Analysing malaria DBLα sequences with Hidden Markov models

Qian Feng

ORCID: 0000-0003-4301-8545

Submitted in total fulfillment for the degree of

Doctor of Philosophy

October 2024

School of Mathematics and Statistics /
Melbourne Integrative Genomics

THE UNIVERSITY OF MELBOURNE

THE UNIVERSITY OF MELBOURNE

Abstract

School of Mathematics and Statistics /
Melbourne Integrative Genomics

Doctor of Philosophy

by Qian Feng
ORCID: 0000-0003-4301-8545

The antigen PfEMP1 (*Plasmodium falciparum* erythrocyte membrane protein 1) plays a key role in the pathogenicity and immune evasion of the deadliest malaria parasite named *Plasmodium falciparum*. This antigen is encoded by a multi-gene family called *var*. *Var* genes evolve rapidly and have extreme diversity, allowing parasites to evade detection from the human immune system and remain in the human bloodstream. Previous studies have shown that recombination is one of the main mechanisms for maintaining the diversity of this gene family.

We focus on the DBLα tag in the DBLα domain of *var* gene. To fully understand recombination and its effects on the evolution of DBLα tags, we study the recombinant tags and their identification. More specifically, we seek to identify, from a large input dataset, which tags are recombinant and which are not.

In this thesis, we firstly develop an algorithm for detecting recent recombinants from a large dataset of unaligned sequences. This algorithm utilises a jumping hidden Markov model (JHMM) developed by Zilversmit et al., and can handle thousands of gene-length sequences. We demonstrate its accuracy through biologically realistic simulations. Applying this algorithm to a real dataset collected from Ghana, we derive a series of novel findings; for instance, non-recombinant tags are more conserved than recombinant ones.

Next, to make this algorithm applicable to the larger datasets that are being generated today, we propose a modified JHMM that constrains recombination jump destinations. We show the accuracy and efficiency of this model through simulation-

s, while the accuracy of identifying recombination with our model is comparable with one using unconstrained model. We demonstrate the utility of this model by applying it to a large longitudinal dataset of DBLα tags from Ghana.

We finally focus upon the classification of upstream sequences (ups) of *var* genes using DBLα tags. We design a profile HMM-based approach to classify DBLα tags into three ups groups, improving existing pipelines which only classify them into two groups. Our method also includes a quantification of group membership probability. We show this algorithm's effectiveness through cross-validation on a global set of DBLα tags; importantly, setting a threshold on inferred probabilities further improves classification performance.

Declaration

I, Qian Feng, declare that this thesis titled, 'Analysing malaria DBLα sequences with Hidden Markov models' and the work presented in it are my own. I confirm that:

- The thesis comprises only my original work towards the Doctor of Philosophy degree except where indicated in the preface;
- due acknowledgement has been made in the text to all other material used; and
- the thesis is fewer than the maximum word limit in length, exclusive of tables, maps, bibliographies and appendices as approved by the Research Higher Degrees Committee.

Signed: Qian Feng

Date: 14/10/2024

Preface

This thesis consists of six chapters, Chapter 1 and 2 are general introduction and literature review, Chapter 6 is the conclusion. Remaining three chapters present research work. All the work in this thesis is original.

- Contributions to each chapter.
  - Contributions to Chapter 1, 2 and 6. Qian Feng was the sole contributor.
  - Contributions to Chapter 4 and 5: The research and written work were conducted by Qian Feng, with guidance from Yao-ban Chan and Heejung Shim.
  - Contributions to Chapter 3: this chapter was a slight modification of our publication in *Bioinformatics* on the 13th of Jan, 2022, entitled "An accurate method for identifying recent recombinants from unaligned sequences". Qian Feng was the first author. Researcher from School of BioSciences, Bio21 Molecular Science and Biotechnology Institute, Department of Microbiology and Immunology, The University of Melbourne, and the Peter Doherty Institute for Infection and Immunity (Kathryn E Tiedje, Shazia Ruybal-Pesántez, Karen P Day), Bioinformatics Division, Walter and Eliza Hall Institute of Medical Research, Parasites and Microbes, Wellcome Sanger Institute, Wellcome Genome Campus (Gerry Tonkin-Hill) and Peter Doherty Institute for Infection and Immunity (Michael F Duffy), Melbourne Integrative Genomics/School of Mathematics and Statistics, The University of Melbourne (Heejung Shim and Yao-ban Chan) all contributed this published paper, with Yao-ban Chan as the corresponding author.
- None of the work in this thesis has been submitted for other qualifications. No third party editorial assistance was provided in preparation of this thesis.
- The research presented in this thesis was funded by China Scholarship Council - University of Melbourne PhD Scholarship, COVID-19 Student Support Grant and Dawson Bursary from the University of Melbourne.

Acknowledgements

I want to express my greatest appreciation to my principal supervisor, Dr. Yao-ban Chan. A person might have many teachers, but only a few of them are remembered and mentioned frequently. Yao-ban belongs to that small group! He is patient and never pushes me to do work, such freedom makes me enjoy research. He also supported me personally and professionally when I was impacted by the travel ban and studying away. He is good at inspiring and providing guidance when delivering knowledge. Moreover, his profession, rigorous attitude and perfectionism toward research improve the quality of my PhD work. Working with such a charismatic supervisor, I feel lucky and very proud to be his student in every minute of my PhD.

I am also grateful to my co-supervisor, Dr. Heejung Shim, who always spares time to provide me with advice and support, both professional and personal. At the initial stage of my PhD, I experienced culture and linguistics shock. Heejung was not only a supervisor but also an elder 'sister'. Her considerate and enthusiasm made me feel warm and a sense of belonging. Without her support, I am not able to go through that tough period. During the PhD, she teaches me how to be a good scientist and team collaborator tirelessly. As a female scientist, she is my role model.

My sincere thanks to the rest of my committee team, Prof. David Balding and A/Prof. Guoqi Qian, for their helpful comments and support. Many thanks to all the collaborators and all the friends in the University of Melbourne.

Finally, I would like to thank my parents for their unconditional love, tolerance and sacrifice, especially during the challenging two years (that I had to work from home) due to COVID-19 pandemic. Thank my cute puppy, Xiaoheizi, for offering me tremendous joy. Thank myself, I have done my best for persuing a PhD degree.

Contents

List of Figures

List of Tables

List of Abbreviations

PfEMP1	*Plasmodium falciparum* erythrocyte membrane protein 1
IEs	infected erythrocytes
TM	transmembrane
DBL	Duffy-binding like
CIDR	cysteine-rich interdomain region
ATS	intracellular acidic terminal segment
IRS	indoor residual spraying with insecticides
HMM	hidden Markov model
OTU	operational taxonomic unit
PCA	principal component analysis
FFP	feature frequency profiles
JS	Jensen-Shannon
HB	homology block
JHMM	jumping hidden Markov model
PAC	product of approximate conditionals
indel	insertions and deletions
ups	upstream sequences
profile HMM	profile hidden Markov model
SCM	sub-pixel confusion matrix
oa	overall accuracy
RMSE	root mean square error
MCL	Markov clustering
NJ	neighbour joining

Chapter 1

Introduction

1.1 Malaria

Malaria is one of the most common infectious diseases [1, 2] caused by parasites. These parasites are generally transmitted to humans by bites from infected mosquitoes, in rare situations, from mother to baby [3]. When the parasites enter the human blood, they will infect and modify the red blood cells and cause the symptoms like high fevers, chills, sweating, headache, and muscle or joint pain.

Without prompt diagnosis and treatment, malaria can be severe and sometimes fatal. After hours or days of the initial symptoms, complications of severe malaria could occur and lead to the failure of different organs. For instance, the blocked blood vessels by parasites in the brain would cause cerebral malaria, which may lead to permanent brain damage and various clinical manifestations with age [4]. Moreover, malaria is one of the leading causes of death in many developing countries.

1.1.1 Disease burden

Though the malaria case incidence and mortality rate declined slowly from 2000 to 2022 (81.0 to 58.4 per 1000 population, 0.30 to 0.14 per 1000 population, respectively), there were still an estimated 249 million malaria cases, and 608,000 malaria-related deaths globally in 2022 [5]. In the past decades, malaria kills an estimated 500,000 persons annually worldwide [6], mostly African children.

Malaria is mostly distributed in tropical and subtropical areas of the world, which are also habitats of mosquitoes with suitable temperatures, humidity and rainfall. Among all endemic countries, WHO African Region saw the most malaria cases (82%) and deaths (95%) in the past two decades [5]; its malaria case incidence and mortality rate could go up to 369.3 and 142.6 per 1000 population.

The most susceptible populations of malaria are children under five, pregnant and travellers who recently travelled to a malaria-transmission country. Children and travellers are the people who do not develop immunity to malaria earlier. In addition, pregnancy decreases the immunity of pregnant women and may further affect the immunity of newborns [7, 8].

Malaria imposes enormous costs on nations, governments, communities, families and individuals. The direct economic cost is estimated US $12 billion annually [9]. Though the total funding for malaria control and elimination (for example, US National Institutes of Health, Bill & Melinda Gates Foundation) increased from 2019 to 2022, the gap between funding received and the resources needed has widened dramatically during the same period [5].

There are several global strategies to accelerate the control and elimination of malaria. World Health Assembly adopted the *Global technical strategy for malaria 2016–2030* (GTS) in 2015 [10]. The GTS aimed for the malaria case incidence and mortality rate to decrease by at least 40% by 2020 and 90% by 2030 compared with the 2015 baseline. Even though considerable progress has been made since 2000, only 25% of the countries (23 of the 93 malaria-endemic countries) achieved this milestone [5]. Regarding malaria elimination, WHO has supported 21 countries with the same goal — eliminate malaria by 2020 through the "E-2020 initiative", unfortunately only eight countries (38%) reported zero indigenous cases [11]. Afterwards, WHO launched the "E-2025 initiative" to support more countries.

Unsurprisingly, the COVID-19 pandemic is one of the reasons why these goals have not been achieved [12–16]. The majority of African countries suffered moderate levels of disruption; as a result, the number of malaria cases and deaths in 2020 and 2021 increased significantly compared with 2019 [5, 17]. To some extent, this pandemic delays malaria control and elimination. All in all, urgent action needs to be taken to reverse the trend; otherwise, future milestones from GTS or "E-2025 initiative" are highly likely not to be met.

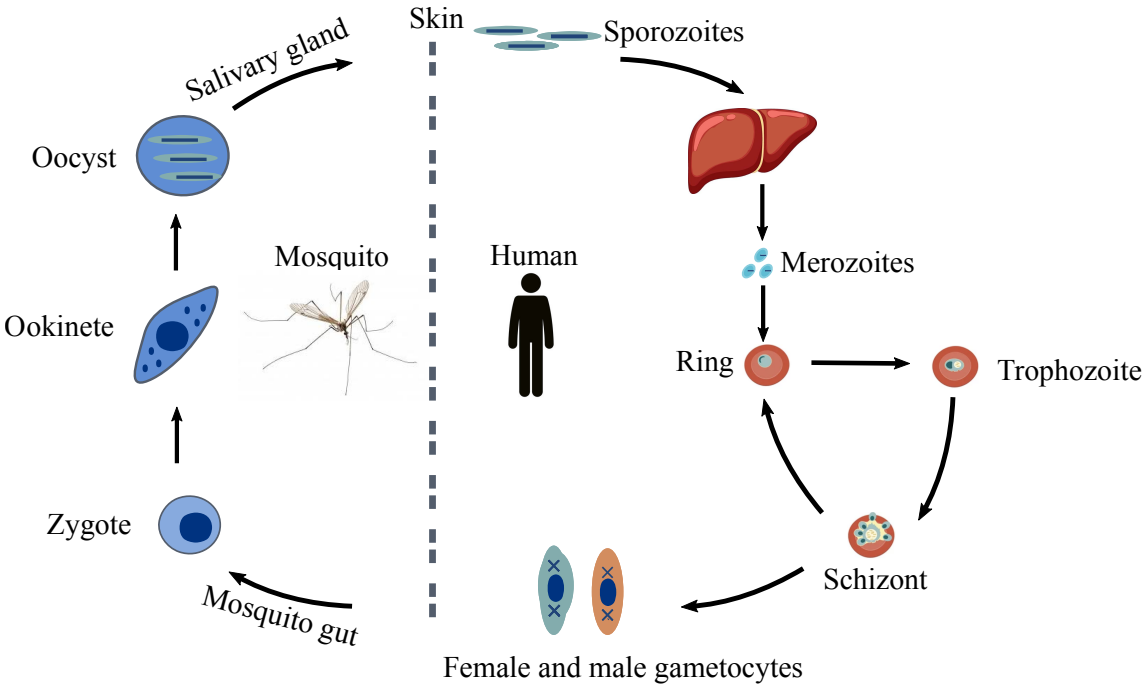


FIGURE 1.1: **Life cycle of the malaria parasite.** In the human infection stage, the sporozoites enter into human liver, and replicate themselves before releasing thousands of merozoites. Merozoites travel in the human blood and infect red blood cells (RBCs). They replicate asexually and transmit from infected RBCs to other uninfected RBCs. A small amount of parasites undergo sexual reproduction and generate female and male gametocytes. In the mosquito infection stage, after taking a meal from infected individuals, gametocytes progress to zygotes in mosquito gut and further form the sporozoites through salivary gland. The infected mosquito could infect a new human host.

1.1.2 *Plasmodium* parasites

Malaria parasites exist in both infected mosquitoes and humans. We show the life cycle of parasites in Figure 1.1. When the parasites reproduce in the human blood after leaving the liver, malaria symptoms start to appear. So the prevention and treatment of malaria are conducted either by vector control — stop the bites of malaria-carrying mosquitoes (for example, using indoor residual spraying, long-lasting insecticidal bed nets, insect repellent on the skin, wearing long-sleeved clothing and long pants), or antimalaria drugs — kill the parasites in human blood. These drugs include artemisinin (the most effective class to the deadliest parasite), chloroquine, doxycycline, mefloquine, quinine, primaquine and halofantrine [18].

Unfortunately, the tenacious survivability of parasites despite the usage of medicine leads to continuously emerging drug resistance, the antimalaria drug efficacy is thus affected in some regions of the world [19–23]. For instance, artemisinin has been

monitored partial resistance in east Africa [5, 19], which may have devastating consequences [20]; chloroquine resistance even spreads to nearly all malaria transmission areas of the world [24].

Given this worrying situation, in our research, we focus on investigating the evolutionary mechanisms of parasites. We believe the knowledge helps inform how the evolutionary systems respond to our attempts to control malaria. For instance, how much do we need to reduce the reservoir of infection in order to see measurable effects. The knowledge also helps slow the course of parasites to develop drug resistance. Therefore, studying parasites' evolutionary mechanisms is of great significance to accelerate the control and elimination of malaria.

Malaria parasites belong to genus *Plasmodium*, and there are more than 200 *Plasmodium* species which can infect birds, reptiles and mammals [25]. Nowadays, five malaria parasites species can infect humans [26].

1. *Plasmodium falciparum* [27–29]. This most prevalent mosquito parasite has caused 300,000 deaths each year and 200 million clinical cases. It can cause severe anemia due to blood loss and fatal cerebral malaria. It is mostly found in tropical and subtropical areas.

2. *Plasmodium vivax* [30]. It is mainly found in Asia, Oceania and Latin America in locations with high human population densities. An estimated 16 million clinical malaria cases are from *P. vivax*.

3. *Plasmodium ovale* [31]. It is prevalent in sub-Saharan Africa where many individuals are negative for the Duffy blood group, as *P. ovale* could infect this special population group. Recently *P. ovale* is found to consist of two distinct species *P. ovale curtisi* and *P. ovale wallikeri* [32].

4. *Plasmodium malariae* [33]. It has a longer life cycle and infection period. If untreated with its infection, chronic infection can sometimes last the host's whole lifetime.

5. *Plasmodium knowlesi* [34]. A zoonotic malaria is found in Southeast Asia, particular in Malaysia. After a 24-hour cycle in the human body, this leads rapidly to severe infection.

Among all the above parasite species, *Plasmodium falciparum* is the deadliest [5], accounting for most malaria cases and deaths worldwide. *P. falciparum* has evolved

from a well-known *Plasmodium* subgenus called *Laverania* [35] which are present in African gorillas, and *P. falciparum* is the only extant species that has transferred from gorillas and adapted to human beings [35, 36]. Moreover, it has developed resistance to nearly all available antimalarial drugs [37]. Though RTS,S and R21/Matrix-M are the only two WHO-recommended vaccines that can kill *P. falciparum* [5], these two vaccines only target children without full protection/eradication and lack corresponding implementation studies [38, 39]. Parasite *P. falciparum* is therefore the subject of our research.

1.1.3 *Plasmodium falciparum* erythrocyte membrane protein 1

Plasmodium falciparum erythrocyte membrane protein 1 (PfEMP1) acts as both an adhesion and antigen protein, playing a key role in the pathogenicity and immune evasion of *P. falciparum* [40]. PfEMP1 is encoded from parasite genomes and expressed on the membrane surface of red blood cells. *P. falciparum*-infected erythrocytes (IEs) could bind to other uninfected red blood cells, making the parasites transmit to other cells and leading to fatal symptoms (for instance cerebral malaria [41]). PfEMP1 is also an antigen, a major target for antibody and human immune response [40]. It aids the parasites in evading the detection of the human immune system. Consequently, the parasites could stay in the human bloodstream for a long time and may lead to waves of parasitemia or human reinfection.

PfEMP1 is encoded by a hyper-variable and large multigene family called *var*. *Var* genes evolve rapidly, millions of new *var* structures might be generated in only a day per infected individual [42, 43]. There are ∼60 (range = 47-90 [44]) *var* genes in each parasite genome [45], around 2/3 of *var* genes are distributed in subtelomeric regions while the remaining 1/3 are found in central chromosome regions [46].

Despite being highly polymorphic within and between individuals [42], *var* genes share the conserved architecture (see Figure 1.2). Each *var* has a two-exon structure. Exon 2 (≈1.5kb) is shorter and more conserved than Exon 1 (≈4-10kb). Exon 2 encodes intracellular acidic terminal segment (ATS) [47], and the first exon encodes the extracellular and transmembrane (TM) region. Exon 1 consists of an N-terminal segment (NTS), a combination of Duffy-binding like (DBL) and cysteine-rich interdomain region (CIDR) domains, followed by a TM region.

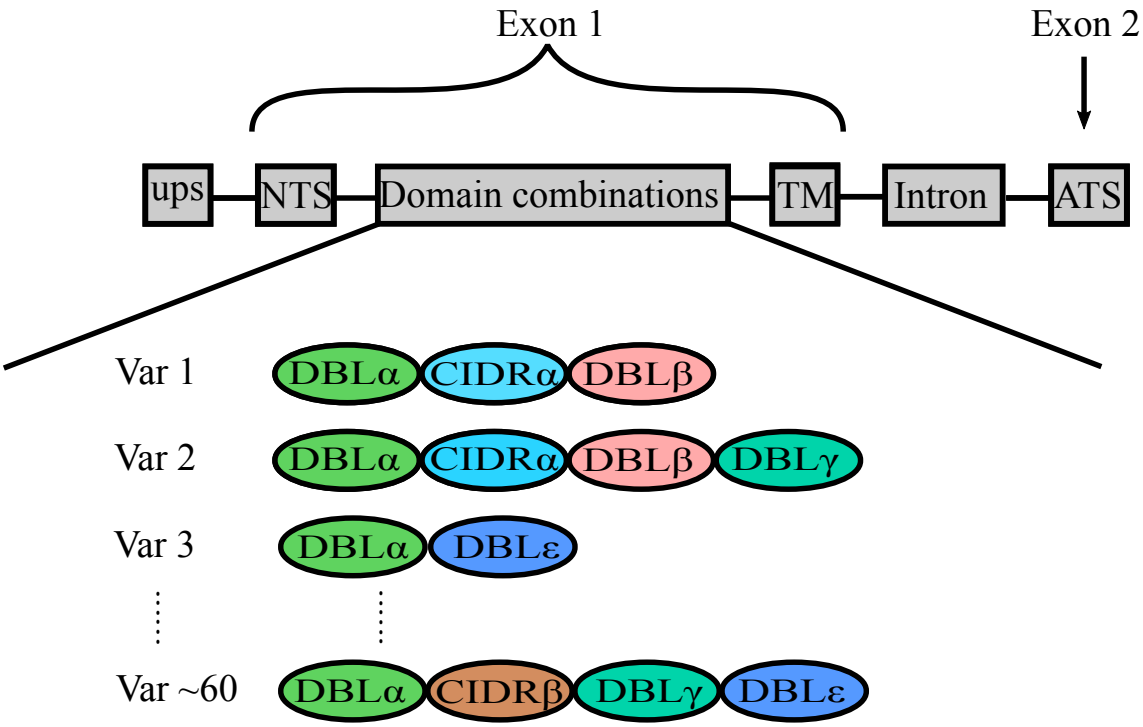
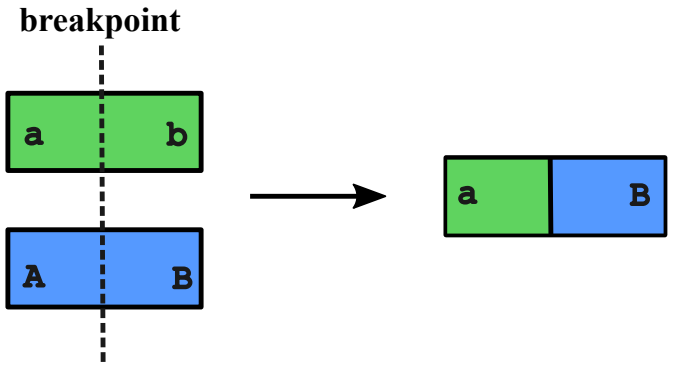


FIGURE 1.2: ***Var* architecture.** The top of the figure shows the architecture of *var* genes. Different *var* genes have different combinations of DBL and CIDR domains. Among all domain classes, the DBLα domain is encoded by all (except *var2CSA* [49–51]) members of the *var* family.

Based on amino acid sequence similarity, the DBL domain is divided into six major classes (DBL$\alpha, \beta, \gamma, \delta, \varepsilon, \zeta$), and the CIDR domain is divided into four classes (CIDR$\alpha, \beta, \gamma, \delta$) [48]. The overall architecture of *var* genes is highly variable regarding the total number of domain classes and their order.

1.1.4 Recombination is one of the main mechanisms for *var* genes

Previous studies [42, 52, 53] have shown that the primary mechanisms for maintaining the diversity of *var* genes are single-base mutation and frequent recombination. Point mutation is evident in highly similar protein regions without changing too much function, and it might partially contribute to parasite evasion of host immunity [54, 55]. Recombination is a process by which pieces of gene sequences are broken and recombined, producing a new combination of genes. The position where gene sequences are "spliced" is called a breakpoint (See the top panel of Figure 1.3).



Parent 1: DIGDIVRGKDLYLGDRKEKVKLEKRLIEIFKNITKNNYSTLK
DLSLEQVREYWWALNREDVWKALTCNAPDKAEYFVYKPDRLR
KFSNSKCGHGEHEVLTNLDYVPQYLR

Parent 2: DIGDIIRGKDLFLGTYQEKKSLEENLKNIFRKLYKELTKYKE
NEAVIKSRYENDGPNYYQLREDWWALNRKEIWKAITCDTEES
DTYFKQSSEGKYSFTNGQCGHNEENVLTNLDYVPQYLR

Parent 3: DIGDIIRGKDLYLGYDDKEKKRRKQLENNLKDIFAKIHSDVT
SGRNKRTNSALQTRYQHDAPDYYQLREDWWTVNRNQVWEAIT
CEAPKDANYFIGSGNKSKSFSNPKCGHNENKVLTNLDYVPQF
LR

Parent 4: DIGDIVRGRDMLRGTGNEKDQLENNLKKIFGKIHEGLTTTRG
KNAEELKARYQDENGGNYYKLREDWWTANRDQVWKAITCSAG
QKDTYFIKPNGRVYSFADAYCGRGDENVPTNLDYVPQFLR

Child: DIGDIIRGKDLYLGDRKEKVKLEENLKNLFKNLYKELTKDKK
NAALQTRYGSDAPDYFQLREDWWTANRDQVWKAITCSAGQKD
TYFIKPNGRVYSFADAYCGRGDENVPTNLDYVPQFLR

FIGURE 1.3: **Visualization of recombination process and an example with DBLα tags.** For the top panel, two genes (represented by green and blue rectangles) are rearranged, and the resulting recombinant is shown on the right side. This figure demonstrates the recombination of genes with a single recombination breakpoint. The bottom panel shows an example of a recombinant DBLα tag (child) and its four parents. All the segments of the child inherited from various parents are colored separately.

Meiotic and mitotic recombination are the two major types of recombination involved in the *var* genes. Both of them can generate novel *var* genes. Meiotic recombination occurs in the mosquito abdomen. Recombination happens between genomes as parasites are in sexual reproduction. Mitotic recombination occurs in human blood, where the parasites spend most of their life cycle. Parasites are in the non-sexual stage, and recombination happens within the same genome. It has been found that the extreme diversity of *var* genes is mainly due to meiotic recombination [53, 56], followed by the mitotic recombination [57]. In our research, although the malaria parasites we aim to analyze are from the human blood, there is *var* gene expression during asexual development of parasites in human blood and *var* gene transcription in the mosquito infection stage [58], so both types of recombination play a role in the evolution of *var* genes.

Recombination is not randomly distributed across the *var* gene. Based on parasite genomes of isolates around the globe, most recombination events occur in DBL domain, followed by CIDR domain, and other regions; in particular, DBLδ is detected to have the most breakpoints [59]. A significant recombination hotspot is found between two DBL subdomains [46]. In addition, recombination tends to happen between more closely related domains due to their high similarity [46, 60].

Despite plenty of research [42, 52, 54, 56, 57, 59–62] about *var* recombination, the evolutionary history of *var* genes is still difficult to describe due to the presence of frequent recombination. In general, when the genealogy is simple and does not involve any horizontal process (such as recombination, horizontal gene transfer, hybridization, introgression or reassortment [63]), to depict the gene sequences' evolution, we use a *phylogenetic tree* where tips represent the present-day genes, and the tree structure (topology) shows the evolutionary descent from a common ancestor. Unfortunately, in the presence of recombination, a *phylogenetic network* is a more appropriate representation of *var* genes' evolutionary history. It is also extremely challenging to construct a phylogenetic network for the sequences with modern-day data size [64, 65]. Furthermore, frequent recombination precludes the phylogenetic analysis. Therefore, quantifying the recombination events and related sequences is a crucial step for understanding the *var* genes' evolution, with implications in malaria interventions.

While we aim to understand the evolution of full-length *var* genes, their complex domain class composition makes it difficult to study. Therefore, we focus on the DBLα domain, which is the most conserved domain [66] among all domain classes.

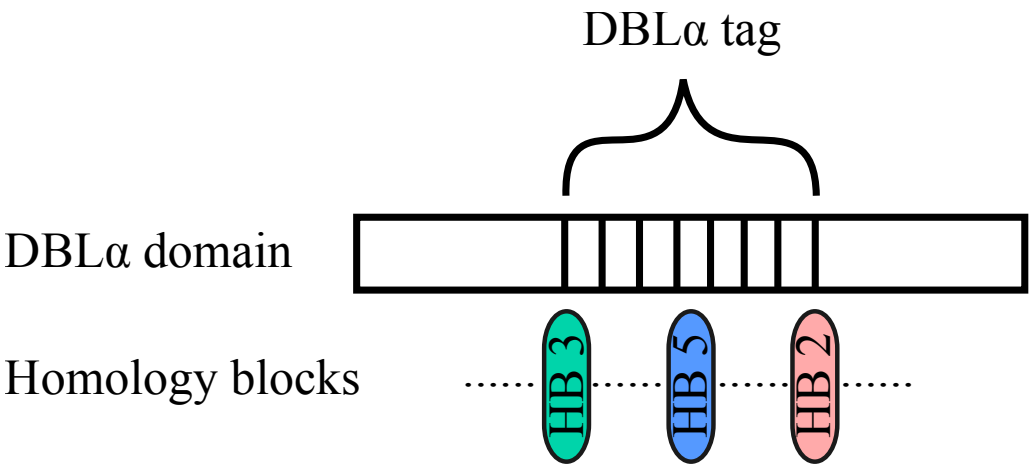


FIGURE 1.4: **DBLα structure.** The top panel shows the DBLα tag is a small region of DBLα domain. The bottom panel is a modification of Figure 1b from [81]. The combination of homology blocks and highly variable regions (dotted line) demonstrate the composition of DBLα tag and its surrounding.

It's encoded by all (except *var2CSA* [49–51]) members of the *var* family. The DBLα domain is also immunogenic and has the potential for vaccination [67, 68]. However, the DBLα domain itself is still highly variable, and a large number of recent studies [62, 69–76] have focused on a small region (100-500 base pairs) of this domain called the DBLα tag (Figure 1.4). Importantly, this tag can provide functional [69] and structural [77, 78] information for the full-length *var* genes. The degenerate primers for DBLα tag amplification from *var* genes have also been designed [56], making extensive tags available. For illustration, we selected five DBLα tags arbitrarily from the 3D7 reference genome [44] and aligned them with ClustalW v2.1 [79] using default parameter settings. We visualized the alignment with Alignment viewer in varDB [80], see Figure 1.5. Taken together, we study the DBLα tag rather than the entire *var* gene or even the whole genome.

