. It should be noted however, that although the RF score is the standard method for tree comparisons, it has a poor resolution and two trees differing solely in the position of one taxon can be maximally different (Penny and Hendy 1985)

5.4 Studies of Sampling Strategies and Ancestor-Descendant Lineages

We evaluate the performance of 5 different methods in correctly inferring ancestor-descendant relationships by using empirical and simulated sequence data. Our results suggest that for inferring ancestor-descendant relationships among serially-sampled data, the MinPD program is an accurate and efficient method, and that traditional ML-based methods, while marginally more accurate, are far less efficient.

5.4.1 Empirical Data: Results

The 31 T7 page Cunningham97 sequences were 2733 nucleotides long (Cunningham et al. 1997), and comprise a much more robust data set than the 89 short sequences (each of length 35 aa) used by Holmes et al. (Holmes et al. 1992). Other features, such as the presence of parallel evolution and the skewing of the mutational bias and the number of invariable sites by the mutagen, nitrosoguanidine, used in the propagation of the T7 page, presented additional challenges to phylogenetic reconstruction methods, especially those based on an assumption of a clock-like rate of evolution. Table 5 shows the A-D Tree performance scores for all five programs on the

two empirical data sets. SeqLink recovered most of the Holmes92 relationships. It fared poorly with the Cunningham97 data set. The poor performance of the algorithm with the T7 page phylogeny may be due to the strong assumptions of the algorithm. MinPD recovered 100% of the Cunningham97 lineage relationships. As for the Holmes92 framework, the one notable difference was with sequence E8 sampled in year 6, for which MinPD chose sequence B (from year 3) as being a closer representative of its ancestral lineage over sequence E1 as postulated by Holmes et al. (Holmes et al. 1992). FastDNAml and the clock-based methods, TipDate and sUPGMA, performed better on the Cunningham97 data set. Nwk2Ances was applied to the output trees of the ML and clock-based programs to calculate the A-D Tree performance score.

Programs	Performance Scores		
	Holmes92	Cunningham97	Average
MinPD	95.65%	100.00%	97.83%
fastDNAml	65.22%	96.43%	80.82%
TipDate	69.57%	92.86%	81.21%
sUPGMA PAL+TD rate	69.57%	75.00%	72.29%
Seq-Link	78.26%	10.71%	44.49%

Table 5. Performance Scored for Empirical Data

5.4.2 Simulated Data: Results

A large number of DNA sequences were generated using a modified version of Treevolve (Grassly et al. 1999) in which special sampling strategies were implemented as described in section 3.2.2. Our modified version of Treevolve performed the following steps:

1 Generate the random tree with different combinations of tree generation parameters: Mutation rate, Recombination Rate, Clock, Number of Leaves (see Table 6).

2 Assign all nodes to sampling periods and randomly sample sequences from sampling periods using specified sampling parameters: Sample Size, Start of Sampling, Number of Periods (see Table 6).

3 Output smaller tree containing only sampled sequences and linking nodes.

Twelve sets of 100 replicates each were generated for different parameter combinations. Parameters were selected based on information from published studies (Shankarappa et al. 1999). These sequences were provided as input for the five programs under consideration. As before, evaluation was based on the A-D Tree Performance Score measure.

Fixed Parameters			
Name		**Settings**	
Model of Evolution		HKY	
Transition/Transversion ratio		4	
Base Frequencies		A=0.3, C=0.2, G=0.3, T=0.2	
Population Dynamic Periods: Period 1 (encountered first when going backwards in time) and Period 2.		Period 1: constant pop. size of 1000000. Period 2: exp. pop. growth.	
Gamma	Shape (Alpha) of the gamma rate heterogeneity	0.5	
SeqLen	Sequence Length	1000	
Periods	Number of Sampling Periods: 10 replicate tests were run to determine an appropriate default number of periods for the 100 replicate runs.	6	
Coalescent rate		1	
Variable Parameters (Tree Generation)			
Name	**Description**	**Values**	**Default**
Mutation Rate	Mutation rate $2*10^{-6}$ was chosen because 10^{-6} caused a severe slow-down of the program fastDNAml. The rate of 10^{-5} is approximately equivalent to that of the HIV virus.	10^{-4}, 10^{-5}, $2*10^{-6}$	10^{-5}
RecRate	Recombination rate.	0, 10^{-8}, 10^{-7}	0
Clock	Data sets were generated either with or without assuming the clock model. For the clock models, time period assignments were correlated to the distance from the root.	Yes, No	No
Leaves	The number of leaves in the tree. (The number of sequences in the tree is leaves*2-1.)	5000, 50000	5000
Variable Parameters (Sampling)			
Name	**Description**	**Values**	**Default**
SSize	Sample Size: the number of sequences per period to be sampled. Values were selected according to the number of sequences available or expected in the public databases	2, 4, 6, 8	8
Start	The time period when sequences were first sampled. The values were selected to observe the effect of sampling closer or farther away from the root of a tree	3, 6, 9	9

Table 6. Simulation parameters used in Treevolve 1.3.2 (sampling strategies version)

Results were analyzed with the standard statistical software package, SPSS 13, by running 2-way ANOVAs on the program performance scores using one of the simulation or sampling parameters as a second variable. As variances were large and mostly overlapping, Post Hoc analysis (Bonferroni, $p<0.05$) was used to determine which differences in means were significant.

The data did not meet the assumptions of a normal distribution or homogeneous variances, suggesting an arcsine-sqrt transformation of the performance score variable. However, it is known that ANOVA is robust to violations of these assumptions as long as the data consists of large and roughly equal-sized samples (Glass *et al.* 1972). Furthermore, boxplots (or normality plots) of the arcsine-sqrt score did not show any marked skewness, and the interaction and Post Hoc results remained identical with or without the transformation; here we report the untransformed performance scores.

We discuss in detail the statistical analysis and the interaction effects that were found to be significant. The graphs are shown in Figure 15.

Clock: Among all our experiments, the only case when the clock-based method, TipDate, outperformed MinPD and ML, was when the data sets were generated assuming the clock-like model of evolution. Even for these data sets, the difference was within a margin of 4-6% (see Figure 15) and was significant from MinPD's performance scores, but not from fastDNAML scores. The effect size, measured by the partial eta-squared value, of 0.23 was the largest among all the interactions.

Recombination Rate: Picking one recombinant donor (parental sequence) out of two or more (a partial match) counted as a correct prediction. This scoring choice explains why the performances were not affected by the higher recombination rates, but also suggests

that the programs placed one of the recombinant donors closer to the reference sequence about as frequently as they did with ancestors of non-recombinant sequences. Effect Size was only 1%. (MinPD's recombination detection feature was turned off for these experiments).

Mutation Rate: MinPD and fastDNAml displayed improved performance at a mutation rate of 10^{-5}, while the clock-based methods achieved their best performance at a faster rate of 10^{-4}, which resulted in highly divergent sequences. At a slower rate of 2×10^{-6} (at 10^{-6} fastDNAml resulted in a severe slowdown) all of the methods had difficulty inferring ancestor-descendant relationships. The effect size was small at 3%.

Leaves: More leaves imply a larger tree and therefore, a smaller sampling rate. The performance scores declined slightly for the larger trees. The effect size was only 1%. The performance scores of fastDNAml for the 50K leaves data sets were significantly better than that of the other programs, including MinPD.

First Sampling Period (Start): Earlier initiation of sampling resulted in sampling of sequences closer to the root. The plots confirmed that inferring relationships is more difficult when sampling starts after sequences have diverged and separated into different lineages. The interaction effect size was 9%.

Sample Size (SSize): The interaction effect was very low (1%). The only significant decline in performance (-5%) for the top three programs occurred when the sample size was increased from 2 to 4. The lower error rates for smaller sample sizes are attributed to the smaller pool of ancestors that the programs end up choosing from.

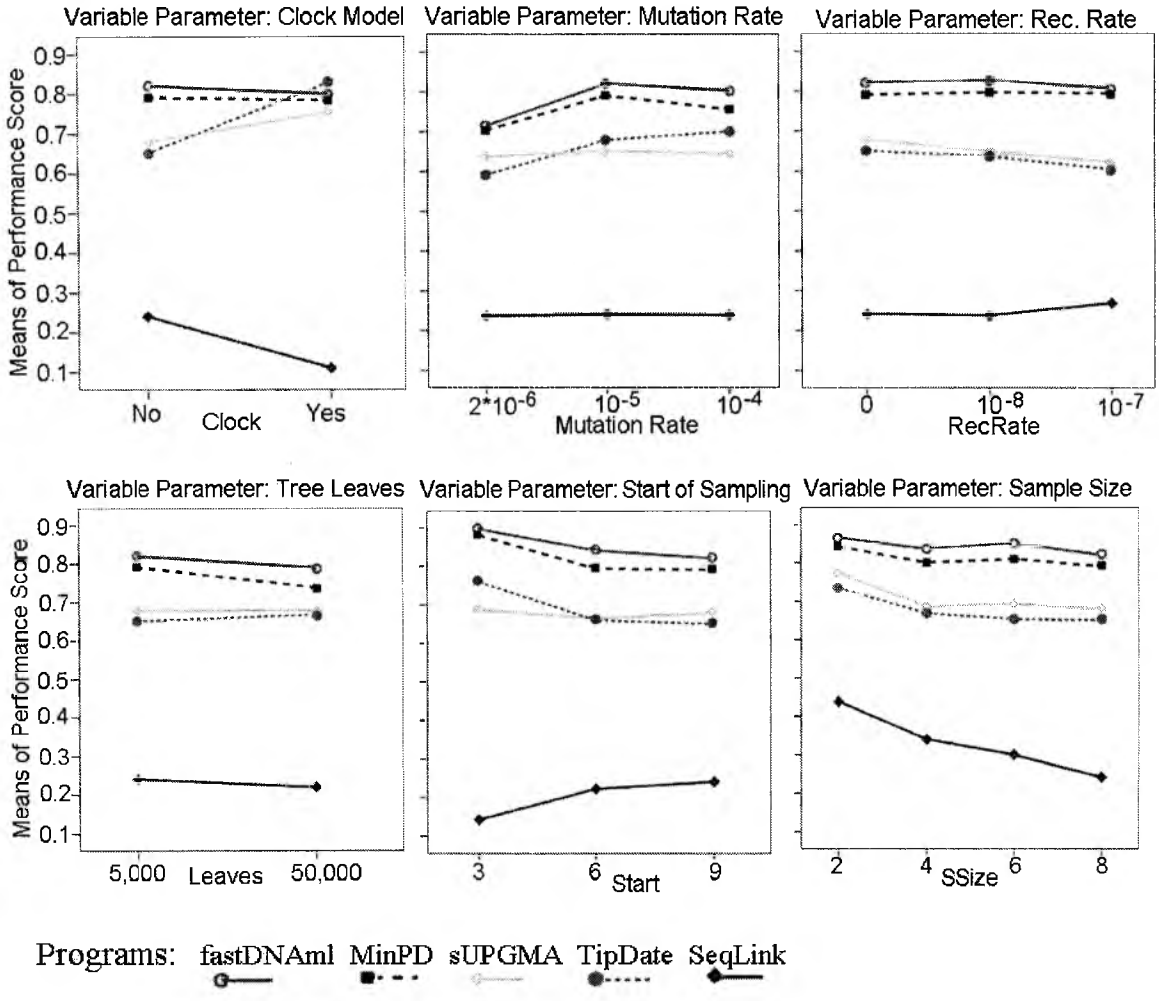


Figure 15. Graphs showing dependence of the performance scores of the algorithms on the different parameters.

Figure 15 shows the A-D performance score for the different algorithms on different selection of parameters. The analyses were based on experiments with 100 replicates generated with a modified version of Treevolve1.3.2. All parameters were fixed except the ones described in the horizontal x-axis. Wherever unspecified, parameters were fixed as follows: Mutation rate: 10^{-5}; Start: 9; Periods: 6; Clock: No; Sample size: 8; Recombination rate: 0; Leaves: 5000, Sequence Length:1000, Model HKY, T_s/T_v:4, Gamma:0.5.

5.5 Studies of the Role of Internal Node Sequences and the Molecular Clock

Since the molecular clock hypothesis is a fundamental assumption for three of the methods, we took a closer look at the effects that the molecular clock assumption has on the performance of the different programs. We applied 7 methods to the data from Sanson's experimental evolution study as the data was obtained using a neutral system of evolution (clock-enforced evolution) (Sanson et al. 2002). In addition to calculating the A-D score (the score used in the previous section), we assessed the performance of the different programs in recovering the true topology. We therefore again analyzed the Cunningham97 data set that featured heterogeneous evolutionary rates (no clock model), this time using the topological distance criterion. In the published results of both experimental evolution studies, ancestral sequences were excluded from the tree reconstruction. In our study, the complete data set consisting of both ancestral as well as terminal sequences was taken into consideration. In our simulation study, we focused on the effects of the clock parameter settings, and on the effect that the presence of internal node sequences has on the performance of the programs. , For the simulated data sets, three groups of 1000 data sets were generated, one with a constant rate of evolution, the other with a variable rate of evolution, and the third with a variable rate and sampling of all internal node sequences.

5.5.1 Empirical Data: Results

Two known phylogenies were chosen for our comparison study with the seven methods mentioned above. The 31 sequences in the Cunningham97 data set were 2733

nucleotides long, while the Sanson02 data set had 37 sequences (which included 6 additional clones of the first 70 cycles) of length 2238 bp. See the introduction section (5.1) for more details on the two experimental data sets.

Programs	Performance Scores			Topological Distance Scores		
	Sanson02	Cunningham97	Average	Sanson02	Cunningham97	Average
MinPD	100.00%	100.00%	100.00%	0.088235	0.071429	0.079832
fastDNAml	100.00%	96.43%	98.22%	0.367647	0.107143	0.237395
Dnapars	100.00%	92.86%	96.43%	0.235294	0.178571	0.206933
TipDate	100.00%	92.86%	96.43%	Same topology as fastDNAml		
BEAST	-	-	-	0.264706	0.214286	0.239496
sUPGMA PAL	96.43%	75.00%	85.72%	0.411765	0.428571	0.420168
Seq-Link	28.75%	10.71%	19.64%	0.647059	0.928571	0.787815
Average	87.50%	77.98%		0.335784	0.321429	

Table 7. Performance and Topology scores for empirical data

Table 7 shows the scores for the seven programs on the two empirical data sets. SeqLink fared poorly in both cases, which may be a result of its strong assumptions. The Cunningham97 data set was especially problematic for SeqLink because for 8 of the sequences from time period 50 there were no ancestors sampled in the previous time period (period 30).

MinPD recovered 100% of the relationships from both lineages and had the best (lowest) topological distance scores. As shown in Figure 16, MinPD displays an evolutionary framework with full lines representing genetic distances and with their extensions of dashed lines indicating linking relationships. FastDNAml performed well with the A-D score for the Sanson data set scoring 100%, but performed poorly based on the topological distance measure for the same data set. As expected the clock-based methods TipDate and sUPGMA performed better on the Sanson02 data set when looking at the A-D score, but the topology distances for sUPGMA were high. The trees output by

these two programs have branch lengths that correlate to the sampling periods, as shown in Figure 16.

DNAPARS ran for an hour on the Cunnnigham97 data set, but for less than 1 second on the Sanson02 data set. The majority rule consensus tree from the DNAPARS list of Cunnnigham output trees was used to compute the topological distance score and for the A-D score calculation the first tree of the output list was used. BEAST topological score was calculated from the majority rule consensus tree of the last 9000 trees, the A-D score could not be calculated as the consensus tree lacks branch length information.

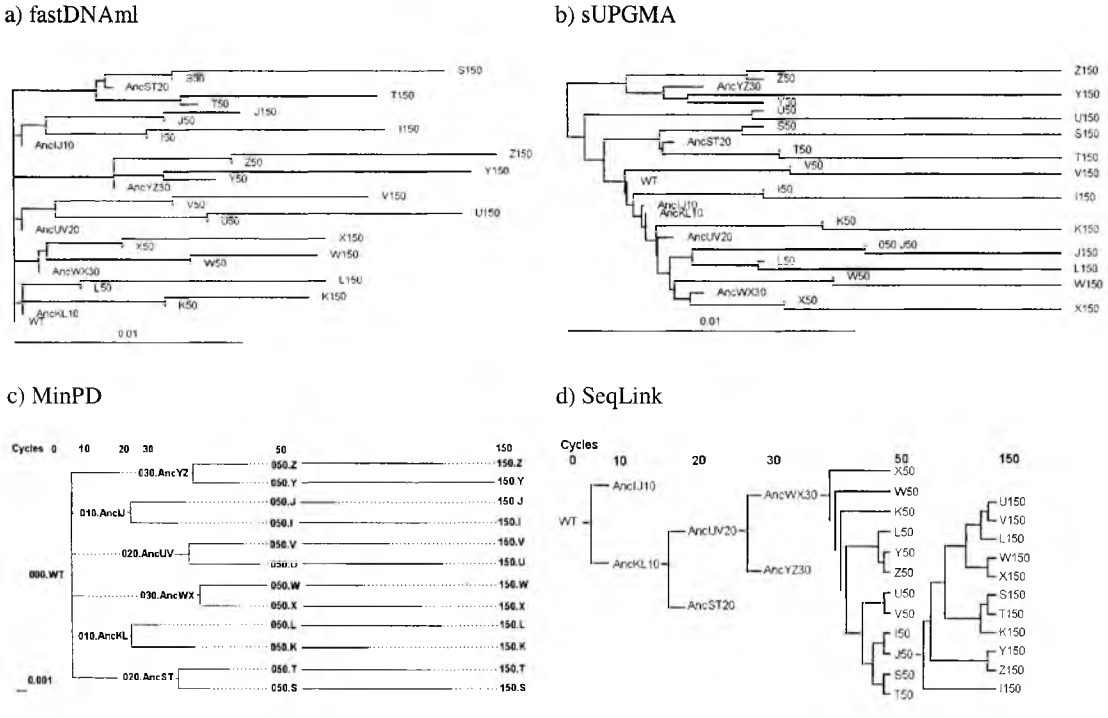


Figure 16. Comparison of *Cunningham97* trees and frameworks

5.5.2 Simulated Data: Results

Three groups of 1000 data sets containing DNA sequences were generated with Serial NetEvolve (Buendia and Narasimhan 2006). One group was evolved with a

homogeneous rate of evolution and one with a heterogeneous rate, while the third included all sequences from the internal nodes of the randomly evolved coalescent trees. These data sets were then provided as input for the six programs under consideration. As before, evaluation was based on the A-D Score and Topological Distance measure. Unlike other simulation programs the topology is not provided by the user and *Serial Treevolve* outputs the randomly generated topology. Fixed parameter settings were set to: sample size per period = 8, periods = 6, mutation rate = 10^{-5}, population size = 10^{6}, exponential rate of 0.0001, model = HKY with rate heterogeneity, $T_s/T_v = 4$, and alpha parameter = 0.5. Variable parameters included: clock = no/yes (variable rates/constant rates), and internal nodes = none/all (i.e., either none or all of the sequences from internal nodes were included), whereas when internal nodes were included, the sample size per period was reduced to 4 to keep the total sample size approximately the same. Default options were: No Clock and No internal Nodes.

Six programs were run on the three groups of simulated data sets and the A-D and topological distance scores were measured. The results were analyzed with the standard statistical software package, SPSS 13, by running 2-way ANOVAs on the scores using the *Clock* or *Internal Nodes* parameter as a second variable. Effect sizes were measured through partial eta-squared values. As variances were large and mostly overlapping, Post Hoc analysis was used to determine which differences in means were significant. The interaction effects between the program scores and the second parameter was significant. Post Hoc analysis was done using the Bonferroni procedure using a threshold P-value of 0.05.

Figure 17 shows the performance of the different methods for different selection of the clock and internal nodes sampling parameters. SeqLink underperformed consistently by a large margin with all data sets and with both comparison measures. SeqLink's strong assumptions may be the reason and will therefore not be included in the discussions below.

Ancestor-Descendant Score

Clock Parameter: For data sets generated with the non-clock model of evolution, MinPD performed better than the other two methods that were specifically designed for serially-sampled data. However, fastDNAml outperformed MinPD by 4%. The clock methods sUPGMA and TipDate improved dramatically with the clock data sets, with TipDate outperforming all programs except FastDNAml by a significant margin of 3-12% (excluding SeqLink). The interaction between program and the Clock parameter accounted for 20% variability in the A-D scores.

Internal Nodes Parameter: When sequences from all internal nodes were provided as input the score improved for all programs except sUPGMA, whose score remained approximately the same. The interaction effect size was 15%.

Topological Score

Clock Parameter: DNAPARS and fastDNAml performed well under this measure regardless of whether or not the generated sequences satisfied the clock model. MinPD's distance score was mediocre in both cases, but as mentioned previously, the program's function is to present ancestor-descendant relationships in a framework structure, which for the purpose of calculating the distance score had to be converted to a tree. The Topological distance score produced a very small effect size of 1%.

Internal Nodes Parameter: As with the empirical study results, MinPD performed well when all internal node sequences were provided, while all other programs had difficulty recovering the true topology. The effect size of 79.5% was the largest among all the interactions.

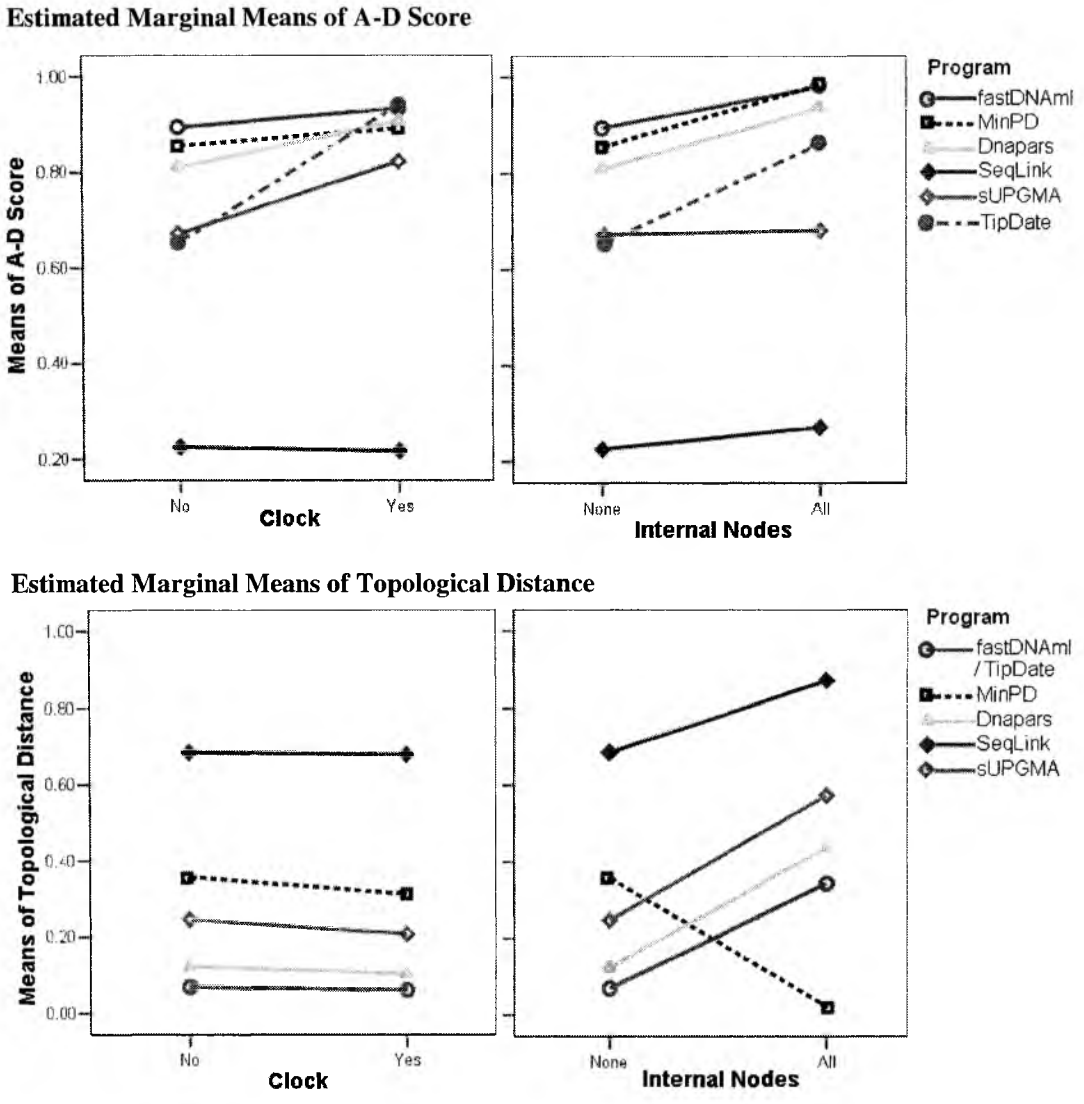


Figure 17. Graph showing dependence of the A-D Score and topology scores of the algorithms on the Clock and Internal Nodes parameters.

5.6 *Discussion*

The results show that for inferring ancestor-descendant relationships, the distance method MinPD outperforms methods specifically designed to analyze serially-sampled sequence data. An important characteristic of MinPD is that it does not require an explicit assumption of a molecular clock. The only method that performed consistently better (but rarely was the difference statistically significant) than MinPD when the evaluation was based on the A-D Score is the ML-based fastDNAml, which was designed for contemporaneous data. It is possible however, that our simulation sampling procedure may have provided a slight advantage to a traditional method like fastDNAml as most sampling started later, i.e., closer to the leaves of the final time line. By starting to sample later we tried to emulate the scenario in which sampling from a patient starts in the later stages of infection when the virus has had ample time to evolve and diverge into different lineages. Due to the similarity in performance of MinPD and fastDNAml, they were often grouped together in the same homogeneous subset by the Tukey range test procedure. MinPD is, however, several orders of magnitude faster than fastDNAml.

Analysis of serially-sampled sequence data is useful for understanding and analyzing the evolution of a virus within a single host. With the increasing ability to sample and sequence large numbers of viral particles from infected patients, it is imperative to develop fast and accurate methods to analyze the resulting sequence data. Table 8 shows the average time taken for 10 runs for each of the seven methods on an input data set with 80 sequences each of length 1000 on a Pentium 4, 2.40 GHz CPU with 512MB of RAM running Windows XP. The variance in the computation time for all the

programs was small with the exception of TipDate, whose time could range from 17 minutes to 2 hours for a set with 80 sequences, and of DNAPARS, which had an average computation time of 5 minutes with the exception of one data set which took over an hour to complete.

Program	Average Time (80 sequences)	Standard Deviation
sUPGMA	<1 second	0
Tipdate	67 minutes	52.44
BEAST (10^6 chain)	141 minutes	4.38
fastDNAml	38 minutes	4.47
DNAPARS	12.8 minutes	23.27
MinPD	<1 second	0
SeqLink	<1 second	0

Table 8. Comparison of computation times for all seven methods

The results of our simulation studies showed that certain sampling practices can adversely affect the outcome of the programs. Sampling from a larger tree increased the A-D Score error rate by about 5%. Sampling early on, before the sequences have considerably diverged, produces better-defined phylogenetic relationships. In contrast, different sample sizes did not significantly affect program A-D Score. The presence of a high rate of recombination did not increase the error rate when partial matches were counted towards the score. There is a decrease in performance for faster or slower mutation rates. As expected the two clock-based methods performed well with data generated by a clock model.

Our studies showed that fastDNAml, DNAPARS and MinPD outperform other programs under the A-D Score measure. As expected the clock-based methods, TipDate, sUPGMA performed well under this measure when data was evolved under the clock model, with TipDate showing the best performance.

MinPD outperformed all the methods by a significant margin under the topological distance measure when internal sequences were also provided as input. FastDNAml and DNAPARS outperformed other methods by a large margin when no internal sequences were provided as input.

The simulation studies corroborated the results of studies with empirical data, in particular the surprising topological score results. As shown by the results with the simulated data, when data sets included the most recent common ancestors for each sequence, tree-based programs failed to place them at basal positions in the tree, resulting in large topological distances from the true tree. In an interesting related study, Huelsenbeck used parsimony to analyze a simulated data set with inputs consisting of "living" taxa and an equal number of "fossil" taxa (with varying evolutionary distance from a randomly chosen direct ancestor of one of the living taxa) (Huelsenbeck 1991). Huelsenbeck concluded that, under certain circumstances, the availability of fossil taxa improves phylogenetic reconstruction, and that the improvement is greater if fossil taxa are closer to the direct ancestors of living taxa. Huelsenbeck's study however, used only small trees with a total of 8 taxa and binary character sequences of length only 100.

In our study MinPD was at an advantage when internal sequences were included in the input as it uses the sampling time information to infer ancestors. We hypothesize that traditional phylogenetic methods that are not restricted by the molecular clock, such as for example parsimony and maximum likelihood, may show an improved performance in their ability to recover the correct topology from serially-sampled data, when sampling times are taken into account and incorporated into their algorithms.

5.7 *Summary*

In this study we investigated and evaluated the performance of seven methods in inferring ancestor-descendant phylogenetic relationships from serially-sampled data in order to achieve goal number 5. We studied two significantly different empirical data sets of experimentally evolved sequence data: The Sanson02 data set consisting of 37 DNA sequences and the Cunningham97 data set consisting of 31 T7 phage sequences, both obtained from known phylogenies produced in the laboratory. We also experimented with simulated data sets to better understand the effect of contrasting evolutionary rate hypothesis and the presence of sequences from internal nodes on the performances of these seven methods. The comparisons were based on two different kinds of scores, the A-D Performance Score, a distance-based measure that evaluates the inference of ancestor-descendant relationships, and the standard Robinson Fould's Topological Distance Score.

The simulation studies corroborated the results of studies with empirical data, in particular the surprising topological score results. As shown by the results with the simulated data, when data sets included the most recent common ancestors for each sequence, tree-based programs failed to place them at basal positions in the tree, resulting in large topological distances from the true tree.

We did not perform comparisons studies on recombination detection in this chapter, as many methods do not support that feature. Due to the lack of standard recombination evaluation criteria we will discuss and propose new comprehensive evaluation criteria in the next chapter.

6 Searching for Recombinant Donors in a Network of Serial Samples

With at least five new recombination detection methods published in the last year, the growing list of over 40 methods suggests that the field of recombination detection is generating a lot of interest. A majority of the methods identify breakpoint positions and putative parental/donor sequences with corresponding probability values. The evaluation procedure of three previous comparison studies did not measure how many sequences were correctly identified as recombinant, or how many breakpoints and donors were correctly identified for the recombination events (Martin *et al.* 2005a; Posada and Crandall 2001a; Wiuf et al. 2001). This chapter addresses goal number 4 and seeks to establish the criteria and mechanisms to perform comprehensive comparison studies on recombination detection tools and phylogenetic tools for serially-sampled recombinant data.

Determining the evolutionary history of a sequence can become quite complex when multiple recombination events are part of its past. Detection of recombinant false positives is a major problem of most evaluation methods. In this chapter we will present an algorithm that identifies breakpoints and donor sequences in a given recombinant network of serially-sampled data. The algorithm, *RecIdentify*, scans the nodes of a randomly evolved phylogenetic network and uses its edge lengths and topology to identify the parental/donor sequences and breakpoint positions for each query sequence. *RecIdentify* findings can be used to evaluate the output of recombination detection programs. *RecIdentify* may also assist in understanding how network size and complexity

may shape recombination signals in a set of DNA sequences. The results may prove useful in the phylogenetic study of serially-sampled viral data with recombination events.

6.1 Introduction

Modeling and detection of recombination is receiving increased attention, as is evidenced by the long list compiled at a popular website (Fan and Robertson 2006). Recombination plays an important role in the evolution of genes and genomes; more importantly, it has a deleterious effects on the accuracy of phylogenetic reconstruction (Posada and Crandall 2002; Schierup and Hein 2000a). Some programs only determine the presence or absence of recombination, without trying to infer recombination breakpoints (Worobey 2001). Such a yes/no answer might be sufficient to decide which sequences to remove from a data set that will be used to infer a phylogenetic tree, thus justifying the evaluation process in many comparison studies of recombination detection methods (Martin *et al.* 2005a; Posada and Crandall 2001a; Wiuf et al. 2001). The more sophisticated programs, however, attempt to detect recombinant signals and donor sequences for the "query" sequences (Lole et al. 1999; Martin et al. 2005b; Strimmer et al. 2003).

Recombination has been largely ignored in the study of evolution due to the lack of practical methods that infer and reconstruct recombinant networks. Another reason is rooted in the belief that recombination may be disregarded when using certain genes such as mitochondrial genes or genes from the Y-chromosome, which were thought not to recombine. However, a recent study has shaken these assumptions (Tsaousis *et al.* 2005). The study made use of the automated recombination detection methods of the package

RDP2 (Martin et al. 2005a; Martin et al. 2005b) to find evidence of the presence of recombination in mitochondrial DNA data sets used in published papers.

Detecting recombinants in a set of input sequences is a difficult problem, and the performance of existing methods with regard to the accuracy of identification of recombinants, donors and breakpoint positions is not known. In this chapter, we consider the simpler problem of identifying the recombinants when the input is a set of serially-sampled sequences and also includes the recombinant network. We show that this seemingly simpler problem is non-trivial. We point out that an efficient solution to such a problem will result in a tool that would be extremely useful for performing experiments with recombination detection.

In this chapter, we discuss a new approach, *RecIdentify*, which seeks to determine the donor sequences and breakpoint positions of a set of simulated serially-sampled sequences that were evolved along a randomly generated network. The algorithm scans the nodes of the network and uses its topology and edge length values to identify the donor sequences and breakpoint positions for each queried sequence. We discuss the conflicts that arise when such networks are large and contain multiple recombination events. The networks for our experiments were generated by the software Serial NetEvolve 1.0 (Buendia and Narasimhan 2006), which attempts to emulate the evolution of recombining viruses such as HIV. The results also shed light on the effects of multiple recombination events on recombination detection of sequences from fast evolving pathogens. Hence, the temporal nature of the data will have a clear influence on the algorithm's strategy. While the observations made here also apply to contemporaneous

data, it is the temporal distribution of the data that will best explain the prevalence or absence of recombination signals in a given sequence.

The reader is referred to the section titled "Notations and Definitions" for precise information on the terminology used in this chapter.

6.2 The RecIdentify Algorithm

Algorithm *RecIdentify* reads in a recombinant network of nodes, some of which are sampled. For each sampled sequence, referred to as the query sequence, it identifies donor nodes from an earlier sampling period.

Input: A recombinant network with edge lengths representing the evolutionary history of a set of serially-sampled nucleotide sequences (sequences are not part of input), along with information on which sequences were sampled and at what sampling period.

Output: Identification of recombinants, breakpoints, and donor sequences from earlier sampling times, for each sampled sequence. A donor sequence is the closest ancestral sequence for some part (or the entire length) of a sampled descendant sequence.

The algorithm *RecIdentify* is composed of two phases.

Phase 1: In a preliminary bottom-up traversal of the network, for each node (both sampled and unsampled), the closest sampled descendants for each sampling time are computed and stored in a list, *DescendantsInfo*.

Phase 2: Subsequently, the sequence at each sampled node is classified as being recombinant or not and its donor sequence and breakpoints are identified. This is achieved by traversing the network from each (sampled) query node up to the root and by inspecting the *DescendantsInfo* list of the sibling node for every node on the path to the

root. The inspected *DescendantsInfo* lists contain all the candidate donors of the query node.

6.2.1 Notations and Definitions

The Network

A recombinant phylogenetic network of serial samples is a directed acyclic graph/network, in which nodes are associated with nucleotide sequences and assigned to sampling times (numerical values) and edges are associated with length values. The direction of every edge is from parent node to child node. As defined below, the nodes in the network may be sampled or unsampled, and either tree nodes or recombinant nodes. When no recombination events are present, the network contains only tree nodes and is a binary tree.

Figure 18 shows an example of a recombinant network with sampled nodes shown as filled circles. Sampled nodes are assigned to one of 3 sampling time points: 3, 6 and 9.

Network properties

- **A network node** may be sampled or unsampled, and is associated with a nucleotide sequence. All sequences in the network are assumed to have the same length.
- **Sampled nodes** are assigned to a sampling time and are identified by an ID whose prefix indicates the sampling time point.
- A **tree node** has only one parent node and at most two children.

- A **recombinant node** is a node at which a recombination event takes place. It has a left and a right parent node and a corresponding breakpoint position. A recombinant node represents the evolutionary event of recombination, in which one part of the sequence (i.e., left of the breakpoint) is inherited from one parent (referred to as the left parent), and the other part (i.e., right of the breakpoint) is inherited from the other parent (referred to as the right parent).
- A **recombinant sequence** is a sequence associated with a node (either tree or recombinant node) resulting from one or more recombination events in its history.
- **A leaf node** is a node with no children nodes.
- **Edge Distance:** The distance value associated with an edge of the network represents the amount of evolutionary divergence between the two incident nodes.

A phylogenetic network is completely specified by its topology and the set of all edge lengths.

A path in a recombinant network

Assume that node x is one of the parents of recombinant node y in the network. Then, depending on the breakpoint, node x is a donor for only part of the sequence at node y. Thus every edge in the network from a parent to a child is associated with a portion of the sequence for which the parent is a donor. Extending this notion, we define a **path** from node v to w in the network to be a sequence of nodes (or edges) along with a specified portion of the sequence (given by a range, i.e., a start point and an end point, in the sequence). More precisely,

$$\text{path}(v,w,s) = (P, s)$$

is an ordered pair with a sequence of vertices $P = \langle n_1,..n_p \rangle$ (with the first and last entries of the sequence being v and w respectively), and an interval range s = [start,end] indicating a start and end point of the sequence linking the nodes v and w.

The nucleotides under consideration are given by the interval range s = [start,end] often flanked by breakpoints on one or both sides. If the given recombinant network has no recombinant nodes, i.e., the given network is a tree, every path is associated with the entire sequence. In a recombinant network, there may be more than one sequence of nodes from a node v to node w, and each must be associated with a disjoint interval range, i.e., portion of the sequence. For example, in Figure 18, path(a,w,[0,700]) = (‹a,b,f,3.O,w›, [0,700]) and path(a,w,[701,sLen]) = (‹a,p,u,3.V,w›, [701,sLen]) are two distinct paths covering disjoint interval ranges of the sequences. Note that the breakpoint is between the 700-th nucleotide and the following one, and sLen is the length of all the sequences. Also note that paths are not directed and may use edges directed either way. In other words, when dealing with paths, we simply ignore the directions of the edges. For example, in Figure 18, the path from node 9.Y to its closest sampled donor, 3.V, involves an edge in both directions.

In Figure 18, sampled nodes are represented by filled circles, unsampled nodes by open circles. Each node is labeled in alphabetical order in the reverse of the order in which the node is processed during phase 1. Each sampled node is identified by a sequence name that starts with a numerical prefix indicating the sampling time point, followed by a dot and an uppercase letter. Recombinant nodes have two parents and are shown with two thick edges leading into them. The numbers above them indicate the corresponding breakpoint position.

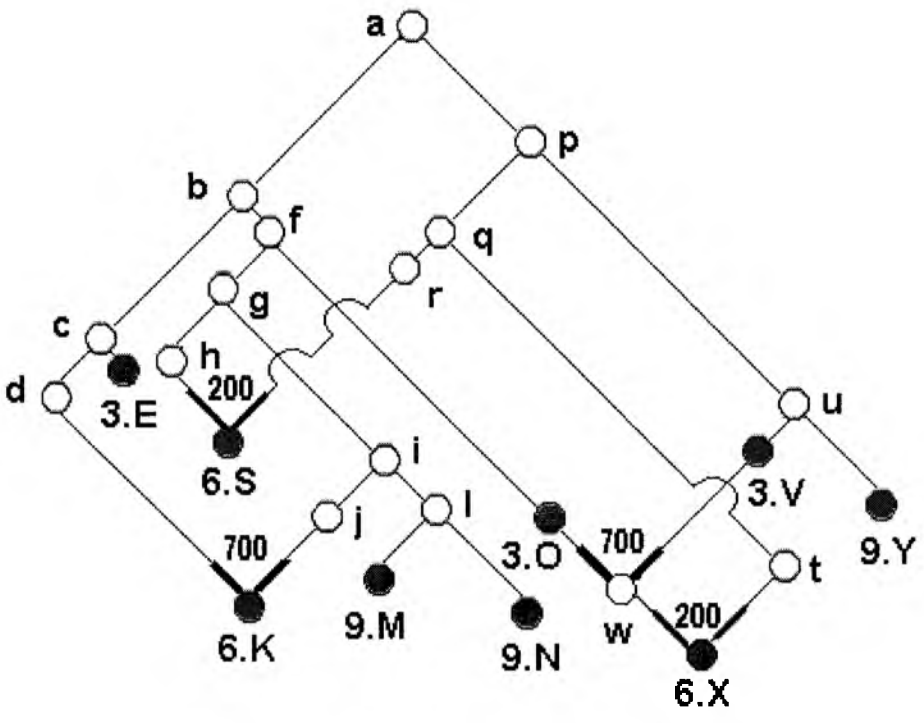


Figure 18. Recombinant network of serially-sampled data with 4 recombination events

6.2.2 Donor and Recipient Lists

As mentioned before, a sampled node with recombinant nodes among its ancestors may have any number of donors. The *DonorList* can be used to store the list of donors of any given node. It stores information about all the donor sequences and the associated breakpoints. The data structure is easily implemented as a linked list of network nodes. The list contains only one network node if only one donor has been identified for the entire sequence. If different nodes have been identified as donor nodes for different parts of the sequence, then the donor list is an ordered list of donors along with the corresponding breakpoints. The *RecipientList*, structured exactly like the DonorList, is used to store a recipients list for any given node. For node x, it stores a list

of all nodes for which x could be a donor (along with the associated breakpoints). Note that such a list is used in Phase I of the algorithm below.

In order to write down the donor or recipient list, we introduce the RDL notation in this chapter and write it as follows:

$[S_1|B_1|S_2|B_2|\ldots|B_{n-1}|S_n]$.

If it denotes a donors list of node x, it expresses the recombinant history of node x as an ordered list of n sampled donor sequences and n-1 breakpoints. S_1 and S_n are the leftmost and rightmost sampled donor nodes, while S_i is the i^{th} sampled donor node. Some nodes may have no donors, or it may have parts of its sequence with no donors. For this purpose, we introduce the concept of a NULL node, which is represented by a dash "—". A NULL node is used to indicate that no donor has (yet) been found for some (or all) portion of a sequence.

For example, during phase 2 (described later) when searching for the donor list of node 9.M, the algorithm traverses upward to the root. When it is at node j, based on the information collected from the subnetwork reachable from node i, the donor is: [—|700|6.K], implying that no donor has yet been found for the first 700 nucleotides. However, when node b is reached, based on the information collected from the subnetwork reachable from node b, the donor list of node 9.M contains the complete information: [3.E|700|6.K]

In Figure 18, *RecIdentify* identifies sequence 6.X as a recombinant sequence using all three of the available distance criteria. Using the RDL notation, the parents of

sequence 6.X can be written down as [3.O|200|6.S]. The sequence at 9.Y is identified as a non-recombinant with [3.V] as the sole donor sequence.

6.2.3 Ancestor Criteria

A key requirement that our method imposes is that for any given sequence, all potential donor sequences must have been sampled at an earlier sampling time point.

6.2.4 Distance Criteria

A node v is said to be closer to node u than to node w for a segment of the sequence s = [start,end], if distance(v,u,s,c) $<_c$ distance(v,w,s,c), where distance(v,u,s,c) is the length of the shortest path between nodes v and u calculated for sequence segment s using distance measure c. In order to compute distances between the sampled data, three different distance measures may be used:

- **Path length measure**: a distance measure based on path lengths,
- **Topology distance measure**: a distance measure based on number of edges on the path, and
- **Combined distance measure:** a measure that combines both path lengths and topology.

The three measures were chosen by taking into consideration the different approaches used in existing recombination detection methods (Martin et al. 2005b; Salminen et al. 1995; Siepel and Korber 1995). A majority of recombination detection methods use a sliding window approach and either pairwise sequence comparisons (path length) or phylogenetic trees to determine location in a tree (topology approach).

A sequence at node v is identified as a donor sequence of the query sequence at node q if distance(v,q,s,c) = $\min_{w \in W, v \neq w}$ distance(w,q,s,c), where W is the set of all sampled nodes from sampling times prior to t_q, the sampling time of q.

Edge Length measure

Let *Len*(*e*) be the length value associated with edge *e*. The distance between sequences at nodes v and q under the edge length criterion is given by distance(v,q,s,lengths) = $\sum_{e_i \in path(v,q,s)} Len(e_i)$. This is thus the path length measure used in traditional graph theory literature. As mentioned before, the directions of the edges on the path are ignored. Ties are broken by using the topology criterion (described below). If ties persist, they are broken arbitrarily.

Topology distance measure

The distance between sequences at nodes v and q under this measure is given by distance(v,q,s,nodes) = |P|-1, where path(q,v,s) = (P,s) is the shortest path between nodes q and v. In other words, the distance value according to this criterion is equal to the number of edges on the shortest path between q and v for the sequence segment s. Ties are broken by using the edge distance criterion. As before, if ties persist, they are broken arbitrarily.

Combined distance measure

This is simply a weighted combination of the distances according to the edge distance measure and the topology distance measure. More formally, the combined

distance is given by: distance(v,q,s,combined) = distance(v,q,s,nodes)*nodePenalty + distance(v,q,s, lengths). Here nodePenalty is a penalty imposed on the number of edges on the path.

In each of the above definitions of the distance measures, the donor sequence for the query sequence at node q is defined as the sequence associated with the node v that has the smallest distance(v,q,s,c) value among all nodes v, where c is one of the three distance measure criteria (path lengths, topology, or combined). In each case, if ties remain, a candidate donor is selected arbitrarily.

Each of the three distance measures may identify different donor sequences. For example, in the network of Figure 18, using the topology distance measure, sequence 6.K will be identified as a recombinant with two donors for two different portions of the sequence; in RDL notation, it may be written down as [3.E|700|3.O]. Note that the node 9.M is from a later time period and cannot be considered as a donor node. The edge distance measure, however, will only identify one donor sequence, [3.E], as it has a smaller edge distance from the right-hand donor (node j) than the sequence at node 3.O. Depending on the nodePenalty value, the combined distance measure could identify the same donor as that with the edge distance measure or that with the topology distance measure. Note that an entirely different donor (i.e., other than the ones identified using the path length or the topology measure is possible (depending on the value of nodePenalty) when the combined measure is used.

6.2.5 The DescendantsInfo list

For the following discussion, let c be one of the three distance measures defined above. With respect to a specific node x and a sequence segment s, we say that $v <_c w$ if x is *closer* to v than to w according to a distance measure c, i.e., distance(x,v,s,c) < distance(x,w,s,c). We also define the relation $\leq_s$ as follows. We say that $v \leq_s w$ if v was sampled no later than w. Finally, we say that node v *dominates* node w, if $v <_c w$ and $v \leq_s$ w. In other words, v dominates w with respect to node x, if v is closer to x and was sampled no later than w.

The *DescendantsInfo* list is an ordered list associated with each node, with one entry for each sampling time. It is sorted by sampling times, and for each sampling time, it contains, for each segment of the sequence, the closest sampled descendants of the node. More precisely, for node x, each element in its associated DescendantsInfo is a RecipientList structure and contains the closest sampled descendant node for each segment (i.e., between two successive breakpoints) of the sequence of x. Besides the identity of the closest node, it also stores the distance to the closest node using the path length measure and the topology distance measure. (Note that the combined distance measure is not stored since it can be computed from the other two measures.) For example, for the network in Figure 18, the *DescendantsInfo* list for node g is [6.S|200|-|700|6.K] for sampling period 6 and [9.M] for sampling period 9.

The elements of the *DescendantsInfo* lists will be used to compute the donor lists containing the recombination history of any query sequence in Phase 2. Thus, if unsampled node x has two children v and w, then the DescendantsInfo list associated

with node v contains potential donors of query nodes in the subnetwork rooted at w, and vice versa.

For tree networks, we know that the RecipientsList has only one node. The DescendantsInfo structure can be further simplified using a simple rule, i.e., **the dominance rule** for two sampled nodes v and w with respect to an ancestor node x and a sequence segment s, which is stated as follows:

if $(v <_c w)$ ***then***

if $(v \leq_s w)$ ***then*** *discard w,*

else *retain w, but after v.*

In other words, the dominance rule discards all dominated nodes from the *DescendantsInfo* list of x and orders the rest by increasing distance from x. Note that if a node x is sampled, its *DescendantsInfo* list contains only one item and that is itself.

As a straightforward consequence of the dominance rule, we observe that the *DescendantsInfo* list of every node in a tree network is sorted in the order of increasing distances and decreasing sampling times.

6.2.6. Phase 1: Storing candidate nodes in the *DescendantsInfo* list

In phase 1, the *DescendantsInfo* list at each node in the network is computed in a bottom-up manner using a simple Depth-First Search (DFS) traversal. Figure 18 shows each node labeled by the order in which it is first visited by this DFS traversal. The objective of phase 1 is to compute, for each node v, a list of closest sampled descendants from each sampling time. This information will be made use of in Phase 2 in the following manner. If a node v is a sibling node of some node in the path from a node x to

the root, then all nodes in v's *DescendantsInfo* list that were sampled prior to x, are potential donors of x. The *DescendantsInfo* list of a node v is computed by a process of "merging" the *DescendantsInfo* lists of its children. This merging procedure is described below in detail.

Merging Procedure

During the DFS traversal, the *DescendantsInfo* list of an unsampled leaf node is initialized to null and that of a sampled leaf node is initialized to contain only one Recipient list, which in turn has only one entry, i.e., itself. For all other nodes, the *DescendantsInfo* list is computed by merging the *DescendantsInfo* of the left and right children of the node. Note that the merging process will merge the RecipientLists from the same sampling period from the two *DescendantsInfo* lists being merged. Before discussing the general case, we discuss two special cases of merging.

Case 1 (Merging at a recombinant parent node): Assume that the current node is a recombinant parent node with only one child. Furthermore, assume without loss of generality, that the parent is a right recombinant donor with breakpoint B. Then the merging process will only retain the part of the *DescendantsInfo* list that is appropriate to the portion of the sequence for which the recombinant parent is the donor (i.e., to the right of breakpoint B). In other words, a NULL node will be added to the left of B in every RecipientList of the *DescendantsInfo* structure of that node, while the portion of the RecipientList to the right of B will be copied over.

Case 2 (Merging at nodes with nonrecombinant children): If a node has two non-recombinant children, then the *DescendantsInfo* structure of that node can be considerably simplified by applying the dominance rule.

Case 3 (Merging Recombinant RecipientLists): The general case of the merging process merges the two *DescendantsInfo* structures (from the two children), each of which is a list of RecipientList structures (one for each sampling time). The merging process merges the RecipientList structures for the same sampling period and is fairly straightforward. The breakpoint list of the merged list is the union of the breakpoint lists of the individual RecipientLists. For each resulting segment (i.e., between successive breakpoints) of the sequence, the merge process retains the closer of the two sequences one from each list. Note that distances are computed according to one of the three predefined distance criteria discussed earlier. In the end, if the closest sequence is the same to the left and right of a particular breakpoint, then the breakpoint is removed and the list is shortened. One additional point to note is that the *DescendantsInfo* structure also stores the distance information. Before merging the two data structures, the distances from the parent to the two children need to be added before the distances are compared and stored.

For example, in Figure 18, node j is a recombinant parent with only one child. Thus, Case 1 is applied resulting in the RecipientList [-|700|6.K]. Case 2 can be applied at node l. Here the edge distance of 9.N is larger, but the topology distance is the same. However in this case the edge distance is applied to break the tie, even if the topology distance is the chosen criteria. The general case 3 is applied at node g, where the

RecipientLists denoted by [6.S|200|-] and [-|700|6.K] are merged resulting in a list denoted by [6.S|200|-|700|6.K]. Note the NULL node in the mid section of the sequence

Unusual example: It is worth noting that in a recombinant network, not every sampled ancestor of a recombinant node is a donor, because it is very much dependent on the location of the breakpoints. The proposed algorithm does take care of this unusual scenario. For example, for the section of a network shown in both Figures 19 (a) and (b), because of the location of the breakpoints, the sequence at node A is not a donor for the sequence at node C. In fact, the *DescendantsInfo* of node A will be empty if node C is a leaf node. In Figure 18, the *DescendantsInfo* list of node 3.V is empty although it appears to be an ancestor of a sampled node 6.X

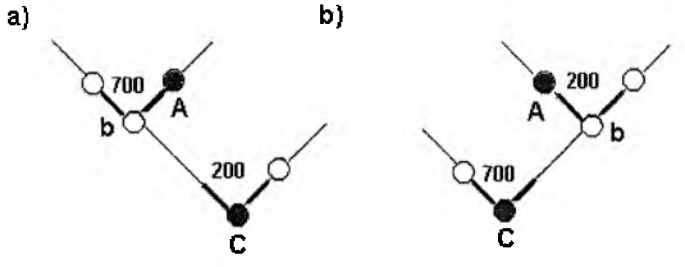


Figure 19. Examples of how subsequent recombination events can affect the *DescendantsInfo* list of node.