Regarding the tag structure, it's known that within the *var* genes, there are some semi-conserved and ancient segments [82] called homology blocks (HBs) [46]. For the DBLα tag and its surrounding, there are three major HBs (HB3, HB5 and HB2), as shown in Figure 1.4. There are also plenty of less frequent HBs within the DBLα tag, such as HB14, HB36, HB54 etc. The numbers after HB represents the frequency of specific HB. The smaller the number, the greater frequency. The most frequent homology block is HB1, HB1 and HB4 are both outside of the DBLα tag [46] and so the most frequent ones related with DBLα tags are HB2, HB3 and HB5. Although the existence of semi-conserved HBs, the detected HBs per DBLα

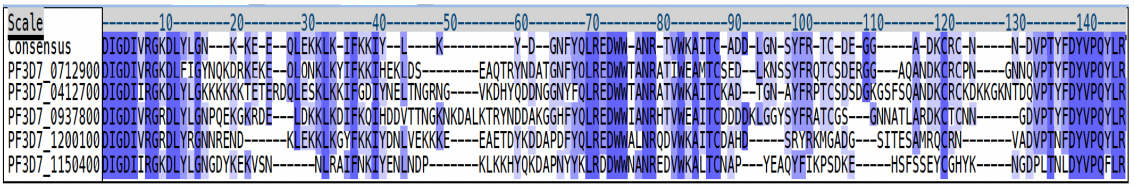


FIGURE 1.5: **An alignment of five DBLα tags from the 3D7 reference genome with ClustalW.** The color scheme was based on conservation level per alignment column. There are conserved motifs at the central and both ends of the DBLα tag.

tag (HB combination) is variable [82]. The regions between HBs are also highly variable, making the alignment of a large number of DBLα tags unreliable.

1.2 DBLα tags from a cross-sectional study in Ghana

Our empirical dataset is from the malaria parasites in a well-designed and implemented study [73, 83, 84] in the Bongo district of Ghana. Researchers assembled DBLα tags from *P. falciparum* confirmed individuals (*isolates*) [73, 83, 84]. This dataset is the only longitudinal data available in the same location for asymptomatic population [85].

In this study, researchers aimed to evaluate how age, geographical locations, seasonality or other relevant factors affect the epidemiology of asymptomatic *Plasmodium falciparum* infections. They selected Ghanaian participants of various age groups residing in two catchment areas of Bongo District (Vea/Gowrie and Soe) where malaria is the most prevalent public disease. The two catchment areas were selected as Vea/Gowrie and Soe represented the irrigated and non-irrigated area separately, and it was hypothesized that the prevalence of *P. falciparum* infections between these two areas might differ [84, 86]. Ghana is characterized by a short wet season (June-October) and a long dry season (November-May); this study conducted a series of surveys at the end of a multi-year wet or dry season. Specifically, a pilot study and six surveys were carried out sequentially from 2012 to 2016. In addition, IRS (indoor residual spraying with insecticides) interventions were also conducted. See Table 1.1 for each survey's completed date, corresponding season, IRS stage and number of DBLα tags. These DBLα tags form our empirical dataset (GenBank BioProject Number: PRJNA396962).

TABLE 1.1: **Summary of empirical data.**

	Sampling time	Season	Intervention	Number of DBLα tags
Pilot	June 2012	Dry	–	35,591
Survey 1	October 2012	Wet	Pre-IRS	134,877
Survey 2	June 2013	Dry	Pre-IRS	82,249
Survey 3	June 2014	Dry	IRS	65,837
Survey 4	October 2014	Wet	IRS	35,378
Survey 5	October 2015	Wet	Post-IRS	44,314
Survey 6	June 2016	Dry	Post-IRS	37,632

1.3 Project aims

With these DBLα tags, we would like to quantify the recombination events in their evolution and identify which tags are recombinant and which are not. These would help us to understand the DBLα tags' recombination patterns and functional changes that occur with recombination, such as the frequency of recombinant tags, the distribution of breakpoint locations and the recombination rate. In addition, separating the recombinant and non-recombinant tags will facilitate direct comparisons between them. These are the fundamental steps for describing the evolution of DBLα tags. Overall, our primary project goal is to understand the evolutionary processes behind DBLα evolution.

The first systematic attempt to quantify the recombination of DBLα domains was performed by Zilversmit *et al.* [54]. They proposed a statistical model called the jumping hidden Markov model (JHMM) and detected the recombination events from the DBLα domains of *P. falciparum* and a closely related species, *P. reichenowi*. A detailed overview is in Section 2.2.2.2. The advantages of the JHMM are that it can estimate the recombination rate and infer the breakpoints from a dataset. However, knowing the breakpoints does not always inform which sequences are the recombinant. Specifically, when applying the JHMM, there are two major issues.

- When input sequences are collected from the same time point, the JHMM cannot identify which sequences are recombinant and which are not.
- When the input data is a mix of sequences which are collected from different time points, though the JHMM can detect the recombinant sequences, it is computationally prohibitive when dealing with a large dataset.

In terms of why the JHMM has such issues, it's due to the identifiability problem with a phylogenetic view. We expand this later in Chapter 3, please see Section 3.5.1.1 for details.

Since we aim to detect the recombination using the data within each survey and across surveys, we need to address both of these issues. Therefore, these two problems form my first and second project goals separately. In the following, I show the main difficulty of each issue and planned analysis in Section 1.3.1 and 1.3.2. Regarding my third project, although we do not focus on analyzing the recombination like previous projects, the recombination still plays an important role in it. The subject of this project is still the DBLα tag. A brief background and associated project goal are provided in Section 1.3.3.

1.3.1 Identifying recombinants from DBLα tags

In this project, we aim to develop an accurate algorithm to detect recombinant DBLα tags, when the tags are collected at the same time. We also aim to apply this algorithm to the Ghana pilot dataset and analyze the recombination patterns.

Although there are plenty of recombinant detection methods [87–113], nearly all of them rely on a multiple sequence alignment or a reference panel of non-recombinants (see our literature review at Chapter 2). Unfortunately, we can not provide a reliable alignment nor reference as the DBLα tags are too diverse for generating high-quality alignments. DBLα tags are diverse in terms of length (e.g. length varies from 110nt to 504nt in our data shown in Table 1.1) and alignment quality by various standard alignment methods [71]. Specifically, the regions between conserved homology blocks are hyper-variable [46, 81], if we force to align DBLα tags with standard alignment methods, long gaps would be introduced at regions with low homology [54]. Take the Gismo [114] alignment of over 30,000 DBLα tags collected from ten countries [62] as an example, its alignment length using translated DBLα tags is 748, while the average sequence length is only 124aa. Different alignment methods generally introduce gaps in different ways, resulting lots of diverse alignments given the same data. As a result, the existing recombinant detection methods fail when dealing with our unaligned DBLα tags.

Additionally, the number of available DBLα tags is always large, as the reservoir of infection is large [5, 115, 116]; in our case, it is over 10,000. Taken together,

these require a novel algorithm to detect recombinants from unaligned sequences (like DBLα tags) within a practical time.

1.3.2 Identifying recombination from a longitudinal dataset of DBLα tags

In this project, we aim to design an efficient algorithm to detect recombination from the longitudinal dataset.

Although the JHMM can already detect the recombination from DBLα domains, the model itself is still not biologically reasonable [54]. We have also validated this. Specifically, the JHMM allows recombination between any two points in a pair of sequences. Our study reveals that recombination only happens at the roughly same normalized locations in a pair of sequence (Appendix A.2.2.4). Because of this, we aim to improve the JHMM by adding constraints into the model. We believe this constrained JHMM would help reduce the number of hidden states of the model, thus reducing the running time. This is greatly beneficial for a much larger dataset, that the conventional JHMM cannot handle.

Regarding the real data application, we focus on detecting recombination from surveys finished at the end of wet seasons, as malaria infection prevalence in wet seasons is usually higher than in dry seasons. After applying our model to this time series data, we aim to analyze whether the recombination patterns change with time.

1.3.3 Classifying malaria *var* genes to ups groups

In this project, we focus on another segment of *var* gene, the upstream sequence (ups). It is located before the NTS region of *var* gene, as shown in Figure 1.2. Based on the ups similarity, *var* genes have been classified into three major groups (upsA, upsB, upsC) [45, 117, 118]. Researchers have analyzed the 2000bp 5' flanking sequence (a *var* gene promoter element [119]) from the upstream sequence and confirmed the ups grouping into three major types by phylogenetic analysis [117]. Figure 1.6 shows an example of three 2000bp upstream sequences belonging to upsA, B and C groups. Different groups of *var* genes are strongly associated with various chromosome locations, transcription orientations [118], and, importantly, disease

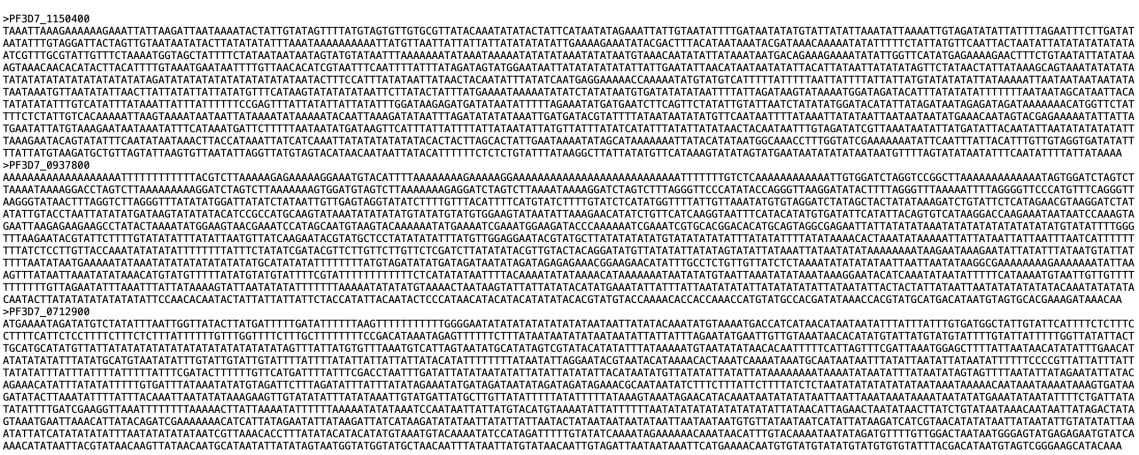


FIGURE 1.6: **Three 2000bp upstream sequences from the 3D7 reference genome which belong to upsA, B and C groups separately.** Three upstream sequences were chosen arbitrarily from 3D7 reference genome [44]. The sequences in this graph from top to bottom belong to upsA, upsB and upsC group separately.

severity. Classifying ups groups of the *var* genes is therefore crucial for preventing and diagnosing malaria.

In this project, we aim to propose an algorithm to classify the *var* genes into ups groups. A natural way is to extract the ups sequences and perform the classification directly. However, the number of publicly available ups sequences is very limited. So we aim to design a method that uses the DBLα tags (instead of ups) for classifying the *var* genes into ups groups.

1.4 Structure of this thesis

The remaining of this thesis is organized as below.

Chapter 2 reviews the literature. In this chapter, I outline the existing methods for identifying recombinants and recombination. In particular, I illustrate the fundamental method for modelling DBLα domain's recombination — Zilversmit's model [54], and its theoretical foundation — Li and Stephens' model [120]. I also provide discussions about these methods' limitations.

Chapter 3 describes our proposed algorithm for identifying recent recombinants from unaligned sequences. This algorithm combines information from mosaic representations of the JHMM and distance-based comparisons to identify recombinants. We

evaluate its performance by extensive simulations and comparisons with other existing recombinant detection methods. Finally, the application to an empirical dataset uncovers a series of biologically meaningful results.

We design an improved JHMM to detect recombination in a larger dataset of DBLα tags. This new model is introduced in Chapter 4. Its effectiveness and efficiency are demonstrated through simulations. Compared with the JHMM, our model generates similar mosaic representations in a much shorter time. Ultimately, we apply our model to the large longitudinal dataset of Ghana, and corresponding recombination patterns are discussed.

Chapter 5 presents a probabilistic algorithm to classify ups groups using DBLα tags. Compared with the existing approach that classifies ups into upsA and non-upsA (upsB/C) two groups, our algorithm could distinguish upsB and upsC group *var* genes and provide different degrees of membership to three ups groups. Through cross validations, we show that our method's classification accuracy is high; setting a cutoff on inferred probabilities further increases the accuracy.

To summarize, I conclude all the work and discuss future research directions in Chapter 6.

Chapter 2

Literature review

This chapter examines the current methods for detecting recombination events and recombinant sequences. There are four main paradigms of recombination and recombinant identification methods. There are also specific studies which identify the recombination of *var* genes. I briefly summarize them in Section 2.1. We stress that none of these approaches is suitable for addressing our project goal — detecting recombination events and recombinant DBLα tags. As previously mentioned, the jumping hidden Markov model (JHMM) is a great tool for modelling the recombination of DBLα tags, I therefore examine it in Section 2.2 after the introduction of basic HMM. This model's limitations are also discussed.

2.1 Existing methods for identifying recombination and recombinants

There are numerous methods for evaluating recombination in a dataset. These methods have several objectives: detection of the absence or presence of recombination, identification of recombinant sequences and their breakpoints, and quantification of population recombination rate along the genome. The first objective is the prerequisite of the latter ones, while the last one is mainly addressed under population genetic models. In this thesis, we focus on the second task. The most exhaustive overview of available recombinant detection tools can be found in [121, 122] and on David Robertson's website (`http://bioinf.man.ac.uk/robertson/recombination/programs.shtml`). These methods can be roughly categorized into

four categories [97]: (1) distance, (2) phylogenetic, (3) substitution distribution, and (4) compatibility. I discuss them individually below.

Distance-based methods These methods are based on a reasonable assumption that any given sequence is like the sequence with which it shares the most recent common ancestor. The similarity between two non-recombinant sequences is uniform throughout, whereas the similarity between a recombinant sequence and its parental sequences would vary from one sequence to the next along the genome. Such shifting of similarity among sequences indicates the presence of recombination.

Pairwise genetic distance is a common way to measure the sequence similarity, and investigating the inversions of distance patterns [98] is the key to distance-based methods. Typically, a sliding-window technique [88–90, 98] is usually used to obtain the partitions of the input alignment, followed by the computation of distances across partitions. Finally, some statistic are used to identify the partitions with evidence of recombination.

There are a variety of distance-based recombination detection tools. `RAT` (Recombination Analysis Tool [87]) is a cross-platform program that plots the similarity vs. genome position. It can examine the suspicious recombinant sequence using the single-sequence viewer or treat each sequence as the potential recombinant with the auto search option. However, it requires user-specified parameters before the analysis, like the maximum number of contributing sequences to a recombinant. These quantities are generally not known in practice. Therefore, it is useful for the initial and fast exploration of recombination. `SimPlot` [88] and `RIP` (Recombinant Identification Program [89]) search each query sequence against a reference panel by partitioning the alignment of all sequences using a sliding-window approach and calculating the percent identity. They generate the equivalent output; the former works on Windows computers, while the latter works on UNIX computers. Although these two tools both provide an adjustable window size, a too-wide or too-narrow window still affects the detection sensitivity [89, 98]. To alleviate this problem, Lee and Sung [90] later introduces a new distance calculation measure and weighting strategy and proposes an accurate and efficient algorithm called `RB-FINDER`. Moreover, its implementation does not require a specific computer system. In general, the speed of distance-based methods is a significant advantage over other types of approaches.

Phylogenetic methods While genetic distance generally correlates well with the evolutionary distance between sequences, it does not always do so [121]. Many algorithms ([91, 92, 94, 95, 123]) instead use phylogeny information to detect recombination signals.

A recombinant query would have diverse evolutionary histories with sequences assembling its parents. Specifically, the corresponding phylogenetic trees differ before and after the breakpoint. Several phylogenetic methods [91–93] first infer the phylogenetic trees across the partitions of sequence alignment and search for discordant tree structures (topologies) between adjacent partitions. The 'jump' from one phylogenetic clade to another is interpreted as evidence of recombination. An early method for identifying the recombinant genome is called `BOOTSCAN` [91]. It searches each query sequence with a set of non-recombinant sequences and makes the bootstrap phylogenetic tree inference for each partition of the alignment. Since this method relies on a pre-defined bootstrap support value cutoff, and it lacks a statistical test of recombination, Martin et al. [92] then proposed a modified approach called `RECSCAN`. It obtains the statistical rigour via calculating the probability of identified recombination regions occurring by chance using a binominal distribution introduced by Martin and Rybicki [109]. Besides, instead of searching the potential recombinant query against the reference panel, `RECSCAN` scans every three sequences (triple) without the requirement of specification of non-recombinant sequences.

More recently, there are methods [94, 95] which directly construct a phylogeny and detect recombination without comparing the tree topologies. For instance, there is an algorithm for detecting recombination of vast SARS-CoV-2 genomes called `RIPPLE` [95]. This method uses a tree structure called mutation-annotated tree [124] and identifies the long branches as the potential recombinant lineages. Then it reconstructs the phylogeny by placing the segments of potential recombinants with maximum parsimony. Moreover, this method conducts a permutation test to check the significance of identified breakpoints. Until now, the phylogenetic-based methods are the most frequently applied category [122] in detecting recombination.

Substitution distribution-based methods Substitution distribution-based approaches look for shifts in patterns of sites across (typically adjacent) sequence alignment partitions to find recombination breakpoints. These methods involve counting nucleotides and constructing a test statistic to determine the degree of relatedness between sequences. The statistic varies based on the algorithms, such as chi-squared

value in MaxChi [96] and CHIMAERA [97], Pearson's coefficient in PHYPRO [98], and exact non-parametric test for 3SEQ [99, 100]. Notably, a sophisticated scan statistic was used in Gubbins [101] to identify the regions with increased density of base substitution. These regions are indicative of recombination. Gubbins does not require the parental sequences of recombinants to be present in the dataset; this is very useful for the dataset with low sampling coverage. In general, methods based on substitution distribution appear to be powerful [125, 126] and become more powerful as the recombination rate increases [97]. This is advantageous for sequences that undergo frequent recombination.

One of the most commonly applied recombination detection programs is RDP4 [102]. It provides seven major methods (phylogenetic-based: BOOTSCAN [91] and RDP [109]; substitution distribution-based: MaxChi [96], CHIMAERA [97], SISCAN [107], 3SEQ [99, 100], GENECONV [108]) for detecting the recombination signals from DNA sequences. This Windows program could remove the recombinant sequences or regions for downstream recombination-free analysis; it also allows for exploring the evolution of recombination events. Its latest version, RDP5 [110] further improves the efficiency and could flag the recombination signals which might be caused by other factors instead of recombination (false positive signals) with methods described in [127, 128].

Compatibility methods A few methods ([103–106]) test for compatibility on a site-by-site basis to detect recombination breakpoints. The basic principle is that if two sites are compatible, the tree topologies on these two sites are the same [129–131]. Reticulate program [103] computes 'neighbor similarity score' for clustering the compatible sites. However, it only returns the informative sites containing recombination signals. Lai and Ioerger [105] later proposed the ACR method to detect the breakpoints accurately. It first uses a compatibility ratio to test the presence of recombination from an input alignment. If there is recombination, it then computes the compatibility score and identifies the site with the minimum compatibility score suggestive as the evidence of recombination. Lai and Ioerger [106] continued to extend this model to ptACR. This method conducts a permutation test on the identified locations by ACR and further improves specificity. The shortcoming of this method is that it cannot handle the alignment with gaps. Therefore, it is not suitable for sequences with extreme diversity.

Unfortunately, none of the above methods is suitable for identifying recombination or recombinants for our large number (see Table 1.1) of diverse DBLα tags. The distance and substitution-based methods have high computational costs for large datasets, and they generally take an alignment as input. DBLα tags are too diverse to have a reliable alignment, especially when there are many such tags. Phylogenetic methods depend strongly on the precision of inferred phylogenies along the sequence alignment. Ultimately, more precise inferred phylogenetic trees lead to a greater capacity to identify recombination signals correctly. This is unavoidably problematic for sequences with a great deal of diversity like DBLα tags, from which it is typically challenging to infer the evolutionary history. Also, DBLα tags' large diversity precludes the most advanced compatibility method ptACR. Therefore, these methods are not suitable for our data.

2.1.1 Methods for identifying *var* recombination and recombinants

In the context of *var* genes, there are few *in vitro* and *in vivo* studies ([42, 59, 61]) for identifying the *var* recombination. Kraemer et al. [61] use the full-length *var* genes from three isolates (3D7, HB3, IT4) and study the sequence similarity with BLASTN for each pair of *var* genes. They next visualize the similarities to find the 'chimeras' representing the recombinants or partial duplications between different genes.

Claessens et al. [42] use slightly more strains (3D7, HB3, Dd2, W2) and conduct 'clone tree' experiments for generating the parasites (clones) at various generations. This *in vitro* evolution generally spans several months. With generated clone trees, the authors perform whole genome sequencing and find chimeras. These potential recombinants are subsequently validated by capillary sequencing of PCR products. They analyze all identified recombination instances and parental genes per recombination breakpoint. By dividing the number of recombination instances by number of involved recombining pairs, an average of 2.4 breakpoints per pair of sequences is reported. However, similar to Kraemer et al. [61], there is a lack of formal statistical tests to identify recombinants and related breakpoints.

More recently, Otto et al. [59] assemble the nearly full-length *var* genes from ∼2400 African and Asian isolates. They focus on Exon 1 of the *var* genes and define a

recombination breakpoint when an almost identical sequence match (>1kb) between two sequences is interrupted by a subsequent less identical match (>0.5kb). They report extensive recombination events of *var* genes. It is not trivial to modify this approach for our much shorter DBLα tags. We also don't have to do this as the jumping hidden Markov model (JHMM [54]) has been designed to identify DBLα recombination.

The hidden Markov model (HMM) is a powerful tool in modelling recombination. It either works as a sole model (like [54, 120, 132–134]) or work with phylogenetic theory (like [93, 94, 135]) for detecting recombination or recombinants. As I mentioned in the Section 1.3 of Chapter 1, the JHMM (a variant of HMM) is the first systematic attempt to quantify the *var* recombination. Compared with other related work ([42, 59, 61]) of *var* genes, the JHMM is a well-designed statistical approach to detecting the recombination events and related breakpoints, it also applies to the DBLα domains, and can handle a relatively large dataset. Therefore, it is necessary to know the corresponding principles. In the following, I introduce basic HMM firstly, followed by a review of HMM's extensive applications in modelling the recombination of various organisms. I finally conclude this section with a detailed review of the JHMM.

2.2 Hidden Markov model in modelling recombination

The hidden Markov model was initially introduced by Baum and his colleagues in the late 1960s [136–140], and was first applied to speech recognition at the 1970s [141–145], where human speech could be recognized and converted to the corresponding text message. The basic HMM theory and details for practical application in speech recognition were then reviewed by Rabiner in 1989 [146]. In the late 1980s, HMM started to be applied to biological sequence analysis [147], and then had become especially popular in bioinformatics [120, 148]. This thesis also focuses on HMM in biological sequences, particularly its application in the sequence alignment.

2.2.1 HMM and its three fundamental problems

The hidden Markov model describes the generation of a sequence of visible symbols ('observations') from a sequence of hidden internal factors ('states'). Therefore, an HMM consists of two layers, one for observation and one for the underlying hidden state. The observation layer is explained by the hidden state layer under HMM framework.

We denote a length L observed sequence as $\mathbf{y} = y_1 y_2 ... y_L$, y_t is the t th symbol of $\mathbf{y}$. Its putative hidden state sequence (also called 'path') is $\mathbf{x} = x_1 x_2 ... x_L$, where the t th hidden state x_t generates symbol y_t. For any $1 \leq t \leq L$, y_t is one symbol from observation set $\mathbf{O} = \{O_1, O_2, ..., O_M\}$, while x_t is one state from hidden state set $\mathbf{H} = \{H_1, H_2, ..., H_N\}$. Here M and N represent the number of all possible distinct observations and hidden states, respectively.

There are two critical assumptions in HMM. Firstly, we assume the hidden path follows a (first-order) Markov chain. The probability of entering into the current hidden state only relies on the hidden state in the previous time point and is independent of all other previous states.

$$\Pr(x_{t+1} \mid x_t, x_{t-1}, x_{t-2}, ..., x_1) = \Pr(x_{t+1} \mid x_t) \tag{2.1}$$

Secondly, the probability of observing y_t only depends on its hidden state x_t at the same time, so it is conditionally independent of any other observations or other states.

$$\Pr(y_t \mid x_1, ..., x_t, ...x_L, y_1, ..., y_{t-1}, y_{t+1}, ..., y_L) = \Pr(y_t \mid x_t) \tag{2.2}$$

With these assumptions, a basic HMM now could be characterized by the following three components [146]:

1. $N \times N$ "transition" matrix $\mathbf{A} = [a_{ij}]$, where $a_{ij} = \Pr(x_{t+1} = j \mid x_t = i)$. Each a_{ij} refers to the probability of hidden state i transits to hidden state j, $i, j \in \mathbf{H}$, each row of $\mathbf{A}$ sums to 1.

2. $N \times M$ "emission" matrix $\mathbf{E} = [e_i(y)]$, where $e_i(y) = \Pr(y_t = y \mid x_t = i), y \in \mathbf{O}$. Each observed symbol comes from a probability distribution over all symbols given the hidden state, $\sum_y e_i(y) = 1$ for a fixed i.

3. Initial probabilities of starting from various hidden states $\boldsymbol{\pi}$. $\pi_i = \Pr(x_1 = i)$, s.t. $\sum_{i \in \mathbf{H}} \pi_i = 1$

Therefore, an HMM is defined by these three parameters

$$\lambda = \{\mathbf{A}, \mathbf{E}, \boldsymbol{\pi}\}. \tag{2.3}$$

Once we have an HMM, there are three basic problems to be addressed in real-world applications. I introduce them with solutions in turn as below.

2.2.1.1 Decoding: Viterbi algorithm

The first problem is that we are interested in the underlying hidden path instead of the observation. The task of predicting hidden state sequence given a particular observation and an HMM is called *decoding*. In HMM, many paths might give rise to a single observation sequence, and each path has a probability. In the decoding task, we want to find out the most probable path that generates this observation, that is, the path with the largest probability. More formally:

Decoding: Given an HMM $\lambda = \{\mathbf{A}, \mathbf{E}, \boldsymbol{\pi}\}$, and a sequence of observations $\mathbf{y} = y_1 y_2 ... y_L$, get a hidden state sequence $\mathbf{x}^* = x_1^* x_2^* ... x_L^*$, s.t.

$$\mathbf{x}^* = \underset{\mathbf{x}}{\operatorname{argmax}} \Pr(\mathbf{y}, \mathbf{x} \mid \lambda).$$

Viterbi algorithm [149] is the most popular solution [148] to this problem, and I introduce it here. This algorithm uses a dynamic programming technique that efficiently computes the most probable path ('Viterbi path'). It only needs to fill in a $N \times L$ trellis. Each row refers to a hidden state, and each column represents a symbol from the observation sequence. Every entry of this trellis sums up all possible transitions to a particular symbol from all hidden states at the previous time.

Specifically, suppose $v_i(y_t)$ represent the probability of the most probable path ending in state i with observation y_t, then the corresponding probability for next symbol y_{t+1} with hidden state j is

$$v_j(y_{t+1}) = \max_i v_i(y_t) a_{ij} e_j(y_{t+1}) \tag{2.4}$$

The probability for the most probable path is calculated by comparing $v_x(y_L)$ over all hidden states, in other words, the last column of the trellis. Regarding the Viterbi path, we record the pointer which approaches it for each trellis entry. Then, an optimal path is found by backtracking until the first symbol. Since there are N transitions to each entry, the complexity of this algorithm is $O(N^2L)$.

2.2.1.2 Likelihood computation: forward algorithm

Likelihood computation: Given an HMM $\lambda = \{\mathbf{A}, \mathbf{E}, \boldsymbol{\pi}\}$, and a sequence of observations $\mathbf{y} = y_1 y_2 ... y_L$, get the likelihood $\Pr(\mathbf{y} \mid \lambda)$.

The second problem in HMM is determining the probability of observing a particular observation sequence given an HMM. Here I describe an efficient approach called the forward algorithm. This algorithm also requires filling in a $N \times L$ trellis where each row refers to a hidden state and each column refers to a symbol of the observation sequence. Suppose the $f_i(y_t)$ (called forward variable) represents the joint probability of observing the partial sequence from y_1 up to y_t, with the hidden state i for symbol y_t, i.e.

$$f_i(y_t) = \Pr(y_1 ... y_t, x_t = i) \tag{2.5}$$

$f_j(y_{t+1})$ is the summation of the probabilities across all hidden states from previous time step transiting to current step.

$$f_j(y_{t+1}) = \sum_i f_i(y_t) a_{ij} e_j(y_{t+1}) \tag{2.6}$$

Therefore, the likelihood of the observation under the HMM parameters is the sum of the last column, $\Pr(\mathbf{y} \mid \lambda) = \sum_{i \in \mathbf{H}} f_i(y_L)$. Comparing this algorithm with the Viterbi algorithm makes it not difficult for us to find the similarities and differences. Overall, these two are very analogous. The major difference is that the forward algorithm replaces all the **max** steps of the Viterbi algorithm with **sum**. Finally, the forward algorithm complexity is also $O(N^2L)$.

Opposite the forward variable is the backward variable. It is also a useful quantity. We denote $b_i(y_t)$ as the backward variable, representing the probability of observing

remaining symbols $y_{t+1}, y_{t+2}...y_L$ given the hidden state i for observing y_t,

$$b_i(y_t) = \Pr(y_{t+1}, y_{t+2}...y_L \mid x_t = i) \tag{2.7}$$

Using both forward and backward variables is the key to solving a practical problem called *posterior decoding*. It aims to determine the most probable hidden state for a particular observed symbol y_t, and has a wide range of real-world applications in various questions. In fact, the product of forward and backward variables helps to obtain the probability of tth symbol with any hidden state given an observation sequence. i.e. $\Pr(x_t = i \mid \mathbf{y})$.

$$\begin{aligned}
\Pr(x_t = i \mid \mathbf{y}) &= \frac{\Pr(\mathbf{y}, x_t = i)}{\Pr(\mathbf{y})} \\
&= \frac{\Pr(y_1...y_t, x_t = i)\Pr(y_{t+1}...y_L \mid y_1...y_t, x_t = i)}{\Pr(\mathbf{y})} \\
&= \frac{\Pr(y_1...y_t, x_t = i)\Pr(y_{t+1}...y_L \mid x_t = i)}{\Pr(\mathbf{y})} \\
&= \frac{f_i(y_t)b_i(y_t)}{\Pr(\mathbf{y})}.
\end{aligned} \tag{2.8}$$

Likewise, there is a backward algorithm. Here we skip the details of this algorithm for reasons of brevity. Please see [148] for a comprehensive introduction. The major utility of the backward algorithm is to check the correctness of programming since it also provides the likelihood of an emitted sequence like the forward algorithm.

2.2.1.3 Learning: Baum-Welch algorithm

Previous decoding and likelihood computation both assume an HMM has already been provided. However, we generally do not know the HMM parameters, i.e., transition matrix $\mathbf{A}$, emission matrix $\mathbf{E}$ and initial probabilities $\boldsymbol{\pi}$. We normally only observe a sequence and might have a possible hidden state set from prior knowledge. Regardless of decoding or likelihood, we all need to get reasonable matrices for an HMM in the first place. The problem for obtaining HMM parameters given observations is called *learning*.

Learning: Given a sequence of observation $\mathbf{y} = y_1 y_2 ... y_L$ and hidden state set, get HMM parameters $\lambda = \{\mathbf{A}, \mathbf{E}, \boldsymbol{\pi}\}$ s.t. $\lambda = \underset{\lambda'}{\operatorname{argmax}} \Pr(\mathbf{y} \mid \lambda')$.

The learning problem is simple when the hidden path for the observed sequence is known. We denote A_{ij} as the count of transitions from hidden state i to j from the data, and $E_i(y)$ as the count of observing symbol y with hidden state i. The transition and emission parameters are obtained by the maximum likelihood estimation method, i.e.

$$a_{ij} = \frac{A_{ij}}{\sum_{j'} A_{ij'}} \text{ and } e_i(y) = \frac{E_i(y)}{\sum_{y'} E_i(y')} \tag{2.9}$$

In practice, we do not know the hidden path of observations. Therefore, we need to learn from observed sequences and get parameters estimated in an iterative way. Here I introduce a well-developed iteration method called *Baum-Welch* algorithm [140]. Since it combines the forward and backward variables, this method is also called the forward-backward algorithm.

The Baum-Welch algorithm is a special case of the Expectation Maximization (EM) technique [150]. With a set of initial parameters, the Baum-Welch algorithm calculates *expected* transition and emission count (E-step); afterwards, it recomputes a new set of HMM parameters with Equation 2.9 (M-step). Overall, this algorithm keeps updating parameters by iteration, and the likelihood of the data with parameters increases simultaneously. At the E-step, from Equation 2.8, we can sum over all positions t to get the $E_i(y)$

$$E_i(y) = \sum_{\{t|y_t=y\}} \frac{f_i(y_t) b_i(y_t)}{\Pr(\mathbf{y})} \tag{2.10}$$

Regarding the expected transition count, we introduce a quantity $\Pr(x_t = i, x_{t+1} = j \mid \mathbf{y})$ representing the probability of adjacent observations with hidden state i and j given an observation sequence [148].

$$\begin{aligned} \Pr(x_t = i, x_{t+1} = j \mid \mathbf{y}) &= \frac{\Pr(x_t = i, x_{t+1} = j, \mathbf{y})}{\Pr(\mathbf{y})} \\ &= \frac{f_i(y_t) a_{ij} e_j(y_{t+1}) b_j(y_{t+1})}{\Pr(\mathbf{y})} \end{aligned} \tag{2.11}$$

Likewise, we are now able to derive A_{ij} by summing over all positions t,

$$A_{ij} = \sum_t \frac{f_i(y_t)a_{ij}e_j(y_{t+1})b_j(y_{t+1})}{\Pr(\mathbf{y})} \tag{2.12}$$

Note that if there are multiple observation sequences, we add the expected count together across all training sequences. Although for brevity, in this thesis I describe the Baum-Welch algorithm with only an observation sequence, a more detailed tutorial with multiple sequences is in [148].

1) **Initialization**: Arbitrary $\mathbf{A}$, $\mathbf{E}$, $\boldsymbol{\pi}$

2) **Recursion**:

   (a) Calculate $f_i(y_t)$ and $b_i(y_t)$ under current HMM parameters.

   (b) Record the likelihood of data with current parameters.

   (c) Compute expected transition count A_{ij} and emission count $E_i(y)$ using Equation 2.12 and 2.10 respectively.

   (d) Update the parameters with Equation 2.9.

3) **Termination**: Stop when the number of iterations exceeds a predefined integer or the change of likelihood is less than a specified threshold.

At the end of this section, I briefly introduce an alternative to the Baum-Welch algorithm called Viterbi training [148, 151]. It is also an iterative method used for the learning problem in HMM. Viterbi training computes the transition and emission counts using only a single path — Viterbi path. By contrast, the Baum-Welch algorithm derives parameters by considering all paths. In each iteration, the Viterbi training algorithm computes the Viterbi path of the training sequence and updates parameters from this path using Equation 2.9. The algorithm stops when the Viterbi path does not change or the number of iterations is larger than a specified number.

Therefore, the essential difference between Baum-Welch and Viterbi training algorithms is that the former maximizes the likelihood, while the latter maximizes the contribution to the likelihood from the Viterbi path. For this reason, Viterbi training is slightly less accurate, but it is between one and two orders of magnitude faster

[152] than the Baum-Welch algorithm and has widespread applications in practice [153].

2.2.2 HMMs in modelling recombination

HMM is a popular tool for modelling recombination in various living organisms, such as bacteria, viruses and parasites [54, 93, 120, 132–135, 154, 155]. Although the complexity of sequences varies across species/genes, the statistical characteristics of HMM still facilitate widespread applications.

Husmeier and McGuire [93] introduced a Bayesian hidden Markov model to detect breakpoint regions in bacteria genes. Recombination causes the genealogies to vary along the sequence (or genome). They treated the tree topology as the hidden state, the change of tree topologies in the inferred optimal path was indicative of recombination. However, the number of input sequences this method could handle is limited. A similar phylogenetic HMM algorithm was proposed to detect recombinant rice genomes [135]. Its biggest bottleneck is still the rapidly growing running time and memory usage when the number of input sequences increases. Didelot and Wilson [94] proposed `ClonalFrameML` to detect recombination breakpoints using the whole genome sequences of bacteria. They reconstructed a ML tree with inferred ancestral sequences of all internal nodes and branch lengths. With this tree, a hidden Markov model was then used to infer the recombination parameters and hidden status (recombinant and un-recombinant) among sites. However, it might underestimate the recombination rate for a dataset with frequent recombination.

Boussau et al. [155] introduced Phylo-HMM to accurately locate the recombination breakpoints and applied it to HIV viruses. In this method, a matrix showing the likelihood of each tree topology at each site was firstly generated. If there is a recombination, the most probable tree topologies for stretches of sites would be different before and after the breakpoint. Therefore, partitioning the matrix into the most likely site segments was next conducted. Note that this method requires an input alignment and the number of trees that best describe this alignment, which are not applicable to our dataset.

Allred et al. [132] proposed incorporating HMM to identify the recombinant alphaviruses and flaviviruses. They trained a HMM on a set of non-recombinant virus genomes so that the model was parameterized with transition/emission probabilities

and a pre-specified recombination parameter. Each virus genome was considered as a hidden state. The Viterbi algorithm was used to find the new genome's most probable path. The change of path from one virus to another indicated a recombination event.

Li and Stephens [120] proposed an HMM to model the haplotype diversity. This model assumed each haplotype was copied from 'reference' haplotypes across all SNP locus along the chromosome. The change of copying haplotypes from one to another represented a recombination event. Schultz et al. [133] then developed an advanced HMM to map each new sequence to its nearest individual sequences in the input alignment. For capturing the information of all members of each subtype, researchers [133] employed the profile HMM [156]. A query sequence's optimal path consisted of a series of subtypes. The breakpoint was inferred by the 'jump' from one subtype to another. Because of these characteristics, this method is called jumping profile HMM. Afterwards, they extended this model to deal with viruses with circular genomes [134].

The first systematic work for modelling the recombination of DBLα domains was conducted by Zilversmit et al. [54]. They proposed the jumping HMM (JHMM) to estimate the recombination rate and detect the recombination breakpoints. Its principle is similar to Li and Stephens [120], but more characteristics are attached to each hidden state, for instance, the position of each specific reference sequence. Importantly, JHMM is the foundation of our work for detecting recombination and recombinants of DBLα tags. Since Zilversmit's model heavily relies on the pair HMM and Li and Stephen's model [120], I illustrate these two models below followed by Zilversmit's model.

Pair HMM In biology, determining whether two biological sequences are related is a basic task [148], and this requires us to assess the similarity of these two sequences that may indicate the functional, structural, or evolutionary relationships. To achieve this, pairwise (sequence) alignment is a way of mapping positions in the two sequences to each other (See Figure 2.1 as an example). Searching for shared domains or motifs between two proteins is a crucial problem, and the alignment of proteins would help to solve it; pairwise alignment of DNA sequences is particularly useful in studying conserved regions and polymorphisms. Generally the alignment with amino acids is more preferable than the alignment with DNA [157]. Overall,

pairwise alignment has become one of the most fundamental operations of bioinformatics [158]. There are a variety of algorithms to get two sequences aligned, such as maximal unique match, word methods, dot-matrix methods and dynamic programming [159]. In this thesis, I describe an extensively used dynamic programming technique called pair hidden Markov model (pair HMM).

```
Y – E R N Q R E D P P L G
Y F E R – – R E H P P L G
---------------------------
M D M M I  I M M M M M M M
```

FIGURE 2.1: **An example alignment of two proteins.** Original two sequences (represented by first two rows) have unequal length. Pairwise alignment introduces gaps (insertions and deletions) and results in aligned sequences with the same length. The bottom row is a hidden state path from a pair HMM point of view.

The pair HMM is a variant of basic HMM. There are three hidden states in pair HMM — match (M), insert (I) and delete (D). Assume two observed sequences $\mathbf{y} = y_1 y_2 ... y_{L_y}$ and $\mathbf{z} = z_1 z_2 ... z_{L_z}$, when the hidden state is M, it emits a pair of symbols (y, z). These two symbols are not necessarily the same. When the hidden state is I, it emits $(y, -)$, representing a symbol in the sequence $\mathbf{y}$ is inserted. When the hidden state is D, $(-, z)$ is emitted, indicating a symbol in the sequence $\mathbf{z}$ is inserted, or conversely, a deletion happens in the first sequence $\mathbf{y}$.

Generally, there is no transition between the insert and delete states. We denote δ as gap opening probability (M to I, M to D), and ϵ as the probability of staying in a gap, or gap extension probability (I to I, D to D). The resulting transition probabilities among hidden states are shown in Table 2.1. Besides, there are two sets of emission distributions for hidden states. If M, the aligned pair of symbols are sampled from a discrete distribution using amino acid or DNA substitution scoring matrix. If I or D, the inserted symbol is sampled from an equilibrium distribution of 20 amino acids or 4 nucleotides. When we compare the general HMM and pair HMM, standard HMM only generates a single sequence, and pair HMM emits an aligned pair of sequences (Figure 2.1).

TABLE 2.1: **Transition probabilities in pair HMM.**

	M	I	D
M	$1 - 2\delta$	δ	δ
I	$1 - \epsilon$	ϵ	-
D	$1 - \epsilon$	-	ϵ

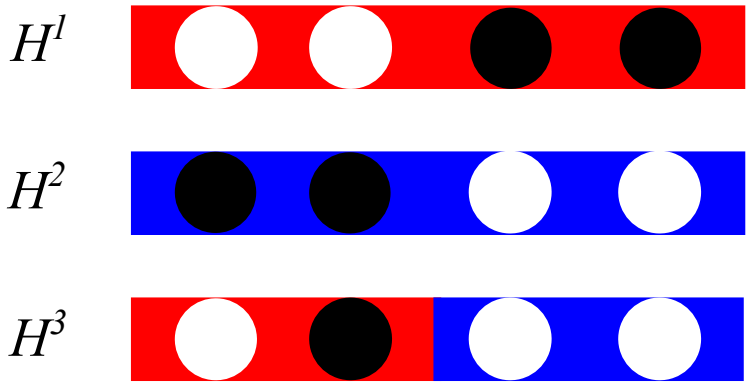


FIGURE 2.2: **Illustration of haplotype is created by an imperfect mosaic of source haplotypes.** This figure is a minor modification of Figure 2 from [120]. Each row represents a haplotype. Each column denotes a biallelic locus, colored by black and white. Haplotype H^3 is inferred to have copied from source haplotype H^1 for the first two alleles (colored by red) and source haplotype H^2 for the remaining two (colored by blue), with a recombination event in between. The second allele from the left of H^3 is imperfectly copied, due to a mutation event.

Nowadays, pair HMM has been widely used in many sequence analysis problems, such as alignment of multiple sequences [160–163], filtering out unreliable alignment columns [160], gene prediction and annotation [164–168]. More recently, there are several studies [169–172] to accelerate the probability calculation with the forward algorithm. Pair HMM is promising for broader applications soon.

2.2.2.1 Li and Stephens' model

Suppose there are N sampled haplotypes typed at L loci, $H^1, H^2, ..., H^N$. For nth haplotype H^n, $n \in \{1, 2, ..., N\}$, we denote its lth locus ($l \in \{1, 2, ..., L\}$) by h_l^n so that $H^n = h_1^n ... h_L^n$. To model recombination among haplotypes, Li and Stephens [120] assumes that each haplotype H^{k+1} is built by an imperfect copying process from 'reference' (source) haplotypes $H^1, H^2, ..., H^k$. See Figure 2.2 as an illustration. With this assumption, a hidden Markov model is used to compute the probability of $\Pr(H^{k+1} \mid H^1, H^2, ..., H^k)$, which further aids the likelihood computation.

The hidden state set is the source haplotype indices $\mathbf{H} = \{1, 2, ..., k\}$. Each locus of haplotype H^{k+1} is considered to be copied from one of the source haplotypes at that site, the specific haplotype is unknown and therefore designed as the hidden state of HMM.

Transition probability The initial step is to choose which haplotype to copy at the first site, Li and Stephens [120] determines the starting point of all the source haplotypes uniformly. Suppose x_l ($1 \leq l \leq L, x_l \in \mathbf{H}$) denote which haplotype H^{k+1} copies at site l, $\Pr(x_1 = x) = \frac{1}{k}$, $x \in \{1, 2, ..., k\}$. After copying a SNP site, the next site is always copied from the next one of the same haplotype; however, with a small probability, it is copied from any other haplotype representing recombination. Generally, the smaller the genetic distance between two sites, the higher the probability that it copies from the same haplotype.

$$P(x_{l+1} = x' \mid x_l = x) = \begin{cases} p_l + \frac{1}{k}(1 - p_l) & x' = x \\ \frac{1}{k}(1 - p_l) & \text{otherwise} \end{cases} \tag{2.13}$$

where $p_l = \exp(-\frac{4Nc_l d_l}{k})$, N is the effective population size, and c_l is the recombination rate per unit of physical distance, d_l is the physical distance between sites l and $l + 1$.

Emission probability After choosing a haplotype, with a certain probability, an exact copy occurs; otherwise, a mutation occurs. Specifically, the emission probability is given by:

$$\Pr(h_l^{k+1} = h \mid x_l = x, H^1, ..., H^k) = \begin{cases} \frac{k}{k+\tilde{\theta}} + \frac{\tilde{\theta}}{2(k+\tilde{\theta})} & h_l^x = h \\ \frac{\tilde{\theta}}{2(k+\tilde{\theta})} & h_l^x \neq h \end{cases} \tag{2.14}$$

where $\tilde{\theta} = (\sum_{s=1}^{N-1} \frac{1}{s})^{-1}$, N is the total number of haplotypes. This $\tilde{\theta}$ results in a prior with an average of 1 for the expected number of mutations per segregating site [120].

Li and Stephens [120] use the following identity to compute the likelihood,

$$\Pr(H^1, H^2, ..., H^N \mid \rho) = \Pr(H^1 \mid \rho) P(H^2 \mid H^1; \rho) ... \Pr(H^N \mid H^1, ..., H^{N-1}; \rho) \tag{2.15}$$

where ρ denotes the recombination parameter. This formula transforms the probability distribution of haplotypes to a series of conditional probabilities. Researchers assumes the first term $\Pr(H^1 \mid \rho)$ is independent of ρ and equals to $1/2^L$, for remaining conditional probabilities, they propose the above HMM framework and calculate

them using the forward algorithm. In this way, maximizing the product of conditional likelihoods yields the estimate of parameter ρ. With estimated parameter, we are able to identify the recombination hotspots.

Overall, this model created by Li and Stephens has many advantages and has been applied to numerous problems [173–183]. It is easy to comprehend and quick to compute; it models both constant and variable recombination rate scenarios; and most importantly, it helps describe linkage disequilibrium (LD) patterns, as it overcomes various limitations of existing methods for modelling LD. Price et al. [173] extended Li and Stephens' model and proposed `HAPMIX` for inferring the local ancestry in admixed populations. However, this model can only infer from two admixed groups, Salter-Townshend and Myers [174] recently improved this model with a method called `MOSAIC`. It modelled the admixture with high precision from an arbitrary number of populations. There are other applications such as the genotype imputations using reference sequences [180, 181], the inference of genealogical history [175, 176, 178] and gene conversion rate estimation [182, 183].

2.2.2.2 The JHMM in Zilversmit's model

Zilversmit et al. [54] proposed the jumping hidden Markov model (JHMM) to uncover the evolutionary relationships of the DBLα domain across species. This method combines Li and Stephens' model to account for recombination effects and pair HMM to handle unequal-length sequences. The key assumption of the JHMM is that each sequence ('target') is constructed from reference ('source') sequences, allowing substitutions, insertions and deletions (indels) in the target sequence and jumps between source sequences representing recombination.

There are similarities and notable differences in the mechanism between the JHMM and Li and Stephens' model. Both models consider each character as a copy of the character of the source sequence. After copying a character from a source sequence, it usually copies the next character from the same source sequence. However, with a small probability, it copies from other characters of any other source sequence. The copy is either the same or a mismatch. However, the major difference between these two models is the range of characters which could be copied. Li and Stephens' model copies character only from that specific site of input alignment, while the JHMM could copy character from everywhere of source sequences.

The hidden state of the JHMM is the position of the character from the source sequence (which is copied). See Table 2.2 for important variables of the JHMM. Specifically, we denote s_k^j as the hidden state at ith character of target sequence x. If $s = M$, the hidden state M_k^j emits character $x(i)$ given $y_k(j)$. If $s = I$, the hidden state I_k^j emits character $x(i)$, representing that $x(i)$ is inserted in x. If $s = D$, the hidden state D_k^j does not emit a character on target, representing a character is deleted at x. When the hidden state emits a character, the character is sampled based on the emission probability distribution.

TABLE 2.2: **JHMM notation.**

Variable	Meaning	Range	Note
i	Index for position in target	$i \in \{1, ..., m\}$	$x(i)$ is the ith character in the target sequence x of length m
k	Index for source sequence	$k \in \{1, ..., n\}$	y_k is the kth source sequence (length is l_k) in dataset Y
j	Index for position in source	$j \in \{1, ..., \max_k l_k\}$	$y_k(j)$ is the jth character in y_k
s	Match, insert and delete states	$s \in \{M, I, D\}$	

The change of elements s, k, j in the hidden state indicates various events. Specifically, the change of source sequence k in the hidden state path represents a recombination event. The switch of s represents an indel event. For instance, change from match to insert/delete state ($M \to I$, $M \to D$) indicates a gap opening event while multiple consecutive insert/delete states ($I \to I$, $D \to D$) infer a gap extension event. Finally, the change of position j suggests copying from one site to another.

Transition probability For the initialization of the hidden state sequence, a position in a source sequence is chosen uniformly at random from all positions. With probability π_M, the path starts with a match state; with probability $\pi_I = 1 - \pi_M$, the path starts with an insert state. The hidden state in the next time step is conditional on the current hidden state. Specifically,

- Suppose the current hidden state is $M_k(j)$, the next hidden state will be
  - a match with the next position, or
  - an insertion with probability δ (gap opening probability), or
  - a deletion with probability δ, or

  - a recombination with overall probability ρ (probability of recombination) across all positions, or
  - a termination with probability τ (called termination probability).

- Suppose the current hidden state is $I_k(j)$, the next hidden state will be
  - a match with the next position, or
  - an insertion with probability ϵ (gap extension probability), or
  - a recombination with overall probability ρ across all positions, or
  - a termination with probability τ.

- Suppose the current hidden state is $D_k(j)$, the next hidden state will be
  - a match with the next position, or
  - a deletion with probability ϵ.

TABLE 2.3: **Transition probabilities from a given hidden state (rows) at a current time step to all possible hidden states (columns) at next time step.** T refers to termination.

	M_k^{j+1}	I_k^j	D_k^{j+1}	$M_{k'}^p$	$I_{k'}^p$	$D_{k'}^p$	T
M_k^j	1-2δ-ρ-τ	δ	δ	$\rho\pi_M/L$	$\rho\pi_I/L$	0	τ
I_k^j	1-ϵ-ρ-τ	ϵ	0	$\rho\pi_M/L$	$\rho\pi_I/L$	0	τ
D_k^j	1-ϵ	0	ϵ	0	0	0	0

Table 2.3 presents the transition probabilities among hidden states. Note that $M_{k'}^p$ (same with $I_{k'}^p$, $D_{k'}^p$) in this table refers to the hidden state representing a recombination, $k' \in \{1, ..., n\}$, $p \in \{1, ..., l_{k'}\}$. The overall number of positions the JHMM could jump to is denoted by L, $L = \sum_{s=1}^{n} l_s$.

Emission probability We denote $e(x(i) \mid y_k(j))$ as the emission probability for observing $x(i)$ given y_k^j from hidden state M_k^j. With probability α (a predefined parameter), it copies the source; otherwise, each other letter has equal probability. We denote $e(x(i))$ as the emission probability for observing $x(i)$ from I_k^j. $e(x_i)$ is the base frequency distribution which is calculated from the character composition of target sequence x.

To train the parameters of the JHMM, δ and ϵ are estimated firstly using the Baum-Welch algorithm (see Section 2.2.1.3), with ρ set to zero. Next, with $\hat{\delta}$ and $\hat{\epsilon}$, a

surface of 'composite' likelihood (product of the likelihood of each target) on a grid of ρ (from 0 to 0.1) is calculated. The one corresponding to the largest likelihood is chosen as $\hat{\rho}$. Default values 0.75, 0.25 and 0.001 are assigned to π_M, π_I and τ separately. Once the parameters are trained, the Viterbi algorithm is then used to find the most likely path of source sequences for each target sequence. Below Figure 2.3 illustrates an instance of mosaic output in which a target is built from three source sequences. The substitution, insertion and deletion are colored brown, red and blue separately.

FIGURE 2.3: **An example mosaic representation of the JHMM.** In this example, the target sequence is aligned to three segments, and each of them is from a source sequence of the reference dataset.

Notably, the mosaic representation provides information on whether the target sequence is recombinant or not, when the source sequences are older than the target sequence. If there is more than one source sequence in the mosaic representation of a target sequence, we can infer this target is a recombinant. In contrast, when the input sequences are from the same time point, we cannot identify the recombinant from mosaic representations any more, since the target and source sequence are both possible to be recombinant (see Section 3.5.1.1 of the following chapter).

The JHMM is the first model that "paints" each DBLα domain with its closest source sequences. Garimella et al. [184] later used this model with all default parameters (default $\delta = 0.025$, $\epsilon = 0.75$, $\rho = 0.0001$) for variant calling. They applied to the sequences from *P. falciparum* crosses and identified nonallelic recombination and mutation patterns. Tonkin-Hill et al. [62] applied Zilversmit's model to explore population structures around the world using DBLα tags from ten countries, a series of strategies were proposed to speed up the parameter estimation due to the infeasible running time posed by large data sizes.

We followed the strategies of [62] and applied Zilversmit's model to our large Ghana pilot data. With the generated mosaic representations, we analyzed a variety of

DBLα recombination patterns. For instance, we explored the recombination frequencies per tag, the breakpoint distribution along the tag, and the jumping distance from one source to another per recombination. We also compared these results with the patterns using global data in [62]. Note that although Tonkin-Hill et al. [62] had already applied the Zilversmit's model to the global data, their analysis about DBLα recombination is limited (due to different research purposes). So we analyzed the recombination patterns from both datasets.

Through the comparison of these recombination patterns, we found very consistent results. The estimated JHMM parameters, breakpoint distributions along the tag, recombination frequencies per tag and jumping distance distributions are all similar across these two datasets, see Appendix A for more details.

2.3 Discussion

This chapter outlines the existing methods for detecting recombination and recombinants among sequences or *var* genes specifically. We summarize that none of them (except the JHMM) is suitable for dealing with our dataset mainly due to the extreme diversity and large sample size. Since the jumping hidden Markov model is the only method that models the recombination of DBLα domains systematically, this chapter introduces basic HMM and examines this JHMM.

The major advantage of the JHMM is that it makes it possible to detect recombination breakpoints from diverse DBLα domains. However, the JHMM was only designed to detect recombination events, not which sequences are recombinant, when the input sequences are from the same time point. For each generated mosaic representation, although the recombination breakpoints are known, it is still unclear whether the target (or source) sequence is recombinant or not. As described in the project aim (Section 1.3), if we could identify recombinant sequences, this would enable us to explore the characteristics of recombinants; clear separation of recombinant and non-recombinant sequences would also facilitate the comparison of their structure and function. In the following Chapter 3, we introduce an algorithm for detecting recombinants when the sequences are from the same time.

The training of the JHMM parameters can become slow when the dataset is large and/or the sequences are long. The JHMM complexity is dominated by the Baum-Welch algorithm used to estimate parameters. Each iteration of the algorithm is

$O(n^2l^2)$ in time and memory [54], where n is the number of sequences and l is the average sequence length. Although Tonkin-Hill et al. [62] proposed to replace the Baum-Welch algorithm with faster Viterbi training algorithm, their impact on model accuracy (estimated parameters, mosaic representations) is unclear. Moreover, the JHMM allows the jump to any sequence position representing recombination. In fact, recombination in DBLα tags does not occur in such an unfettered manner (see Appendix A.2.2.4). Therefore, constraints must be added to make this model more time efficient and biologically relevant. This has prompted us to propose a revised JHMM described in Chapter 4.

Chapter 3

An accurate method for identifying recent recombinants from unaligned sequences

Abstract

Motivation: Recombination is a fundamental process in molecular evolution, and the identification of recombinant sequences is thus of major interest. However, current methods for detecting recombinants are primarily designed for aligned sequences. Thus they struggle with analyses of highly diverse genes, such as the *var* genes of the malaria parasite *Plasmodium falciparum*, which are known to diversify primarily through recombination.

Results: We introduce an algorithm to detect recent recombinant sequences from a dataset without a full multiple alignment. Our algorithm can handle thousands of gene-length sequences without the need for a reference panel. We demonstrate the accuracy of our algorithm through extensive numerical simulations; in particular, it maintains its effectiveness in the presence of insertions and deletions. We apply our algorithm to a dataset of 17,335 DBLα types in *var* genes from Ghana, observing that sequences belonging to the same ups group or domain subclass recombine amongst themselves more frequently, and that non-recombinant DBLα types are more conserved than recombinant ones.

Availability: Source code is freely available at `https://github.com/qianfeng2/detREC_program`.

3.1 Introduction

Recombination, the exchange of genetic materials between two molecular sequences, is a fundamental evolutionary process in viruses, prokaryotes, eukaryotes, and even between kingdoms. The biological mechanisms of recombination, which differ across different species, lead to the creation of novel 'mosaic' sequences in which different regions have distinct evolutionary histories.

In population genetics, recombination plays a central role in shaping the patterns of linkage disequilibrium, and thus recombination identification is of importance for estimating recombination rates, quantitative trait loci and association studies [120, 185]. Recombination also explains a considerable amount of the genetic diversity of human pathogens [186–188], such as the malaria parasite *Plasmodium falciparum* [42, 189] or protozoan parasites [190]. It plays a central role for parasites to escape from host immune pressures, or adapt to the effects of antiparasitic drugs. Characterisation of recombination events in these pathogens would aid in the understanding of these evolutionary mechanisms.

Many methods have been developed for identifying recombination events and/or recombinants (e.g., [97, 99, 109, 123, 191], see [121] for a review). They can be roughly characterised into four paradigms:

1. Distance-based methods [89, 192, 193] look for inversions of distance patterns among the sequences. They usually employ a sliding-window approach to estimate distances and are generally computationally efficient.

2. Phylogenetic methods [109, 187, 194] look for discordant topologies in adjacent sequence segments, which is taken as a sign of recombination.

3. Compatibility methods [103] test for phylogenetic incongruence on a site-by-site basis.

4. Substitution distribution-based methods [96, 97, 99] use a test statistic to examine adjacent sequence segments for signals of recombination.

Nearly all available methods require a multiple sequence alignment; this is commonly available for population genetic datasets which have relatively low intra-population diversity, but may be unreliable for datasets with higher diversity. Likewise, many of the most commonly used methods, such as RDP [109] or 3SEQ [99], are triplet-based;

that is, they test for recombination signals in each possible triplet of sequences, which can become slow as modern-day datasets grow larger and larger. Finally, some (though not all) methods (e.g., [89]) require a reference panel of known non-recombinant sequences, which potential recombinants can be compared against. We aim to develop a method which works directly on sequences without requiring a full multiple sequence alignment or a reference panel, and is fast enough to be practical for large datasets.