6.2.7. Phase 2: Identification of donor nodes

During this search phase the closest donor sequences have to be determined for each sampled query sequence. The resulting DonorList structure is returned as the answer. When searching for the donors of a query sequence, the DonorList is initialized to the empty list. Then the network is traversed along all paths from the query node to the

root. At every node, the DonorList is "merged" with all RecipientLists from the *DescendantsInfo* list of their sibling nodes (if any) after adding the distance from the node and its sibling to their parent. Only entries that are at a closer distance than the current answer stored in DonorList are merged. Care is taken to only use those RecipientLists that correspond to sampling times earlier than that of the query sequence. The search traverses a simple path if no recombination events are encountered along the way; otherwise the search process splits up. Whenever two of these paths meet, the DonorLists are merged again. If we assume that the network has a unique root, then all the paths will meet. The process is stopped when the root of the network is reached or the distance traveled in the network exceeds the distance of the best answer stored in DonorList. Note that the merging process described in the previous section is now being reused in this phase, albeit with the modification that RecipientLists from all relevant sampling periods are merged. Also note that if a node along the path is a sampled node itself, then that node is also merged into the DonorList and the search process aborted for the path with the sampled node. Details of the code are provided in Figure 20.

Special Cases

During Phase 2 the donor sequences for each "query" sequence is determined. A complex network structure may obscure recombination signals and lead to ambiguous and incorrect results. Recombinant sequences may be identified as non-recombinants, while non-recombinants may get identified as recombinants. Figures 21 (a) and (b) show examples of both these cases. In both figures, the sampled query node is marked with the label Q. In Figure 21 (a), no recombination event is present in the path from Q to the root.

```
GetClosestDonor(current node, sibling node, answer DonorList)
If current node is NULL
        Then return          // it's beyond the root so stop
If current node is sampled
        Then merge current node with answer
Else If sibling is not NULL
        For all RecipientLists in DescendantsInfo of sibling node
                If sampling time is earlier than query sampling time
merge RecipientList with answer according to distance criteria
    If current node not sampled
        If current node is recombinant
If c
                Then    GetClosestDonor(leftParent of current node, NULL, left answer)
                        GetClosestDonor(right Parent of current node, NULL, right answer)
                        attach left answer to the left of right answer before current node's bkp. position
   Else
                        GetClosestDonor(left parent of current, sibling of current, answer)
Else
                    GetClosestDonor(left parent of current, sibling of current, answer)
```

Figure 20. Algorithm GetClosestDonor

A non-recombinant query sequence Q may nevertheless be identified as a recombinant with donor list: [A|200|B|700|C], since the closest sampled sequences from a previous sampling period (nodes A and C) are descendants of recombinant parents. In contrast, in Figure 21 (b), Q that will be identified as a non-recombinant with B as the only donor and will not be identified as a recombinant sequence since the closest sampled node (node B) is not a recombinant.

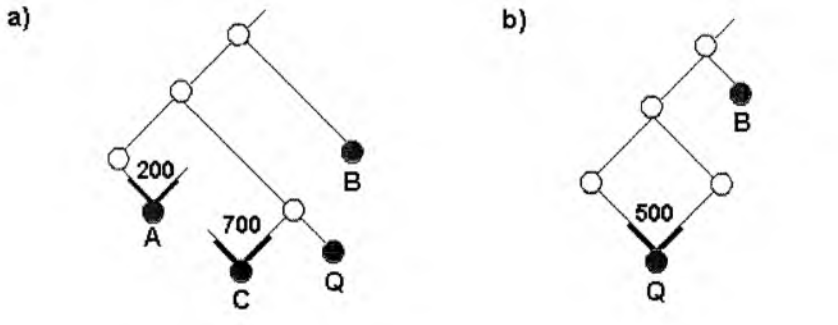


Figure 21. Examples of ambiguous sequence identification

6.2.8. Time Complexity

Phase 1

The merge process at each node merges t pairs of lists, where t is the number of sampling times. Each list can, in the worst case have as many breakpoints r as the number of recombination events in the network. If n is the number of nodes in the network, then DFS-traversal of the network, which performs O(n) merges runs in time O(trn). Note that the number of edges in the network is linear in n (since each node has at most two edges leaving and at most two edges entering. Also, note that because of recombination events, the number of nodes in the network can be arbitrarily larger than the number of leaves or number of sampled nodes in the network.

In the best case scenario, n = s , all nodes are sampled nodes. Time complexity is O(n) because "every" sampled node contains itself and only itself in its DescendantsInfo list, therefore t=1 and r=1.

Phase 2

GetClosestDonor procedure is called for each sampled node that is not from the first sampling time point. In the worst case, this could take O(trn) for each sampled sequence and O(trsn) overall. However, this time can be improved, when the search is carried out for all sampling times at once instead of for each queried sequence. During the search the result per sampling time is stored at each visited node in the path. The search will start at the leaves as before but will stop whenever it encounters a node that contains the result for a path that has already been searched previously. Since, with this

improvement, the processing (i.e., merging) has to be performed at most once at each node, the time complexity as in Phase 1, is O(trn). In Figure 18 the answer for 9.M: [6.S|200|3.E|700|6.K] will be stored in node l. Then the answer for node 9.N will be found after traversing only one node upward towards the root.

Again in the best case, n=s, and the time complexity is O(n) because the donor will be found after 1 step. The donor is the sampled parent node.

6.3 Discussion

Given a set of aligned, serially-sampled sequences and a simulated network that describes the phylogenetic relationships between the sequences, the questions we ask in this chapter are how to classify the sequences into recombinant or non-recombinant sequences and how to identify donors and breakpoint positions, while using an approach that is compatible with the methodologies of published recombination detection tools. We sought to determine the kind of recombination events that complicated the determination of the past history of a sequence and how this complication affects the recombination detection process.

A recombinant network of serially-sampled data, explaining the true history of the data set, was used as the input structure. The edge lengths in the network used represent the number of substitutions per site, but do not represent time (i.e., the molecular clock model was not assumed). Recombination events in the network appear as two edges merging into one recombinant node, as opposed to tree edges that split from one node into two edges. Thus, there are tree and recombinant edges in the network. An example of such a network can be seen in Figure 18

Distance and topology approaches in the determination of donor sequences and breakpoints

For each (query) sequence the algorithm *RecIdentify* determines if the sequence has one or more donor sequences among the sampled data. Breakpoint positions are also determined for recombinant sequences. The identification process is heavily dependent on the chosen distance criteria, which in turn is inspired by the two main classes of recombination detection methods. A number of recombination detection methods, including RIP (Siepel and Korber 1995), use genetic distance or pairwise sequence comparisons to identify recombinant sequences; others, like the popular bootscanning method, use phylogenetic trees and rely solely on tree topology to obtain their results (Lole et al. 1999; Salminen et al. 1995). To conform to these two approaches, three distance criteria are used in this chapter: one based on path length, a second based on topological distance (network position), and a third measure based on a combination of both path length and topology.

The path length measure and the topology measure tend to agree whenever all the edges in the network are of roughly the same length. If this is not the case, the path length measure identifies fewer sequences as being recombinant, and finds fewer breakpoints as it discards donor sequences positioned at the end of longer edges. Ties resulting from the use of one measure may be broken with the help of another measure. This approach reduces the number of sequences identified as recombinants, especially when topology ties are broken by applying the distance measure.

The distance and topology measures return results that often differ in the choice of donor sequences and breakpoint position. The combined measure is based on the path lengths with a penalty for the number of nodes traversed. Like the topology measure it tends to identify more sequences as recombinants, since it is less discriminating about distances and ends up with more ties.

Based on our experiments, we conclude that the combined measure may be a more realistic measure.

Effects of complex phylogenetic network structures

A phylogenetic network that describes the evolutionary history of a set of recombinant taxa may obscure the actual historical evidence if too many recombination events are dispersed throughout the network or if subsequent recombination events are located in the same path from the root to a sampled sequence. Some of the more remarkable cases are briefly discussed here and presented in more detail in the Methods section.

Recombinant sequence may be identified as a non-recombinant when a sequence at a recombinant node has unsampled left and right parental nodes and the closest sampled donor node is an ancestor of both the unsampled parents (see Figure 21b).

Non-recombinant sequences may be identified as recombinant if a non-recombinant query sequence has different sequences identified as donors for different parts of the sequence, but no recombination event took place on the path from the root to the node corresponding to the query sequence. This case is less intuitive than the previous case, and occurs when the closest donors of the query node are sequences from subtrees

(with recombination events) of siblings of its unsampled ancestors. Recombinant nodes only inherit a segment of the sequence from each parent node (see methods section), therefore they can only be identified as donors for the segment corresponding to the path that joins one parent node and the query node. (See Figure 21(a) for an example.)

The closest sampled nodes may not always be donor candidates of a query sequence. If the path from the closest sampled node to the query sequence contains more than one recombination event, the query sequence may not have inherited any part of the sequence from the sampled node. For example, in Figures 19 (a) and (b), A is the closest sampled ancestral node with respect to C, but it in both cases it is not a donor for any part of the sequence associated with C.

The cases discussed above make it clear that the identification of donors in a given recombinant network is nontrivial and requires a careful algorithm that recognizes the evolutionary implications built into such a structure.

The testing procedure

In a more complex network with a larger number of sampled sequences, the identification of recombinant sequences, donor sequences and breakpoints ceases to be trivial. We proposed a special algorithm, *RecIdentify*, to carry out these tasks. The resulting implementation can be conveniently used to evaluate the performance of recombination detection methods.

Most recombination detection methods only offer graphical outputs from which the user is left to decide whether or not a given sequence is recombinant. New methods

are being developed that replace the visual output with a list of the inferred recombinant sequences along with statistical significance values associated with the inference.

We also investigated the effect of the application of the three distance measures to identify closest sampled donor sequences on the output of recombination detection methods. Our findings show that more sequences are identified as recombinant in a phylogenetic network, when the topology measure is chosen. This leads us to believe that recombination detection methods that are based on tree topology alone may be identifying more sequences as recombinants, perhaps leading to false positives.

The algorithm presented assumes that all the sequences associated with the nodes in the network are of the same length. While this makes the discussion convenient, it is not a necessary assumption. If the sequences are not of the same length, then a global multiple alignment could provide the reference positions required in the algorithm

The program Serial NetEvolve 1.0 (Buendia and Narasimhan 2006) creates a randomly generated coalescent network of serially sampled sequence data, and outputs its structure and the alignment of the sampled sequences. Sequences from both internal and terminal nodes may be present. Sequences are assigned to different sampling time points. The network output of Serial NetEvolve may be analysed with *RecIdentify* and the results compared to the results obtained by recombination detection methods that analysed only the set of aligned sequences. A comprehensive comparison study of this kind would consist of comparing the donors and breakpoint matches in addition to computing the number of true positives and false positives and other performance measures computed

using these values. While such a testing procedure is outside the scope of this current chapter, it lays the groundwork for new comprehensive comparison studies.

6.4. Summary

We presented an algorithm that classifies sequences from a phylogenetic network of serial samples into two categories: recombinant or non-recombinant. For a recombinant sequence the donor sequences and breakpoint positions are also identified. The objective of the identification procedure is to use the results to achieve an evaluation of recombination detection methods that is more comprehensive than in previous studies (Martin et al. 2005a; Posada and Crandall 2001a; Wiuf et al. 2001) as described in Goal 4 of this dissertation.

The different cases that are encountered in a complex recombinant network during the identification of donors require an algorithm that recognizes the evolutionary implications built into such a structure. We have learned that a complex recombinant history may obscure the phylogenetic signals of the data if only a few sequences are available/sampled. Navigating the network in search of recombinant donors for a query sequence involves setting up criteria for deciding when a taxon is to be called a donor. Clearly, the donor must have been sampled at an earlier time period; however, the most influential criterion is related to the distances between sequences in the network. The distance criteria chosen in the *RecIdentify* algorithm were inspired by the two main classes of recombination detection methods, those based on sequence comparison (genetic distance) and those based on phylogenetic network topology. It is evident from

the findings related to the choice of distance criteria that the selection of recombinant donors hinges greatly on the underlying distance measure used.

In the next chapter (section 7.3.2) we will present and discuss the results of a comprehensive comparison study that uses the evaluation criteria proposed in this chapter to compare recombination detection methods.

7 Sliding MinPD

This chapter addresses the goal of developing computational tools to automatically detect recombination in serially-sampled data and displaying the recombination events within an evolutionary network structure as specified in Goal 2. It also addresses Goal 1 as it presents an improved version of the preliminary MinPD version discussed in Chapter 4. Another goal addressed in this chapter is Goal 6, i.e., *the evaluation of methods to study serially-sampled recombinant sequence data through extensive computer simulation studies.*

Traditional phylogenetic methods assume tree-like evolutionary models and are known to perform poorly when provided with sequence data from recombining, fast-evolving viruses. Furthermore, these methods assume that all the sequence data are from contemporaneous taxa, which is not valid for serially-sampled data. A more general approach is needed — one that is mindful of the sampling times of the input sequences and that reconstructs the viral evolutionary relationships in the form of a network structure with implicit representations of recombination events. The underlying network organization may reveal unique patterns of viral evolution and could help explain the emergence of disease-associated mutants and drug-resistant strains, with implications for patient prognosis and treatment strategies. The method we developed, referred to as Sliding MinPD, reconstructs evolutionary networks of serially-sampled sequences by combining minimum pairwise distance measures with automated recombination detection based on a sliding window approach. The method was tested using simulated data and was also applied to a set of serially-sampled HIV sequences from a single patient. The

type of recombinant networks output by the Sliding MinPD method are referred to as *serial evolutionary networks*, and are a generalization of the evolutionary framework introduced by Holmes et al. in his 1992 study (Holmes *et al.* 1992).

7.1 Introduction

As described in section 2.2 of this dissertation, RNA viruses exploit numerous genetic and evolutionary mechanisms to ensure their survival; some of these include high mutation rates, high yields (increased replication), and short replication cycles that produce a diverse array of mutants. For some RNA viruses, enormous numbers of progeny are produced after a single cycle of replication and almost every new genome differs from every other (Flint *et al.* 2000f). This implies that because RNA viruses evolve fast, when their sequences are sampled at time intervals of a year or less, they already exhibit several substitution patterns in contrast to what one would observe in slow-evolving genes from higher organisms. It is this aspect of their evolution that demands for special attention and a different approach when studying their evolutionary relationships. RNA viruses are known for their role in a number of diseases, including respiratory infections by the SARS virus, and liver disease caused by Hepatitis C virus. Retroviruses are a class of RNA viruses and have received a lot of attention in the last twenty years, mainly due to the role of one of its notorious members, HIV-1, in the AIDS epidemic.

It is common practice to use traditional phylogenetic methods in the study of viral evolution (Franco *et al.* 2003; Rambaut *et al.* 2004). Traditional phylogenetic analysis has also been applied to HIV-1 sequences sampled serially from the same patient over a

period of several years (Shankarappa *et al.* 1999). It is necessary however to point out that such methods were conceived for the study of contemporaneous data. Furthermore, these standard methods do not consider the effects of genetic recombination in shaping the evolutionary relationships. Recombination is a critical mechanism in the evolution of HIV-1 and it is thought to help the virus escape from immune pressures and adapt to the effects of antiviral therapies.

An ever-growing number of recombination detection tools have been published in recent years. A comprehensive, up-to-date list of over 40 programs can be found at http://bioinf.man.ac.uk/recombination/programs.shtml. A few of the programs in that list were included in two comparison studies carried out in 2001 (Posada and Crandall 2001a; Wiuf et al. 2001). The two largest groups of recombination detection methods fall into two categories of which we picked one each to implement in Sliding MinPD. One group of tools, which includes the online tool RIP (Siepel and Korber 1995), uses pairwise sequence comparisons; another large group uses phylogenetic trees and the popular bootscanning method, and is implemented in Simplot and RDP2 (Lole *et al.* 1999; Salminen *et al.* 1995). Two different classes of methods that were included in the study by (Posada and Crandall 2001a) were (a) compatibility methods that test for partition phylogenetic incongruence on a site-by-site basis, and (b) nucleotide substitution distribution methods that examine the sequences for a significant clustering of substitutions or fit to an expected statistical distribution. The bootscanning and RIP methods were chosen to be incorporated in Sliding MinPD as they represent the earliest and most popular approaches on which many of the newer methods have been built upon. A modification of the bootscanning method was published more recently (Martin *et al.*

2005a), as was a method called RAT (Recombination Analysis Tool), which is comparable to the RIP method, but uses genetic distances (Etherington *et al.* 2005). A third method that combines both methods was implemented in RDP2 (Martin *et al.* 2005b) and also incorporated in Sliding MinPD. The goal of these methods was to identify the putative recombinant ancestor sequences of a query sequence from a set of aligned sequences. A total of three existing methods were implemented in Sliding MinPD and adapted to the requirements of the input data by automating the determination of the ancestral donor sequences from previous sampling times for different parts of a sequence and for each query sequence not from the first sampling time point.

The Sliding MinPD strategy allowed us to construct an evolutionary network, which is a modification of the Holmes et al. evolutionary framework (Holmes et al. 1992). In a 1992 study, Holmes et al. created an "evolutionary framework" to express the inferred ancestor-descendent relationships in HIV sequence data that was serially sampled from a single patient. Holmes et al. stated that "unlike most molecular phylogenies, real ancestors may be present in the data and the framework expresses the postulated ancestor-descendent relationships" (Holmes et al. 1992). Sliding MinPD constructs an "evolutionary network" that represents recombination events and ancestor-descendant relationships using an approach that emulates and automates the recombination detection process. This method may be applied to fast-evolving, recombinant pathogens such as HIV-1, to better understand the evolutionary history of the virus and how this history correlates with selective pressures, the emergence of drug-resistant strains and disease progression.

Several methods that estimate the phylogenetic relationship of serially-sampled data have been published since 2000 (Buendia and Narasimhan 2004; Drummond and Rambaut 2003; Drummond and Rodrigo 2000; Ogishima *et al.* 2001; Rambaut 2000; Ren *et al.* 2001); also see Chapter 4. The performance of the different methods (including a preliminary version of MinPD) was compared in the study reported in Chapter 5 (Buendia et al. 2006a). Of the programs in the comparison study, only MinPD took recombination into account although the feature was turned off in the study cited above. Sliding MinPD is an improvement of the original MinPD with several significant modifications, including the use of a sliding window approach, a choice of three different recombination detection approaches (based on distance or topology comparisons), and the option of calculating statistical significance of the predictions (in terms of bootstrap values).

7.2 Methods

Sliding MinPD combines a minimum pairwise distance approach with automated recombination detection to study the ancestor-descendant relationships of serially-sampled nucleotide sequences. Our method presents the results in an evolutionary network structure that respects the time order of the sampled data, represents genetic distances and linking relationships, and indicates recombination events and breakpoint positions.

7.2.1 Recombination Detection in Sliding MinPD

The standard recombination detection methods rely on a visual approach in which it is left to the user to decide (by looking at the graphical output) if a sequence is a recombinant and to determine the ancestral donor sequences and location of the breakpoint positions. In Sliding MinPD the identification of recombinants, ancestors and breakpoints is automated with no need for user input. Three existing methods, all of them using the **sliding window** approach, were implemented as user options in Sliding MinPD. The basic approach is the same for all three methods: a sliding window is moved along the aligned input sequences, and at every position the query sequence is compared to each of the background representatives with the goal of finding the "closest" (or most similar sequence) using an appropriate scoring measure. After the window has traversed the alignment from left to right, for any pair of sequences, a plot of the measure of closeness between them can be obtained as a function of the position in the alignment. A comparison of the plots involving the query sequence reveals which background representative the query sequence most resembles at any given position. Recombination breakpoints can be found at the intersection of the appropriate plots. The difference between the three methods lies in the scoring mechanism used, as described below.

- Recombination Identification Program (RIP): Similarity between two sequences is quantified as the percentage of identical base pairs (Siepel and Korber 1995). Sliding MinPD does not use a similarity score, but instead uses a corrected distance measure that incorporates rate heterogeneity and substitution patterns to correct for the estimation bias of counting only mismatches among sites.

- The standard Bootscanning method (SB): Bootstrapped phylogenetic trees are built for each window segment and finally the bootstrap value for placing the query sequence with each of the reference sequences/sequence groups is tabulated and plotted along the sequence. It requires a minimum of 4 sequences (Lole et al. 1999; Salminen et al. 1995). In Sliding MinPD, Neighbor-Joining trees are constructed from the bootstrapped corrected distance matrices calculated during the RIP process (see above). The pairwise sequence position within the tree is stored in a topology distance matrix. Note that the resulting distances are dependent on which other sequences are in the reference set.

- The distance Bootscanning method (B-RIP): This alternative approach to the standard Bootscanning was implemented in RDP2 (Martin et al. 2005b). Here only the bootstrapped corrected distances are calculated and plotted in the graph (instead of constructing the trees and calculating the position within the trees). Note that the resulting distances are dependent on which other sequences are in the reference set.

All three methods have been previously implemented elsewhere. The two bootscanning methods were implemented in RDP2 (Martin et al. 2005b). The recently developed RAT implements a RIP method using genetic distances (Etherington et al. 2005). The significant modification introduced in Sliding MinPD is the automation of these methods, thus eliminating user interaction to identify recombinant sequences and to determine the ancestral donor sequences and breakpoints. The automated identification of recombinants is necessary for the reconstruction of the evolutionary network. The performance of these automated determination process is tested extensively on simulated data and will be discussed in detail in the results section of this chapter.

Sliding MinPD has three phases. In the first one, every sequence that is not from the first sampling time point is deemed a query sequence and the pairwise distances (corrected distances or topological distances) are calculated for every pair of sequences for the entire length of the sequences (step [1] of algorithm). This step is required for the identification of only one ancestral donor for the whole length of a sequence. This donor is chosen if in a later step the sequence is identified as being a non-recombinant. The process is straightforward for the RIP option as it involves calculating a distance matrix for the current input of aligned sequences. For B-RIP and SB, however, the process involves creating bootstrap replicates of the entire sequence alignment and calculating a distance matrix for each replicate (step [1b] of algorithm). When the SB option was selected, NJ trees have to be constructed for each replicate and the positions of the sequences have to be stored in a topology distance matrix (step [1c] of algorithm). In the second phase, the ancestors and breakpoints for the recombinant sequences are automatically determined. It is in this phase that a query sequence is identified as a recombinant or not. The same procedures from phase 1 are carried out, but this time for every window along the alignment (steps [2a-b] and [3a-c] of algorithm). For non-recombinants the sequence at minimum genetic distance is chosen as the ancestor sequence. In the final phase of the algorithm the evolutionary network is constructed (step [4] of algorithm).

An evolutionary network has both tree edges and network edges. Tree edges are edges that link a single ancestor sequence to a descendant query sequence (indicating that it is the closest ancestor for the whole length of the sequence). Network edges are edges

that join two or more ancestor sequences with one descendant query sequence and represent a recombination event.

7.2.2 Algorithm

Algorithm *Sliding MinPD*

1. **For each** pair of sequences s_i and s_j in input S calculate distances for the whole alignment
    a. **if** *RIP* option, **then** compute the distance matrix Dist(s_i,s_j)
    b. **if** *SB* or *B-RIP* option, **then** bootstrap input S. For each bootstrap replicate S_b, compute the distance matrix Distb(S_b,s_i,s_j).
    c. **If** *SB* option, **then do**
    use Distb(S_b,s_i,s_j) to create NJ trees per bootstrap replicate S_b and store topology distances in distance matrix Distb(S_b,s_i,s_j).
2. **For each** pair of sequences s_i and s_j slide a window along the alignment, and for every window wx **do**
    a. if *RIP* option, then do step [1a] for window wx and compute distance Distw(s_i,s_j,wx)
    b. else if *SB* or *B-RIP* option, then do step [1b] for window wx and compute average distance in Distw(S_b,s_i,s_j,wx)
3. **For each** query sequence s_i **do**
    a. **if** (s_i is judged to be a recombinant) **then** identify its closest ancestor sequences and the corresponding breakpoints.

b. else choose as ancestor of s_i the sequence at minimum distance from it among sequences sampled at all previous times.

c. If *SB* or *B-RIP* option, **then** identify the associated bootstrap value.

4. **For each** set of sequences with the same chosen ancestor, construct a NJ tree with the chosen ancestor as the outgroup.

In Step [3a] above, the automated recombination detection test is performed on all query sequences **s** that are not from the first sampling time point. In what follows, we assume that there are w different possible positions for the sliding window as it slides along the alignment.

***Sliding MinPD* Recombination Detection Test for query sequence s**

1. **For each** of the w windows of s, select the sequence that is closest to it in that window. Let s_i be the sequence chosen from the i^{th} window. Sequence s_i is said to **dominate** in the i^{th} window. Put all selected sequences in a list called **Candidates**. (These sequences are candidates for being potential ancestors of s.)
2. **For each** pair of sequences s_i and s_j in Candidates
    a. Construct distance vectors for each sequence s_i. In other words, the distance vector of s_i contains the distance of every window of s_i to the corresponding window of s.
    b. if the Pearson Correlation Coefficient (PCC) between the distance vectors of two sequences s_i and s_j is above a distance threshold, **then** discard the

sequence s_i or s_j whichever has the larger average distance in the two windows where the two sequences dominate.

3. **If** *SB* or *B-RIP* option, **then do**

   a. **for each** s_i from Candidates do pair it up with **s** and **do** steps [1b-c] from the main Sliding MinPD algorithm after replacing the reference set S by the set Candidates.

   b. **for each** window wx, **do** calculate and store bootstrap values $Boot(Candidates_b,s_i,s,wx)$.

   c. discard from the set Candidates all sequences s_i with short bootstrap spikes or spikes below a given threshold (to reduce number of false positives).

   d. **If** any sequences were discarded, **then** redo steps [3a-b] with the updated Candidates.

   e. **for each** breakpoint **bkp**, **do** calculate average bootstrap value for the corresponding combination of left and right ancestor sequences $val(s,s_i,s_j,bkp)$. **If** the highest average bootstrap value for combinations of left and right ancestor sequences $val(s_i,s_j,bkp) > Boot\_threshold$) **then** identify **s** as recombinant with s_i and s_j as the ancestor sequences with breakpoint **bkp.**

4. **if** *RIP* option, **then**

   a. **For each** of the w-1 breakpoints **bkp** and all sequences in **Candidates do** calculate minimum average distance value for combinations of left and right ancestor sequences $val(s_i,s_j,bkp)$.

   b. **If** ($min(val(s_i,s_j,bkp)) < RIP\_threshold$) **then** identify **s** as recombinant with s_i and s_j as the ancestor sequences with breakpoint **bkp.**

Steps [1] and [2] identify sequences that are at minimum distance in each of the w windows. If two sequences have very similar distance vectors (from s), then it is probably because they are very similar. In this case, one of them is discarded in Step [2b], since either one of them would serve the purpose. Similarity of two distance vectors is computed using the Pearson Correlation Coefficient, which is high if the vectors are highly correlated (i.e., similar). Step [3] is for the SB and B-RIP options only. In step [3a-b], the steps [1b-c] from the main algorithm are carried out for each window and the pool of candidates for being recombinant donors is reduced once more in step [3c-d].