We focus on the specific application of detecting recombinants in the *var* genes of *Plasmodium falciparum*. These genes express the *Plasmodium falciparum* erythrocyte membrane protein 1 (PfEMP1), which is the main target of the human immune response to the blood stages of infection. The *var* genes are a large and diverse gene family (up to 60 copies per genome), and high levels of diversity in the *var* genes have been observed in a single parasite genome, as well as small local populations [46, 74, 195, 196]. This diversity is driven primarily by homologous recombination [42], and so an accurate identification of *var* recombinants is critical to understanding the evolution of the system.

We focus on the DBLα domain, which is the only domain encoded by all (but one) members of the *var* multigene family. This domain has been found to be immunogenic [67] and is crucial to understanding acquired immunity and potential for vaccination [68]. Unfortunately, the DBLα domain is highly variable in terms of both length and sequence composition, with datasets [62] containing tens of thousands of disparate sequences. Under these conditions, multiple sequence alignments constructed from these datasets are very unreliable, and a phylogenetic tree is not an appropriate representation of their evolutionary history due to frequent recombination. Thus, it is difficult to reconstruct an explicit evolutionary history of the DBLα domain.

The first systematic attempt to map out recombination in *var* genes was performed by Zilversmit et al. [54], who developed a method based on a jumping hidden Markov model (JHMM) to align a sequence to its nearest relations in a reference dataset, allowing jumps between sequences which represent recombination events. They used this method to "paint" each sequence according its nearest relations. However, this method does not identify the recombinant sequences themselves, only recombination events. An explicit identification of recombinants and non-recombinants would enable direct comparison between them, helping to determine the effect of recombination on the structure and function of the gene.

Because each sequence is considered individually, the JHMM is limited to the detection of 'recent' recombination events; that is, recombinations whose signal can be found only in one sequence in the dataset. In contrast, a single more ancient recombination may leave traces in multiple sequences, hindering the ability to detect them. It is thus an unavoidable consequence that any method based on the information provided by the JHMM is limited to the detection of recent recombinants, i.e., the descendants of recent recombinations.

In this paper, we develop a new method to identify recent recombinants in a large dataset of sequences, that does not require a multiple sequence alignment. This method exploits the information produced by the JHMM method, combining it with a distance-based comparison to identify recombinants. Extensive simulations confirm the accuracy and applicability of our method, in particular in the context of sequences with insertions and deletions. We also show that our method is more accurate than many currently used methods. Finally, we apply our method to a large dataset of DBLα sequences, producing several new biological results concerning the patterns of recombination in this domain.

3.2 Methods

We propose a novel method to detect recombinant sequences in a set of protein or DNA sequences for which a full multiple alignment is difficult to construct or unreliable. It takes as input a set of homologous sequences, and outputs the sequences that are identified as recombinant, their putative parents, and the corresponding breakpoints.

See Figure 3.1 for a graphical overview of our method. It consists of the following steps:

1. We apply the JHMM method of Zilversmit et al. [54] to represent each sequence as a 'mosaic' of segments from other sequences in the dataset.

2. We identify 'recombinant triples' which contain a recombinant segment and its two parents. The mosaic representations provide pairwise alignments for each of these triples, which we then complete to three-way alignments with the MAFFT algorithm [197].

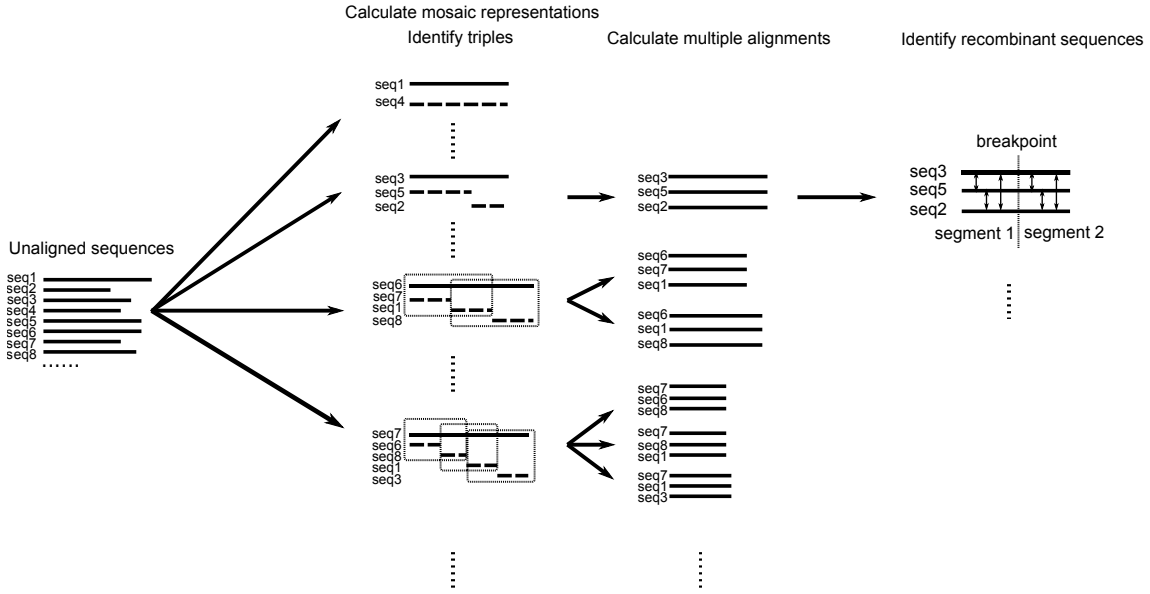


FIGURE 3.1: **A schematic of the algorithm.** From an input set of unaligned sequences, we first use the JHMM method to represent each sequence as a mosaic of other sequences. Next, we identify triples of segments, consisting of a recombinant segment and its two parents, and complete their alignment with the MAFFT algorithm. Finally, we identify the recombinant in each triple using a distance-based approach.

3. Using a distance-based approach, we identify the recombinant sequence in each triple.

Note that extant sequences are identified as the 'parents' of the recombinant; more accurately, we identify the descendants of the ancestral sequences which were the parents of the recombination.

We discuss each step in detail in the following sections.

3.2.1 Calculating mosaic representations

We first use the jumping hidden Markov model of Zilversmit et al. [54]. In this model, each character in a 'target' sequence is considered to be a copy from a character in a sequence in a reference set ('source' sequences). The hidden state of the Markov model is the (position of the) character which is copied. The copy may be imperfect, representing mutation. After a character is copied, the next character in the target sequence is usually copied from the next character in the same source sequence. However, with small probabilities:

- the source character may switch to any character in any position in another sequence, representing recombination;
- the model switches to an 'insertion' state, where the target character is chosen randomly and the source character does not move;
- the model switches to a 'deletion' state, where the source character moves forward without being copied.

If the model is in an insertion or deletion state, it continues in this state until (with a small probability per character) we return to copying characters from the current source sequence.

We first estimate the parameters of the model, following Tonkin-Hill et al. [62]. The parameters are the probabilities of gap initiation δ, gap extension ϵ, and recombination (source switching) ρ. We first set ρ to zero, and compute maximum likelihood estimates for δ and ϵ with the Baum-Welch algorithm (see [146]). We then calculate the composite likelihood of all sequences for all values of ρ over the interval $[0, 0.1]$ under the estimated $\hat{\delta}$ and $\hat{\epsilon}$, and choose the value of ρ which maximises this likelihood as our estimate $\hat{\rho}$.

Finally, we calculate the Viterbi path for each target sequence to find the most probable sequence of hidden states (copied characters, insertions, and deletions). The result is a 'mosaic' alignment for each sequence to a series of segments from the other sequences in the dataset. An example of this can be seen in Figure 2.3 of Chapter 2.

For large-scale datasets, training the JHMM model is a significant bottleneck for our method. We again follow Tonkin-Hill et al. [62], and use the Viterbi training algorithm [152] in place of the Baum-Welch to estimate δ and ϵ, and calculate the composite likelihood over 1000 randomly selected sequences to estimate ρ. This allows us to analyse large datasets (such as the DBLα dataset in Section 3.3.2) in a practical timeframe with only a small loss in accuracy.

3.2.2 Identifying recombinant triples and calculating multiple sequence alignments

For each breakpoint in each sequence, we identify the triple of the target sequence and the two sequences which contain the source segments before and after the breakpoint as a 'recombinant triple', that is, the two parents and the child of a recombination. This results in a list of recombinant triples, some of which may refer to the same recombination event. The JHMM method only provides a pairwise alignment of each target segment to one source segment. We take these pairwise alignments and add the corresponding segment from the remaining source sequence in the triple, using the MAFFT algorithm [197]. For each triple, this results in a multiple alignment of the segments surrounding the breakpoint. See Supplementary Figure 3.23 for an overview of this process.

Note that we require a sufficient sequence length on either side of the breakpoint in order to calculate distances accurately. Moreover, we observe in practice that short source segments resulting from the JHMM method tend to be artifacts of the method, rather than representing multiple consecutive recombinations. To address this, we exclude triples for which the aligned segment on either side of the breakpoint has length less than 10, which we found to be a suitable threshold in practice.

3.2.3 Identifying recombinant sequences

We now apply the well-known principle [96, 97, 99] that two non-recombinant sequences will have a similar evolutionary distance all along the sequence; that is, the distance between the two sequences does not change before and after a recombination breakpoint in a third sequence. Conversely, the distance between a recombinant sequence and another sequence does change at a breakpoint. Using a distance-based method here allows us to avoid an expensive tree or network inference step and thus scale our method to many sequences.

We calculate, for each recombinant triple $\{a, b, c\}$, the evolutionary distance between each pair of segments before and after the breakpoint. We use here the BLOSUM62 distance [198] for amino acids and Hamming (mismatch) distance for DNA sequences (these could in principle be substituted by a large variety of ways to calculate evolutionary distance). We denote these distances by D_1 and D_2 for the

first (pre-breakpoint) and second (post-breakpoint) segment respectively. The pair with the smallest absolute difference in distance before and after the breakpoint are inferred to be the two non-recombinant sequences, while the third is inferred to be recombinant. Formally, we have

$$\text{recombinant} = \{a, b, c\} \setminus \underset{\{x,y\}\subset\{a,b,c\}}{\operatorname{argmin}} |D_1(x, y) - D_2(x, y)|.$$

This method identifies one recombinant from each recombinant triple; note that one recombination may generate one or more triples, but the identified recombinant from each of these triples should be the same. We apply this to all triples identified above, generating a list of recombinants in the entire dataset and their putative parents.

3.2.4 Calculating support values

In addition to identifying recombinant sequences, we can also measure the uncertainty in our identification by using bootstrapping. For each multiple alignment of a triple, we resample characters in the alignment (columns) within each segment, with replacement. This provides us with a resampled alignment, and we generate 100 replicates per triple. We then run our distance-based method to identify the recombinant for each replicate. The proportion of replicates which infer the same recombinant as the original alignment is the support value of this detection. The larger the support value, the more certain we are of the detection.

3.3 Results

3.3.1 Simulations

We conducted extensive simulations to evaluate the effectiveness of our method. Our simulation protocol is as follows:

1. Simulate a tree (genealogy) under the coalescent (without recombination) using `msprime` [199].

2. Evolve amino acid sequences from a common ancestor along the tree using `Pyvolve` [200]. If insertions and/or deletions are required, we use `INDELible` [201] instead.

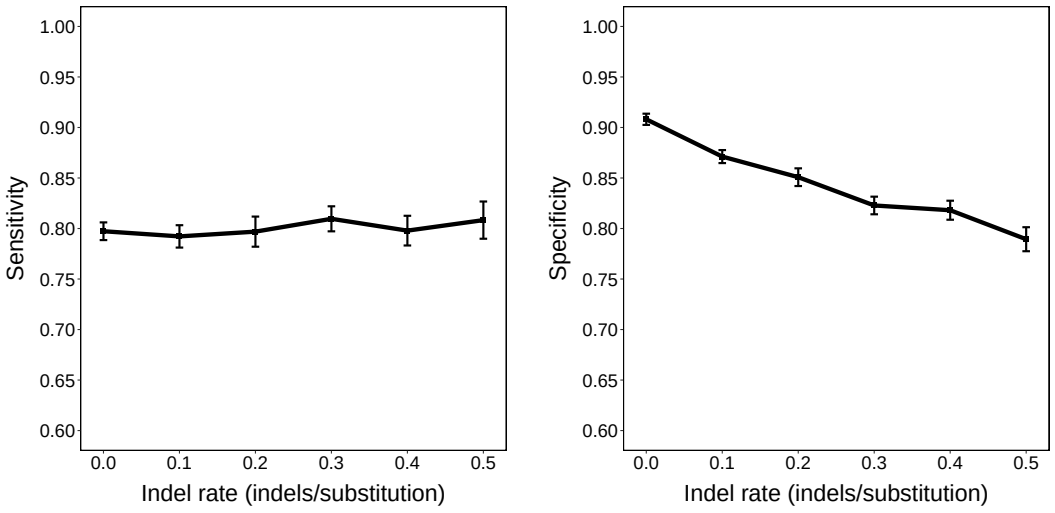
3. Generate recombinant sequences from two or more randomly chosen sequences in the dataset, with breakpoints chosen uniformly at random along the genome. The parent sequences are removed from the dataset.

This simulation produces a dataset which can be clearly separated into recombinants and non-recombinants. Manually performing the recombination step guarantees that we have only recent recombinants, which our method is designed to detect. Moreover, the non-recombinants are guaranteed to have no ancient recombination events in their history. Note that while we do not evolve our sequences further after recombination, we do remove the parents from the dataset, which produces a similar effect: their nearest extant relations in the dataset are evolutionarily separated from the recombinant sequence. In our simulations, we simulate both equal-length sequences (no indels), and unequal-length sequences with indel events, generating unaligned input.

There are a wide variety of parameters which could potentially affect the performance of the method. We vary the proportion of recombinant sequences in the dataset; the number of recombinations per recombinant; the number of sequences in the dataset; the sequence length; the mutation rate; and the substitution model. For simulations with insertions and deletions, we also vary indel rate and size. To keep our simulations tractable, we only vary one parameter at a time, keeping the remainder fixed at default values (Supplementary Tables 3.3 and 3.4). For each parameter combination, we simulate 100 datasets and run our method on each dataset in turn.

Our results are shown in Supplementary Section 3.5.2. In summary, we find that the method enjoys good performance, with most parameter settings offering both sensitivity and specificity above 70% (and often much higher). For the simulations without indels, we find that sensitivity increases with the number of recombinations, sequence length, and mutation rate, while staying stable with respect to the other parameters. Specificity decreases (usually slightly) as the proportion of recombinant sequences, number of recombinations, sequence length, and mutation rate increase.

An important feature of our method is its ability to accept unaligned sequences as input. When we include indels in the generating process, we can see (Figure 3.2)



(a) (b)

FIGURE 3.2: **Mean sensitivity and specificity (with 95% confidence intervals) for varying indel rate.**

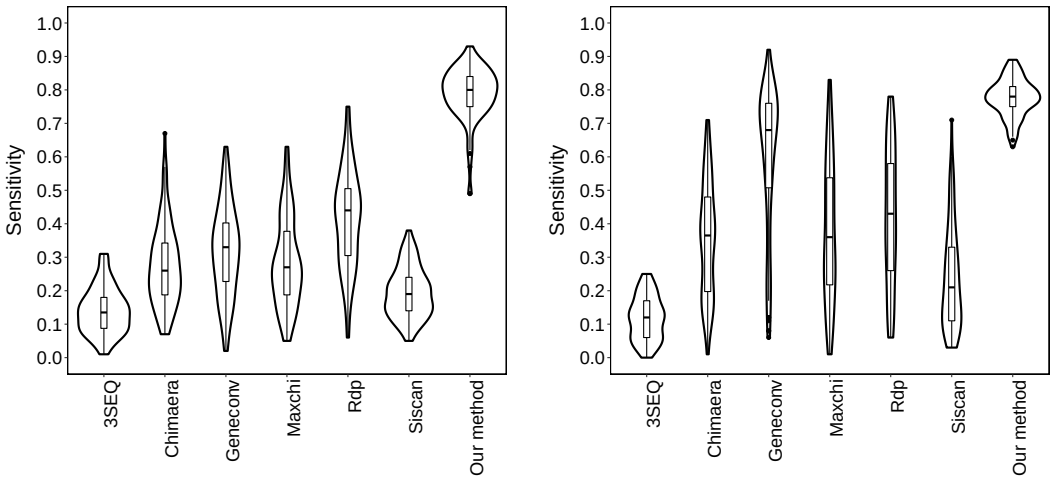
that both sensitivity and specificity remain relatively unaffected, with a moderate decline in specificity as indel rate increases. This indicates that our method is robust to indels even when the indel rate is large.

We also compared our method with a number of popular recombinant detection methods, after aligning the simulated sequences. We note that these methods only accept aligned sequences, making a direct comparison potentially biased one way or the other (depending on whether the sequences have indels or not). Despite this, we can see (Figure 3.3) that our method enjoys the highest sensitivity overall of finding recombinant sequences when we matched the specificity of other methods to that of our method, whether or not indels are included in the sequences. For more details, see Supplementary Section 3.5.2.2.

Finally, we studied the distributions of the support values for true and false detections, and the accuracy of the JHMM methods in our simulations (Supplementary Sections 3.5.2.4 and 3.5.2.5).

3.3.2 Analysis of DBLα sequences from a cross-sectional study in Ghana

Population genetic studies of *var* genes have focused on sequencing the DBLα domain, since nearly all *var* genes (except *var2CSA* [49–51]) encode a single DBLα



(a) (b)

FIGURE 3.3: **Distribution of sensitivity (for matched specificity) for different recombinant detection methods on simulated datasets with (left) and without (right) indel events.**

domain. We applied our method to detect recombinants and breakpoints in a dataset of DBLα sequences collected from individuals with microscopically confirmed *P. falciparum* infections (*isolates*) living in the Bongo District, in the Upper East region of northern Ghana (GenBank BioProject Number: PRJNA396962) [70, 85]. This dataset consists of 35,591 previously published DBLα sequences collected from 161 isolates, which were clustered into 17,335 representative DBLα 'types' of average length 125aa (s.d. 8.4aa). Of these, we detected 14,801 (85.4%) to be recombinant. See Supplementary Section 3.5.3.1 for more details.

3.3.2.1 DBLα sequences from the same ups group recombine more frequently

The upstream promoter sequences of each *var* gene can be classified into three main ups groups, upsA, upsB, and upsC [46]. Earlier studies on a much smaller dataset [61], based on sequence similarity, proposed that *var* gene recombination preferentially occurs within the same ups group. Using our method, which to our knowledge is the first systematic attempt to detect recombinants in *var* genes in natural parasite populations, we found considerable evidence supporting this hypothesis. Our results are summarised in Table 3.1.

TABLE 3.1: **Proportions of recombinations from the same ups groups and DBLα subclasses.** Expected proportions are given in brackets. All p-values are highly significant ($< 2.2 \times 10^{-16}$) except for the entry marked in red ($p = 0.2734$).

	Parent-child	Parents	Family (parents-child)
UpsA vs. upsB/C	99.7% (92.5%)	98.9% (85.0%)	98.5% (85.0%)
UpsA, B and C	85.3% (75.4%)	65.5% (50.9%)	51.1% (50.9%)
DBLα	58.8% (53.9%)	31.0% (7.9%)	20.6% (7.9%)

We calculated the proportion of recombination triples which have one parent and the child, both parents, and both parents and the child belonging to the same ups group ('Parent-child', 'Parents', and 'Family' in Table 3.1). With one exception, we found that the parents and/or the child of a recombination were significantly more likely ($p < 2.2 \times 10^{-16}$ from χ^2 tests) to belong to the same ups group, compared to a (conservative) null model where the parents have independent groups, but the child shares the group of one of its parents. (Under a more liberal model where the child group is also independent, all p-values are highly significant.) Our results strongly reinforce the conclusions of earlier studies, and provide more precision with the division into three ups groups.

We also considered the proportions of identified recombinants in each ups group. We found that there was a significant difference in the proportions of recombinants in the three groups ($p = 2.193 \times 10^{-7}$ from a χ^2 test), with upsA having the least proportion of recombinants, and upsC the most (82.3%, 84.9%, and 87.6% from A, B, and C respectively).

3.3.2.2 Proportions of recombination differ among domain subclasses

DBLα sequences can also be classified according to sequence similarity into 33 subclasses (DBLα0.1–24, DBLα1.1–8, DBLα2) [46]. These subclasses are strongly associated with ups groups; however, they also provide greater resolution in dividing the sequences. We thus repeated our earlier analyses with regards to the subclasses. As with ups group, we found a significant (all $p < 2.2 \times 10^{-16}$) increase in recombinations with one parent and the child, both parents, and both parents and the child from the same domain subclass (Table 3.1).

We next considered the proportions of identified recombinants in each subclass (Figure 3.4). We identified seven subclasses (DBLα0.1, 5 and 11 were too high, while

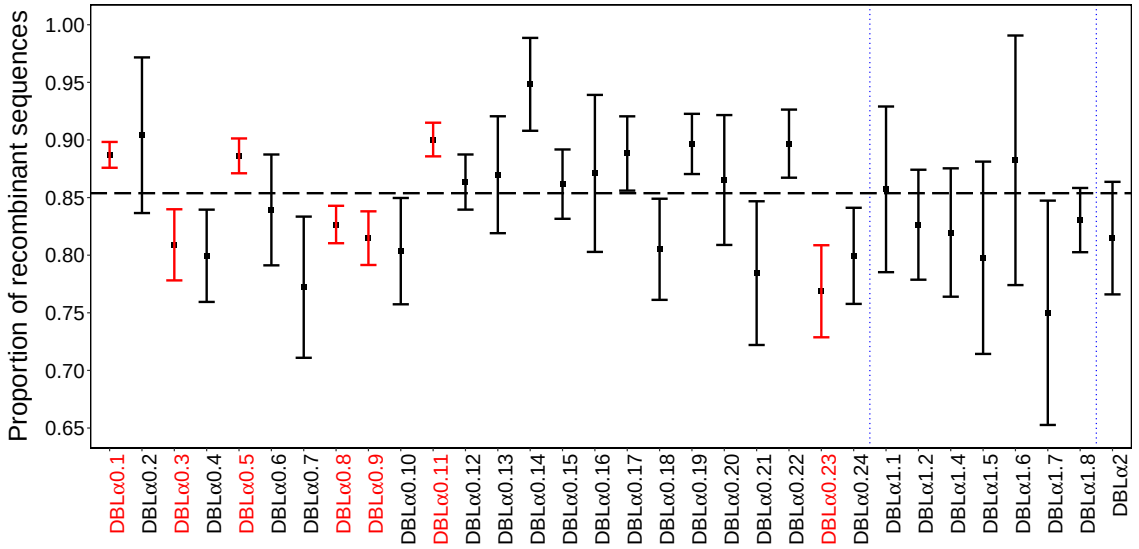


FIGURE 3.4: **Proportions (and 95% confidence intervals) of recombinants for each DBLα subclass.** Subclasses which are significantly different from the overall average (under a correction for multiple testing) are highlighted in red. The horizontal dashed line displays the overall proportion of recombinant sequences in the entire dataset.

DBLα0.3, 8, 9 and 23 were too low) which were significantly different from the average under a Bonferroni correction for multiple testing. Of particular note is the DBLα0.1 subclass, which has been noted to involve more recombinations than other subclasses [42]. We suggest that these subclasses should be explored further to determine if there are some biological factors that may explain these results.

We also investigated the proportion of recombinants among individual isolates, and among the two broad catchment areas in the Bongo District (Soe and Vea/Gowrie) that the isolates were collected from. We did not detect any significant differences here (see Supplementary Section 3.5.3.2).

3.3.2.3 Non-recombinant DBLα types are more conserved than recombinant types

It is well known [74, 202] that some DBLα types are highly conserved (appear in many different isolates) in a population (or even globally, 62). On the other hand, many other types only appear rarely, or even once. We hypothesise that non-recombinant types are more "stable" than recombinants, and thus may be more highly conserved.

TABLE 3.2: **Proportions of frequent (larger than the threshold) recombinant and non-recombinant DBLα types.**

Threshold	5	10	15	20
Recombinants	17.5%	4.5%	2.1%	1.3%
Non-recombinants	21.0%	6.0%	2.3%	1.6%
P-value (χ^2 test)	0.006	0.047	0.666	0.634

We investigated this hypothesis via the recombinants identified by our method. Firstly, we compared the observed frequencies in the dataset of the recombinants to the non-recombinants; we found that non-recombinants occurred significantly more often (average 4.2 vs. 3.7, $p = 0.021$ from a Wilcoxon rank sum test).

We also considered if there is a difference in the proportions of frequent DBLα types in recombinants and non-recombinants. As the frequencies of types are highly right-skewed (see Supplementary Figure 3.26), we thresholded the frequencies at various levels to determine if there were particular frequencies where an effect could be noticed. The results are in Table 3.2. We found that for a threshold frequency of 5, there were significantly fewer frequent recombinants than non-recombinants; however, this effect becomes less noticeable for larger thresholds. This suggests that there is a high proportion of recombinants which appear very few times in the dataset; these are potentially relatively recent recombinants, which may have not been established in the population.

3.3.2.4 Breakpoint positions are associated with homology blocks

It is known that a number of semi-conserved homology blocks (HBs) occur frequently in *var* genes [46]. These HBs recombine at exceedingly high rates [53, 56], and are known to be useful in predicting disease severity [82]. We thus investigated the patterns of recombination in DBLα types in relation to these homology blocks.

The positions of recombination breakpoints, as found by the JHMM method, are shown in Figure 3.5. Of particular note is:

- The recombination rate is not constant throughout the sequence, but displays three distinct peaks spaced in roughly equal intervals. These peaks clearly correspond to the three most frequent homology blocks, HB5, 14, and 36, with the height of the peak also corresponding to the frequency of the HB.

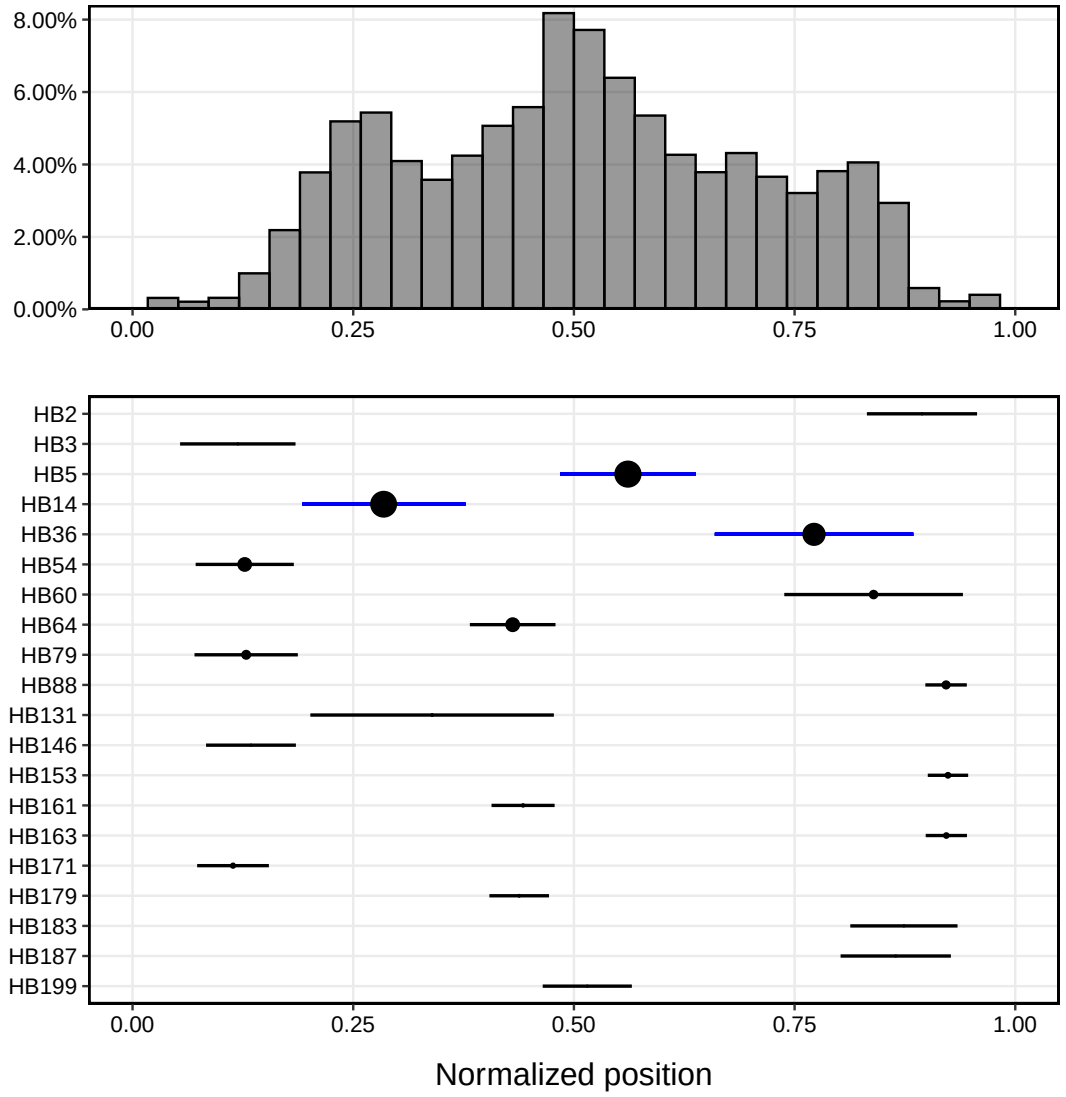


FIGURE 3.5: **Positions of recombination breakpoints.** (Top) The histogram of relative breakpoint positions of recombinations. (Bottom) The position of the most common homology blocks, with circle size proportional to frequency. The three most frequent homology blocks (HB5, 14, and 36) are highlighted in blue.

- The frequency of breakpoints drops sharply towards either end of the sequence. This is an artifact of the method and does not imply that the recombination rate is lower there; we cannot recognise a recombination which is close to one end of the sequence.

This reinforces the biological theory that recombination occurs within short identical segments [203].

3.4 Discussion

In this paper, we have developed a statistical method to detect recombinant sequences from a large set of genetic sequences without requiring a multiple alignment or a reference panel. We can also assess the reliability of the inferred recombinants with a bootstrapping-based tool. Simulations show that our method performs very well even when there is a high recombination rate, long sequences, or a large dataset. Crucially, it maintains its accuracy in the presence of insertions and deletions, where methods that require an alignment would normally fail. In a study of DBLα domains of *var* genes, comparisons between recombinant and non-recombinant DBLα types reveal a series of biologically meaningful results; we find evidence for the hypothesis that recombination is more frequent within ups groups [61], but also find that it is more frequent within domain subclasses. We also find novel results that recombinants differ from non-recombinants both in their representation in domain subclasses, and in their levels of conservation.

While our method is not strictly an alignment-free tool, it carries several advantages over methods based on a full multiple sequence alignment. Our method mostly aligns segments which are closely related to each other, thus increasing the reliability of the alignments; as datasets increase in size and variability, it will become more difficult to construct a reliable full alignment for all sequences. Moreover, our method only attempts to align three sequences at once, again saving time and increasing reliability. By identifying recombination triples directly from the JHMM, our method also avoids having to examine all possible triples of sequences one by one.

As noted above, our method is designed to only detect recent recombinants, which have not yet diverged in the dataset. For example, if a more ancient recombination produces a lineage that diverges into two sequences, they will be preferentially matched to each other by the JHMM, and it is possible that no recombination will be detected. The initial clustering of DBLα tags into types at 96% similarity (a standard part of the preprocessing pipeline [204]) may help in this regard, as the lineages must diverge beyond this threshold to be distinguished. The use of different clustering thresholds may affect the results, potentially unlocking access to signals of older recombinations.

Note that it is uncertain how long a recombinant will remain recent for, and this may well depend on sampling coverage and sample size. For example, although recombination events have been reported on timescales of several years [42], a recombinant

may continue to be 'recent' for far longer than that. The Ghana dataset studied in this paper is the first of a longitudinal dataset collected over several seasons, which may give insight into the frequency and patterns of recombination on epidemiological timescales; this is the subject of current work.

Furthermore, there is an implicit assumption that recombinations do not 'interact' with each other, i.e., that they are sufficiently far apart either in the evolutionary network or in the genome that we can decompose the dataset into recombinant triples and assess those independently. This is a strong (and perhaps unrealistic, in the context of genes which have a high recombination rate) assumption which we make in order to obtain a tractable algorithm. As seen from our results, we do appear to obtain good accuracy with our detections even in cases where this assumption might not hold; assessing the exact impact of this assumption on our results is also the subject of future work.

Although our methods are motivated primarily by the highly recombinant *var* genes, our approach is not restricted to these genes, but could be used for any genes which are recombinant but lack a reliable alignment or reference panel. The scalability of our method means that it will be applicable even to large datasets, thus holding great promise for broader applications.

Acknowledgements

We would like to thank Mun Hua Tan and Darren Martin for informative discussions and helpful feedback, and Martine Zilversmit and David Posada for kindly providing us with source code for their programs.

Funding

This study was supported by the Fogarty International Center at the National Institutes of Health (Program on the Ecology and Evolution of Infectious Diseases), Grant number: R01-TW009670 to KPD. Salary support for KET was provided by R01-TW009670 and The University of Melbourne. SRP was supported by a Melbourne International Engagement Award from The University of Melbourne. Salary support for MFD was provided by The University of Melbourne. QF has been supported by the China Scholarship Council.

3.5 Supplementary materials

3.5.1 Methods

3.5.1.1 Identifiability: a phylogenetic perspective

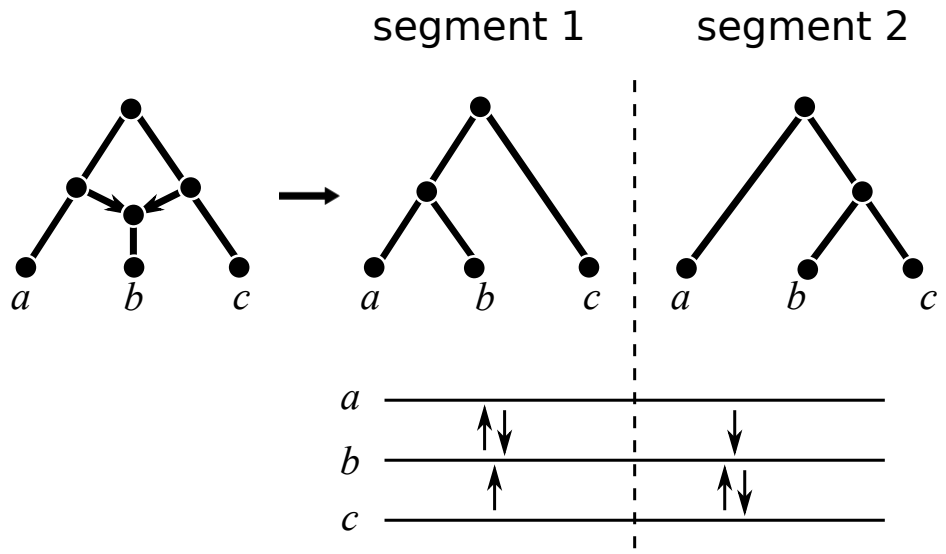
An important component in our method is the ability to identify which member of a triple is the true recombinant. It is important to note that the JHMM method does *not* identify the recombinant, but instead finds the (segments of) extant sequences which are the most closely related to the target sequence.

This can be illuminated by considering an explicit phylogenetic network [205] with three aligned sequences and one recombination as an example, as shown in Figure 3.6. Here, we can translate a phylogenetic network to the corresponding mosaic representations, assuming the JHMM method estimates the distances between sequences perfectly. It can be seen that the same mosaic structure can result from networks with different recombinants.

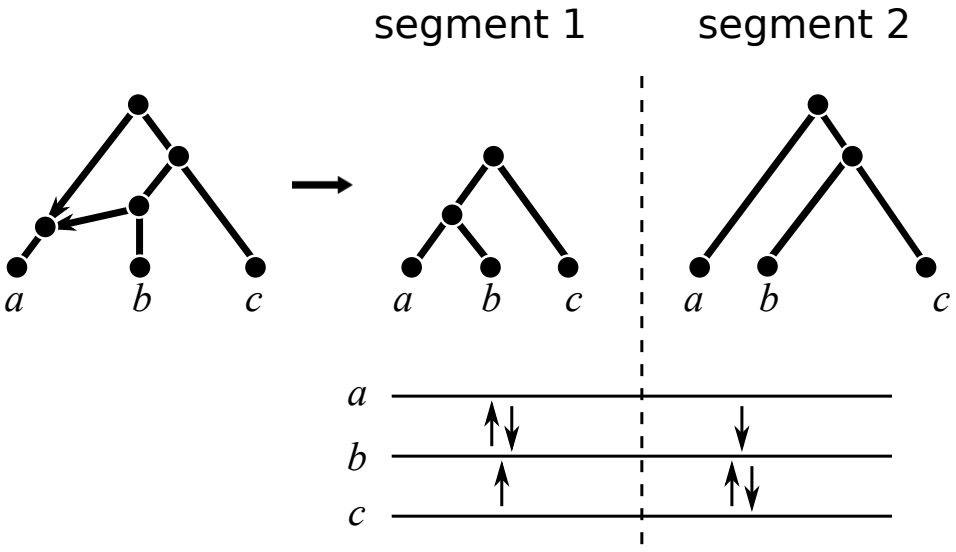
In fact, as discussed at length by Pardi and Scornavacca [206], this is an unavoidable problem with the identifiability of phylogenetic networks; networks cannot be distinguished solely by the topologies of displayed trees, which the output of the JHMM method is dependent on. The solution, as given in that paper, is to use (inferred) branch lengths to distinguish between the networks, and thereby identify the recombinant.

When the phylogenetic network only consists of three sequences and one recombination (as in Figure 3.6), it is easy to translate the network to the JHMM output, and thus use it to find the recombinant. However, the problem rapidly becomes much more complicated with more sequences and/or recombinations, and indeed for ancestral recombinations (predating a divergence) it's not even clear how to define an extant 'true recombinant'. To avoid this problem, we only identify triples of sequences, and assume that only one recombination occurs in the recent evolutionary history of each triple. For large datasets, we are essentially assuming that recombinations are 'sufficiently far apart' either in the network or in the genome that they do not interact with each other.

From a phylogenetic perspective, we can see that when this assumption holds, identifying only triples breaks down a complicated network into repeated cases of a three



(a) Sequence b is the recombinant



(b) Sequence a is the recombinant

FIGURE 3.6: **Identifiability of networks from the JHMM output.** Here, two networks with different recombinants produce the same profile tree topologies, and thus the same JHMM output. The JHMM output is depicted below the profile trees, with arrows from each target segment pointing to the matching source segment (so, for example, if b is the target sequence, it is matched to source sequence a in segment 1 and c in segment 2 in both cases). Both cases produce identical JHMM output: in particular, sequence b is matched to two different source sequences even though it is not necessarily the recombinant.

sequence–one recombination network, for which we can identify the recombinant. See Figure 3.7 for an example of this.

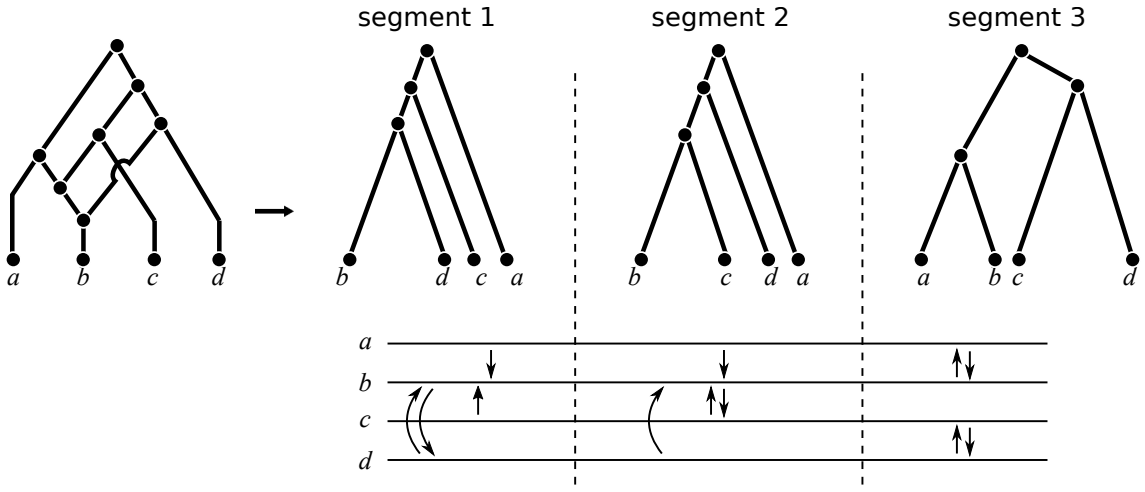


FIGURE 3.7: **Decomposing a network into triples.** At the first breakpoint, the triple $\{b, c, d\}$ is identified from target sequence b, while at the second breakpoint, $\{a, b, c\}$ is identified from sequence b, and $\{b, c, d\}$ from sequences c and d. In all cases, distance-based recombinant identification will obtain the correct recombinant (b at both breakpoints).

3.5.2 Simulation results

3.5.2.1 Effect of parameters

The parameters that we vary in the simulations, and their ranges, are shown in Tables 3.3 and 3.4. We vary one parameter at a time, holding the remainder at default values (shown in bold in the tables). We now consider the effect of each parameter in turn.

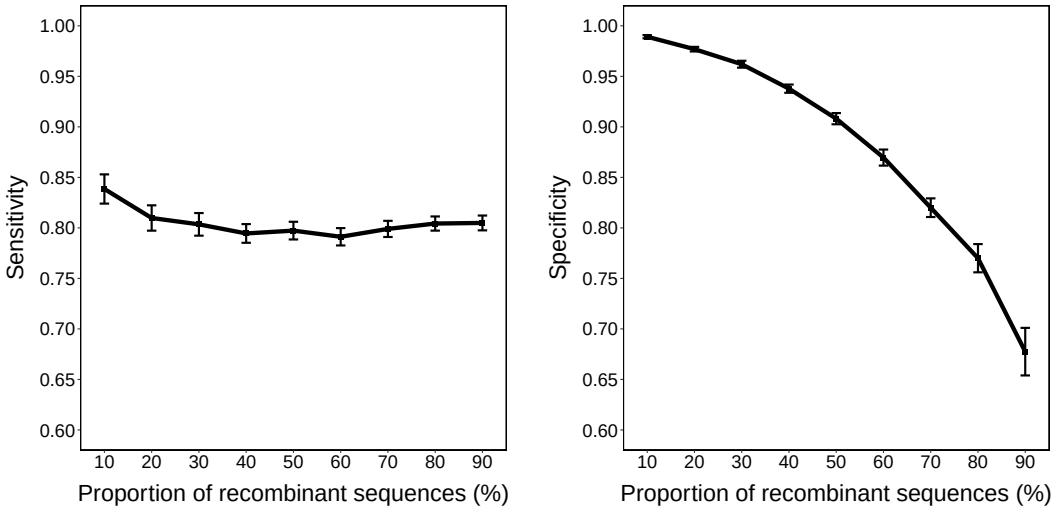
TABLE 3.3: **General simulation parameters (no indels).** We vary each parameter in turn while holding the others fixed at the default values (in bold).

Parameter	**Values**
① Proportion of recombinant sequences (%)	10, 20, 30, 40, **50**, 60, 70, 80, 90
② Average number of recombinations per recombinant sequence	1.0, 1.1, 1.2, 1.3, 1.4, **1.5**, 1.6, 1.7, 1.8, 1.9, 2.0
③ Dataset size (sequences)	100, 150, **200**, 250, 300, 350, 400, 450, 500
④ Sequence length (AA)	100, 150, **200**, 250, 300, 350, 400, 450, 500
⑤ Mutation rate (substitutions/site/coalescent unit)	0.1, 0.2, 0.3, 0.4, **0.5**, 0.6, 0.7, 0.8, 0.9, 1.0
⑥ Amino acid evolution model	AB [207], DAYHOFF [208], JTT [209], LG [210], MTMAM [211], **WAG** [212]

TABLE 3.4: **Indel simulation parameters (default values in bold).** Insertions and deletions are simulated at the same rate, with lengths according to a negative binomial distribution with variance 10.

Parameter	Values
⑦ Indel rate (expected number of indels/substitution)	0.1, 0.2, **0.3**, 0.4, 0.5
⑧ Mean indel size (AA)	3.7, 5.2, **6.0**, 6.6, 7.0

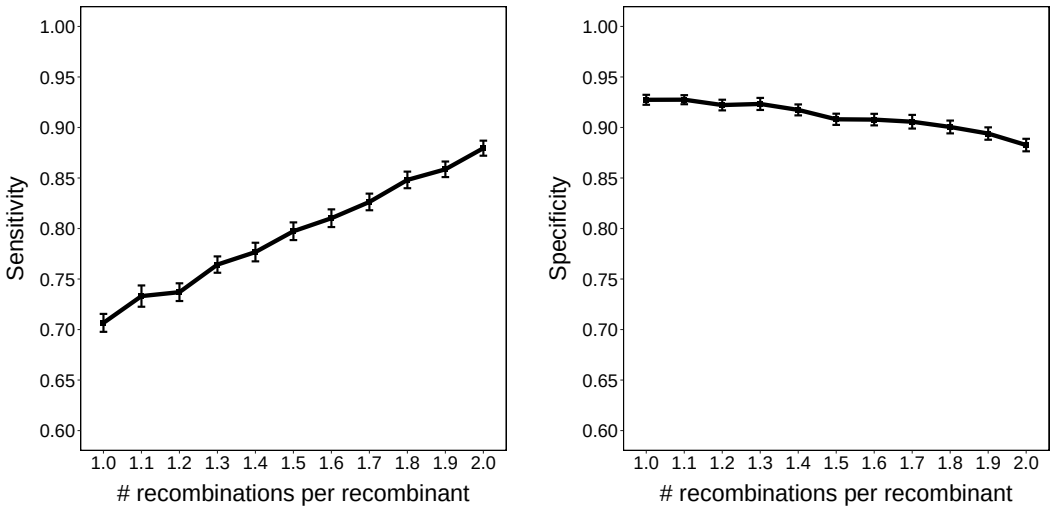
Recombinant proportion As the proportion of recombinants increases, sensitivity is stable at around 80%, while specificity decreases (Figure 3.8). Here, more recombinant sequences result (correctly) in a higher number of recombinations detected. It appears that the proportion of true recombinants extracted from the recombinant triples remains largely the same (constant sensitivity); however, there are proportionally more false detections as the number of non-recombinants decreases, resulting in a lower specificity.



(a) (b)

FIGURE 3.8: **Mean sensitivity and specificity (with 95% confidence intervals) for varying proportions of recombinant sequences.**

Number of recombinations per recombinant As shown in Figure 3.9, the datasets where there are more recombinations per recombinant sequence appear to have a higher sensitivity, and slightly lower specificity. As with recombinant proportion, an increase in the number of recombinations results (correctly) in more inferred recombinations; unlike that case, the number of true recombinants remains the same here. It appears that the 'extra' detections are mostly correct, which results in a greater proportion of true positives (sensitivity increases) and a relatively stable specificity.



(a) (b)

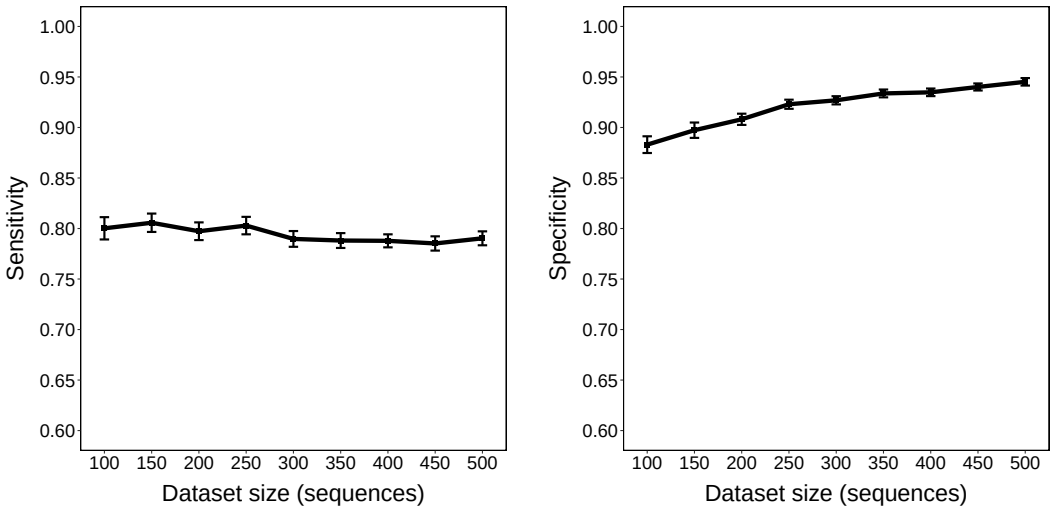
FIGURE 3.9: **Mean sensitivity and specificity (with 95% confidence intervals) for varying numbers of recombinations per recombinant sequence.**

We also conducted a further analysis by matching the distribution of the number of recombinations per recombinant to the Ghana dataset (see Supplementary Section 3.5.3.4 and Figure 3.27 for more details). Our results indicate that, despite a low specificity (40.0%), a high sensitivity (89.0%) still demonstrates the applicability of our algorithm to real data.

Dataset size Dataset size does not appear to have a drastic effect on the sensitivity of the method, while specificity increases slightly (see Figure 3.10). It is to be expected that performance increases slightly as information accumulates across a larger dataset, but it is unclear why this is only expressed in the specificity here.

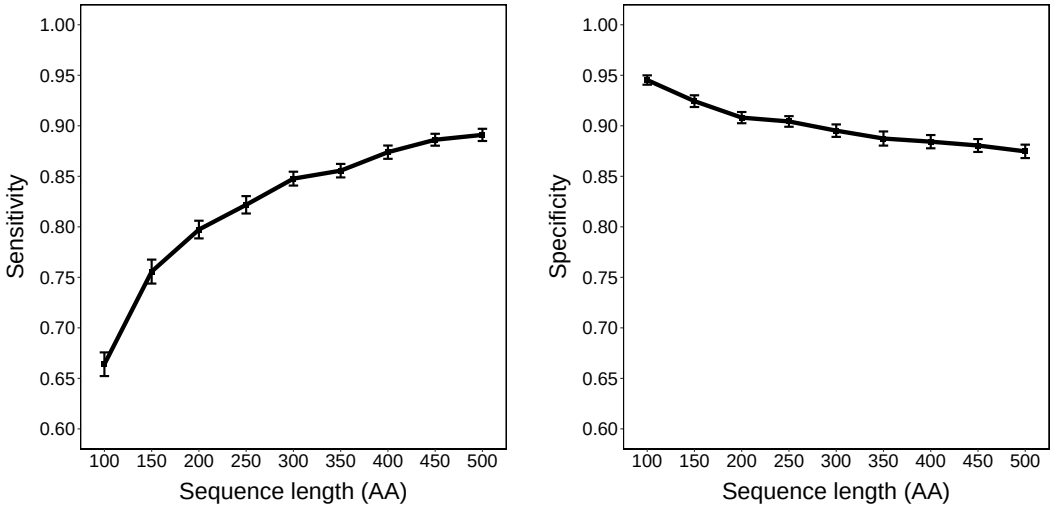
Sequence length Datasets with longer sequence length have much higher sensitivity, and slightly lower specificity (Figure 3.11). It seems (Figure 3.12) that as sequence length increases, the number of recombinations detected also increases, even though the true number of recombinations remains the same. This increase in detections, combined with a fixed percentage of recombinants, results in a effect similar to that seen for the number of recombinations per recombinant: an increase in sensitivity and a slightly decreasing specificity.

Mutation rate As the mutation rate increases, the sensitivity of the method rapidly increases before levelling out (Figure 3.13). This makes sense, as if the



(a) (b)

FIGURE 3.10: **Mean sensitivity and specificity (with 95% confidence intervals) for varying dataset size.**



(a) (b)

FIGURE 3.11: **Mean sensitivity and specificity (with 95% confidence intervals) for varying sequence length.**

number of substitutions is too low, the sequences are difficult to distinguish from each other, which makes the results from the JHMM unreliable. Conversely, as the number of substitutions grows, it also becomes more difficult to identify sequences which are closely related to each other, resulting in a decrease in specificity.

Evolution model The method appears to be robust to the stochastic model of amino acid evolution (Figure 3.14).

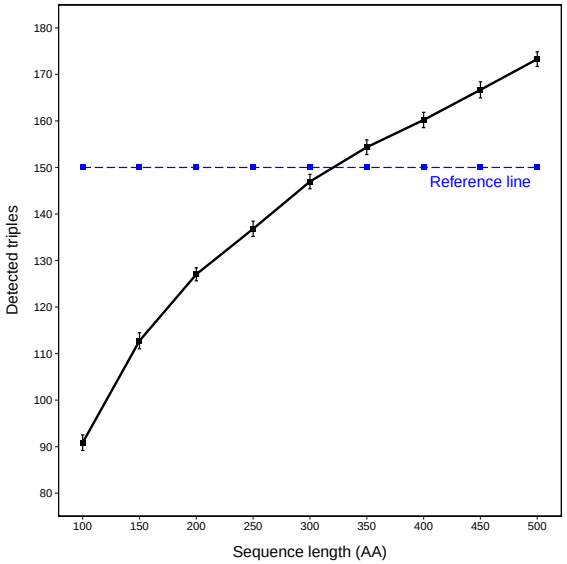
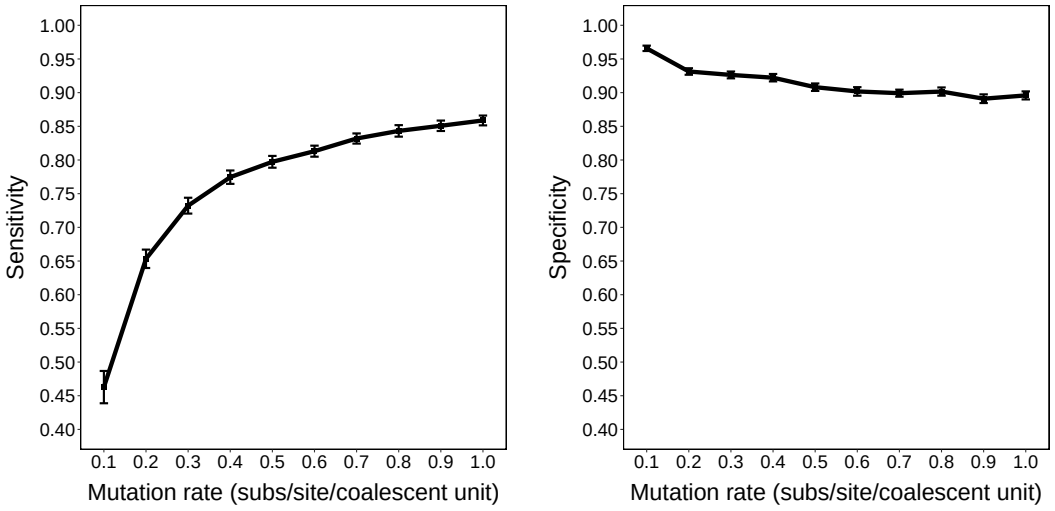


FIGURE 3.12: **The number of recombinant triples detected by our algorithm for varying sequence length.** The reference line indicates the true number of recombinant triples in the dataset.

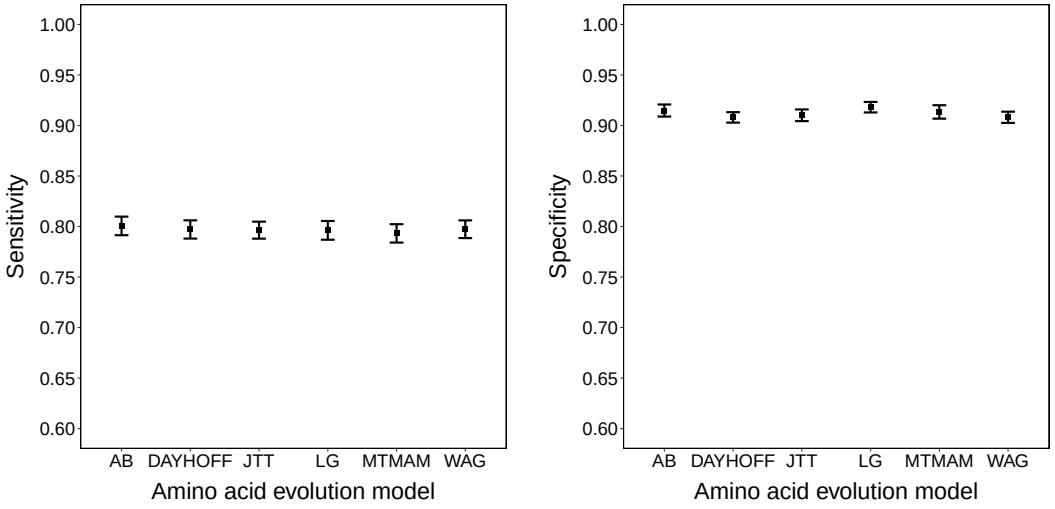


(a) (b)

FIGURE 3.13: **Mean sensitivity and specificity (with 95% confidence intervals) for varying mutation rate.**

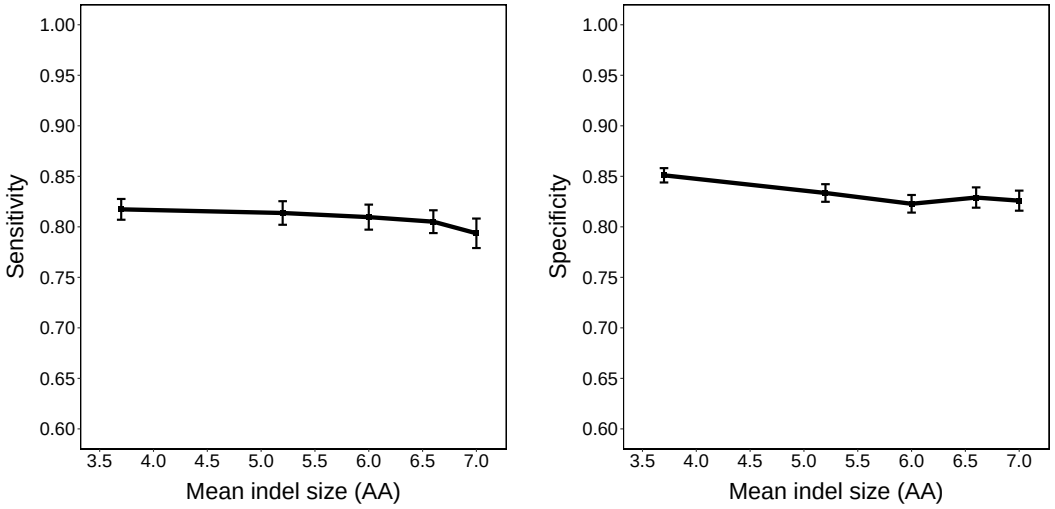
Indel size When indels are included in the generating process, accuracy is not affected by the size of the indels (Figure 3.15).