When the bootscanning options SB or B-RIP are selected, then an intermediate step [3c] is needed to discard false positives with short bootstrap spikes or weak bootstrap spikes. An increase in the number of false positives was observed when such sequences were left in the candidate pool. A bootstrap spike is defined as a segment of consecutive windows in which a given sequence has a higher bootstrap value than the other sequences. If the spike lasts less than 100 nucleotides (or the number of windows that covers this length), it is probably an artifact of the bootstrap procedure. If the spike is not high enough, then the sequence can't be considered a candidate for recombinant ancestor.

In step [3e] the combinations of left and right ancestor sequences with a high average bootstrap value that exceed the bootstrap threshold are identified as recombinant donors and the query sequence is identified as a recombinant. Step [4] is carried out only for the RIP option. In step [4], the combinations of left and right ancestor sequences with a minimum distance that is less than the RIP threshold are identified as recombinant donors and the query sequence is identified as a recombinant.

As an alternative procedure to the standard bootstrap, a procedure that we called the "bootknife" was also implemented. The same procedure was implemented in RDP2 (Martin et al. 2005b). With the bootknife a percentage p of the sites (25% to 50%) is picked at random and removed and is replaced by other randomly picked sites. The remaining sites (roughly half of the sites) are left untouched.

7.2.3 Detecting Multiple Breakpoints

The algorithm we have presented above will find recombination events with only one breakpoint. (Note that we use the terms "breakpoint" and "crossover point" interchangeably). We propose a method based on the weighted interval scheduling algorithm (Kleinberg and Tardos 2005) to find recombination events with one or more crossovers. We define a chain of windows to be a set of consecutive windows that would be obtained if one were to slide a window over a portion of the sequence. If a chain of windows corresponds to an interval, and if its average bootstrap value over the interval corresponds to its weight, then it is easy to see that the problem of finding an optimal set of (multiple) breakpoints corresponds to the problem of finding an optimal set of weighted intervals. Each chain has a start point, the first window position, and an end point, the first window position plus the nucleotides covered by the length of the chain. The weight per chain is either the average modified distance (RIP) or the average bootstrap value (SB or B-RIP) of the chain. With more crossovers, more spikes are possible, and therefore a penalty is assessed to short spikes. The penalty decreases when longer chains are chosen and is subtracted from the average bootstrap value. This helps to keep the number of false positives at a minimum. The *modified* distance for the RIP

option is modDist = *Largest*-Dist, where *Largest* is the largest distance value of all windows. This allows for a standardized algorithm that can be applied to all three options (in which the maximum distance is preferred). Section 4 of the Sliding MinPD recombination detection test will perform the following steps:

4. **For each** of the w-1 breakpoints **bkp,** with w the number of windows, and all sequences in **Candidates** do
    a. Prepare chains that end at bkp. There are m=n*bkp chains c_i that end at bkp, with n the number of sequences in **Candidates and** $1 \leq i \leq m$.
    b. Calculate maximum average modified distance or bootstrap value for bkp as M[bkp] := max(weight(c_i) + M[start(c_i)]) .

The position start(c_i), where a chain c_i starts, is a position where another chain ends, and is therefore an earlier breakpoint position. The range of the inspected breakpoints goes from 1 to w-1. M[start(c_i)] contains the best result up to breakpoint start(c_i). The penalty function p is designed to decrease the weight in the following manner:

weight = weight - weight *p.

The penalty is a function of the length of the chain and tends to zero for larger chains as its purpose is to diminish the effect of spikes. We therefore propose to use the following exponential penalty function

$$p = \frac{(a-1)^x}{a},$$

where x is the length of the chain of consecutive windows c_i and $a = \log_2 w$.

7.3 Sliding MinPD Results

The performance of Sliding MinPD was tested with simulated sequence data generated by the program Serial NetEvolve 1.0; which was described in Chapter 3. Serial NetEvolve is a simulation tool that generates random recombinant networks and evolves serially-sampled nucleotide sequences along the network. Seven data sets of 100 replicates were generated, each data set with different rates of recombination. Sliding MinPD was run repeatedly on the data sets with different combinations of parameters (window size, step size for the sliding windows, choice of detection method, PCC value threshold, bootstrap value threshold, and choice of crossover option).

7.3.1 A-D Network Score

The evaluation process for Sliding MinPD is based on the recombination detection principle of identifying parental sequences for a given query sequence. In accordance with the concepts used in the two main types of recombination detection methods (Posada and Crandall 2001a), the ancestor-descendant evaluation score, referred to as the A-D Score, also uses two different measures: one based on path length, the other based on topological distance (network position). For a query sequence q in the true network, the closest parental sequence s was determined using two different distance measures as follows:

- Topology measure: the number of nodes traversed from q;
- Path length measure: the total length of the path traversed from q.

If two sequences are at the same distance from q, then the alternative criteria is used as a tie-breaker. For example, if two sequences are at the same distance from q with

respect to the topology measure, then the distance with respect to the path length criterion is used to break the tie.

The A-D score was calculated as the percentage of correctly inferred ancestor-descendant tuples (q, s_1, ... , s_w), where s_1,...,s_w are the closest ancestors of query sequence q for different portions of the sequence. If q is non-recombinant then there is only one ancestor s_1 for the whole length of the sequence, and if Sliding MinPD identifies q as a non-recombinant with only one ancestor a_1, then it is counted as a true negative (TN). If s_1 matches a_1, the match is counted towards the TN A-D score. If q is a recombinant sequence and is identified as such by Sliding MinPD, then it is counted as a true positive (TP). false positives (FP) and false negatives (FN) are determined correspondingly and express the association between the presence or absence of recombination. The A-D score goes further and calculates the percentage of ancestors identified by Sliding MinPD that match the true ancestors. The A-D Score for the other values are calculated in the same way as that of the TN A-D score described above. The BKP Score is calculated as the number of times a breakpoint is identified within a distance of n nucleotides to the left or right of the true breakpoint position. The default interval size n was set to 60.

7.3.2 Analysis of Simulation Study

Serial NetEvolve 1.0 was used to generate data sets with 7 different recombination rates (see graphs). The sequence length was set to 1000, the model of evolution to HKY, with the rate heterogeneity alpha parameter set to 0.5, mutation rate of 0.00001, internal node sampling rate of 0.5, and exponential rate of 0.0005. Breakpoints

were added uniformly between the positions 150 and 850, the breakpoint margins. A sample size of 8 sequences per sampling time was chosen with 6 sampling time points, which resulted in a sequence alignment containing 48 sequences, of which 8 were from the first sampling period. Sliding MinPD was then used to identify the recombinants among the 40 query sequences and to determine the closest ancestors for parts of or the entire length of the sequence.

Specificity (SP), sensitivity (SE) and positive predictive values (PPV) were calculated for each combination of the parameters used for Sliding MinPD. The sensitivity values ranged between 0.5 and 0.7, with an increase in the number of false positives for the larger sensitivity values. The goal of the simulation study then was to find the settings for which Sliding MinPD could obtain the highest specificity and positive predictive values, albeit at the expense of lower sensitivity values. The parameters at which the program performed best were found to be different for different recombination detection options.

Table 9 shows the results for the default parameters of the 3 recombination detection options. The default parameters were chosen from the results of the simulation study as the parameters for which Sliding MinPD performed the best.

Rec. rate	Option	TN	TP	FN	FP	SP	SE	PPV	TN A-D Score	TP A-D Score	FN A-D Score	FP A-D Score	BKP (60) Score
LOW	BRIP	9635	1359	923	83	0.991	0.596	0.942	0.895	0.653	0.862	0.855	0.480
	SB	9725	1301	861	113	0.989	0.602	0.920	0.795	0.596	0.698	0.726	0.467
	RIP	9637	1358	924	81	0.992	0.595	0.944	0.896	0.658	0.858	0.926	0.514
HIGH	BRIP	7284	2467	2161	88	0.988	0.533	0.966	0.916	0.588	0.879	0.795	0.380
	SB	7396	2421	2063	120	0.984	0.540	0.953	0.814	0.537	0.712	0.758	0.371
	RIP	7303	2532	2096	69	0.991	0.547	0.973	0.914	0.599	0.882	0.855	0.414

Table 9. Benchmark results of the simulation studies

The default values for SB and B-RIP were: 100 bootstrap replicates, a window size of 200, a step size of 20, and a seed of -3. The default PCC threshold values were set to 0.4 for B-RIP and RIP, and 0.2 for SB. The default bootstrap thresholds were 88 for SB and 90 for B-RIP. The default values for the RIP option were: a window size of 100 and a step size of 30. All three options used the TN93 distance and an alpha parameter of 0.5. Six different recombination rates were used, namely $1x10^{-8}$ (10%), $2x10^{-8}$ (18%), $3x10^{-8}$ (27%), $4x10^{-8}$ (30%), $5x10^{-8}$ (39%), and $6x10^{-8}$ (44%); the parenthesized percentage values correspond to the rough percentage of sequences that were recombinant. The first three recombination rates were classified as being "Low", while the last three were classified as "High". The results for these recombination rates are shown in Table 9. The values of TN (true negatives), TP (true positives), FN (false negatives), and FP (false positives) were obtained as described previously at the end of section *A-D Network score*. The TN A-D (resp. TP A-D) Score is the percentage of true negative (resp. true positive) ancestors that matched the ancestor(s) identified by Sliding MinPD. FN A-D Score is the percentage of FN ancestors matched and is calculated when Sliding MinPD identifies only one ancestor for a sequence that is recombinant (FN). FP A-D Score is calculated correspondingly (see also previous section on the *Network A-D Score*). The BKP Score gives the percentage of breakpoints that were correctly identified (a margin of error of 60 nucleotides was allowed for the breakpoint position predictions).

The graphs in Figure 22 show various measures of performance for different choices of the parameters. Based on the results, the best combination of parameters was compiled for the analysis of the empirical data (see Chapter 8). The evaluation procedure of previous comparison studies consisted of measuring the number of data sets for which

the presence or absence of recombination were correctly identified (Posada and Crandall 2001a). These procedures did not measure how many sequences were correctly identified as recombinant, or how many breakpoints and donors were correctly identified for the recombination events. The evaluation study we carried out is therefore unique in that it attempts to go beyond a simple pass or fail test for the three methods that were implemented in Sliding MinPD.

Program options and evaluation criteria: The three different program options as well as the 2 different evaluation criteria were tested for different recombination rates (Figure 22a-b). Overall, the standard bootscan (SB), an option that uses phylogenetic trees to compute the answer, performed better when the topology evaluation criterion was applied on data sets with low recombination rates. A slight decrease in the specificity values could be observed when the methods were evaluated with the topology criterion, however this also caused an increase in the sensitivity scores. The sensitivity score fell dramatically for larger recombination rates. In all of the following tests, note that the SB method was always used in conjunction with the topology measure, and the RIP and B-RIP methods were used in conjunction with the path length measure.

Window and Step Sizes: Different combinations of window and step sizes were chosen for comparing the three program options (Figure 22c). The X-axis shows combinations of window sizes (top) and step sizes (bottom) ordered by increasing number of windows. B-RIP and SB showed improved performance at the default window size of 200 and step size of 20. The RIP option however was drastically affected by the choice of these parameters, with lower specificity and higher sensitivity for the smaller step sizes, and higher specificity and lower sensitivity for larger step sizes. The default

values chosen for RIP in all other studies were a window size of 100 and a step size of 30. The RIP performance can be explained by the observation that it does not generate bootstrap replicates and that therefore the distance values between windows fluctuate more for larger step sizes when the sequences are sufficiently divergent (default mutation rate: 0.00001) and the distances are not averaged over all bootstrap replicates. The study also shows that smaller window sizes negatively affected the sensitivity but improve the specificity values of all options, but specially that of SB and BRIP.

Crossovers: We tested our algorithm that is able to detect more than one crossover (recombination event) with different recombination rates (Figure 22d). The detection of multiple crossovers algorithm improved the performance of B-RIP and SB considerably, with the sensitivity increasing by as much as 5 percentage points for a small decrease in specificity; it did, however, have an adverse effect on RIP. The sensitivity values of RIP decreased considerably, while the specificity values increased marginally. The BKP scores for B-RIP and SB improved by an average of 5% and more so for the higher recombination rates (data not shown). It is possible that the weight penalty adversely affected the computation of RIP, in which no bootstraps are performed.

Bootstrap Threshold: As has been observed in many studies, an optimal calibration of the parameters is difficult because of the delicate balance between sensitivity and specificity values. The bootstrap study highlights this point very clearly (Figure 22e). With higher bootstrap thresholds the specificity increases but the sensitivity decreases. Based on this study the bootstrap threshold of 88 was chosen as the default value for SB and a threshold of 90 for B-RIP with the goal of keeping the sensitivity at a value close to

0.60 and the specificity at a value close to 0.99 for the data sets generated with low rates of recombination.

PCC Threshold: This study shows the effect of different PCC threshold values on the selection of an appropriate pool of donor candidates (Figure 22f). The three options show very different levels of performance for the different threshold values. The RIP option is highly affected by this parameter, with the specificity falling and sensitivity increasing for higher threshold values. A high threshold value reduced the pool of candidates only minimally. The SB option appeared to be at an advantage for very low threshold values, which greatly reduces the pool of candidates.

BKP Score: The BKP score is clearly affected by the choice of the recombination rate, with SB performing rather poorly at higher rates (Figure 22g). However, when the evaluation interval around the breakpoint position (i.e., the region around the breakpoint that was analyzed to perform the analysis) was increased (from 60 sites to 80 sites, and then to 100 sites), the performance of SB improved markedly and was better than that of the other methods for 100 sites. The small graph (Figure 22g) shows a comparison between all three options for different breakpoint intervals averaged over all recombination rates.

Bootstrap/Bootknife Comparison: The three options with their respective default values were tested with the standard bootstrap and with the bootknife method (described in the algorithms section). In Figures 22h-i, each of the three options was tested with the two bootstrap procedures (label is used BKN for bootknife, and BST for the standard bootstrap). Our results show that the bootknife procedure is marginally better, thus justifying its choice as the default option for the other comparison studies reported here.

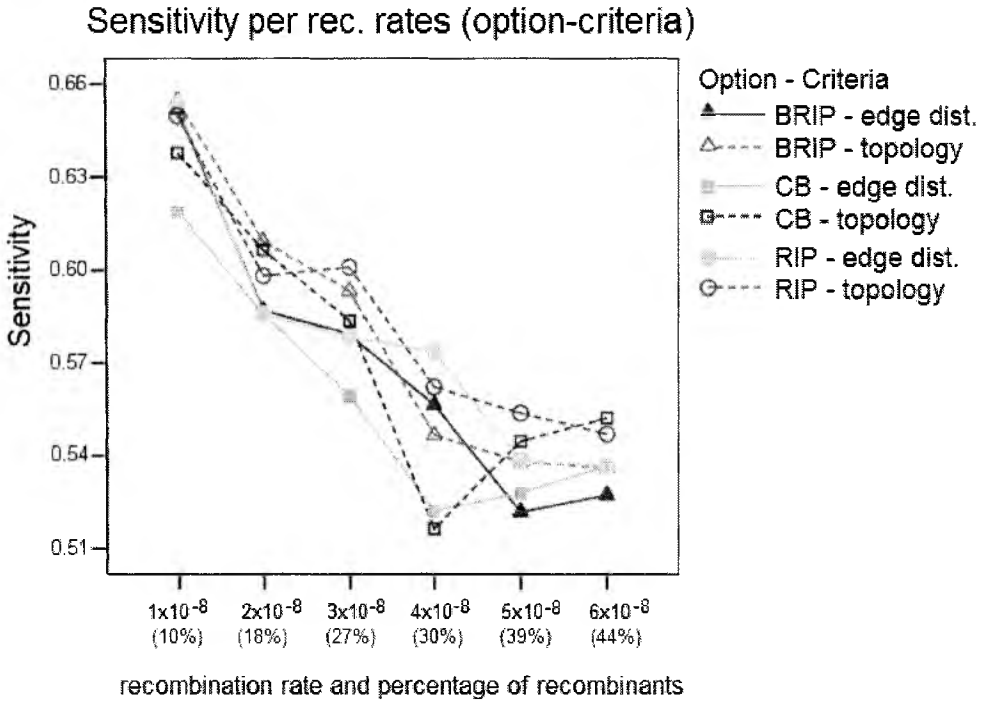
Corrected Distance Models: We compared the effect of using one of the three different distance formulas, a setting in Sliding MinPD (Figure 22j): Jukes Cantor 69 (JC69), Kimura-2-Paramter (K2P), and Tamura Nei 93 (TN93). The results show that the choice of a distance model affected the performance of the different options, with TN93 showing the best results for all three options. RIP was especially affected by the choice of distance formula. This gives evidence to the robustness of the bootstrap process, which balances out substitution effects that are misinterpreted by the simpler distance models.

Low and High Mutation Rates: This study compared the effect of simulating data with a different mutation rate, a setting in Serial NetEvolve (Figure 22k-l). Posada and Crandall used mutation rates in the range of 0.0000025 to 0.00005, while the default value of our simulations is 0.00001. Our results are comparable to the Posada and Crandall study for the low mutation rate (0.0000025). The methods perform poorly with sequence data that has low divergence. This observation with regards to low mutation rates extends to all phylogenetic tools. The option RIP was ran with a different window size of 200 and a step size of 50 as the default settings returned very poor results. This again proves that the bootstrap process (not part of RIP) imparts robustness to the detection process. Our studies showed that too high a mutation rate (0.00005) also caused a decrease in performance of the different detection options, with RIP especially affected. RIP was ran with a window size of 160 and a step size of 20, settings that are known to increase this options' sensitivity. This decrease in performance was not observed in the Posada and Crandall study, but as stated before, their study did not involve identification of recombinant sequences and donor sequences, but only measured the presence or

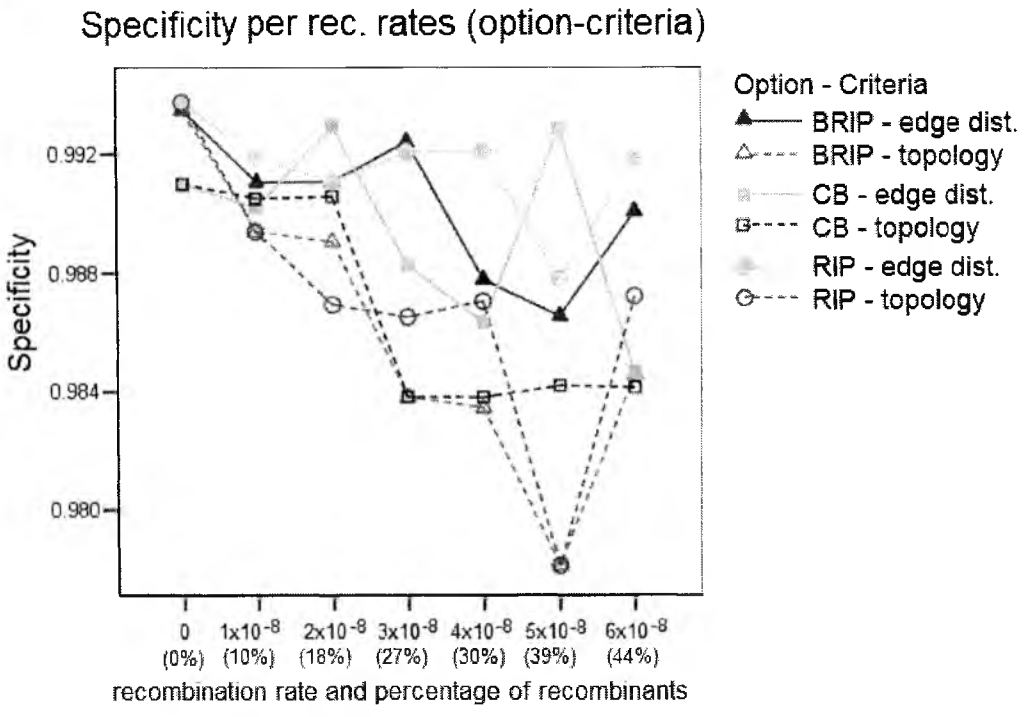
absence of recombination in a data set. The two studies are therefore not easily comparable.

Rate Heterogeneity Parameter: This study (Figure 22m) compared the effect of simulating sequence data with a homogenous mutation rate across all sites (alpha parameter = ∞) to that of simulating sequence data with a gamma rate heterogeneity (alpha parameter =0.5). The rate heterogeneity parameter is a setting in Serial NetEvolve, not in Sliding MinPD. To evaluate the power of the different recombination tests, Posada and Crandall (Posada and Crandall 2001a) generated data with homogenous rates, which as our results show, improve the recombination detection performance of the different options, with the specificity, sensitivity and positive predicative values clearly improving.

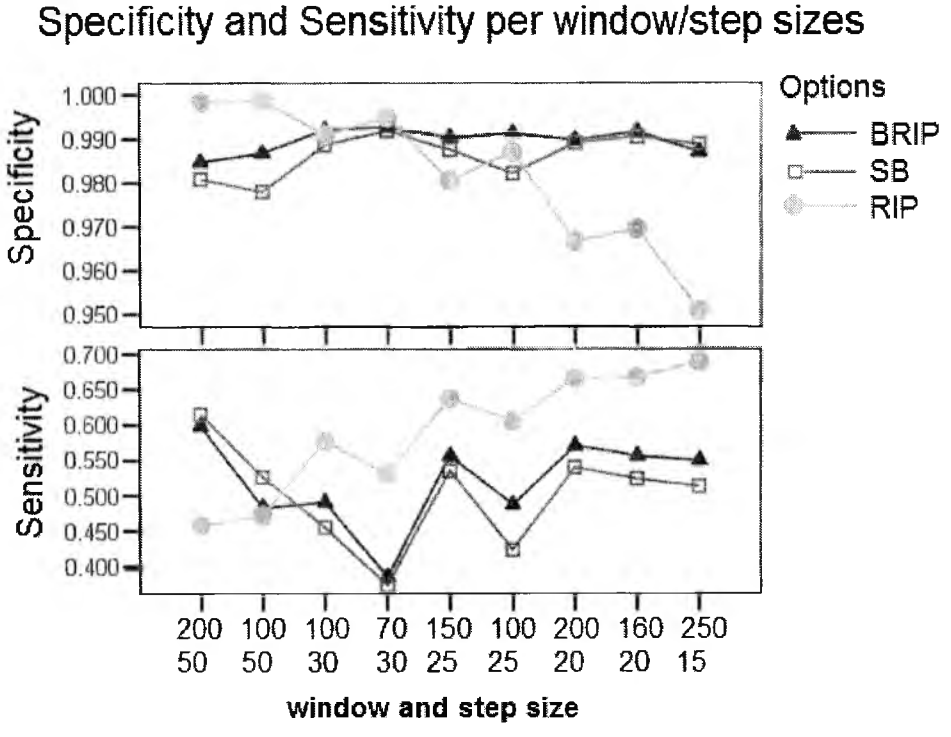
a)



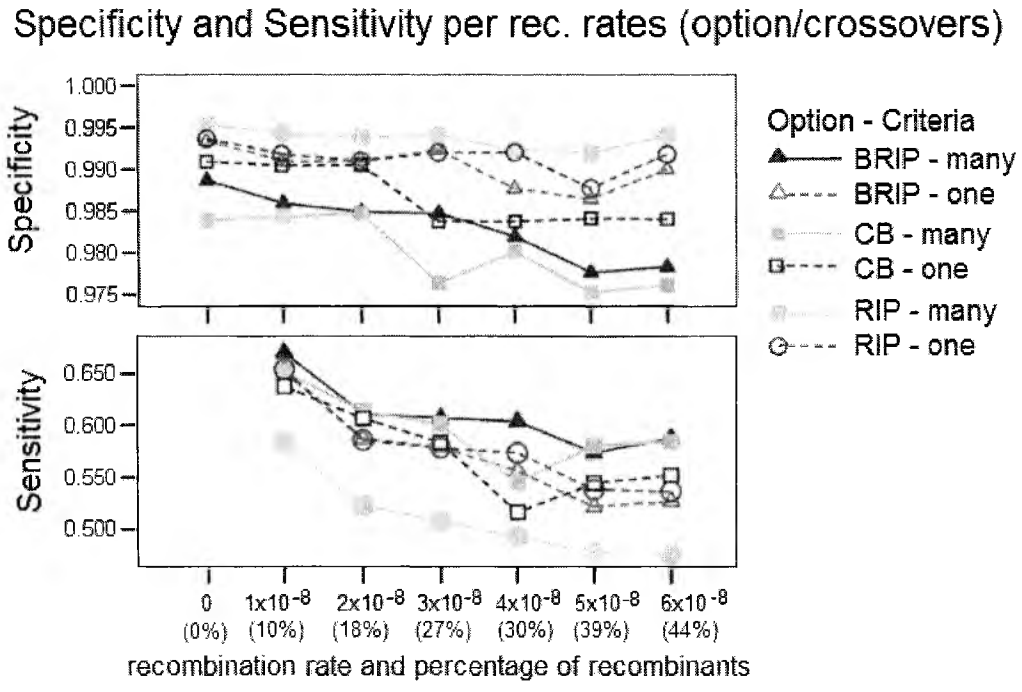
b)


Specificity per rec. rates (option-criteria)
Specificity
0.992
0.988
0.984
0.980
0 (0%)
1x10-8 (10%)
2x10-8 (18%)
3x10-8 (27%)
4x10-8 (30%)
5x10-8 (39%)
6x10-8 (44%)
recombination rate and percentage of recombinants
Option - Criteria
BRIP - edge dist.
BRIP - topology
CB - edge dist.
CB - topology
RIP - edge dist.
RIP - topology

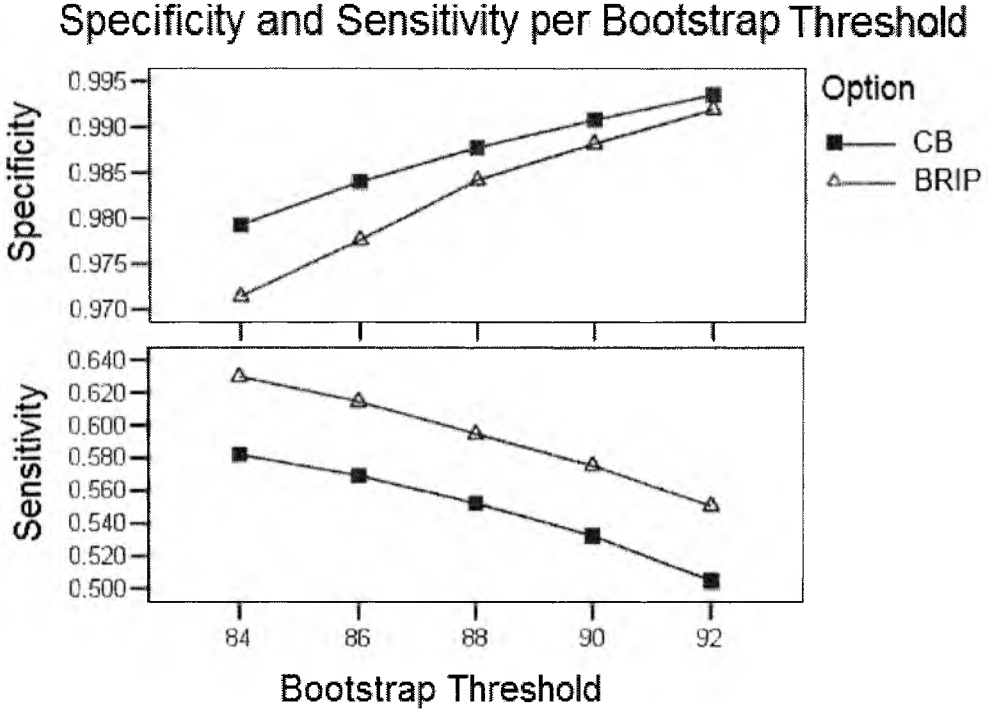
c)