Running time As expected, the only parameters which affect the running time of the algorithm are dataset size and sequence length. In Figure 3.16, we show the running time of the simulations for each replicate (without parallelisation; see below). The running time appears to grow quadratically with respect to both dataset



(a) (b)

FIGURE 3.14: **Mean sensitivity/specificity (with 95% confidence intervals) for each model of amino acid evolution.**

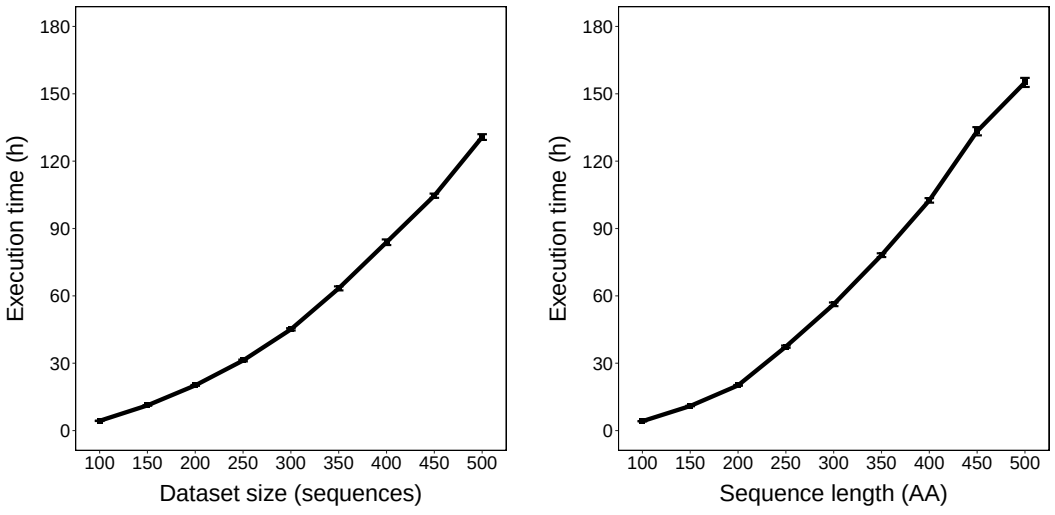


(a) (b)

FIGURE 3.15: **Mean sensitivity and specificity (with 95% confidence intervals) for varying indel size.**

size and sequence length (the slopes of regressions on the log-log data are 2.09 and 2.28 respectively). This compares favourably to many recombinant detection methods which are based on examining all triples of sequences, and are thus $O(n^3)$ in the dataset size.

While the total running time becomes quite large at even moderate dataset sizes, the algorithm is easily parallelisable in a relatively naive way. The main computational task of the algorithm is in the determination of Viterbi paths for every sequence in



(a) (b)

FIGURE 3.16: **Average running time per replicate (and 95% CIs) for varying dataset size (left) and sequence length (right).**

the dataset with respect to all other sequences. This is used for both training the JHMM, and calculating its final output. By computing the Viterbi paths for each sequence in parallel, we can achieve massive savings in real time; for example, the *var* gene dataset can be analysed in a tractable amount of time even with many more sequences.

On the other hand, this parallelisation does not produce any benefits as the length of the sequences grow longer. Thus our algorithm is more suited to the analyses of massive datasets of relatively short sequences.

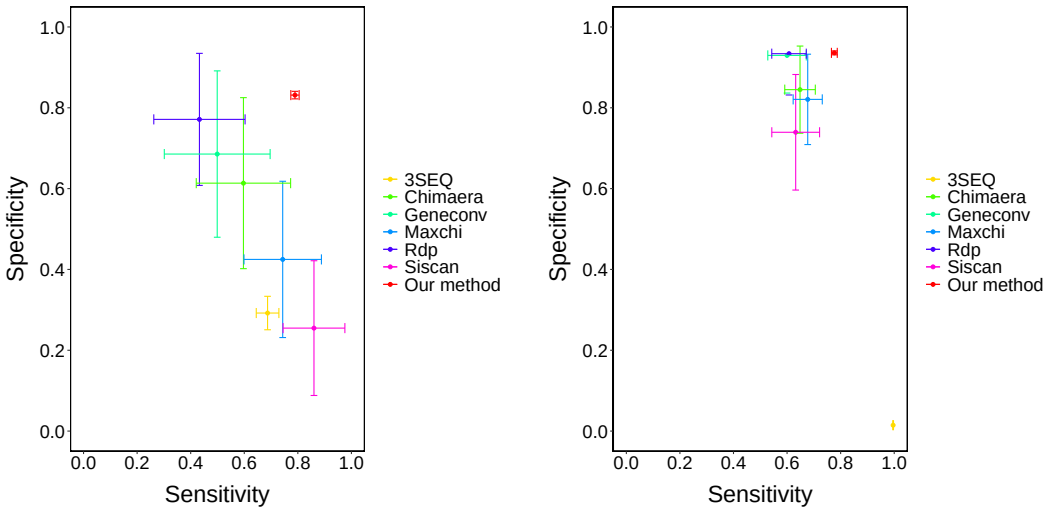
3.5.2.2 Comparison with other methods

We compared our method with the recombinant detection methods 3SEQ [99, 100], Chimaera [97], GeneConv [213], MaxChi [96], RDP [109], and Siscan [107]. All but the first method are implemented in RDP5 Beta 6 [110].

As these methods mostly accept aligned DNA sequences as input, we simulated DNA sequences with length 200nt under the F81 substitution model [214]. Other parameters followed the default simulation settings in Tables 3.3 and 3.4. We simulated both with and without indel events, then aligned the resulting sequences with MUSCLE v3.8.31 [215] for methods requiring an alignment.

As our method does not utilise p-values, in order to ensure a fair comparison we thresholded the p-values output by other methods so that the specificity (false detection rate) of all the methods are the same. We then compared the sensitivities of each method.

In addition, we also compared both the sensitivity and specificity of all the methods for their default settings (Figure 3.17). With indels simulated, our method is clearly superior in both sensitivity and specificity, as expected; with no indels simulated, methods based on aligned sequences perform better than before, but our method is still superior.



(a) With indels

(b) Without indels

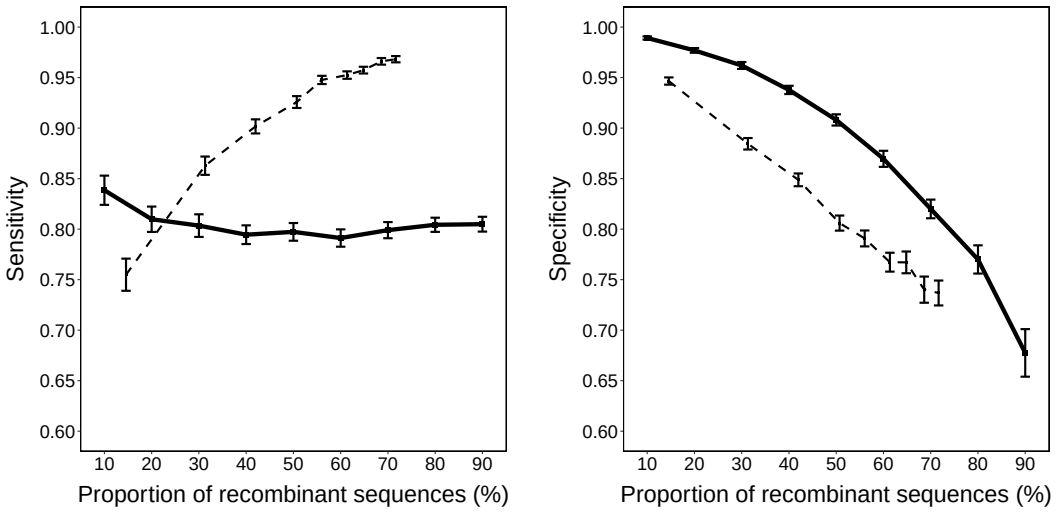
FIGURE 3.17: **Sensitivity and specificity (and 95% CIs) under default parameters with and without indel events.**

3.5.2.3 Ancient recombinations

Our simulations are designed to only contain 'recent' recombinations, that is recombinations which only descend to one sequence in the dataset. This allows us to have complete control over the proportion and make-up of recombinants, and to unambiguously distinguish between recombinants and non-recombinants. On the other hand, it is possible that our method may be hindered by the presence of ancient recombinations which descend to a number of sequences in the dataset.

To test this, we used the `msprime` software to simulate sequences under the full coalescent with recombination; this allows recombinations to occur throughout the

evolutionary history of the sequences. We use default values for the simulation parameters (apart from the proportion of recombinant sequences and average number of recombinations per recombinant), and vary the recombination rate, producing varying proportions of recombinant sequences in the dataset. We then determined each sequence as a recent recombinant if and only if it is the only extant descendant of an ancestral segment produced by a recombination (i.e., a segment surrounding the recombination breakpoint).



(a) (b)

FIGURE 3.18: **Sensitivity and specificity (and 95% CIs) for recent recombinations only (solid lines) and recombinations allowed throughout (dashed lines), for varying recombination rate.**

We observe that our method still retains a lot of power to detect recent recombinants under this scenario, with slightly higher sensitivity and slightly lower specificity compared to our previous simulations. Indeed, the sensitivity improves with the recombination rate; it appears, rather pleasingly, that our method has some ability to even detect the signal of older recombinations.

3.5.2.4 Support values

In addition to detecting recombinants, we also calculate support values for each detection using bootstrapping. Here, we verify that the calculated values are indeed effective for this purpose. For our simulations, we calculate the support values for each of the correct detections, as well as each of the false positives. The distributions of the support values for the default parameters are shown in Figure 3.19. Here, we

can see that there is a clear separation between the distributions of support values for the true and false positives; while the values for both are relatively high, the support values for true detections are overall much higher. Similar patterns are seen among all the remaining parameter settings (Figures 3.28–3.35).

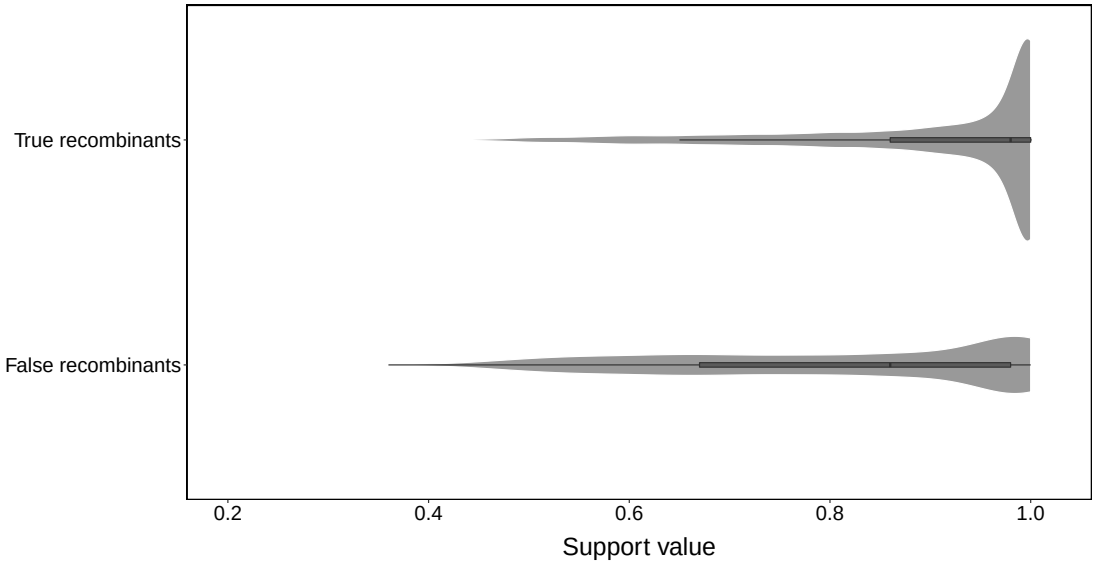


FIGURE 3.19: **Distributions of support values under default parameters without indel events.**

This suggests that we can use a threshold on the support value to refine our detections. This is reasonable if we wish to reduce false positives; however, in practice we found that applying a threshold also reduced true positives (as expected) to an extent which lowered the overall accuracy of the method, so we have elected not to apply it here. Instead, we suggest that the support value be used to assess the confidence which should be placed in individual recombinant detections of interest.

3.5.2.5 Accuracy of the JHMM method

The JHMM method of Zilversmit et al. [54] forms a key part of our method to detect recombinants. Until now, there has not been a systematic study of the accuracy of this method. Two key outputs of this method are the locations of the inferred recombination breakpoints, and the estimated recombination parameter ρ. Here, we study the accuracy of these inferences for our simulated datasets.

Recombination breakpoints For each recombination, we calculate the distance between the true and inferred breakpoints. For ease of comparison, we restrict this

analysis to the case where each recombinant sequence has exactly two parents (one recombination), which avoids the problems of matching breakpoints in the same sequence to each other.

We find in general (see Figure 3.20) that the breakpoints are very accurately inferred, with 38.4% of all breakpoints inferred exactly, and 75.0% being at most 5AA from the true value. There is also a slight but noticeable positive bias, where the inferred breakpoints tend to be slightly to the right of the true breakpoints (Figure 3.21). This can be best explained by noting that the JHMM method infers the best (Viterbi) path from left to right, and recombinations are considered relatively unlikely; hence a recombination will tend to be inferred slightly later than it actually is, particularly if both parents' sequences are identical around the breakpoint.

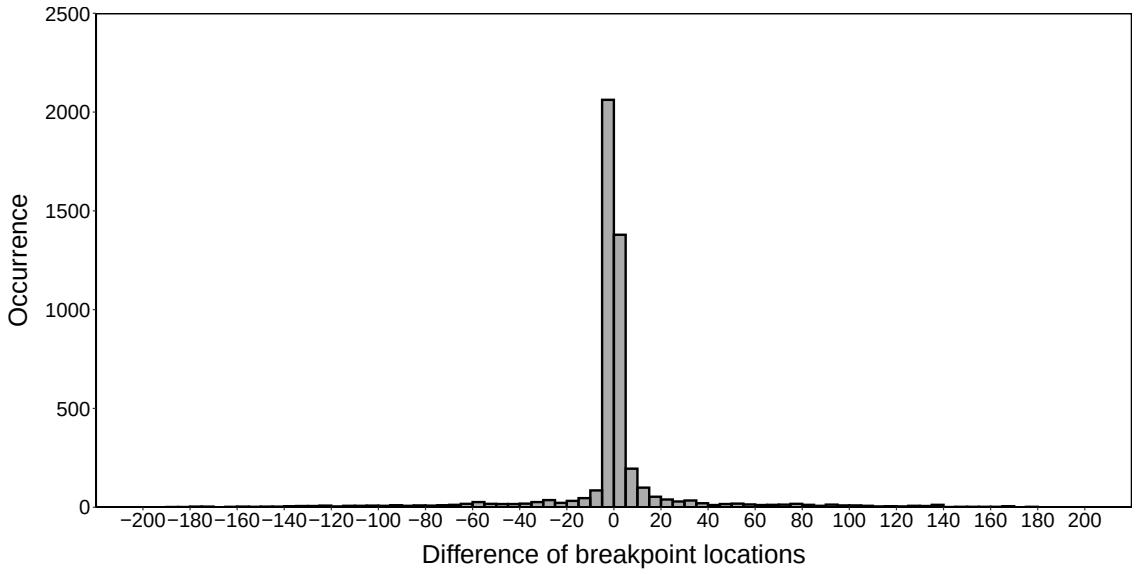


FIGURE 3.20: **Breakpoint inference error of the JHMM method under default simulation parameters.**

Finally, we note that the breakpoint accuracy appears to be very robust to indel events; this is expected, since the method explicitly accounts for these events.

Recombination rate The parameter ρ, the probability of switching between source sequences after any character, is directly related to the recombination rate in the dataset (although it does not provide a rate in terms of time dimension). As such, an accurate estimate of ρ is valuable for molecular phylogeneticists. We observe in our simulated datasets (Figures 3.36-3.39) that the inferred values of ρ provide an accurate estimate of the recombinaton rate.

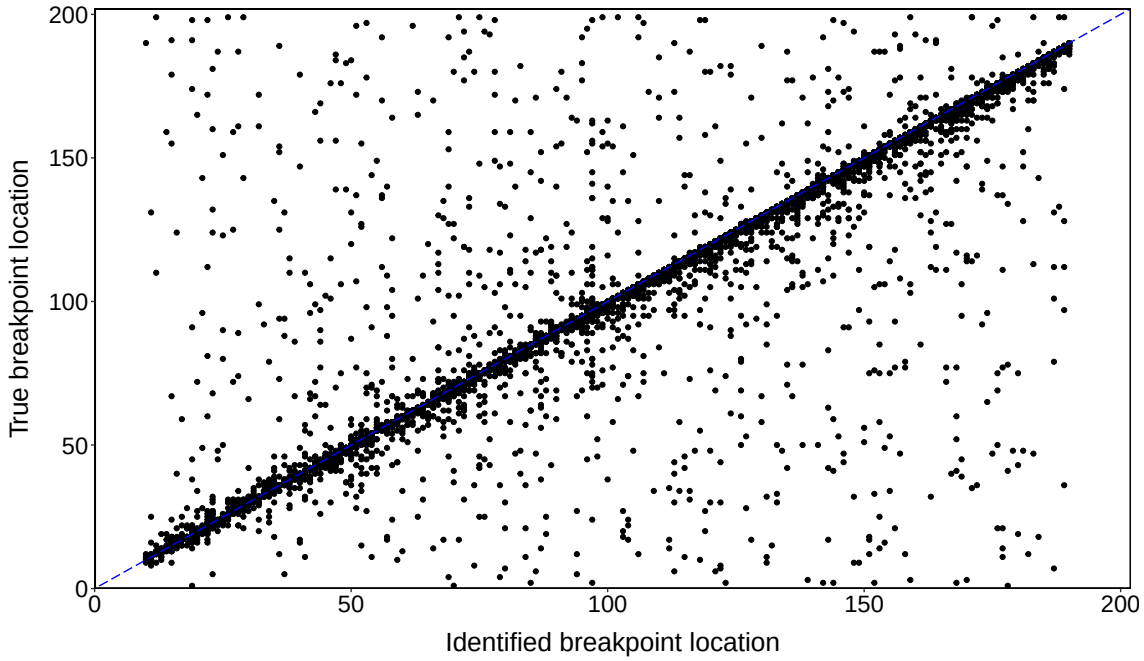


FIGURE 3.21: **Breakpoint inference of the JHMM method under default simulation parameters.** Most points cluster around the line $y = x$, indicating a high accuracy of breakpoint inference. However, there is a slight positive bias in the identified breakpoint location, particularly for breakpoints which occur later in the sequence.

On the other hand, the inferred ρ can also be affected by mutation rate (Figure 3.22) and (to a lesser extent) indel events (Figures 3.40-3.41); here, an increasing rate of indel events leads to some of them being mistaken for recombination, distorting the inference of the recombination rate. This indicates that the use of the JHMM to infer the true recombination rate has the potential to be inaccurate.

3.5.3 Analysis of DBLα sequences from a cross-sectional study in Ghana

3.5.3.1 Data handling

Details on the study population, data collection procedures, and epidemiology have been published elsewhere [73, 83, 84].

Preprocessing We follow the standard pipeline used in [62, 74]. The DNA sequences were first translated into protein sequences, and removed if the resulting sequence contained a stop codon. The protein sequences were then clustered with

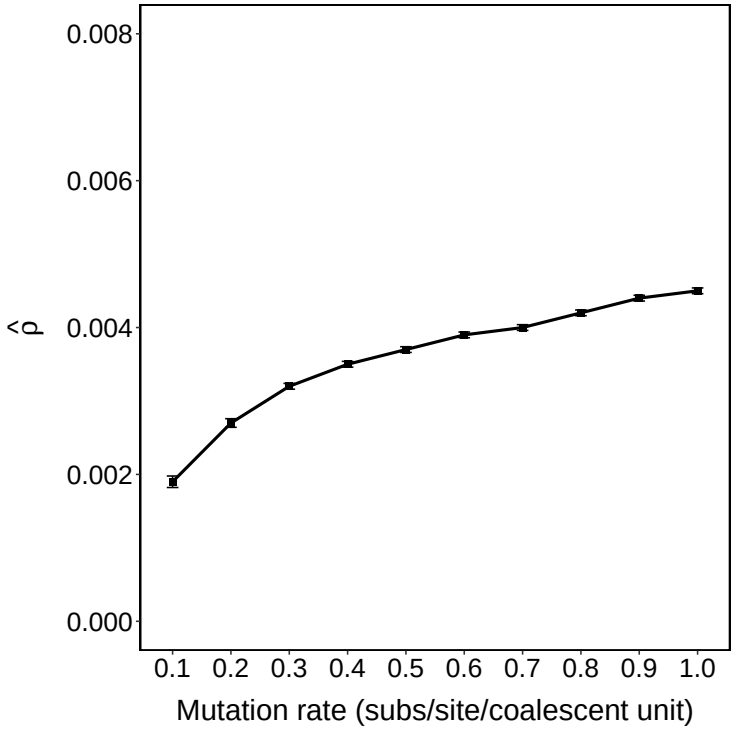


FIGURE 3.22: **Estimated ρ (and 95% CI) with varying mutation rate (but constant number of recombinations).**

the Usearch software (v8.1.1861) [216] with a 96% sequence similarity cutoff [204]. The cluster centroids were then taken as a representative sequence for the clusters, which are known as DBLα *types*. This results in a dataset of 17,335 types, each of which may appear in several isolates.

Identifying recombinants We applied our method to this dataset to detect recombinant types. We detected 14,801 (85.4%) of the DBLα types to be recombinant.

The analysis was run on a high performance cluster at the University of Melbourne (72 Intel(R) Xeon(R) Gold 6254 CPU cores @ 3.10GHz, 768GB RAM). The computation of Viterbi paths for each sequence, which is necessary for both estimating parameters and identifying recombinants, can be performed in parallel; we computed Viterbi paths for 30 sequences (against all other sequences in the dataset) at a time on one core (578 subsets total). The total time taken was 943 minutes; this is broken down in Table 3.5. By far the largest bottleneck is the computation of the mosaic representations of the sequences (both parameter estimation and computation of the Viterbi paths); once this was completed, the remaining steps are very efficient even for a dataset of this size.

TABLE 3.5: **Time and memory consumption of the algorithm on the Ghana dataset.**

	JHMM parameter estimation	**Viterbi paths**	**Recombinant identification**
Time (minutes)	644.8	294.9	2.7
Memory (GB)	21.3	21.2	0.1

3.5.3.2 Recombinant proportions across isolates and catchment areas

We investigated the proportion of recombinants among individual isolates to determine if there were certain isolates with an elevated or reduced proportion of recombinants. Excluding isolates with less than 20 DBLα types [62, 74] resulted in a total of 158 isolates with a mean of 217.1 DBLα types per isolate (range 33–833). We tested if the average proportion of DBLα types in each isolate was equal to the overall dataset proportion with a t-test with a Bonferroni correction for multiple testing. There were no isolates which had a significantly different proportion of recombinants under this test (see Figure 3.25).

In addition, 133 isolates (82.6%) were from two catchment areas: Soe and Vea/-Gowrie. A χ^2 test showed no significant difference between the proportion of recombinants from these two areas ($p = 0.992$).

3.5.3.3 Detection of HBs in recombinant and non-recombinant DBLα types

The location of homology blocks (HBs) in each sequence was obtained using the VarDom server [46] with the default cut-off of 9.97 as a threshold to define a match. For each HB, we averaged the leftmost and rightmost relative positions of each occurrence in a sequence to obtain the overall location of the HB.

We identified in total 41 different HBs in the database (mean 5.5, range 1–10 HBs per sequence). HBs are numbered based on the frequency of occurrence [46], with HB1 the most frequent. We found that the frequency of HBs in our dataset also decreased with the numbering, with the exception of HB2 and HB3; these HBs are frequent, but lie partially outside the DBLα tag boundaries, making it difficult to positively identify them in the dataset. The most frequent HBs in our dataset were HB5, HB14, and HB36.

To compare sequences directly based on HBs, we used the pairwise HB similarity [82]. This is defined as the number of HBs shared between any two sequences, divided by the average number of HBs within a sequence.

We discovered that the number of HBs in recombinant sequences were significantly higher than in non-recombinant sequences (5.5 vs. 5.3, $p < 2.2 \times 10^{-16}$ from Wilcoxon rank sum test). Furthermore, the proportion of sequences containing "important" HBs (5, 14, and 36) were also significantly different between the two groups (83.9% vs. 78.5%, $p = 1.859 \times 10^{-11}$ from χ^2 test), indicating that recombinants tend to have more conserved building blocks. Finally, we found that recombinant sequences had higher pairwise HB similarities with each other than non-recombinants (0.629 vs. 0.618, $p < 2.2 \times 10^{-16}$ from Wilcoxon rank sum test).

3.5.3.4 Matching recombination numbers to real data

We performed an additional simulation to match the distribution of the number of recombinations per recombinant sequence to the Ghana data. To do this, we applied the JHMM method to the Ghana data, and extracted the number of source segments matched to each target sequence (Figure 3.27). From this Figure, we observe that it is extremely rare to have a sequence match to 8 or more source segments (i.e., 7 recombinations), so we do not allow this to happen in our simulations.

The primary difficulty here is that the JHMM method appears to slightly overestimate the number of recombinations, which means that if we simulate recombinations in exactly the same proportion as found from the Ghana data, the JHMM method produces a recombination frequency which is slightly too high. To accommodate this, we tested five sets of probabilities in simulation, and selected the probabilities of (0.02, 0.30, 0.21, 0.23, 0.14, 0.11, 0.00) for 1–7 source segments (0–6 recombinations) for each sequence. This produced a distribution of numbers of recombinations which was similar to the Ghana data.

3.5.4 Figures and tables

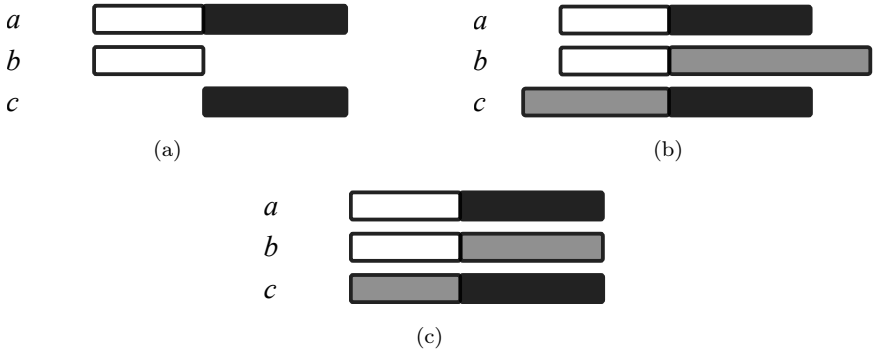


FIGURE 3.23: **An overview of calculating a multiple sequence alignment with MAFFT.** (a): A segmental pairwise alignment generated by the JHMM method. Segments from sequence a are aligned to segments from sequences b and c respectively. (b): Using MAFFT, we include the corresponding segment from the third sequence into the pairwise alignment on either side of the breakpoint. (c): By trimming the alignments, we generate a multiple alignment.

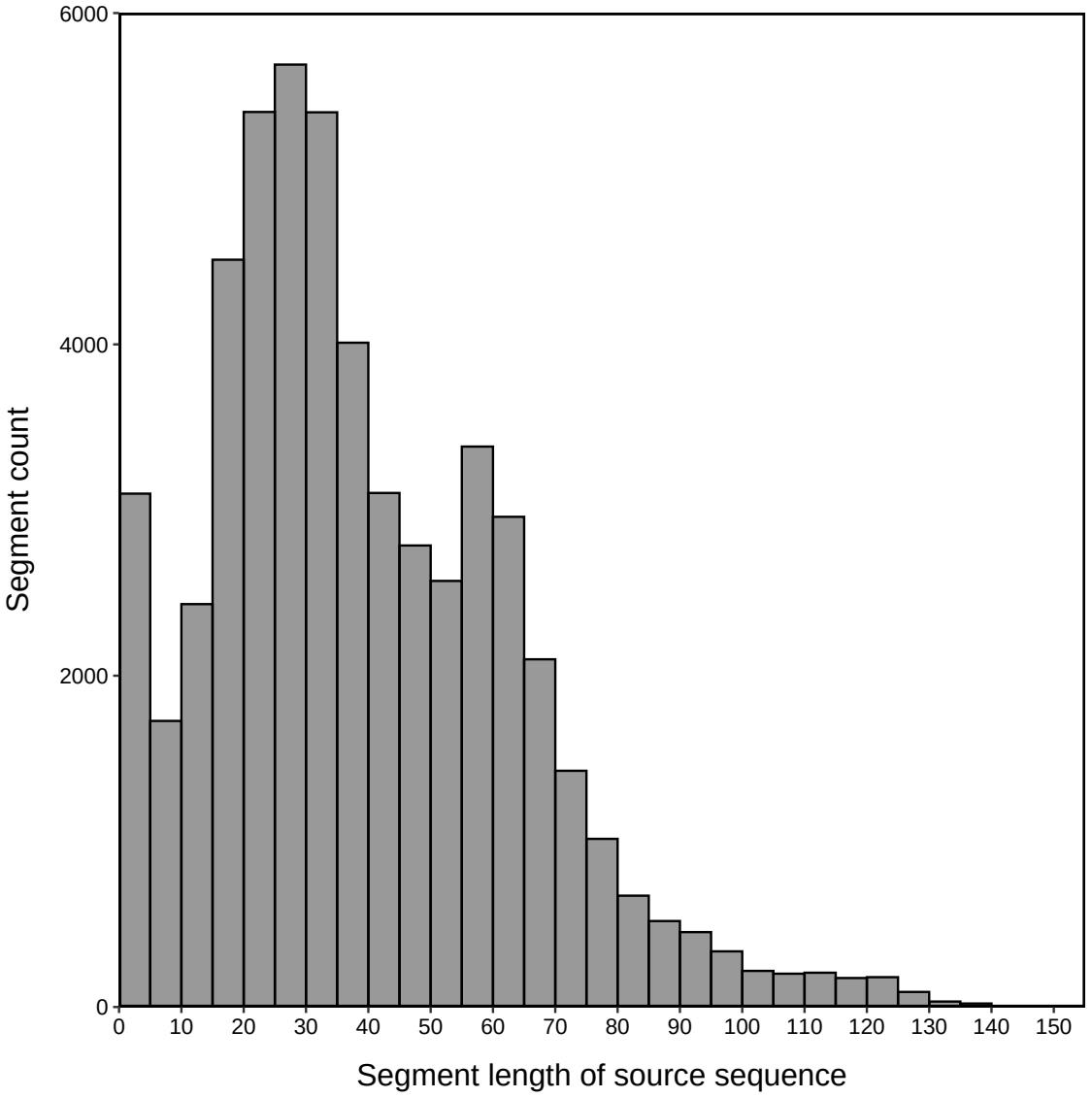


FIGURE 3.24: **Distribution of source segment length in mosaic representations of Ghana data.** There is a peak of source segments which are less than 5AA, which appear to be the artifacts of the JHMM method.

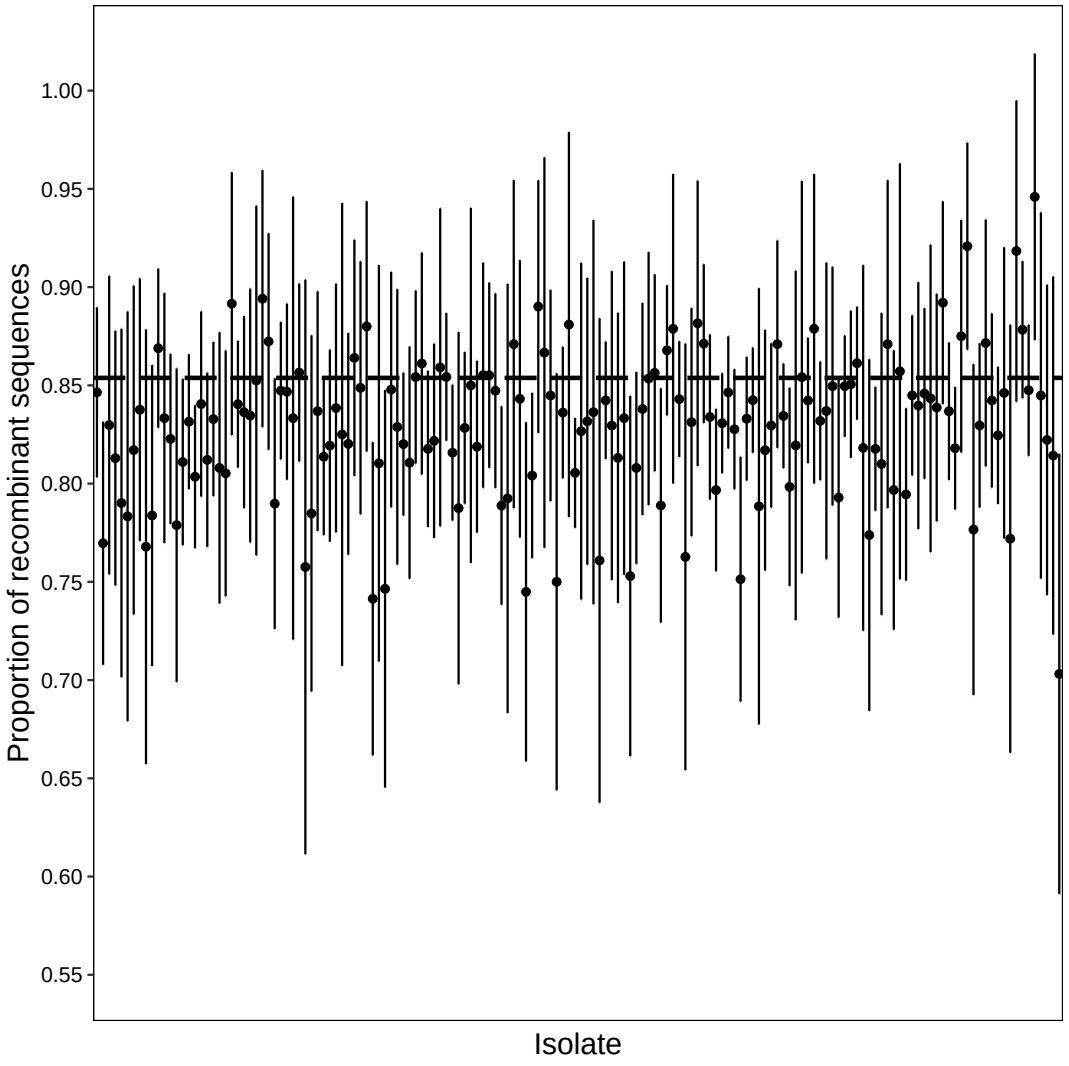


FIGURE 3.25: **Proportions (and 95% confidence intervals) of recombinants for each isolate.** The horizontal dashed line displays the overall proportion of recombinant sequences in the entire dataset. Each vertical bar represents an isolate, and we remove isolate names from x-axis for brevity.

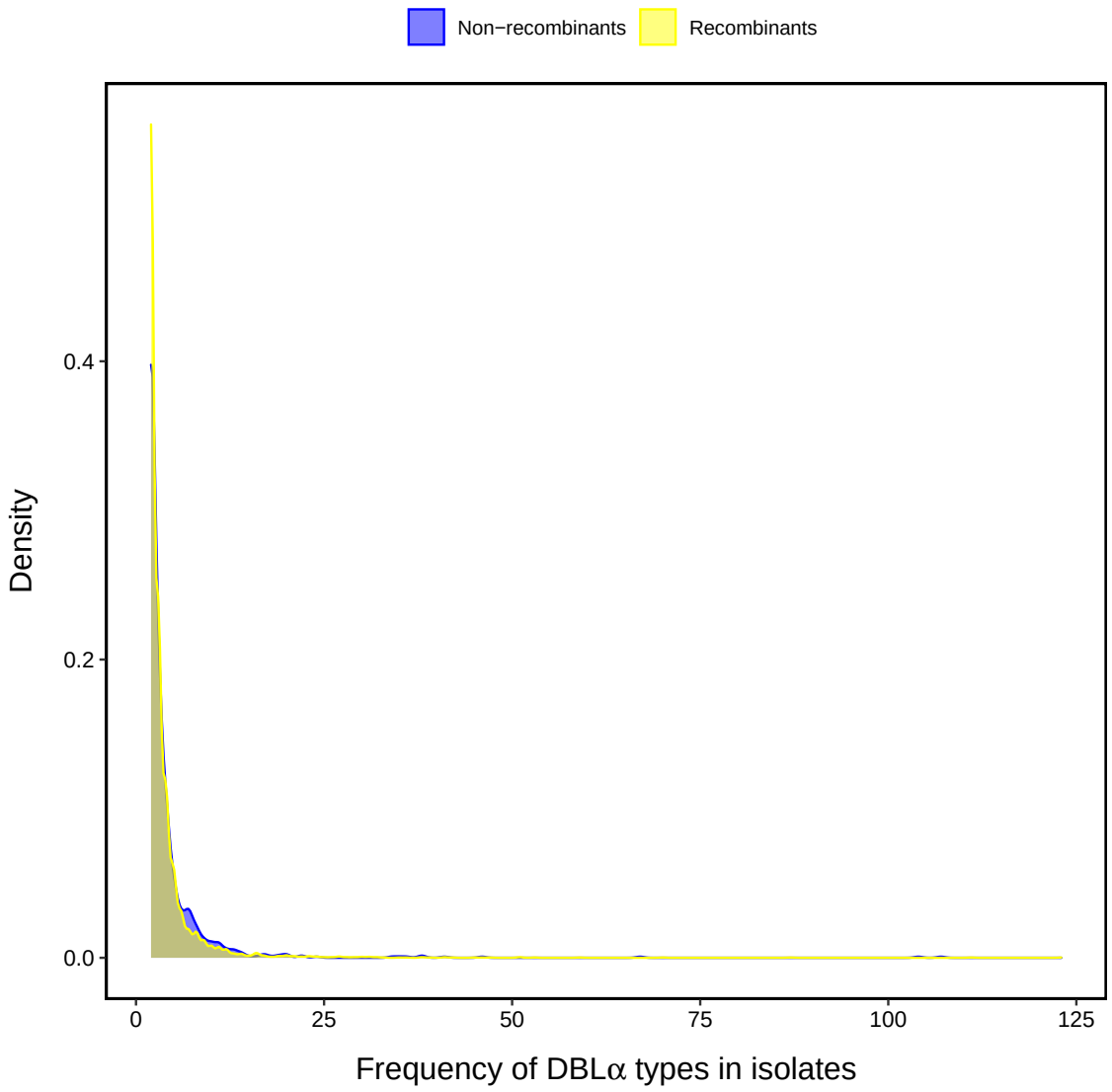


FIGURE 3.26: **Frequency of DBLα types in the isolates of the Ghana dataset.**

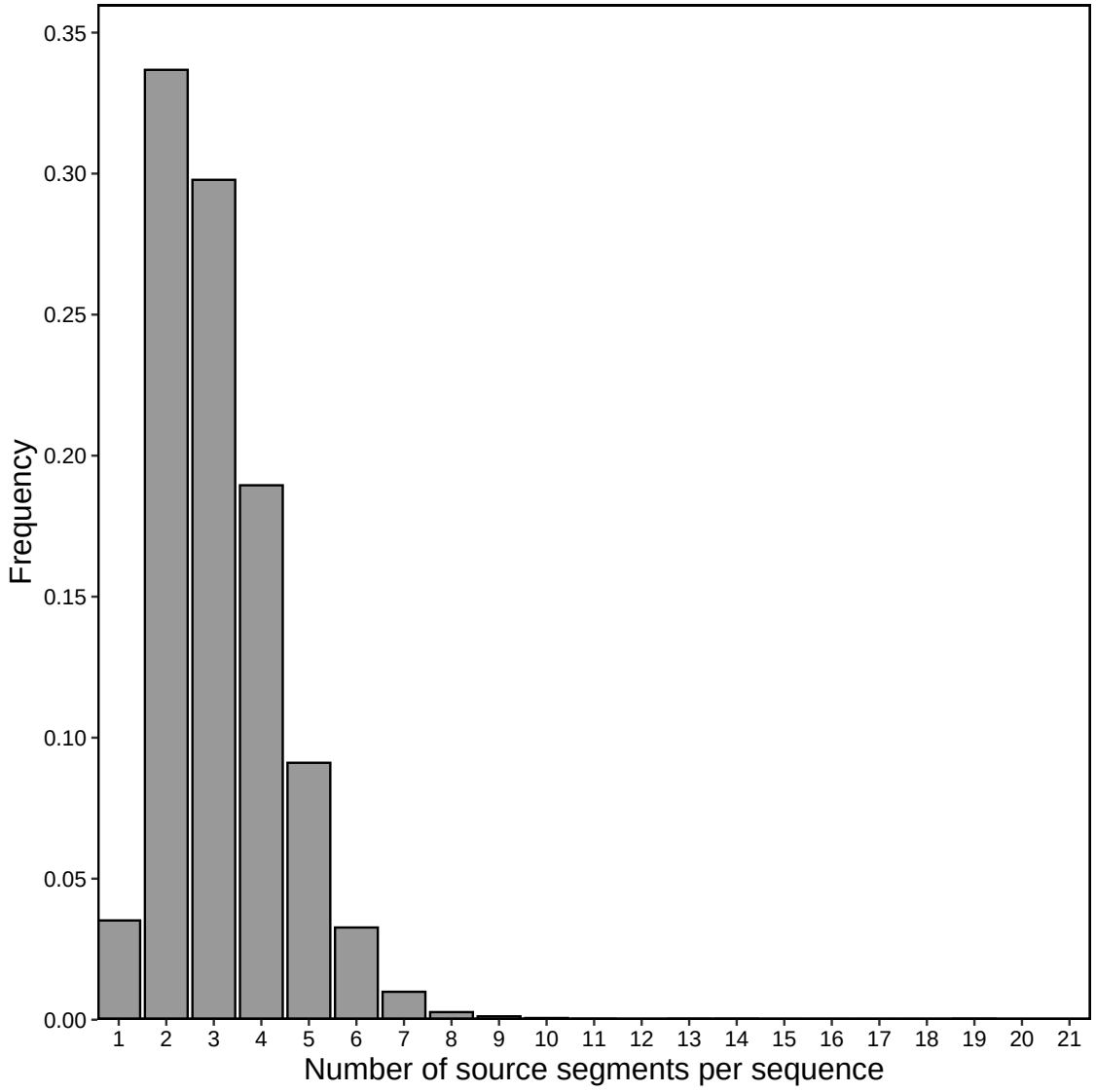


FIGURE 3.27: **Distribution of source segment count from the JHMM output in the Ghana data.**

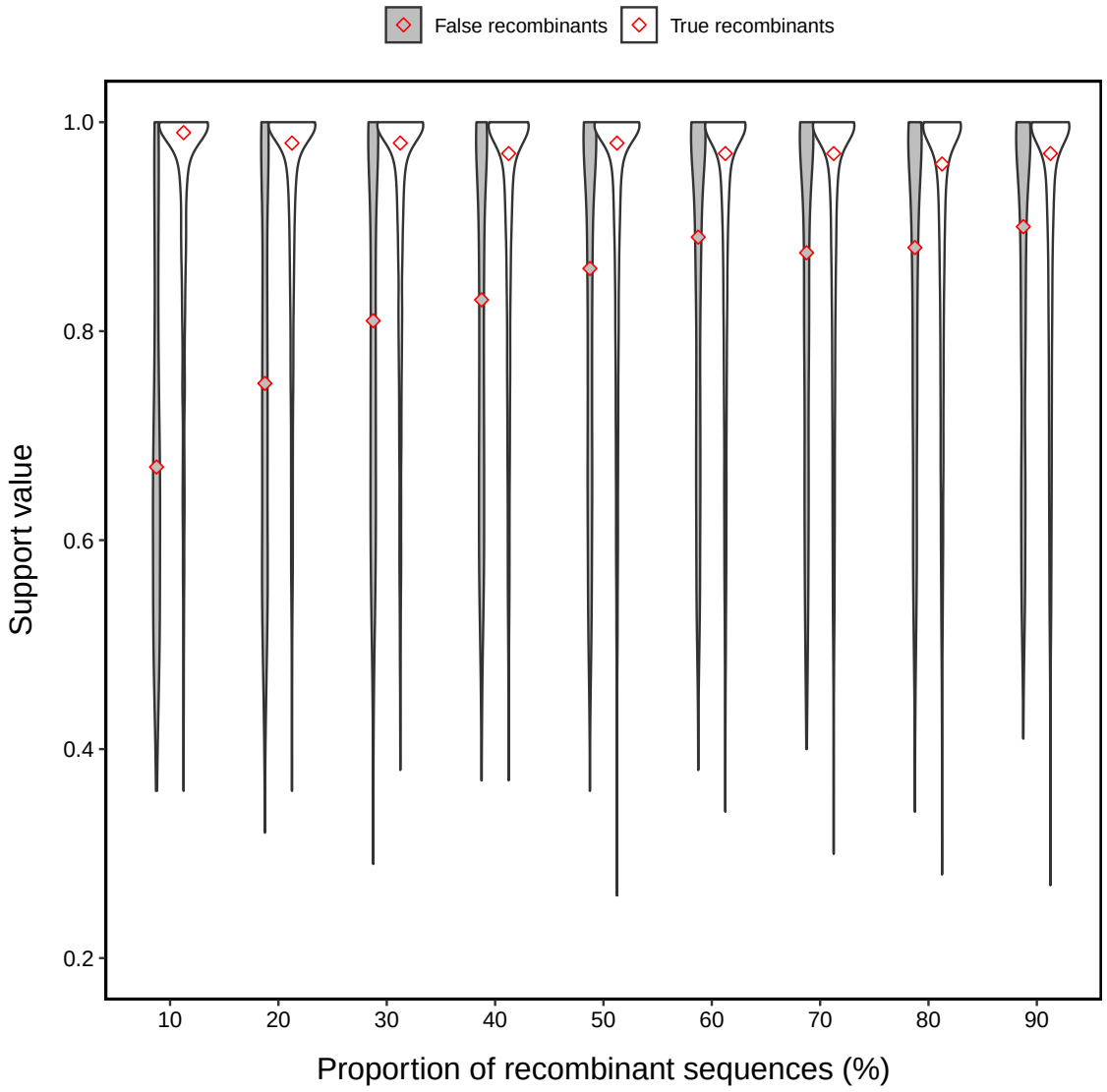


FIGURE 3.28: **Distribution of support values for varying proportions of recombinant sequences.** Red points represent the median of support values (same hereinafter).

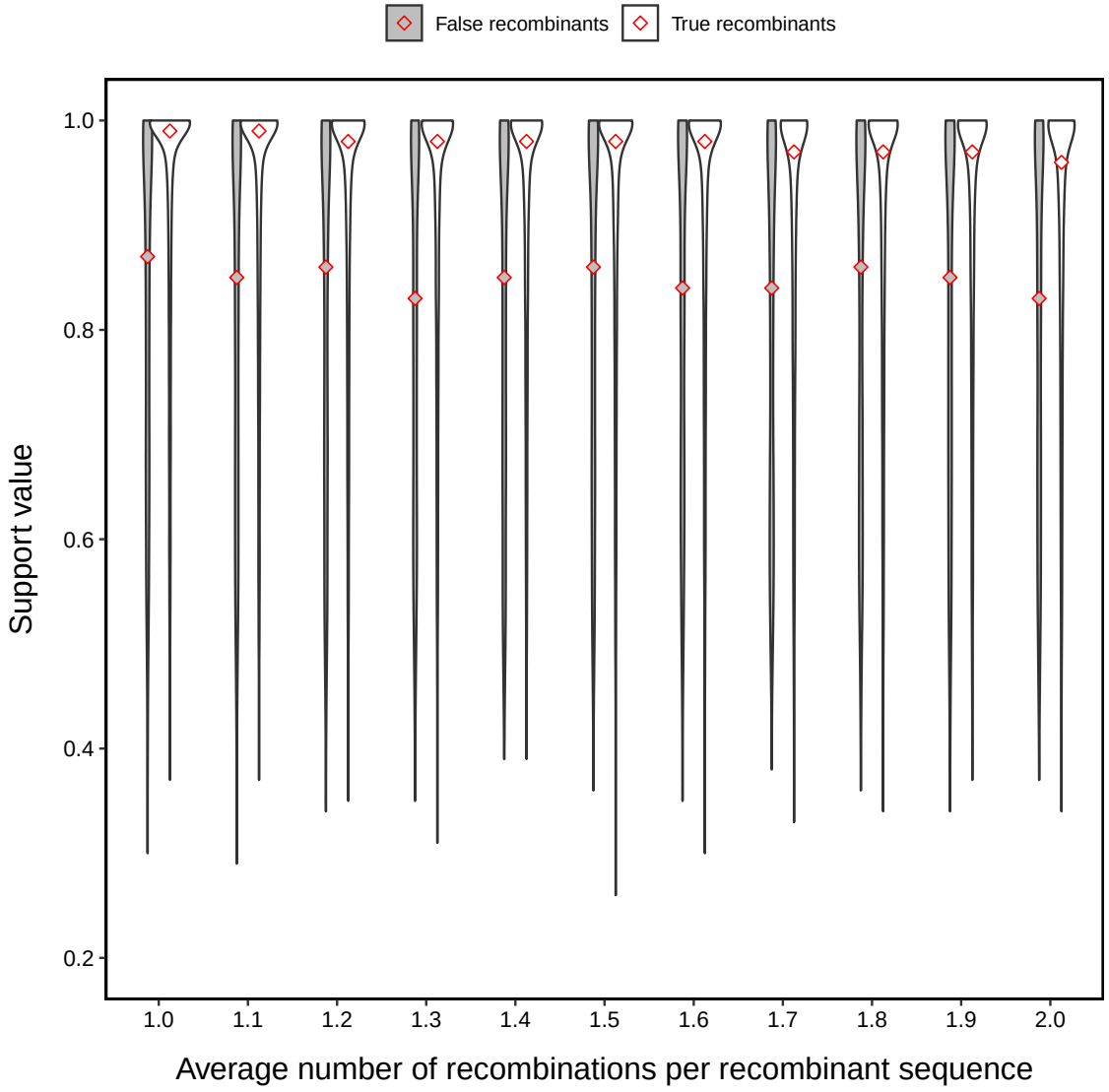


FIGURE 3.29: **Distribution of support values for varying numbers of recombinations per recombinant sequence.**