Specificity and Sensitivity per window/step sizes
Specificity
1.000
0.990
0.980
0.970
0.960
0.950
Sensitivity
0.700
0.650
0.600
0.550
0.500
0.450
0.400
200 50
100 50
100 30
70 30
150 25
100 25
200 20
160 20
250 15
window and step size
Options
BRIP
SB
RIP

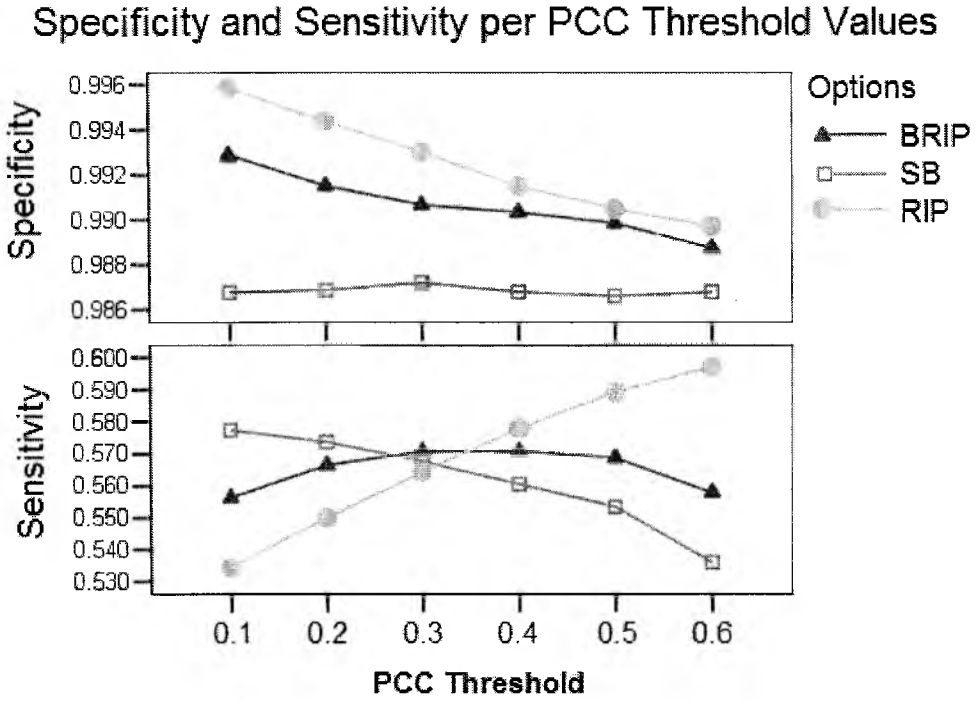
d)


Specificity and Sensitivity per rec. rates (option/crossovers)
Specificity
1.000
0.995
0.990
0.985
0.980
0.975
Sensitivity
0.650
0.600
0.550
0.500
Option - Criteria
BRIP - many
BRIP - one
CB - many
CB - one
RIP - many
RIP - one
0 (0%)
1x10-8 (10%)
2x10-8 (18%)
3x10-8 (27%)
4x10-8 (30%)
5x10-8 (39%)
6x10-8 (44%)
recombination rate and percentage of recombinants

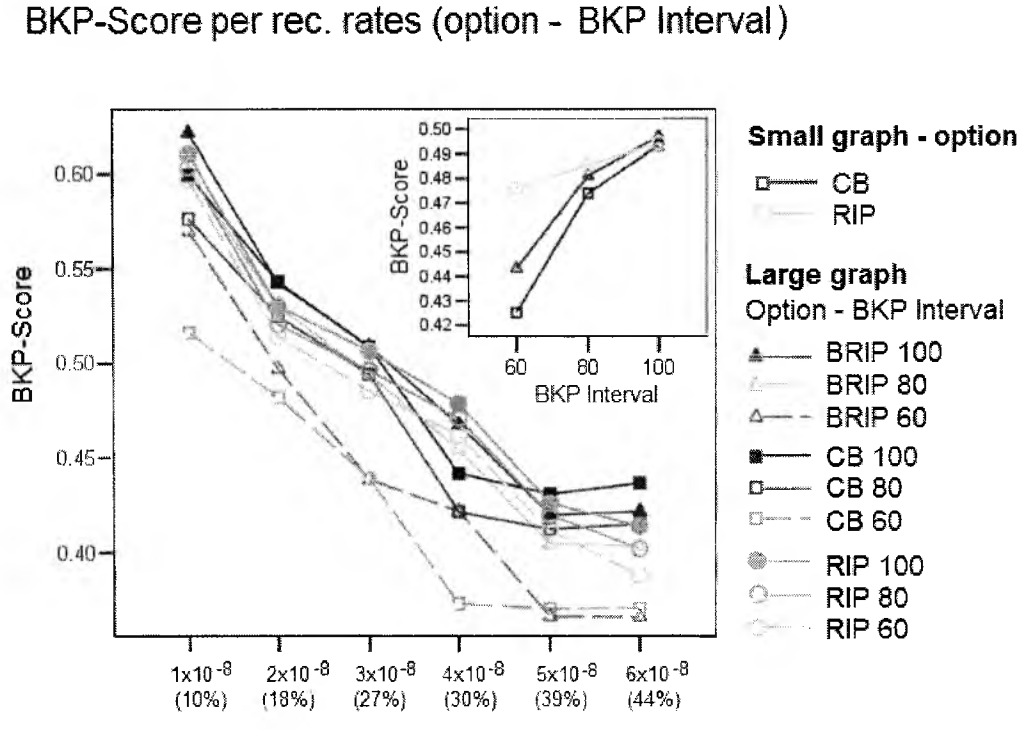
e)


Specificity and Sensitivity per Bootstrap Threshold
Specificity
0.995
0.990
0.985
0.980
0.975
0.970
Sensitivity
0.640
0.620
0.600
0.580
0.560
0.540
0.520
0.500
Option
CB
BRIP
84
86
88
90
92
Bootstrap Threshold

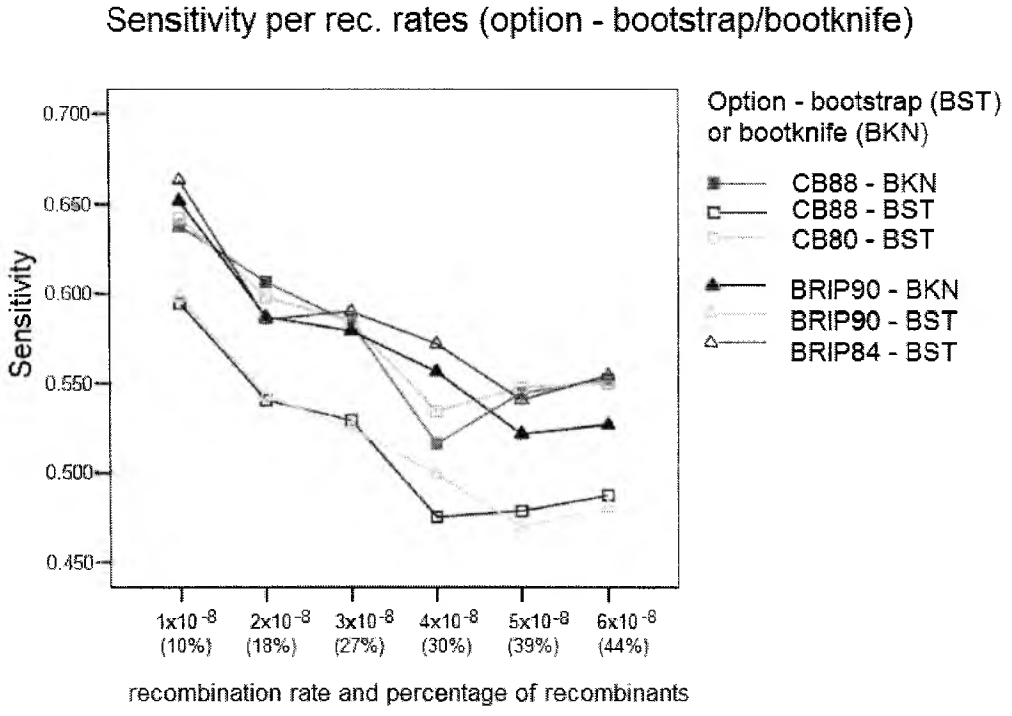
f)


Specificity and Sensitivity per PCC Threshold Values
Specificity
0.996
0.994
0.992
0.990
0.988
0.986
Sensitivity
0.600
0.590
0.580
0.570
0.560
0.550
0.540
0.530
0.1
0.2
0.3
0.4
0.5
0.6
PCC Threshold
Options
BRIP
SB
RIP

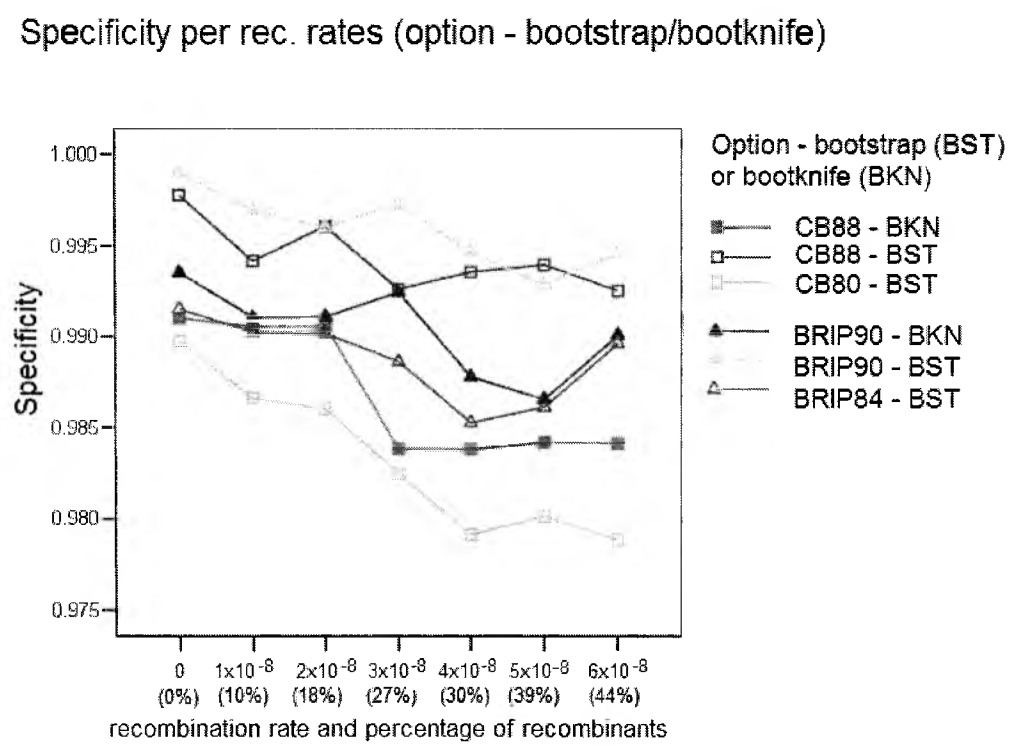
g)


BKP-Score per rec. rates (option - BKP Interval)
BKP-Score
0.60
0.55
0.50
0.45
0.40
BKP-Score
0.50
0.49
0.48
0.47
0.46
0.45
0.44
0.43
0.42
60
80
100
BKP Interval
1x10-8 (10%)
2x10-8 (18%)
3x10-8 (27%)
4x10-8 (30%)
5x10-8 (39%)
6x10-8 (44%)
recombination rate and percentage of recombinants
Small graph - option
CB
RIP
Large graph
Option - BKP Interval
BRIP 100
BRIP 80
BRIP 60
CB 100
CB 80
CB 60
RIP 100
RIP 80
RIP 60

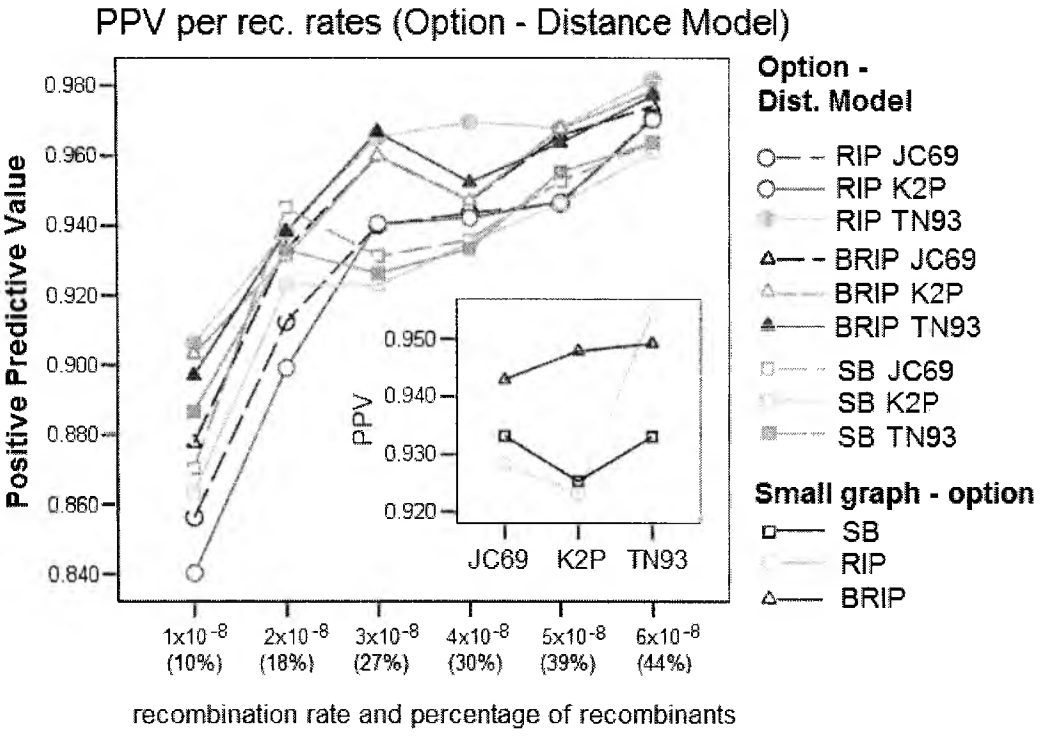
h)


Sensitivity per rec. rates (option - bootstrap/bootknife)
Sensitivity
0.700
0.650
0.600
0.550
0.500
0.450
1x10-8 (10%)
2x10-8 (18%)
3x10-8 (27%)
4x10-8 (30%)
5x10-8 (39%)
6x10-8 (44%)
recombination rate and percentage of recombinants
Option - bootstrap (BST) or bootknife (BKN)
CB88 - BKN
CB88 - BST
CB80 - BST
BRIP90 - BKN
BRIP90 - BST
BRIP84 - BST

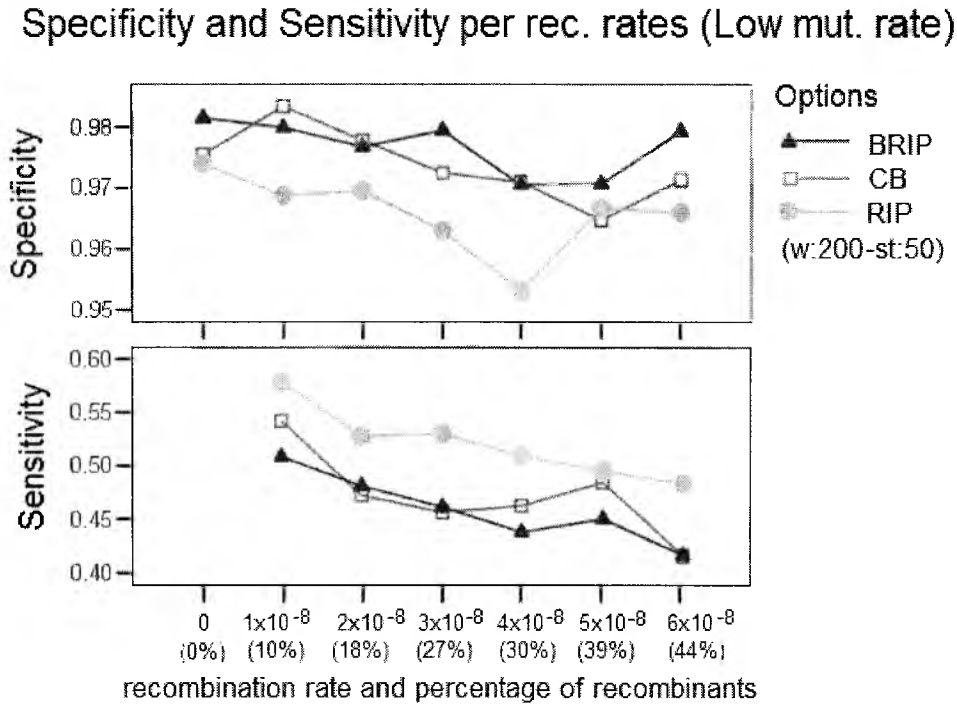
i)


Specificity per rec. rates (option - bootstrap/bootknife)
Specificity
1.000
0.995
0.990
0.985
0.980
0.975
0 (0%)
1x10-8 (10%)
2x10-8 (18%)
3x10-8 (27%)
4x10-8 (30%)
5x10-8 (39%)
6x10-8 (44%)
recombination rate and percentage of recombinants
Option - bootstrap (BST) or bootknife (BKN)
CB88 - BKN
CB88 - BST
CB80 - BST
BRIP90 - BKN
BRIP90 - BST
BRIP84 - BST

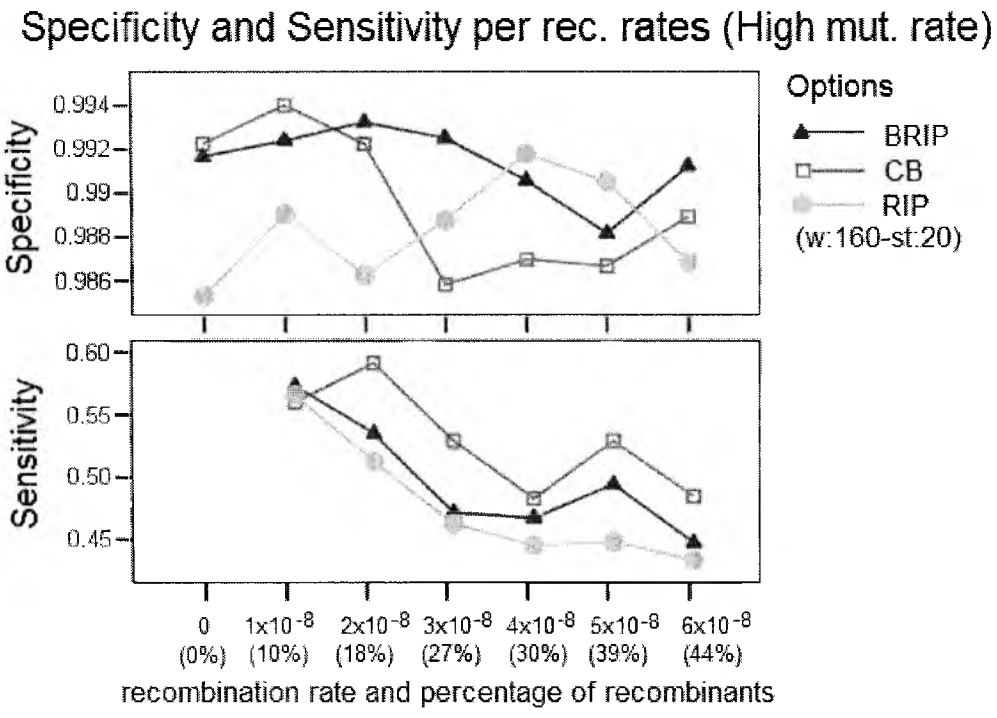
j)


PPV per rec. rates (Option - Distance Model)
Positive Predictive Value
0.980
0.960
0.940
0.920
0.900
0.880
0.860
0.840
PPV
0.950
0.940
0.930
0.920
JC69
K2P
TN93
1x10-8 (10%)
2x10-8 (18%)
3x10-8 (27%)
4x10-8 (30%)
5x10-8 (39%)
6x10-8 (44%)
recombination rate and percentage of recombinants
Option - Dist. Model
RIP JC69
RIP K2P
RIP TN93
BRIP JC69
BRIP K2P
BRIP TN93
SB JC69
SB K2P
SB TN93
Small graph - option
SB
RIP
BRIP

k)


Specificity and Sensitivity per rec. rates (Low mut. rate)
Specificity
0.98
0.97
0.96
0.95
Sensitivity
0.60
0.55
0.50
0.45
0.40
0 (0%)
1x10-8 (10%)
2x10-8 (18%)
3x10-8 (27%)
4x10-8 (30%)
5x10-8 (39%)
6x10-8 (44%)
recombination rate and percentage of recombinants
Options
BRIP
CB
RIP
(w:200-st:50)

l)



m)

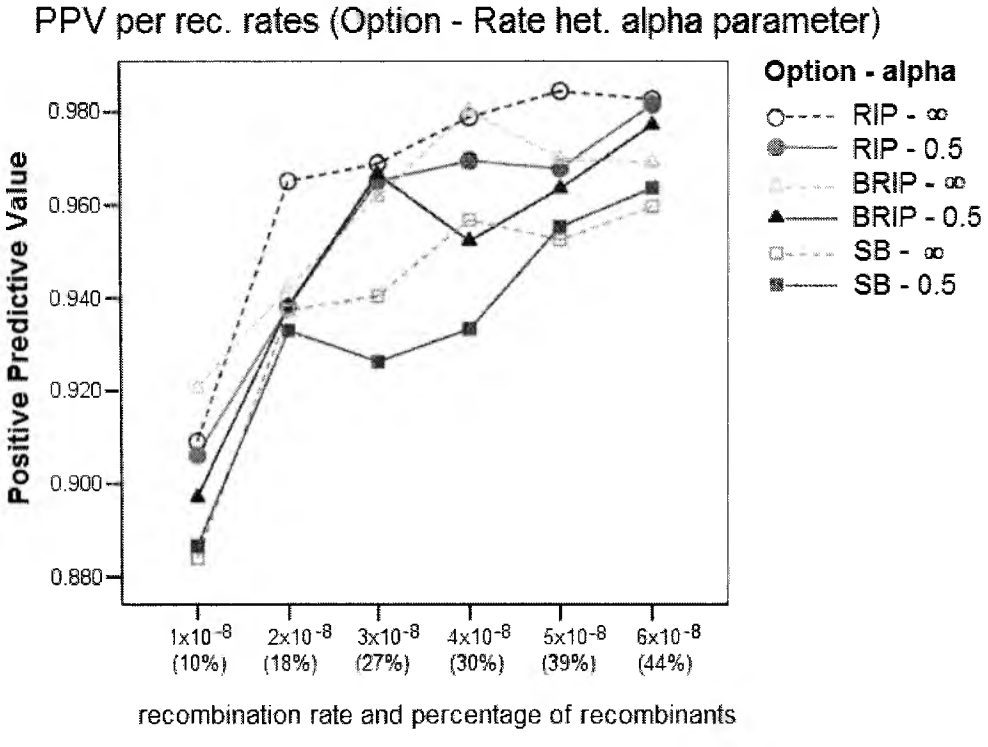


Figure 22. Graphs representing the results of simulation studies with Sliding MinPD

7.4 Summary

In this chapter we have described a new method, Sliding MinPD, that implements a recombination detection approach to study the phylogenetic relationship of serially-sampled data. Three of the dissertation goals were achieved with this method and with the comparison studies performed. Sliding MinPD constructs an evolutionary network reflecting these relationships. Three recombination detection methods were implemented in Sliding MinPD as user options and the results of an extensive comparison study using computer simulations are provided, resulting in an evaluation of recombination detection methods that is more comprehensive than in previous studies (Posada and Crandall 2001a; Wiuf et al. 2001). Detecting recombinants in a set of input sequences is a difficult problem, and the performance of existing methods with regard to the accuracy of identification of recombinants, donors and breakpoint positions is not known. Previous studies did not measure specificity or sensitivity values, or how many breakpoints and donors were correctly identified for the recombination events. The main reason why such evaluation studies were not carried out is that the existing detection methods only offer graphical outputs from which the user is left to decide whether or not a given sequence is recombinant. Sliding MinPD implements three recombination detection methods and automates the detection process replacing the visual output with a list of the inferred recombinant sequences along with statistical significance values associated with the inference. Sliding MinPD uses this output to construct a network that describes the evolutionary relationship of the data.

Applying our method on simulated data returned an average sensitivity score of 0.60 for low recombination rates (up to 27% of recombinants) and few false positives with all rates (specificity rates of 0.99). About 50% of breakpoint positions were correctly detected for the low recombination rates, this rate increased by an average of 6% when the interval to the left and right of the true breakpoint position was increased from 60 to 100 in the evaluation process. We presented two versions of our algorithm — one that detects only one crossover and another that detects multiple crossovers. The multiple crossovers alternative also increased the accuracy of detection of breakpoint positions for two of the options (SB and B-RIP).

Sliding MinPD is a method that can be applied to recombining, fast-evolving viruses such as HIV. While traditional phylogenetic methods assume a tree-like model of evolution, MinPD seeks to recover the evolutionary relationships within a network structure that represents recombination events. Sliding MinPD is a versatile tool, which offers different options of recombination detection. An analysis study of serially sampled data with the multiple options implemented in Sliding MinPD provides a complete approach in which the main aspects of serially-sampled viral data are considered, such as the temporal nature of the data, the ancestor-descendant relationships and the detection of recombinants.

8 Analysis of Serially-Sampled HIV Data Sets

The tools developed in this dissertation were intended to help in the study of the evolution of rapidly evolving viral species within host environments, and to shed light on the evolutionary mechanisms and patterns of evolution. In this chapter we present the results of applying the Sliding MinPD method on real, serially-sampled, viral sequences.

8.1 Introduction

As discussed in chapter 2, some of the remarkable properties of RNA viral populations are their large population sizes within host systems, their high replication rate, and short generation time. Recombination in RNA viruses is another process that creates population diversity and is significant in that it produces genomes with selective growth advantage (Flint et al. 2000b). The human immunodeficiency virus (HIV) is a RNA retrovirus with a high mutation rate (0.2 errors per genome per cycle) and an even higher recombination rate (3 events per genome per cycle). HIV has been determined to be the cause of the AIDS disease, with 5 million people becoming infected with HIV in 2002 alone and, of which, 70% live in sub-Saharan Africa (Rambaut *et al.* 2004).

In 1999 a groundbreaking study on HIV evolution was published (Shankarappa *et al.* 1999). The study performed phylogenetic and statistical analysis on sequence data (viral DNA and plasma RNA) from viral samples that were collected at recurring time intervals from nine patients over a span of 8 to 12 years. Henceforth we will refer to the HIV data set as the *Shankarappa99* data set.

In this chapter we will analyze the *Shankarappa99* data set using data for eight of the nine patients. We anticipate this data set to provide an ideal source for the testing of our method, Sliding MinPD, with the goal of studying the viral evolutionary relationships, evolutionary patterns, splitting and merging of lineages, and determining how these correlate with the disease status of the patient. Our analysis provides insight into within-host viral evolution and helps find patterns that may explain the emergence of harmful mutants associated with disease progression.

8.2 Data Sets and Algorithms

The Shankarappa99 data set was made available at GenBank and the HIV Los Alamos database. Two hand-aligned versions of the data set were made available from the URL: http://www.cebl.auckland.ac.nz/~hros001/HIVpossel/, as supplementary material for the work of Ross and Rodrigo, who used the data to study immune-mediated positive selection driven by HIV-1 molecular variation (Ross and Rodrigo 2002). Both versions were available in the PAUP Nexus format and were separated into subsets, each corresponding to a single patient. One version was a gapped, hand-aligned file of the entire Shankarappa99 data set (separated into lists by patients), where gaps signified insertions or deletions (Ross and Rodrigo 2002). The length of the sequences was 786 nucleotides. The second data set contained a gap-balanced alignment of each of the subsets of the same data. Gaps had been removed from this data set in a "balanced" manner, i.e., such that codon alignments were preserved. Both the data sets, gapped and gap-balanced, were aligned against reference HIV sequences (HIV-1 type B accession numbers K03455, M17451, U63632, and U21135) from the Los Alamos database. The

sampled sequences corresponded to amino acid positions 342 to 594 in the reference sequences. The data sets of eight patients were then analyzed using Sliding MinPD. The Nexus files were converted to aligned files in FASTA format, as required by Sliding MinPD. Duplicate sequences from the same sampling point were removed as they represented the same ancestral sequence, retaining just one copy, but marked with an **x** followed by the number of duplicate copies. Thus "11x2" from Figure 24 refers to two copies of sequence 11 from the sampling time point 20. Duplicate sequences from different sampling points were left untouched in the data set; we will comment on this decision later in this chapter.

The sequences in the Shankarappa99 data set from the same time point had a relatively low level of diversity (average genetic distance: 0.03) and a low level of divergence relative to an initial founder sequence from the first sampling period (average genetic distance: 0.05). The exceptions were the data sets for patient 9 at all sampling points, and the sequences for patients 2, 3, and 7 for the last sampling times (Shankarappa et al. 1999). Low divergence causes a drop in performance in most recombination detection methods (Posada and Crandall 2001a). The number of false positive predictions increases for sequence data with low divergence. We provided the alignment files, two for each patient, as input to Sliding MinPD, which ran each file using the three different recombination detection options: B-RIP (Bootscan RIP), SB (standard Bootscan) and RIP, producing 6 result files for each patient. In a final step, a consensus file was manually created by combining information from the 6 result files using a consensus strategy and a keen awareness of the strengths and weaknesses of the three methods used (based on the experiments and analyses from Chapter 7). The settings for each option of

the program were the same as the default options presented in Chapter 7, with the exception of the RIP program, which was executed with a window size of 200 and a step size of 50, to compensate for the low divergence rates in the data sets. The default options are as follows — corrected distance: TN93, window size: 200, step size: 20, bootstrap replicates: 100, bootstrap threshold: 90, PCC of 0.2 (SB) and 0.4 (RIP and B-RIP), rate heterogeneity alpha shape parameter: 0.5, and a bound of 1 for the number of crossovers. The results for non-recombinant predictions were less compromised given that MinPD (the predecessor of Sliding MinPD's RIP option) has been proven to perform well for data with low mutation rates (Buendia et al. 2006a).

A consensus file of the 6 result files was created manually by looking at each ancestor-descendant relationship and by observing the following guidelines that apply to recombinants and non-recombinants alike:

- Majority rule: The relationship chosen by a majority (at least four) of the six methods was chosen as the consensus. If this rule was not applicable, then we proceed to the next rule.
- Consensus by bootstrap support: The result with the highest bootstrap value was chosen.
- Consensus by priority: The results by B-RIP method on the aligned data were given highest priority because of its superior performance and because it outputs the statistical bootstrap support for each estimate. RIP was given the second highest priority for the ancestor choice, when all options agreed on a non-recombinant relationship. The results by SB had the second highest priority when choosing

recombinant relationships. The results for the gap-balanced data were given secondary consideration when choosing the consensus ancestor-descendant relationships, as the length of the alignment was often different from that of the gapped alignment and breakpoint positions could not be easily matched.

A computational tool that constructs the serial evolutionary network from (a manually created) consensus file containing the ancestor-descendant relationships and an input of aligned sequences was developed. The consensus file was created manually from the 6 results files by applying the consensus rules specified earlier. The network builder generates the collection of Neighbor-Joining trees using the same procedure as in Sliding MinPD and according to the chosen relationships in the consensus file. The Neighbor-Joining trees are used to draw a single HIV-1 serial evolutionary network that reflects the consensus result of the previously obtained 6 results.

8.3 The Networks

The serial evolutionary networks of the 8 patients were manually drawn using the visualization approach, described in detail in Section 4.3.2. A short summary of the guidelines used for drawing and interpreting the graphs is given below:

- The sampling times in units of "months" from the time of seroconversion (the time when the production of antibodies started in response to the infection) are shown for every sequence at the top of each network directly above it. Thus, all sequences from the same sampling time are aligned vertically under the corresponding time.
- Full lines indicate branch lengths. Thus longer full lines indicate greater evolutionary distance. Dotted lines indicate linking relationships and are used merely to retain the

vertical alignments enforced on sequences sampled from the same time point (as mentioned in the previous item). Finally, dashed blue lines indicate recombination-linking relationships between donors and recombinants.

- Recombinant sequences are underlined in blue.
- Breakpoint positions appear after a slash. If the left donor is at the top (bottom, respectively) end of the recombination-linking dashed line, this is indicated by a forward (backward, respectively) slash followed by the breakpoint position.
- X4-mutants appear in red and are marked with an elevated small **x** after the sequence, as in the taxon $14(96)^{x}$ from Figure 24. The X4 mutations can be identified by a K (Lysine) or R (Arginine) located at amino acid positions 40, 53, or 54 of the protein sequence alignments.
- An equal sign "=" next to a node indicates that the sequence is identical to an ancestral sequence from a previous sampling period. For example, the sequence 18(100) of patient 2 at 91 months is identical to sequence 18(82) from the 80 months sampling period.

The new method, Sliding MinPD, outputs additional values of importance, such as statistical bootstrap support values. The bootstrap values provide statistical significance scores for the ancestor-descendant predictions. When the program has to choose between highly similar sequences the bootstrap values may be low, signifying that the results are less conclusive. Due to the value of this information it was necessary to incorporate it in the network representation. Besides the guidelines summarized above, the changes/additions to the network representation are as follows:

- The bootstrap values are added in parenthesis after the sequence ID.

- The bootstrap values correspond to the ancestor-descendant relationship and not to the clades or topological position in the subtrees. The bootstrap value represents the support given to the choice of linking a specific query sequence to the ancestor at the root of the immediate subtree. Thus, the taxon labeled 43(97) at 61 months in Figure 24 corresponds to a bootstrap value of 97 for the ancestor-descendant relationship between taxon 43 and 08 at 40 months.
- Unlike the bootstrap values of recombinant sequences, the bootstrap values of non-recombinant sequences were calculated by using the data set containing all sequences up to the previous sampling point of the query sequence.
- Bootstrap values of recombinant sequences were from computations involving only a few sequences from the pool of donor candidates. A threshold of 90 was used in the computation.

- The sequence IDs have been shortened to decrease the amount of clutter and to make space for the bootstrap values. In their shortened form, they are unique within a sampling point, but not across the whole serial evolutionary network. For example, the GenBank ID of p2c005-03 is shortened as 03 in the network of patient 2 under the sampling time 5 months.

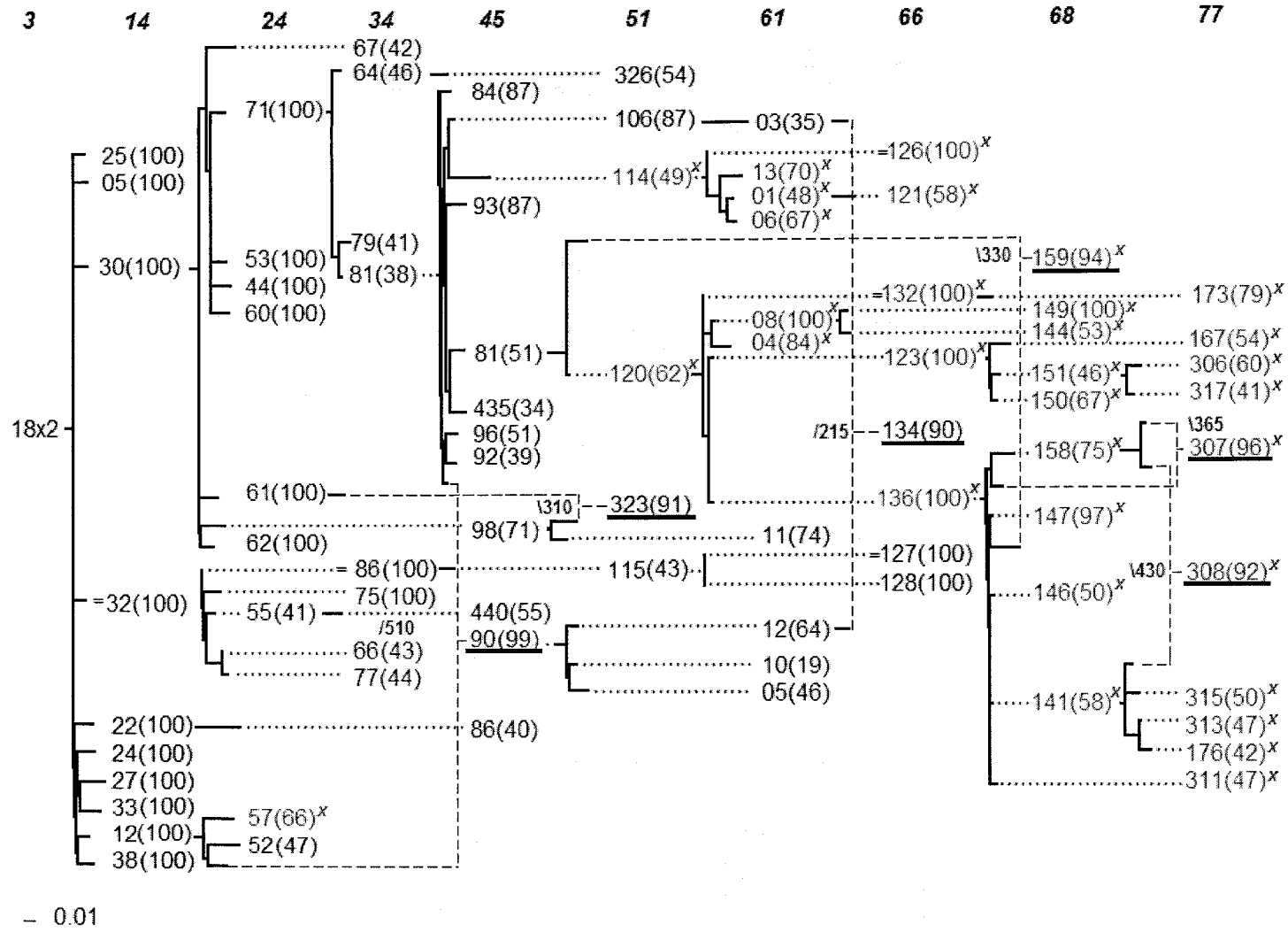


Figure 23. Serial evolutionary network of patient 1

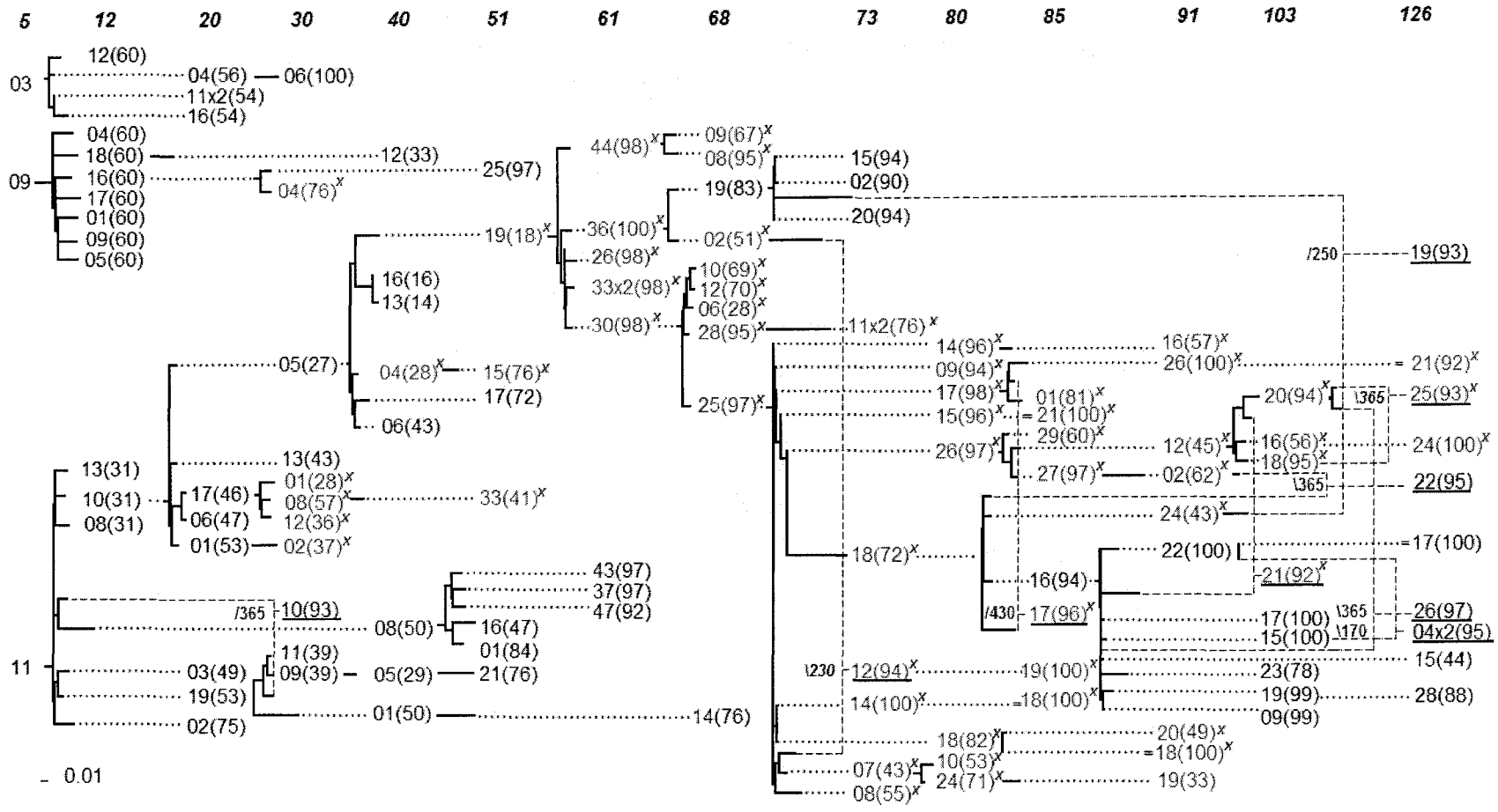


Figure 24. Serial evolutionary network of patient 2

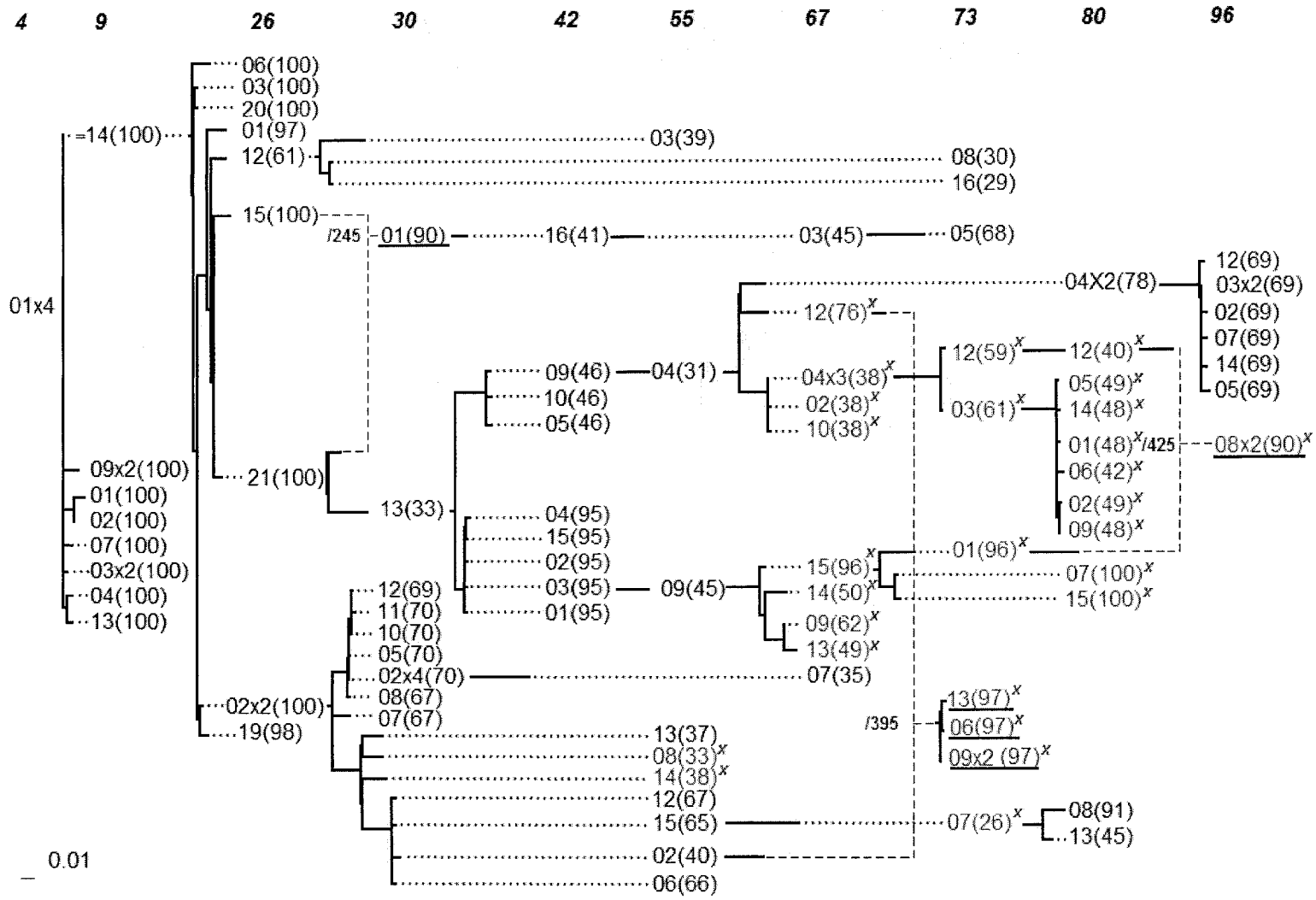


Figure 25. Serial evolutionary network of patient 3

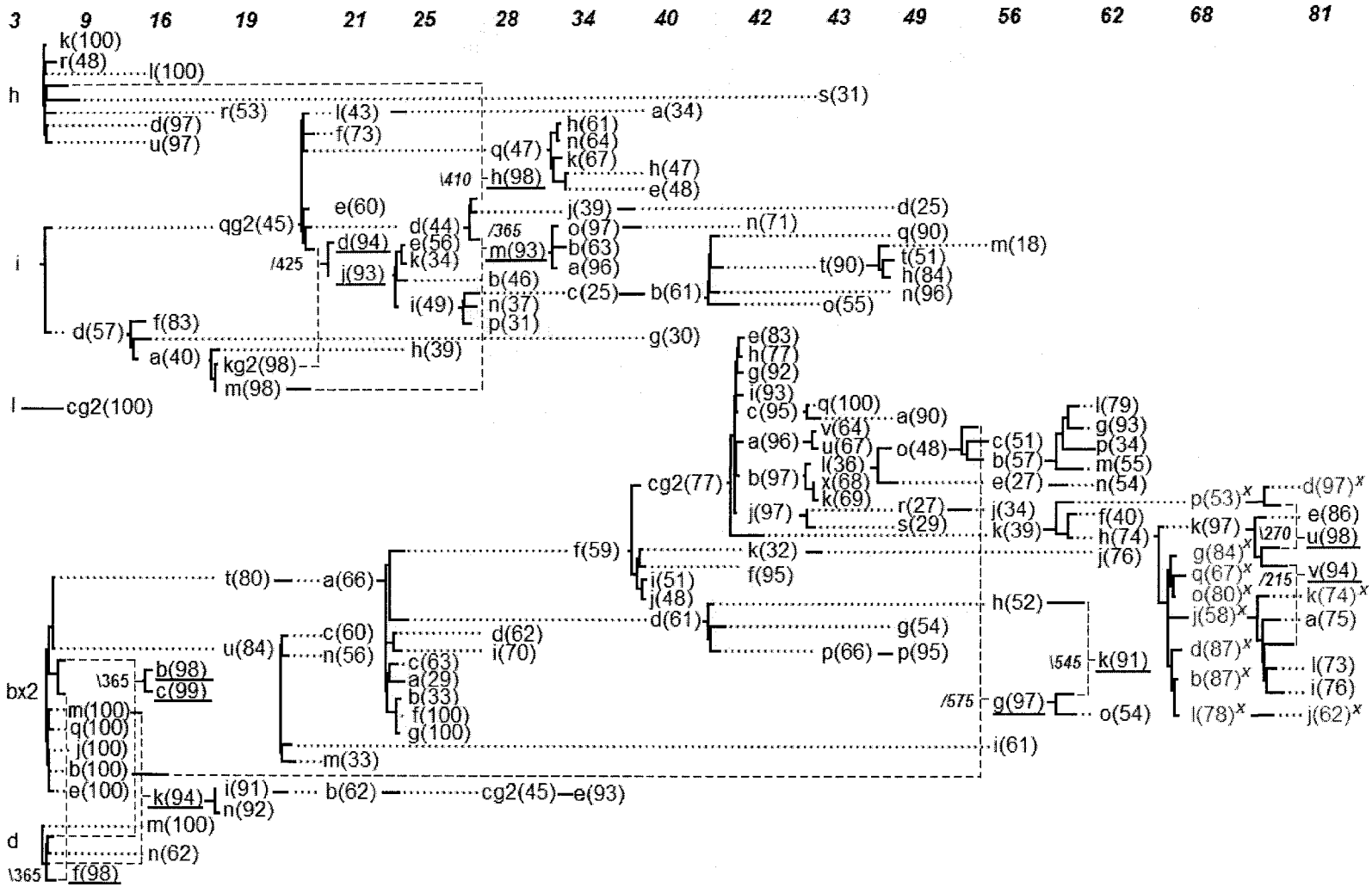


Figure 26. Serial evolutionary network of patient 5

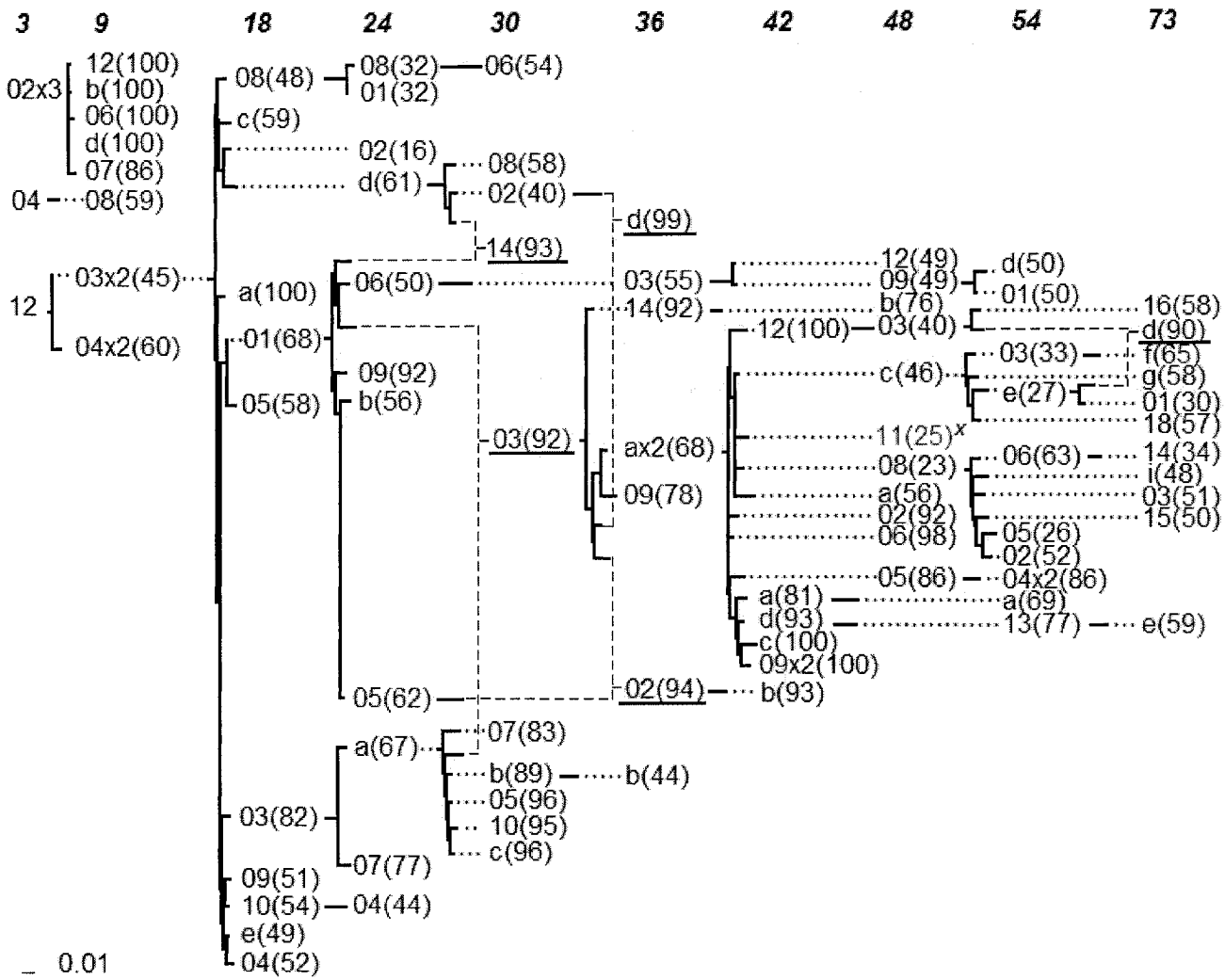


Figure 27. Serial evolutionary network of patient 6

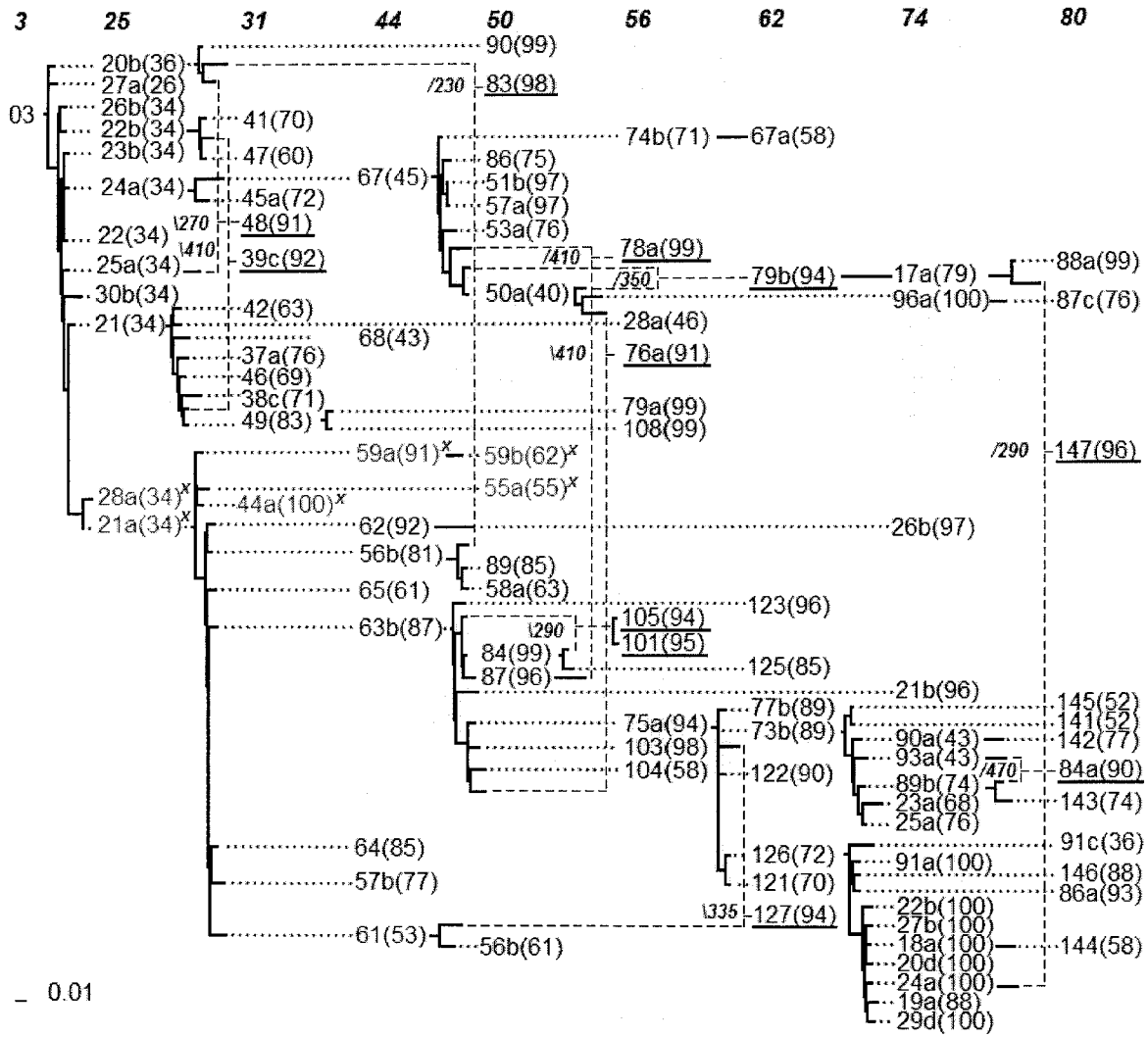


Figure 28. Serial evolutionary network of patient 7

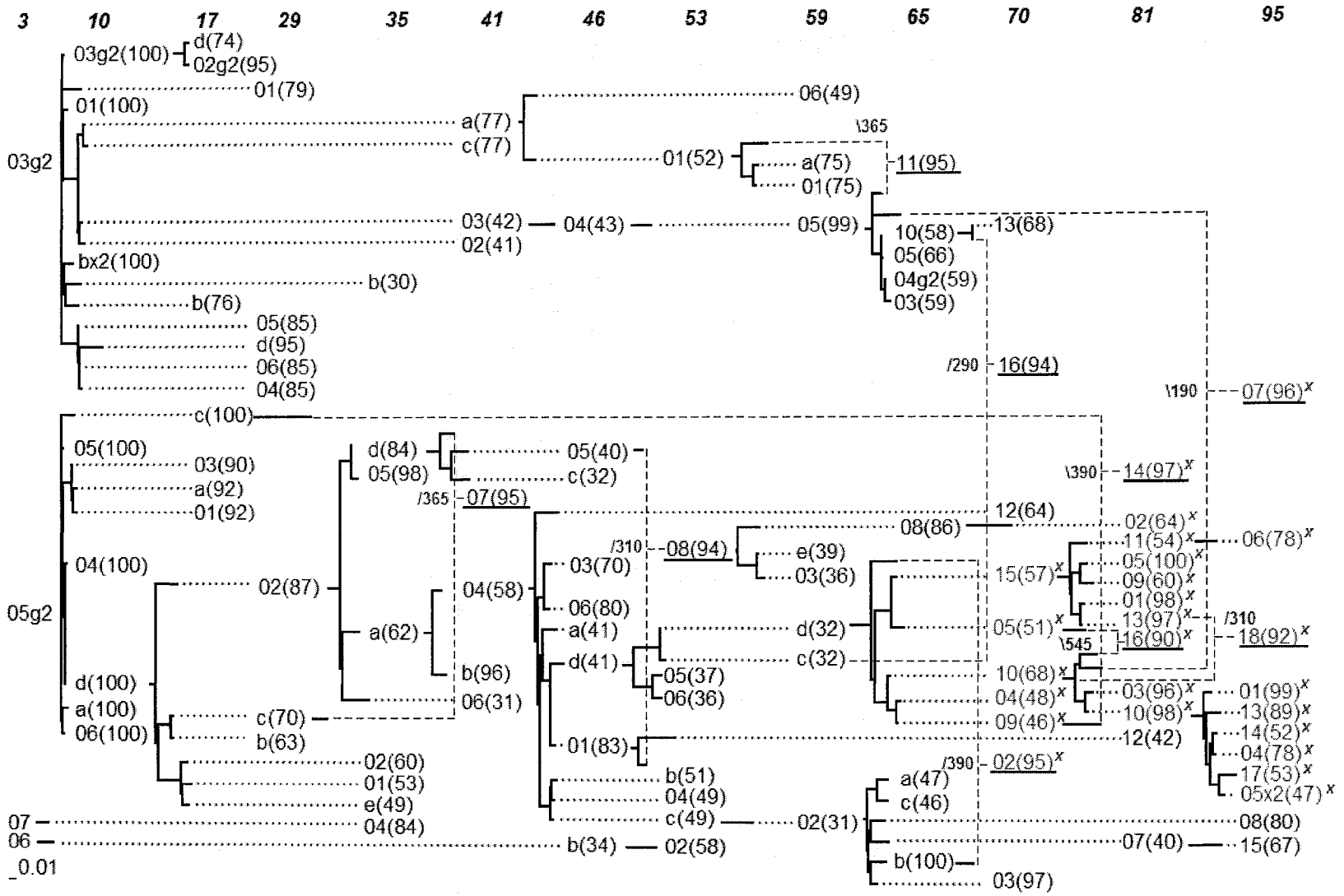


Figure 29. Serial evolutionary network of patient 8

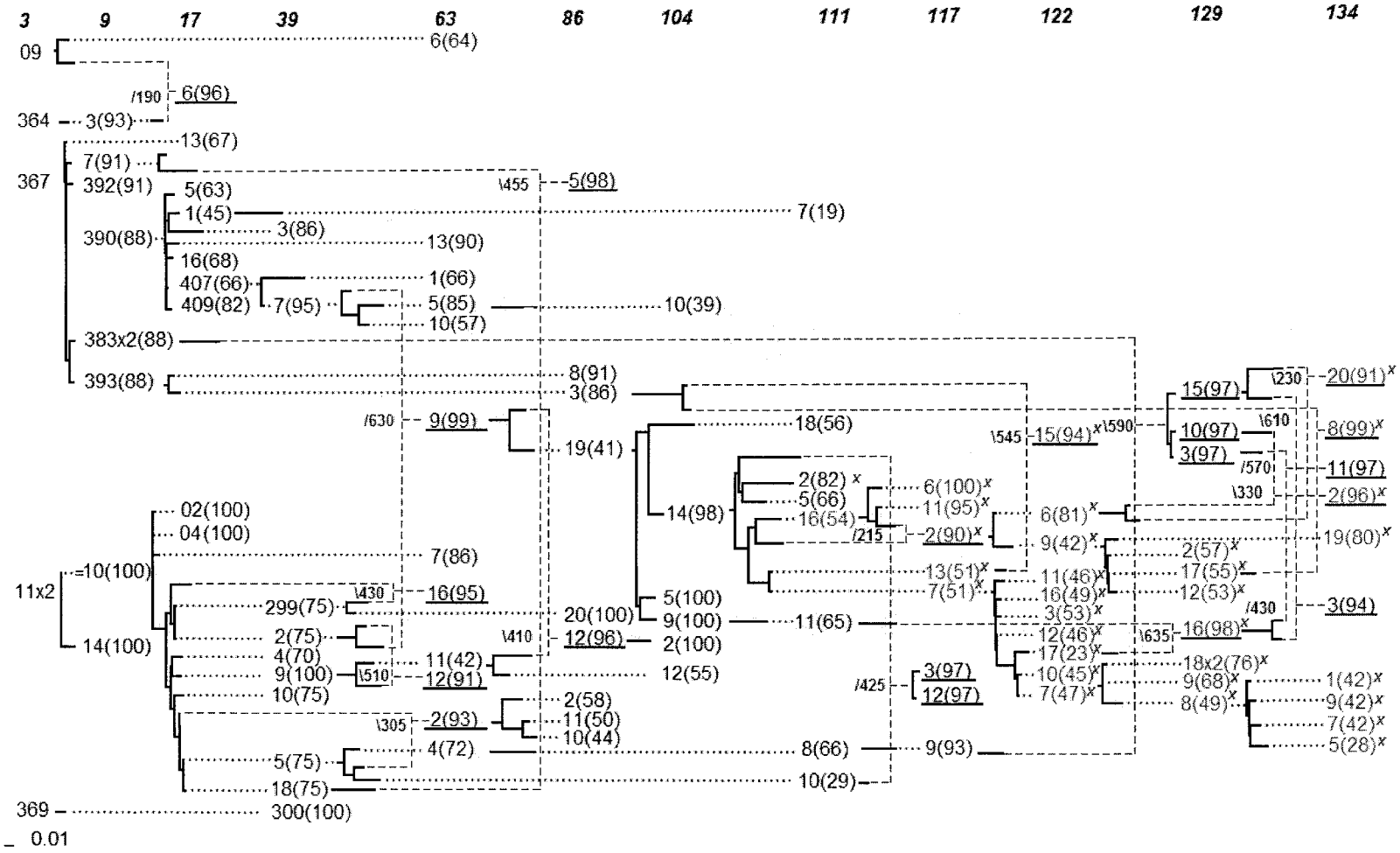


Figure 30. Serial evolutionary network of patient 9

8.4 Discussion

In Chapter 4, the network of patient 2 (labeled as patient *S* in that chapter), was constructed using the MinPD method (see Figure 13). This chapter presents a serial evolutionary network produced by the improved method, Sliding MinPD (see Figure 24). We will therefore start the discussion by analyzing the differences between the two results. For this discussion, we will use the complete ID of the sequences as it appears in Genbank. The name includes the patient information, which appears as "pX" for patient X, followed by the information on sampling time, which appears as "cN" for time period N months since seroconversion, followed by a dash, and a ID number unique within that sampling period. For example p2c51-19 refers to sequence with ID 19 sampled at 51 months from patient 2. A suffix of "x" followed by a number n indicates that the same sequence was sampled n times during the same sampling time point. A suffix of "g" indicates that a duplicate was only found in the gap-balanced alignments. The information about the health status of the patients, the therapy administered to each, and the $CD4^+$ T-Cell counts were obtained from published results (Shankarappa et al. 1999).

8.4.1 Comparison of two networks for patient 2

The two networks, one obtained by applying MinPD and the other obtained by applying Sliding MinPD were compared. They were found to be similar when considering the splitting of lineages, and showed some differences with respect to the identification of recombinant sequences. As in the network of Figure 13, up to the 51-months sampling period there were three different lineages. Also like in Figure 13 one lineage ceased to exist at that point with sequence p2c51-25 as its last sampled sequence.

As shown in Figure 13, another lineage with the last sampled representative as p2c68-14 disappeared at 68 months. After 68 months both networks showed no distinct lineages, with clear indications of the occurrence of recombination events. As suggested by both networks, the X4 mutation that was widespread in most sequences sampled later appears to have descended from p2c61-30 and its putative ancestor p2c51-19. However, the two methods disagreed on whether or not the sequence p2c61-30 was a recombinant. Although the RIP option in Sliding MinPD continued to identify this sequence as a recombinant, the other methods did not. In particular, the gap-balanced version of B-RIP suggested that sequence p2c51-19 was the only donor with a bootstrap support of 98%. As for the donors of sequence p2c51-19, two of the results, the ones using the SB and B-RIP options, suggested that it was a recombinant sequence. However, the two methods disagreed on the donors (Left:p2c30-12 and Right:p2c40-08 versus Left:p2c30-01 and Right: p2c40-01); a majority of the other results identified sequence p2c30-05 as the only donor, albeit with low bootstrap support. Although a recombinant history would have made more sense because of its X4 mutation, both networks linked it to a non-X4 ancestor. It is worth pointing out that the original MinPD initially identified a large number of potential recombinants, but only a few were added to the network after doing additional analysis using the Bootscanning program Simplot (Ray 2003). As a majority consensus rule was used to choose recombination events in Sliding MinPD, which gives priority to the B-RIP method, the recombination results were quite different in the resulting network. The networks suggest an increase in recombination activity in the later sampling periods. This coincided with a decrease in the number of CD4 T-Cells. More importantly, it also coincided with the start of the antiretroviral drug therapy at the 103rd

month period. In fact, the largest number of recombinants (with strong recombination signals) were found during the last sampling period after the patient started taking three different drugs. These drugs included Zidovudine- ZDV, Lamivudine-3TC (which the patient stopped taking at 120 months), and Stavudine-d4T (which was continued until the 126 months period). An interesting recombination event was suggested for sequence p2c126-19 with a putative left-side donor of p2c68-19, which was sampled almost 5 years earlier, and which was from a different lineage and did not carry the X4 mutation. As will be seen with other patients (patients 8 and 9), the appearance of recombinant sequences that join separate lineages seems to be a recurring theme typically after the start of retroviral drug therapies.

8.4.2 Analysis of networks for other patients

Patient 9

Only Sliding MinPD was applied to the data set from patient 9, as well as to the other data sets discussed in this section (Section 8.4.2). The network of patient 9 suggests the resurgence of reservoir virus as suggested by the recombination events at 129 months. The presence of different types of HIV reservoirs cells that harbor dormant virus has been discussed in many studies (Chun and Fauci 1999; Imamichi *et al.* 2001). The persistence of reservoirs of HIV, including latently infected, resting $CD4^+$ T-cells that can give rise to infectious HIV after a period of latency, has posed a sobering challenge to the long-term control or eradication of HIV in infected individuals receiving highly active antiretroviral therapy (HAART).

Sequences p9c09-383x2 (with 2 identical copies sampled at 9 months) and p2c09-393 with progeny p9c86-3 are putative dormant strains of which a few might have escaped latency through the low degree of ongoing viral replication that is thought to occur in latently infected, resting $CD4^+$ T cells. We hypothesize that reservoir strains that co-infect a cell already infected with virus from the dominant quasispecies, may replicate and get assembled together with a copy of the current strain into a new viral particle. The viral particle with 2 different parental RNA strands may infect another cell and recombine during replication.

One aspect that becomes apparent is the elevated number of recombinants among the X4-mutants, some of which produce strains without the X4 mutation. It is interesting to note that the earliest putative ancestor of the lineage with a large number of X4-mutants is a recombinant itself, namely the sequence p9c63-9 (with 99% bootstrap support), and has donors from two separate lineages, one that ceased to exist at 111 months and the other that persisted in recombination events through 129 months. Yet another feature of interest is that many recombinant sequences in patient 9 are themselves putative parental donors contributing to other recombination events. Patient 9 was on Zidovudine (ZDV) therapy since before the 63 month sampling time point and was the only patient (among the ones considered in this work) with data spanning a long period of therapy of over 6 years. It is possible that a consequence of the drug therapy, which seeks to inhibit reverse transcriptase activity, is an increase in recombination events, which is known to happen by the switching of templates during the reverse transcription phase. Acceleration of recombination events is perhaps a stress response or an evolutionary mechanism used by the virus to explore ways of achieving drug resistance.

Patient 8

As with the network of patient 9, we can observe a putative recombination event between a possible dormant strain from an early period and a later X4 mutant strain resulting in p8c81-14. The putative right-hand donor and early strain p8c17-c did not have any direct descendants that inherited the whole length of its sequence. As discussed previously, this may be evidence of a pattern with dormant viral strains involved in recombination events with strains from later periods.

As with the networks of other patients, one can observe that the X4 sequences were grouped into their own lineage. However, in the network for the data set from patient 8, the recombinant sequences from 70 months and later had one donor that was from a different lineage. The recombination event generating the sequence p8c95-07 joined two separate lineages of which the last representative was sampled at 70 months. Patient 8 started on antiretroviral therapy at the 81 months period after a drastic decline in the number of $CD4^+$ T-cells. The administration of drugs started rather late (at 81 months) as opposed to patient 9 and there was not enough data to reflect that the same effect of increased recombination occurred with the infected viral population in this patient.

Patient 3

This patient started on ZDV therapy at the 67 months sampling period. A change in the pattern of the linking relationships can be observed in the sampling periods before and after the drug therapy. In the periods before the drugs were administered, most sequences from the same sampling time point clustered together in the same subtrees, sharing the same ancestor and creating a very "orderly" structured network. However, this order broke down after the drug therapy was started. The strain p3c30-13, which is the putative ancestor of the lineage that appears to have survived until 96 months, and which contains a majority of the X4 mutants, had low bootstrap support (33) and a large genetic distance (see full line branch length) to its putative ancestor (i.e., sequence p3c26-21) suggesting that a close ancestral sequence was not sampled for this strain. This patient developed AIDS at the last sampling period and died a year after that (at 9.1 years after seroconversion).

Patient 1

This patient was never administered antiretroviral drug therapy. Although the number of X4 strains fell to close to zero (data not shown) after the 77 months sampling point, the patient's $CD4^+$ T-cell count continued to decline dramatically and the patient died at 9.1 years after seroconversion. There are little remarkable details in the network of patient 1. Not many recombinant sequences, nor recombination events involving different lineages.

Patient 5

There are two remarkable observations to be made on the network for the data set from patient 5. The first one is that there are two lineages that coexist until the 56 months sampling time point and these are not joined via any recombination events. The other observation is related to the recombinant sequence p5c56-g, which as observed in other patient networks represents a recombination event between a possible dormant virus from a very early period and a current strain. Patient 5 was started on therapy after the 62 months sampling time point and passed away 8.1 years after seroconversion. Unlike other patients there's repeated sampling of the virus in short intervals, such as during the sampling time points 21, 25, and 28, and 40, 42 and 43. If the sequences are too similar it will cause the predictions to be less statistically significant as shown by bootstrap values below 70.

Patient 6

There is little that is noteworthy about this network, as there is only one discernible lineage until the last sampling point. The recombinant sequence p8c30-03 is the putative ancestor of a majority of the sequences sampled after time point 36. Furthermore, the only X4-genotype sequence has a very low bootstrap support indicating that no clear ancestor relationship was found for that strain. Therapy started early at 36 months and the patient died of AIDS at 7.1 years after initial detection of the virus.

Patient 7

There are 2 major lineages that are however not sufficiently distinguishable because of repeated putative recombination events between them and also because of single strains that are linked with neither one of these lineages, such as p7c56-79a, p7c56-108, and specially p7c74-26b . The X4 genotype appeared early and disappeared early, and as observed in all other patients it was part of the largest surviving lineage. Therapy started early at 44 months and the patient died at 9.1 years after initial detection of the virus.

8.5 Summary

In this chapter, we have applied our method Sliding MinPD to "real" sequence data from the HIV-1 envelope region of eight patients sampled serially over a period of many years. We discussed the representation of our results in the form of a serial evolutionary network and we made inferences on the patterns of evolution of the different strains. The evolution of HIV within host environments is very different from general HIV evolution (say, within a host population). There is strong evidence that natural selection is the driving force of within-host evolution. Within-host HIV phylogenies have a strong temporal structure, reflecting the successive fixation of advantageous mutations and the extinction of unfavorable lineages (Rambaut et al. 2004). This can be observed in all the serial evolutionary networks presented in this chapter. Each of the networks obtained from the sequence data for the different patients appears to be qualitatively different and reveals different aspects of the evolutionary relationships among the data.

It is important to mention the role that the visualization of the results plays in understanding the different patterns of within-host viral evolution. First, with our network representation, we can locate any viral strain easily due to the temporal positioning within the network, where strains sampled at the same sampling point are vertically aligned. Second, it allows us to better understand the evolutionary relationships of the X4 strains. By displaying recombination information we can study another kind of relationship among the serial samples. The data of all patients included X4 mutant strains, which are known to be associated with a higher rate of $CD4^+$T-cell decline, and therefore, a more rapid progression to AIDS (Rambaut et al. 2004). The network representation allowed us to take a closer look at the evolution of these strains and to connect our observations with the known facts and published results by researchers who study molecular viral evolution.

A few of the recurring patterns observed in the 8 patient's networks are:

1. The X4-strains always belong to the largest and longest surviving lineage (all patients).
2. The founder strain of the largest lineage (the one containing the X4 genotype) is a recombinant or has very low bootstrap support, which often happens when the methods are split on whether the strain is or is not a recombinant or on their choice of an ancestor. This observation is evidence of the unique status of these strains in the networks (sequence p1c34-81 in patient 1 has low bootstrap support, as do p2c51-19 in patient 2, p7c25-21a in patient 7, p8c41-04 in patient 8, p3c30-13 in patient 3, p6c30-03 in patient 6 is a recombinant sequence, as well as p9c63-9 in patient 9).

3. Recombination events that involve strains sampled at much earlier times suggest the existence of dormant viruses escaping latency and that recombine with viruses from the later populations (patients 2, 5, 8, 9).

4. Recombination events linking two different major lineages is a recurring event in several patients (patients 3, 7, 9, 8).

The findings indicate that there is a possibility of increased fitness when patterns 2 and 3 are observed as patient 8 and 9 are the only surviving patients (during the length of the study) whose $CD4^+$ T-cell count remained above 0 and even rebounded slightly in the late stages of the study when the number fell dramatically. There was no discernable pattern related to the administration of antiretroviral drugs, possibly also because with the exception of 3 patients all patients started with therapy relatively late (at the last 2 sampling time points).

The visualization of the results played an important role in understanding the different patterns of evolution of the viral data sampled from eight patients. With our network representation, we located certain viral strains more easily due to the temporal positioning within the network allowing for a better understanding of the evolutionary relationships of for example the disease inducing X4 strains. The network representation allowed us to take a closer look at the evolution of these strains and to connect our observations with the known facts and published results by biologists and virologists that study the molecular viral evolution. It is our belief that the use of our serial evolutionary network to represent the relationships inferred by our method Sliding MinPD will

improve the understanding and modelling of the within-host molecular evolution and epidemiology of recombining RNA viruses such as HIV.

9 Conclusion

Our studies confirm that our newly developed distance-based phylogenetic tools are both accurate and efficient in analyzing serially-sampled sequence data, detecting recombination and inferring the complex evolutionary network to represent the sequence data. Such evolutionary networks can be of great assistance in the understanding and modeling of the molecular evolution of recombining RNA viruses. We start with a general discussion of the main accomplishments of this dissertation and then discuss how each of the sub goals mentioned in the introductory chapter were addressed.

9.1 Accomplishments of the dissertation

The main objective of this dissertation work was to develop new methods to study the evolution of recombinant, serially-sampled viral populations within a single host. Our study is a contribution to the ongoing research on infectious viral diseases and on the role of within-host viral evolution on the health of the infected patient and their prognosis. We have developed a suite of versatile tools that help to provide new insights into the evolutionary relationships of recombinant serially-sampled data. The main tool called Sliding MinPD performs a combined analysis of the different aspects of viral evolution. The soundness, efficacy, accuracy, and utility of our method were demonstrated using extensive experimentation with both real data (from experimental evolutionary studies) as well as simulated data. Sliding MinPD was also carefully and comprehensively compared with other competing methods, even on a feature-by-feature basis. The accuracy of Sliding MinPD compared favorably with the best methods on individual features. Furthermore, Sliding MinPD remains unique in terms of the overall array of features it

offers. Its strongest feature, however, remains its efficiency. It is orders of magnitude faster than methods that use maximum likelihood or Bayesian techniques, while remaining competitive in terms of accuracy. We expect the utility of Sliding MinPD to grow enormously as serially-sampled data sets start to grow larger and larger in size. Most of the competing methods are not geared to handle data sets that are tens or hundreds of times larger than existing data sets.

To perform the large volume of simulation experiments required for this dissertation, another sequence generation tool was developed called Serial NetEvolve.

There was also a clear need for comprehensive evaluation techniques to assess recombination detection methods; the previous ones were insufficient. The results of our comparison studies should prove significant to the phylogenetic community and the many researchers currently working on the important area of recombination detection.

Another important contribution of this dissertation is a novel way of representing the networks obtained by using MinPD and Sliding MinPD on serially-sampled, recombinant sequence data by displaying them in a structure we called an *evolutionary network*. We also demonstrate its value by making critical biologically relevant inferences from the resulting networks with relative ease.

Last, but not the least, we used the Sliding MinPD method to infer patterns of evolution and predict the evolutionary relationships of real viral data sampled serially from HIV patients.

9.2 Addressing the goals of the dissertation

We articulated a number of goals for this dissertation. We describe below how these goals were addressed.

Goal 1: *to develop new computational tools to find ancestor-descendant relationships between serially-sampled data and to construct an evolutionary network for them.*

We developed a method to analyze serially-sampled sequence data based on minimum pairwise distance computations (aptly named "MinPD"), which infers ancestor-descendant relationships (by identifying the sequences at a minimum distance) and detects recombination (by dividing the sequence into equal-sized fragments and identifying sequences whose corresponding fragments were at a minimum distance). The MinPD's features were tested with simulated data that showed it performed well with recombinant as well as with non-recombinant data. The results of MinPD were presented in an "evolutionary network" structure that places the sequences sampled at the same time in different columns to show the relationships between the taxa. MinPD was also successfully applied to the HIV-1 envelope gene sequences of a patient.

Sliding MinPD, an enhanced version of MinPD, was specifically designed to analyze recombinant serially-sampled data and to also construct an evolutionary network. This method helped us to address the second goal of the dissertation, as discussed below.

Goal 2: *to develop computational tools to automatically detect recombination in serially-sampled data and display the recombination events within an evolutionary network structure*

We considered the problem of detecting recombinant sequences in a set of serially-sampled sequences. Instead of dividing the sequence into equal-sized fragments, we chose a sliding-window approach. Three standard methods were implemented as user options in the program. An important feature was the elimination of user intervention, an undesirable feature of several of the earlier methods. Of the three methods implemented, one method used bootstrapped phylogenetic trees and used changes in the pattern of the location/closeness of the taxa in the trees to find recombinant sequences. Another method used bootstrapped genetic distances to find a change in the choice of sequence at minimum distance between windows along the sequence. The third option, which is a generalization of the fragment analysis approach of MinPD, used a sliding window, but without bootstrap. The results of the program showed that careful calibration of the parameters could lead to high specificity (0.99) and high positive predictive values (>0.90). Sliding MinPD outputs the evolutionary network with additional information: the bootstrap support for linking a descendant with the inferred ancestor(s).

Goal 3: *to build tools to generate realistic synthetic data sets*

We developed a tool called Serial NetEvolve to simulate the evolution of serially-sampled recombinant viruses. Inspired by its predecessor, Treevolve, Serial NetEvolve simulates the evolution of serially-sampled recombinant viral sequences along a

randomly generated coalescent tree or network. An array of additional features make Serial NetEvolve superior to competing programs. Other features of this program include: (1) the option to output serially-sampled sequences, (2) the option to use variable rates of evolution instead of the clock model, (3) the option to sample from the internal nodes of the tree or network, (4) the option to output the topology of the tree or network created, and (5) the option to output the network using a newly developed format called the NeTwick format. Serial NetEvolve was used in two major comparison studies. The resulting tool was invaluable in helping to perform the large volume of experiments performed as part of this dissertation.

Goal 4: *to establish the criteria and mechanisms to perform comparison studies on recombination detection tools and phylogenetic tools for serially-sampled recombinant data*

The AD-Score and RF-distance score were devised to effectively compare the tools to predict ancestor-descendant relationships and predicted network topologies. Due to the complexity of recombination detection approaches, special evaluation procedures were established. Previous studies did not compute specificity and sensitivity values, and more importantly, did not assess the breakpoints and donors that were correctly identified (Posada and Crandall 2001a; Wiuf et al. 2001). We developed a tool, called RecIdentify, which parsed the information contained in a network generated by Serial NetEvolve to identify recombinant sequences, their donors and breakpoints. The tool proved valuable in evaluating recombination detection tools for synthetically generated data sets.

Goal 5: *to evaluate the performance of methods to study serially-sampled data*

We performed two comprehensive evaluation studies that used biological data from empirical evolution studies with "known" phylogenies and data simulated with Serial NetEvolve. Seven methods were included in the comparison studies. The results showed that for inferring ancestor-descendant relationships, the distance method MinPD outperforms methods specifically designed to analyze serially-sampled sequence data. An important characteristic of MinPD is that it does not require an explicit assumption of a molecular clock. The simulation studies corroborated the results of the studies with empirical data. In particular, the topological score results were surprisingly good. As shown by the results with the simulated data, when data sets included the most recent common ancestors for each sequence, tree-based programs failed to place them at basal positions in the tree, resulting in large topological distances from the true tree.

Goal 6: *to evaluate methods to study serially-sampled* ***recombinant*** *sequence data through extensive computer simulation studies.*

Extensive computer simulations were carried out, resulting in an evaluation of recombination detection methods that is more comprehensive than in previous studies. The main reason why such evaluation studies were not carried out is that the existing detection methods only offer graphical outputs from which the user is left to decide whether or not a given sequence is recombinant. Sliding MinPD implements three recombination detection methods and automates the detection process replacing the visual output with a list of the inferred recombinant sequences along with statistical

significance values associated with the inference. Applying our method on simulated data returned an average sensitivity score of 0.60 for low recombination rates (up to 27% of recombinants) and few false positives with all rates (specificity rates of 0.99). About 50% of breakpoint positions were correctly detected for the low recombination rates; this rate increased by an average of 6% when the interval to the left and right of the true breakpoint position was increased from 60 to 100 in the evaluation process.

Goal 7: *to apply the newly developed tools and techniques to "real" data sets and to interpret the results in biologically meaningful ways.*

We applied our method Sliding MinPD to sequence data from the HIV-1 envelope region of five patients sampled serially over a period of many years. We discussed the representation of our results in the form of an evolutionary network and we made inferences on the patterns of evolution of the different strains. We could observe a strong temporal structure, reflecting the successive adaptation of advantageous mutations and the extinction of unfavorable lineages in the evolutionary networks presented and studied in this dissertation. Every structure representing the evolutionary relationships of the patients was different and revealed different aspects of the evolutionary relationships among the data.

The visualization of the results played an important role in understanding the different patterns of within-host viral evolution. With our network representation, we located certain viral strains more easily due to the temporal positioning within the network allowing for a better understanding of the evolutionary relationships of for

example the disease inducing X4 strains. The network representation allowed us to take a closer look at the evolution of these strains and to connect our observations with the known facts and published results by biologists and virologists that study the molecular viral evolution.

Putting all the pieces together, the end result is an effective and useful tool for studying the evolution of serially-sampled recombinant sequence data.

References

Allain, J., Dong, Y., Vandamme, A., Moulton, V., and Salemi, M. (2000). *Evolutionary rate and genetic drift of hepatitis C virus are not correlated with the host immune response: studies of infected donor-recipient clusters*. J. Virol. 74:2541-9.

Anderson, C. N. K., Ramakrishnan, U., Chan, Y. L., and Hadly, E. A. (2005). *Serial SIMCOAL: A population genetics model for data from multiple populations and points in time*. Bioinformatics 21:1733-1734.

Buendia, P., Collins, T., and Narasimhan, G. (2006a). *Reconstructing Ancestor-Descendant Lineages from Serially-Sampled Data: A Comparison Study*. International Conference on Computational Science (IWBRA06), Reading, UK. 807-814.

Buendia, P., Collins, T., and Narasimhan, G. (2006b). *The Role of Internal Node Sequences and the Molecular Clock in the Analysis of Serially-Sampled Data*. Submitted to International Journal on Bioinformatics Research and Applications.

Buendia, P. and Narasimhan, G. (2004). *MinPD: Distance-based Phylogenetic Analysis and Recombination Detection of Serially-Sampled HIV Quasispecies*. Proc. IEEE Comput. Sys. Bioinform. Conf., Stanford, CA

Buendia, P. and Narasimhan, G. (2006). *Serial NetEvolve: A flexible utility for generating serially-sampled sequences along a tree or recombinant network.* Bioinformatics 22:2313-2314.

Chun, T. and Fauci, A. (1999). *Latent reservoirs of HIV: obstacles to the eradication of virus*. Proc. Natl. Acad. Sci. U. S. A. 96:10958-61.

Cox, A. L., Mosbruger, T., Mao, Q., Liu, Z., Wang, X.-H., Yang, H.-C., Sidney, J., Sette, A., Pardoll, D., Thomas, D. L., and Ray, S. C. (2005). *Cellular immune selection with hepatitis C virus persistence in humans*. J. Exp. Med. 201:1741-1752.

Cunningham, C. W., Jeng, K., Husti, J., Badgett, M., Molineux, I. J., Hillis, D. M., and Bull, J. J. (1997). *Parallel molecular evolution of deletions and nonsense mutations in bacteriophage T7*. Mol. Biol. Evol. 14:113-6.

Drake, J. W. and Holland, J. J. (1999). *Mutation rates among RNA viruses*. Proc. Natl. Acad. Sci. U. S. A. 91:4821-4824.

Drummond, A., Nicholls, G., Rodrigo, A., and Solomon, W. (2002). *Estimating mutation parameters, population history and genealogy simultaneously from temporally spaced sequence data*. Genetics 161:1307-1320.

Drummond, A. and Rambaut, A. (2003). *BEAST v1.0*: http://evolve.zoo.ox.ac.uk/beast/

Drummond, A. and Rodrigo, A. G. (2000). *Reconstructing genealogies of serial samples under the assumption of a molecular clock using serial-sample UPGMA (sUPGMA).* Mol. Biol. Evol. 17:1807-1815.

Drummond, A. and Strimmer, K. (2001). *PAL: An object-oriented programming library for molecular evolution and phylogenetics.* Bioinformatics 17:662-663.

Etherington, G., Dicks, J., and Roberts, I. (2005). *Recombination Analysis Tool (RAT): a program for the high-throughput detection of recombination.* Bioinformatics 21:278-81.

Fan, J. and Robertson, D. (2006). *Links to recombinant sequence analysis/detection programs*: http://bioinf.man.ac.uk/recombination/programs.shtml

Felsenstein, J. (1981). *Evolutionary trees from DNA sequences: A maximum likelihood approach.* J. Mol. Evol. 17:368-376.

Felsenstein, J. (1999). *The Newick tree format*: http://evolution.genetics.washington.edu/phylip/newicktree.html

Felsenstein, J. (2004). *PHYLIP (Phylogeny Inference Package)* version 3.6. Distributed by the author. Department of Genetics, University of Washington, Seattle,

Flint, S. J., Enquist, L. W., and Krug, R. M. (2000a). *Foundations of Virology.* Pp. 3-21. Principles of Virology. ASM Press, Washington.

Flint, S. J., Enquist, L. W., and Krug, R. M. (2000b). *Genome Replication and mRNA production by RNA Viruses.* Pp. 163-196. Principles of Virology. ASM Press, Washington.

Flint, S. J., Enquist, L. W., and Krug, R. M. (2000c). *Multiple Facets of Human Immunodeficiency Virus Pathogenicity.* Pp. 631-632. Principles of Virology. ASM Press, Washington.

Flint, S. J., Enquist, L. W., and Krug, R. M. (2000d). *Retroviruses.* Pp. 762-764. Principles of Virology. ASM Press, Washington.

Flint, S. J., Enquist, L. W., and Krug, R. M. (2000e). *Reverse Transcription and Integration.* Pp. 207-208. Principles of Virology. ASM Press, Washington.

Flint, S. J., Enquist, L. W., and Krug, R. M. (2000f). *Virus Evolution and the Emergence of New Viruses.* Pp. 717-748. Principles of Virology. ASM Press, Washington.

Franco, S., Gimenez-Barcons M, Puig-Basagoiti F, Furcic I, Sanchez-Tapias JM, Rodes J, and JC., S. (2003). *Characterization and evolution of NS5A quasispecies of hepatitis C virus genotype 1b in patients with different stages of liver disease.* J. Med. Virol. 71:195-204.

Glass, G., Peckham, P., and Sanders, J. R. (1972). *Consequences of failure to meet assumptions underlying the fixed effects analysis of variance and covariance.* Review of Educational Research 42:237-288.

Grassly, N., Harvey, P., and Holmes, E. (1999). *Population dynamics of HIV-1 inferred from gene sequences.* Genetics 151:427-438.

Guindon, S., Rodrigo, A. G., Dyer, K. A., and Huelsenbeck, J. P. (2004). *Modeling the site-specific variation of selection patterns along lineages.* Proc. Natl. Acad. Sci. U. S. A. 101:12957-12962.

Hasegawa, M., Kishino, H., and Yano, T. (1985). *Dating the human-ape splitting by a molecular clock of mitochondrial DNA.* J. Mol. Evol. 22:160-174.

Hillis, D. M. (1999). *Phylogentics and the study of HIV.* Pp. 106-111 *in* K. A. Crandall, ed. The Evolution of HIV. The John Hopkins University Press, Baltimore.

Hillis, D. M., Bull, J. J., White, M. E., Badgett , M. R., and Molineux, I. J. (1992). *Experimental phylogenetics: generation of a known phylogeny.* Science 255:589-592.

Holmes, E. C., Zhang, L. Q., Simmonds, P., Ludlam, C. A., and Brown, A. J. (1992). *Convergent and divergent sequence evolution in the surface envelope glycoprotein of human immunodeficiency virus type 1 within a single infected patient.* Proc. Natl. Acad. Sci. U. S. A.:4835-4839.

Hudson, R. R. (1983). *Properties of a neutral allele model with intragenic recombination.* Theor. Popul. Biol. 23:183-201.

Huelsenbeck, J. (1991). *When are fossils better than extant taxa in phylogenetic analysis?* Syst. Zool. 40:458-469.

Huelsenbeck, J. and Ronquist, F. (2001). *MRBAYES: Bayesian inference of phylogenetic trees.* Bioinformatics 8:754-5.

Imamichi, H., Crandall, K., Natarajan, V., Jiang, M., Dewar, R., Berg, S., Gaddam, A., Bosche, M., Metcalf, J., Davey, R. J., and Lane, H. (2001). *Human immunodeficiency virus type 1 quasi species that rebound after discontinuation of highly active antiretroviral therapy are similar to the viral quasi species present before initiation of therapy.* Journal of Infectious Diseases 183:36-50.

Johnston, E., Zijenah, L., Mutetwa, S., Kantor, R., Kittinunvorakoon, C., and Katzenstein, D. (2004). *High frequency of syncytium-inducing and CXCR4-tropic viruses among human immunodeficiency virus type 1 subtype C-infected patients receiving antiretroviral treatment.* J. Virol. 77:7682-8.

Jukes, T. H. and Cantor, C. R. (1969). *Evolution of protein molecules.* Pp. 21-123 *in* H. N. Munro, ed. Mammalian Protein Metabolism. Academic Press, New York.

Kalinina, O., Norder, H., and Magnius, L. (2004). *Full-length open reading frame of a recombinant hepatitis C virus strain from St Petersburg: proposed mechanism for its formation.* J. Gen. Virol. 85:1853-1857.

Kimura, M. (1980). *A simple model for estimating evolutionary rates of base substitutions through comparative studies of nucleotide sequences.* J. Mol. Evol. 16:111-120.

Kingman, J. F. C. (1982). *The coalescent.* Stochastic Process. Appl. 13:235-248.

Kleinberg, J. and Tardos, E. (2005). *6.1 Weighted Interval Scheduling: A Recursive Procedure.* Pp. 864. Algorithm Design. Addison Wesley.

Korber, B., Muldoon, M., Theiler, J., Gao, F., Gupta, R., Lapedes, A., Hahn, B., Wolinsky, S., and Bhattacharya, T. (2000). *Timing the ancestor of the HIV-1 pandemic strains.* Science 288:1789-96.

Kumar, S., Tamura, K., and Nei, M. (2004). *MEGA3: Integrated Software for Molecular Evolutionary Genetics Analysis and Sequence Alignment.* Briefings in Bioinformatics 5:150-163.

Lin, C., Lin, K., Luong, Y., Rao, B., Wei, Y., Brennan, D., Fulghum, J., Hsiao, H., Ma, S., Maxwell, J., Cottrell, K., Perni, R., Gates, C., and Kwong, A. (2004). *In Vitro Resistance Studies of Hepatitis C Virus Serine Protease Inhibitors, VX-950 and BILN 2061.* Journal of Biological Chemistry 279:17508-14.

Liu, S., Mittler, J., Nickle, D., Mulvania, T., Shriner, D., Rodrigo, A., Kosloff, B., He, X., Corey, L., and Mullins, J. (2002). *Selection for human immunodeficiency virus type 1 recombinants in a patient with rapid progression to AIDS.* J. Virol. 76:10674-84.

Lole, K. S., Bollinger, R. C., Paranjape, R. S., Gadkari, D., Kulkarni, S. S., Novak, N. G., Ingersoll, R., Sheppard, H. W., and Ray, S. C. (1999). *Full-length human immunodeficiency virus type 1 genomes from subtype C-infected seroconverters in India, with evidence of intersubtype recombination.* J. Virol. 73:152-60.

Martin, D. P., Posada, D., Crandall, K., and Williamson, C. (2005a). *A modified bootscan algorithm for automated identification of recombinant sequences and recombination breakpoints*. AIDS Res. Hum. Retroviruses 21:98-102.

Martin, D. P., Williamson, C., and Posada, D. (2005b). *RDP2: recombination detection and analysis from sequence alignments*. Bioinformatics 21:260-262.

Meintjes, P. L. and Rodrigo, A. G. (2005). *Evolution of relative synonymous codon usage in Human Immunodeficiency Virus type-1*. Journal of Bioinformatics and Computational Biology 3:157-68.

Nei, M. and Kumar, S. 2000. *Molecular evolution and phylogenetics*. Oxford Univ. Press

Nickle, D. C., Jensen, M. A., Shriner, D., Brodie, S. J., Frenkel, L. M., Mittler, J. E., and Mullins, J. I. (2003). *Evolutionary indicators of Human Immunodeficiency Virus type 1 reservoirs and compartments*. J. Virol. 77:5540 - 5546.

Ogishima, S., Ren, F., and Tanaka, H. (2001). *Reconstruction and analysis of within-host longitudinal HIV-1 evolution by a distance-based sequential-linking algorithm*. Chem-Bio Informatics Journal 1(2):73-83.

Olsen, G. J., Matsuda, H., Hagstrom, R., and Overbeek, R. (1994). *fastDNAml: A tool for construction of phylogenetic trees of DNA sequences using maximum likelihood*. Comput. Appl. Biosci. 10:41-48.

Page, R. D. and Holmes, E. C. (1998a). *Inferring Molecular Phylogeny*. Pp. 172-227. Molecular Evolution: A Phylogenetic Approach. Blackwell Science Ltd.

Page, R. D. and Holmes, E. C. (1998b). *Measuring Genetic Change*. Pp. 135-171. Molecular Evolution: A Phylogenetic Approach. Blackwell Science Ltd.

Page, R. D. and Holmes, E. C. (1998c). *Models of Molecular Phylogeny*. Pp. 228-279. Molecular Evolution: A Phylogenetic Approach. Blackwell Science Ltd.

Penny, D. and Hendy, M. D. (1985). *The use of tree comparison metrics*. Syst. Zool. 34:75-82.

Posada, D. and Crandall, K. (2002). *The effect of recombination on the accuracy of phylogeny reconstruction*. J. Mol. Evol. 2002:396–402.

Posada, D. and Crandall, K. A. (2001a). *Evaluation of methods for detecting recombination from DNA sequences: computer simulations*. Proc. Natl. Acad. Sci. U. S. A. 98:13757-62.

Posada, D. and Crandall, K. A. (2001b). *Selecting models of nucleotide substitution: an application to human immunodeficiency virus 1 (HIV-1).* Mol. Biol. Evol. 18:897-906.

Quinones-Mateu, M., Gao, Y., and Ball, S. (2002). *In Vitro subtype recombinaints of Human Immunodeficiency Virus Type 1: Comparison to Recent and Circulating In Viva Recombinaint Forms.* J. Virol. 76:9600-9613.

Rambaut, A. (2000). *Estimating the rate of molecular evolution: Incorporating non-contemporaneous sequences into maximum likelihood phylogenies.* Bioinformatics 16:395-399.

Rambaut, A. and Grassly, N. (1997). *Seq-Gen: an application for the Monte Carlo simulation of DNA sequence evolution along phylogenetic trees.* Comput. Appl. Biosci. 13

Rambaut, A., Posada, D., Crandall, K. A., and Holmes, E. C. (2004). *The causes and consequences of HIV evolution.* Nat. Rev. Genet. 5:52-61.

Ray, S. (2003). *SimPlot for Windows* 3.4.

Ren, F., Ogishima, S., and Tanaka, H. (2001). *A new algorithm for analysis of within-host HIV-1 evolution.* Pacific Symposium on Biocomputing, Hawaii 595-605.

Ren, F., Ogishima, S., and Tanaka, H. (2003). *Longitudinal phylogenetic tree of within-host viral evolution from noncontemporaneous samples: a distance-based sequential-linking method.* Gene 317(1-2):89-95.

Robinson, D. F. and Foulds, L. R. (1981). *Comparison of phylogenetic trees.* Math. Biosci. 53:131-147.

Rodrigo, A. and Felsenstein, J. (1999). *Coalescent approaches to HIV-1 population genetics.* Pp. 233-272 *in* K. Crandall, ed. Molecular Evolution of HIV. Johns Hopkins University Press.

Rodrigo, A. and Steel, M. (2004). *DIMACS Working Group on Phylogenetic Trees and Rapidly Evolving Diseases*: http://dimacs.rutgers.edu/Workshops/WGPhylogeneticTrees/DIMACS_report.doc

Rodrigo, A. G., Goode, M., Forsberg, R., Ross, H. A., and Drummond, A. J. (2003). *Inferring Evolutionary Rates Using Serially Sampled Sequences from Several Populations.* Mol. Biol. Evol. 20:2010-2018.

Ross, H. A. and Rodrigo, A. G. (2002). *Immune-mediated positive selection drives Human Immunodeficiency Virus Type 1 molecular variation and predicts disease duration.* J. Virol. 76:11715-11720.

Saitou, N. and Nei, M. (1987). *The neighbor-joining method: a new method for reconstructing phylogenetic trees*. Mol. Biol. Evol. 4:406-25.

Salminen, M., Carr, J., Burke, D., and McCutchan, F. (1995). *Identification of recombination breakpoints in HIV-1 by bootscanning*. AIDS Res. Hum. Retroviruses 11:1423–1425.

Sanson, G., Kawashita, S., Brunstein, A., and Briones, M. (2002). *Experimental phylogeny of neutrally evolving DNA sequences generated by a bifurcate series of nested polymerase chain reaction*. Mol. Biol. Evol. 19:170-178.

Schierup, M. and Hein, J. (2000a). *Consequences of recombination on traditional phylogenetic analysis*. Genetics 156:879–891.

Schierup, M. and Hein, J. (2000b). *Recombination and the molecular clock*. Mol. Biol. Evol. 17:1578–1579.

Seo, T.-K., Thorne, J. L., Hasegawa, M., and Kishino, H. (2002). *Estimation of effective population size of HIV-1 within a host: A pseudomaximum-likelihood approach*. Genetics:1283-1293.

Shankarappa, R., Margolick, R. B., Gange, S. J., Upchurch, D., Farzadegan, H., Gupta, P., Rinaldo, C. R., Learn, G. H., He, X., Huang, X.-L., and Mullins, J. I. (1999). *Consistent viral evolutionary changes associated with the progression of HIV 1 infection*. J. Virol. 73:10489-10502.

Shriner, D., Shankarappa, R., Jensen, M., Nickle, D., Mittler, J., Margolick, J., and Mullins, J. (2004). *Influence of random genetic drift on human immunodeficiency virus type 1 env evolution during chronic infection*. Genetics 166:1155-64.

Siepel, A. and Korber, B. (1995). *Scanning the Database for Recombinant HIV-1 Genomes*. Pp. 35-60. Human Retroviruses and AIDS Compendium, Part III. Los Alamos National Laboratory.

Strimmer, K., Forslund, K., Holland, B., and Moulton, V. (2003). *A novel exploratory method for visual recombination detection*. Genome Biology

Swofford, D. L. (2000). *PAUP*: Phylogenetic Analysis Using Parsimony (and Other Methods). Version 4*. Sinauer Associates, Sunderland, Massachusetts.

Swofford , D. L., G J. Olsen, Waddell, P. J., and Hillis, D. (1996). *Phylogenic Inference*. Pp. 407-514 *in* B. K. Mable, ed. Molecular Systematics. Sinauer &Associates, Sunderland, MA.

Tsaousis, A. D., Martin, D. P., Ladoukakis, E. D., Posada, D., and Zouros, E. (2005). *Widespread recombination in published animal mtDNA sequences*. Mol. Biol. Evol. 22:925-933.

Williamson, S. (2003). *Adaptation in the env Gene of HIV-1 and Evolutionary Theories of Disease Progression*. Mol. Biol. Evol. 20:1318-1325.

Wiuf, C., Christensen, T., and Hein, J. (2001). *A simulation study of the reliability of recombination detection methods*. Mol. Biol. Evol. 18

World Health Organization. (2002). *The hepatitis C virus*: http://www.who.int/csr/disease/hepatitis/whocdscsrlyo2003/en/index2.html

Worobey, M. (2001). *A novel approach to detecting and measuring recombination: new insights into evolution in viruses, bacteria, and mitochondria.* Mol. Biol. Evol. 18:1425-1434.

VITA

PATRICIA BUENDIA

2001-2006	Ph.D. Candidate in Computer Science, Florida International University
1995	Technical University of Munich, Germany Master of Science, Computer Science, May 1995 Thesis Title "Incremental Mix-fix Parsing Generator"
1990	Technical University of Munich, Germany Bachelor of Engineering, Computer Science, Minor in Mathematics, May 1990

PUBLICATIONS

Buendia, P., Collins, T., and Narasimhan, G. (2006). Reconstructing Ancestor-Descendant Lineages from Serially-Sampled Data: A Comparison Study. International Conference on Computational Science (IWBRA06), Reading, UK. 807-814.

Buendia, P. and Narasimhan, G. (2004). MinPD: Distance-based Phylogenetic Analysis and Recombination Detection of Serially-Sampled HIV Quasispecies. IEEE Computational Systems Bioinformatics Conference, Stanford, CA.

Buendia, P. and Narasimhan, G. (2006). Serial NetEvolve: A flexible utility for generating serially-sampled sequences along a tree or recombinant network. Bioinformatics, *in Print*.

Buendia, P., Collins, T., and Narasimhan, G. (2006). *The Role of Internal Node Sequences and the Molecular Clock in the Analysis of Serially-Sampled Data.* Submitted to International Journal on Bioinformatics Research and Applications.

POSTERS

Buendia, G. Narasimhan, G. (2004). Using MinPD to study within-host HIV evolution. International Conference on Bioinformatics and its Applications (ICBA'04), Fort Lauderdale, Florida, December 2004.

AWARDS

Minority Biomedical Research Support (MBRS) Fellowship: 2002-2006

PROFESSIONAL EXPERIENCE

1997 - 2001	Vice President, Software Development & Research; Vision Lab Telecommunications Inc., Miami (Now Venali, Inc.)
1992-1997	Software Developer at SCG Munich eV and at SHS Munich Gmbh