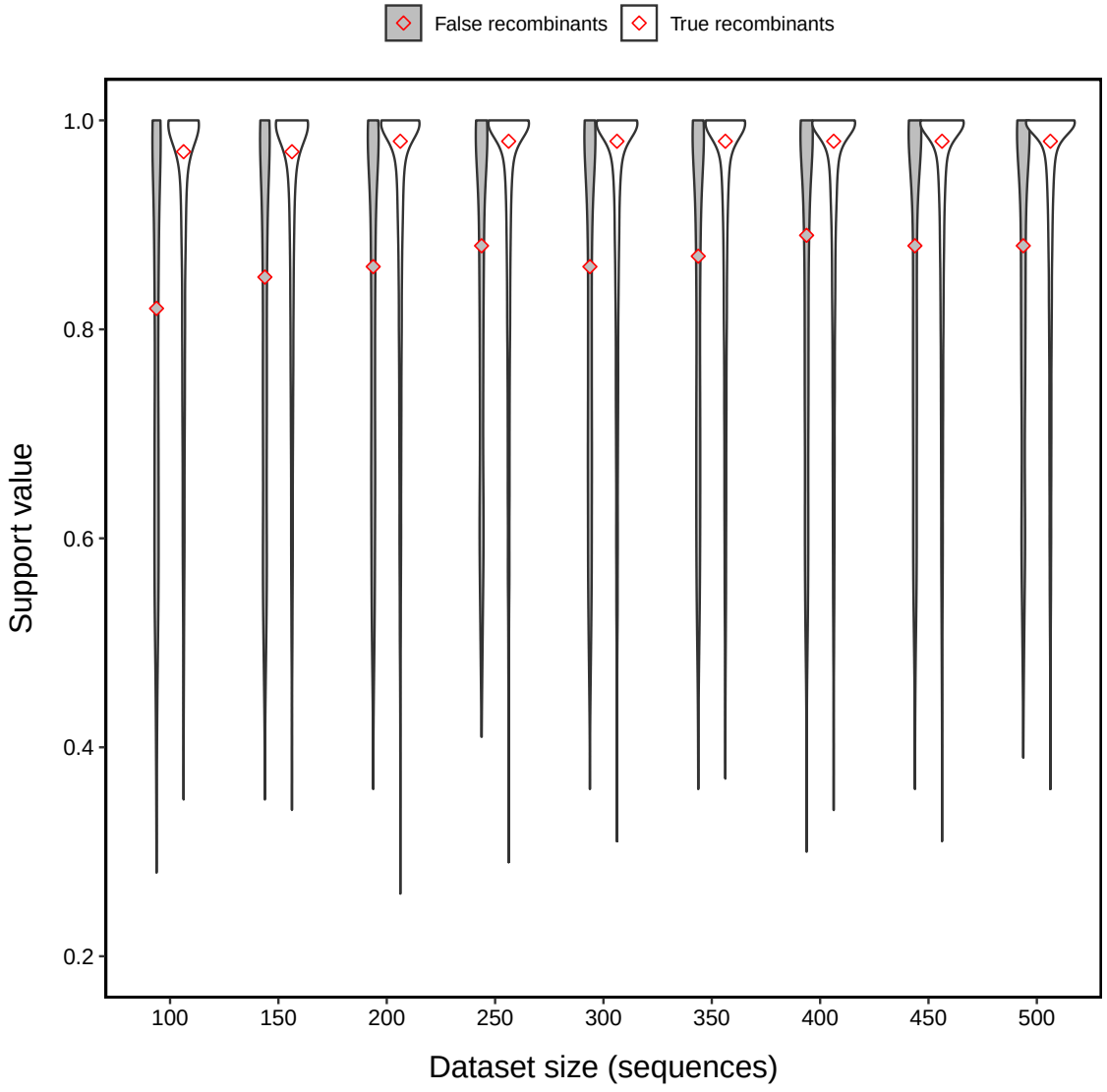


FIGURE 3.30: **Distribution of support values for varying dataset size.**

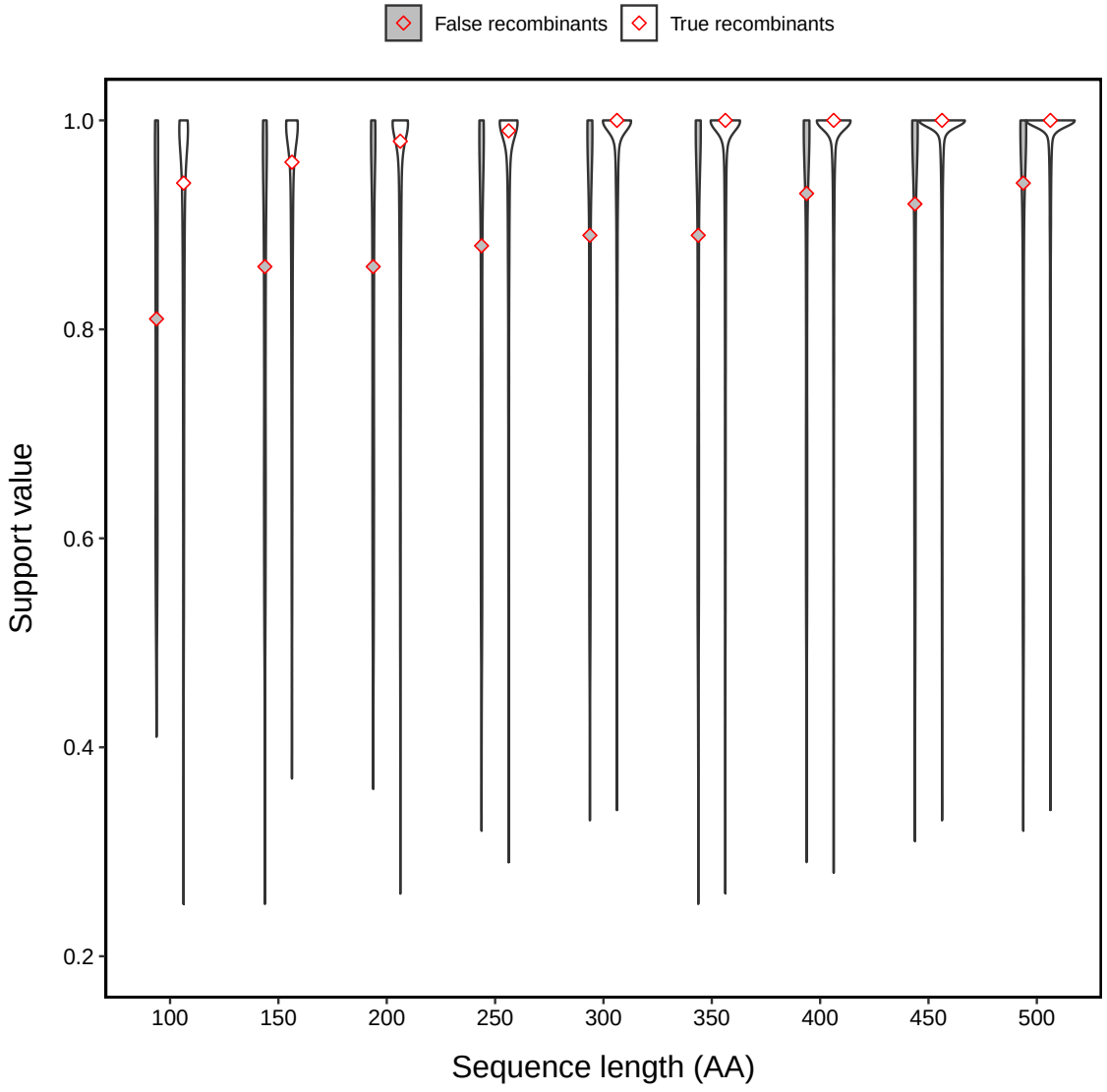


FIGURE 3.31: **Distribution of support values for varying sequence length.**

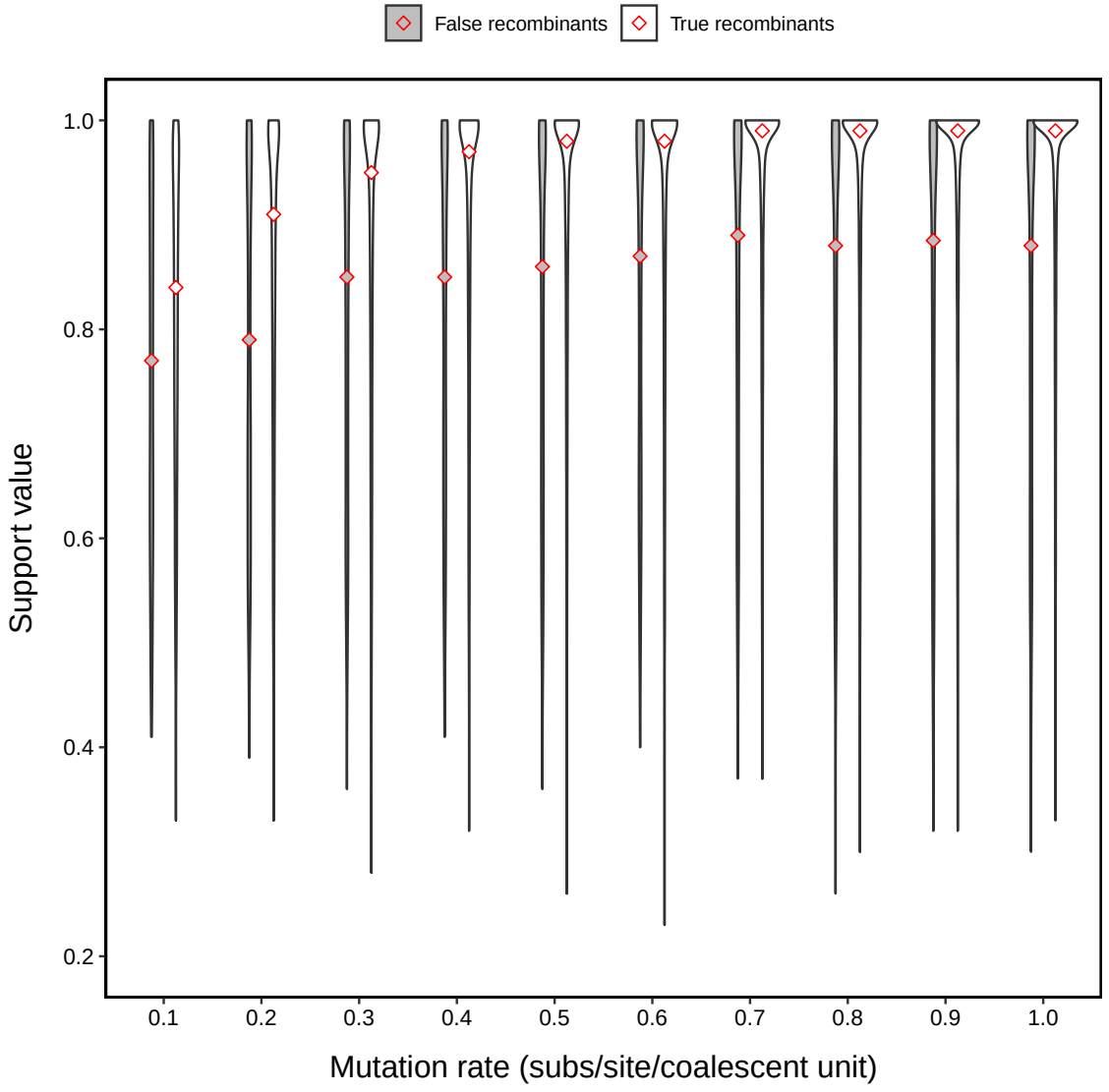


FIGURE 3.32: **Distribution of support values for varying mutation rate.**

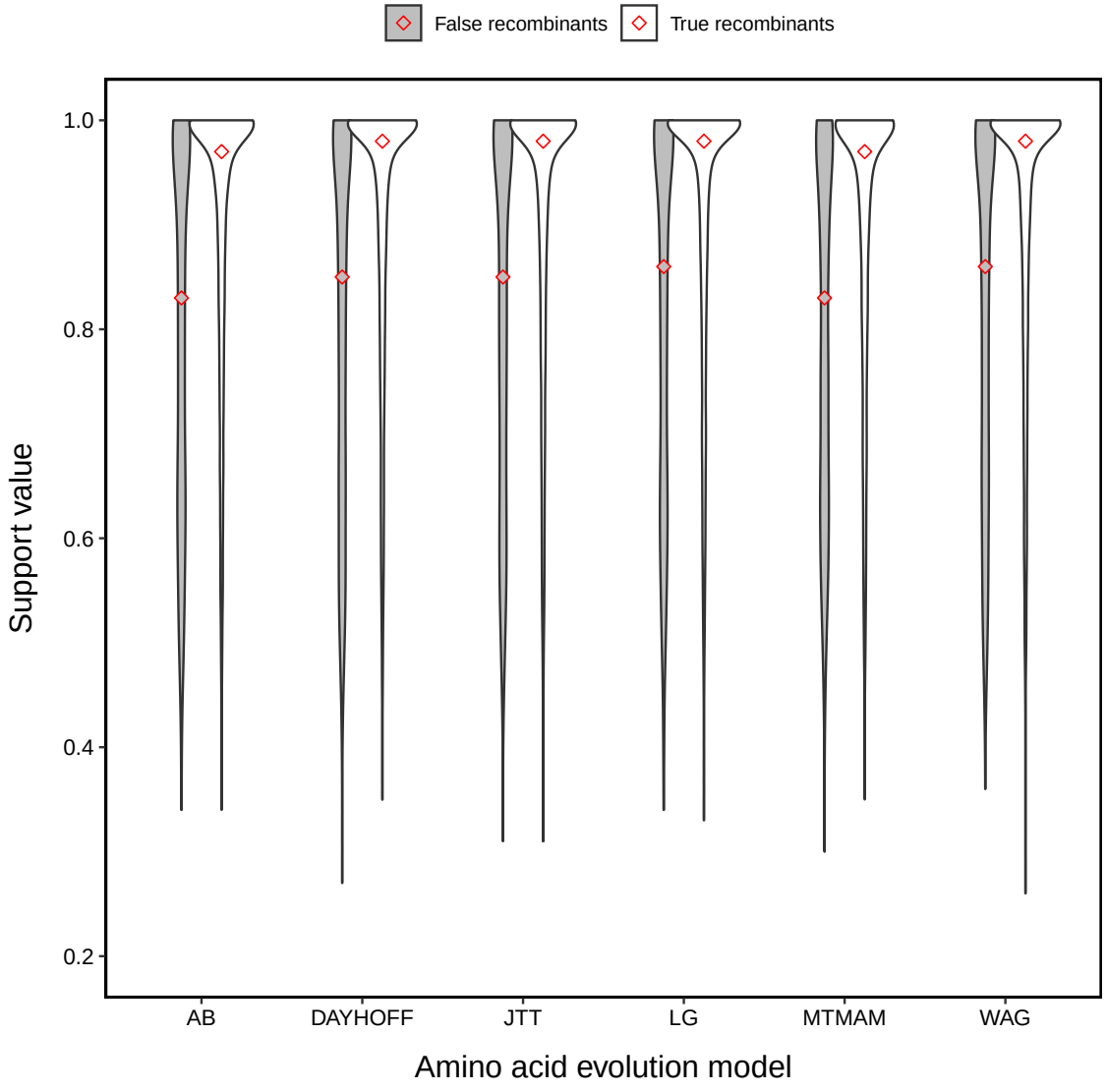


FIGURE 3.33: **Distribution of support values for different models of amino acid evolution.**

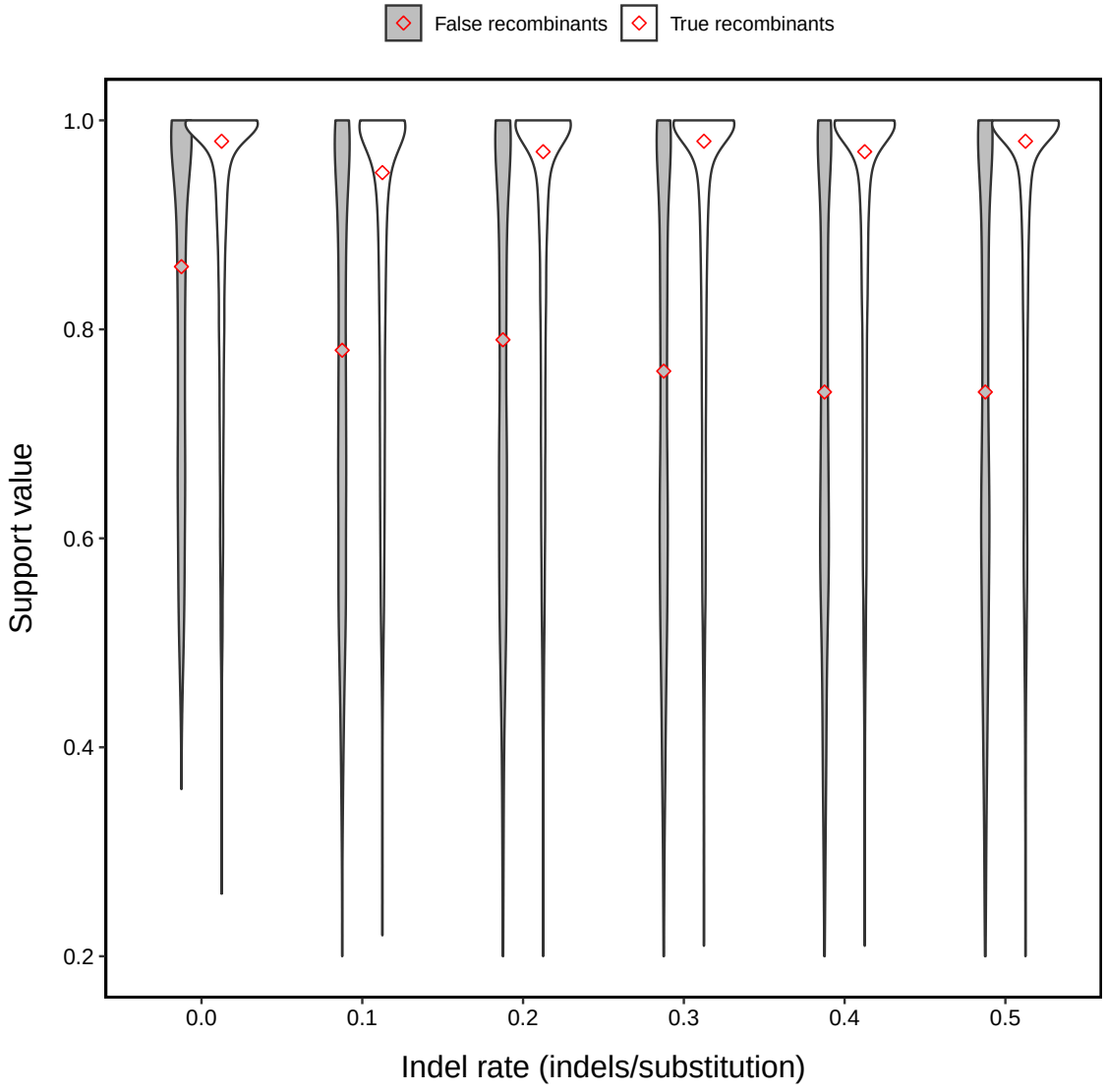


FIGURE 3.34: **Distributions of support values for varying indel rate.**

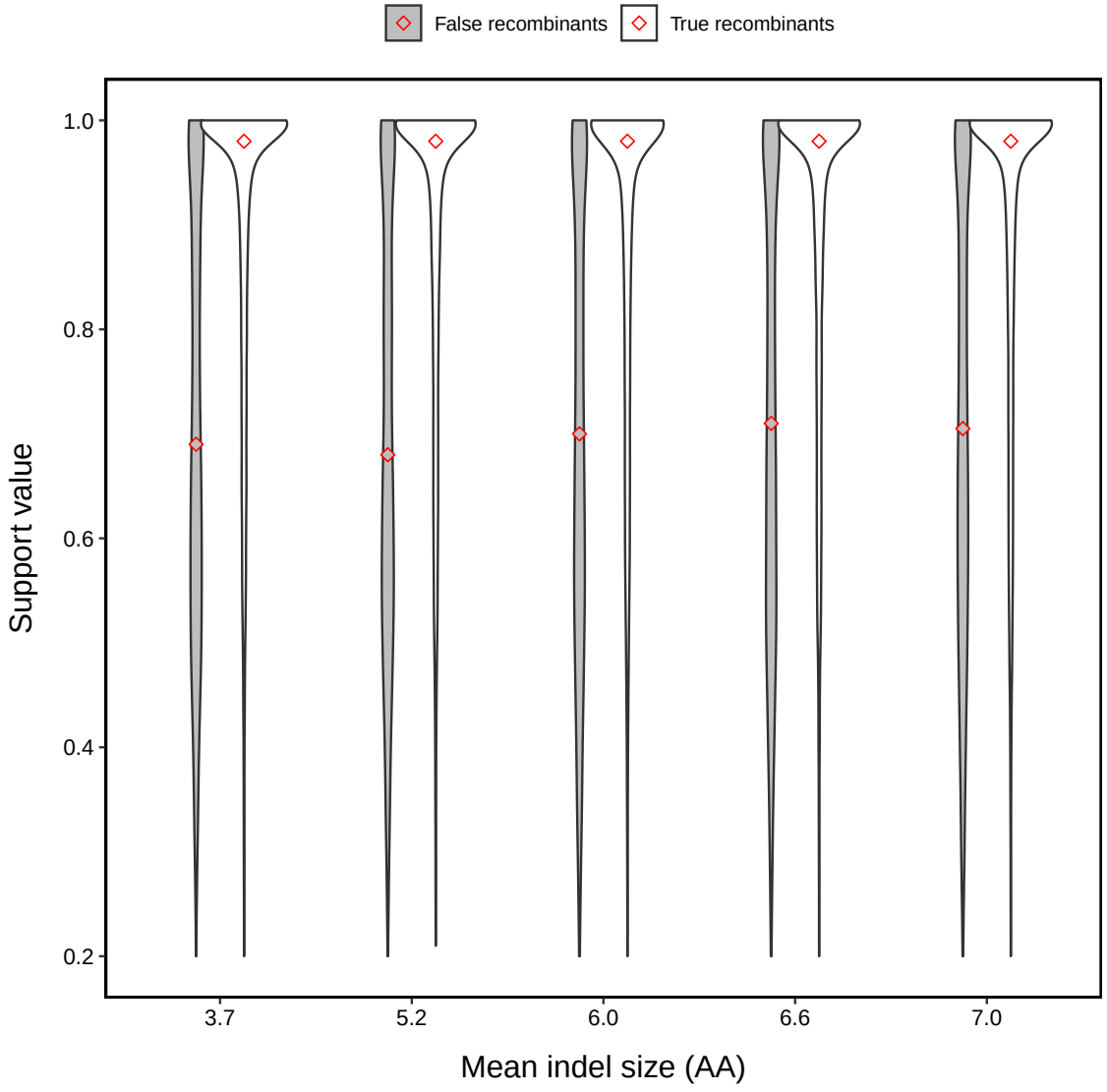


FIGURE 3.35: **Distributions of support values for varying indel size.**

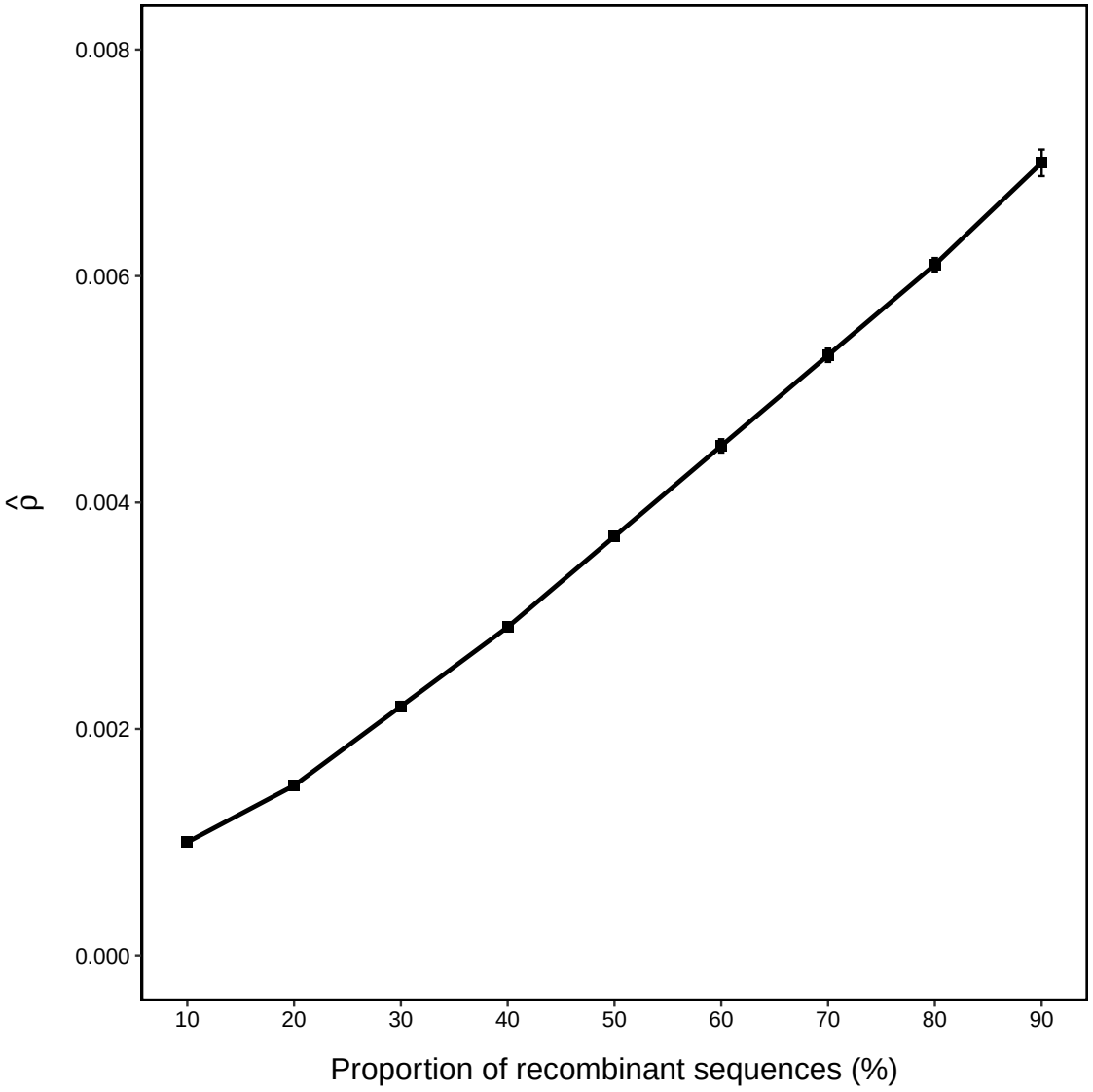


FIGURE 3.36: **Estimated ρ (and 95% CI) for varying proportions of recombinant sequences.** Some CIs are too short to be visible (similarly for Figures 3.37–3.39). $\hat{\rho}$ appears to grow linearly with the proportion of recombinant sequences, as expected.

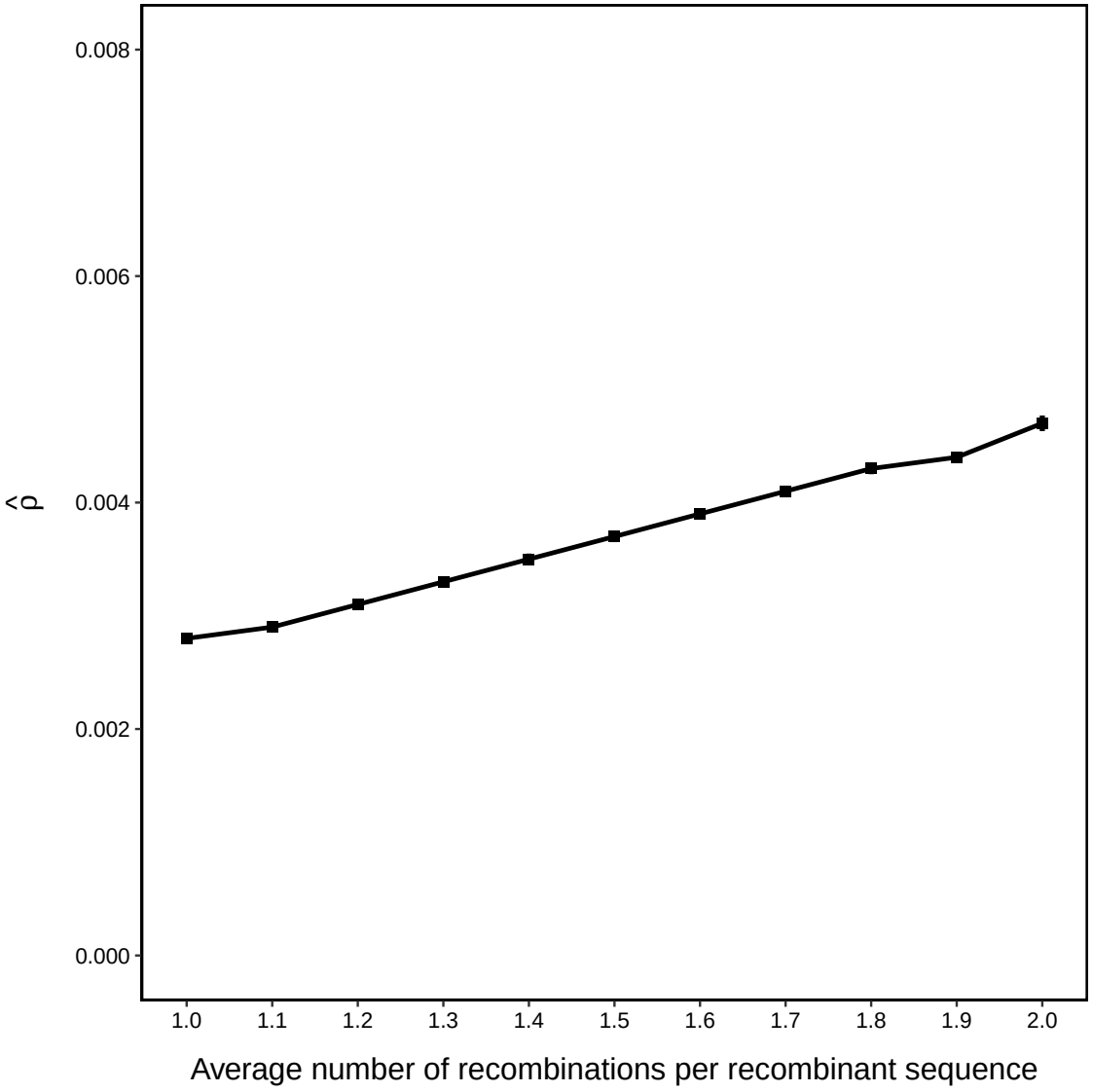


FIGURE 3.37: **Estimated ρ (and 95% CI) for varying number of recombinations per recombinant sequence.** $\hat{\rho}$ appears to grow linearly with the number of recombinants per sequence, as expected.

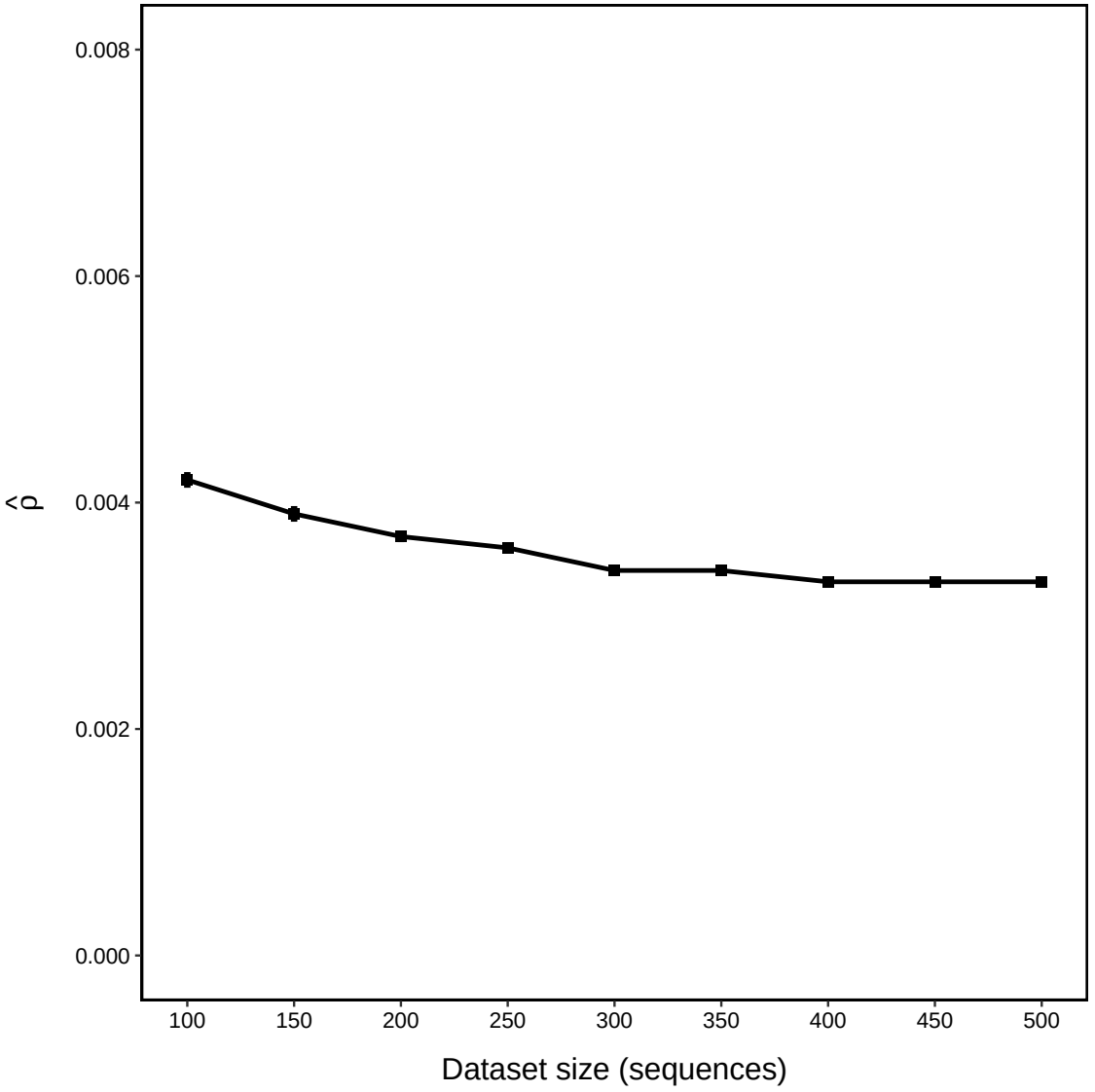


FIGURE 3.38: **Estimated ρ (and 95% CI) for varying dataset size.** $\hat{\rho}$ decreases slightly with increasing dataset size, although the recombination rate remains constant.

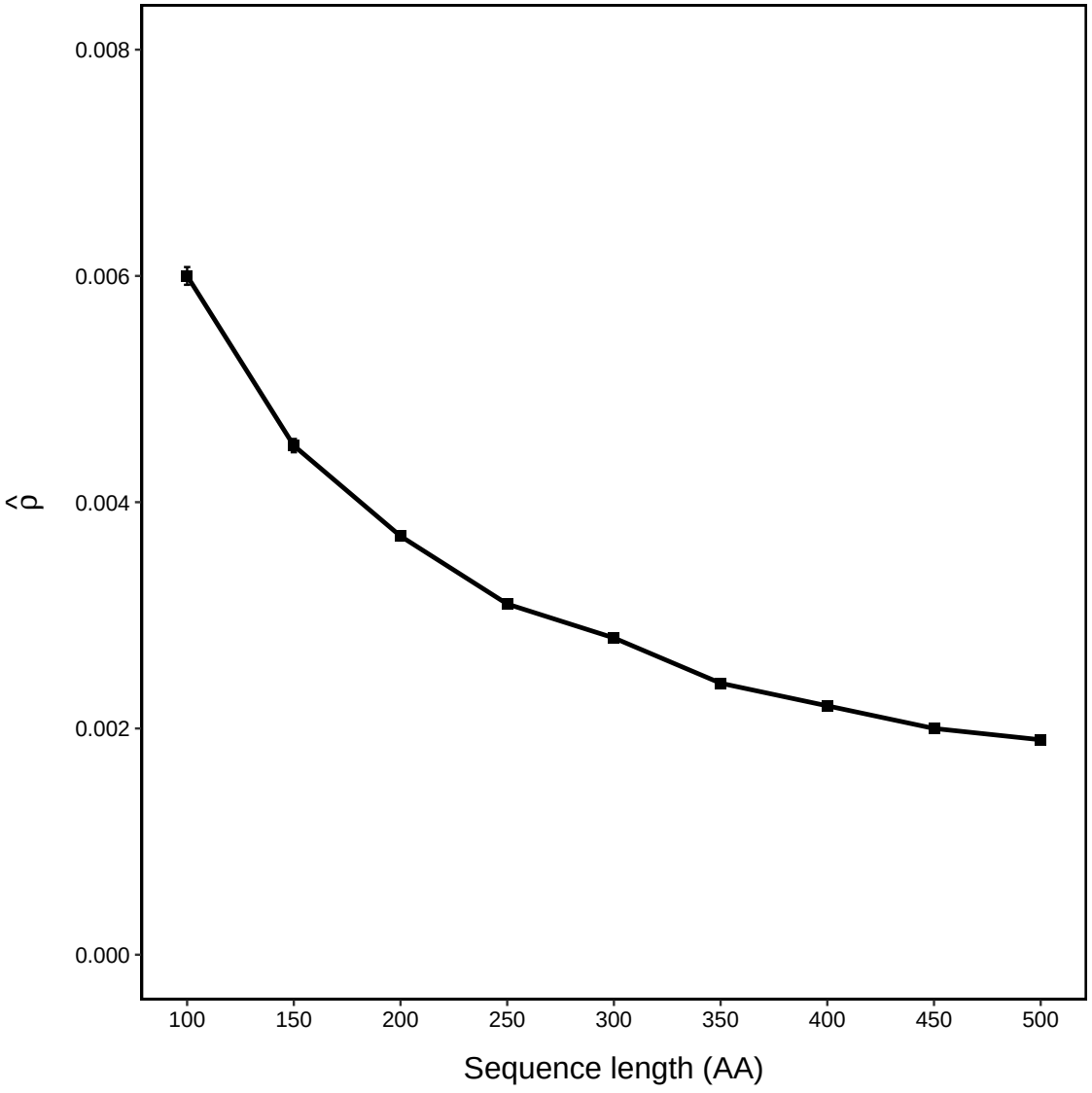


FIGURE 3.39: **Estimated ρ (and 95% CI) for varying sequence length.** $\hat{\rho}$ decreases in inverse proportion to the sequence length, as expected.

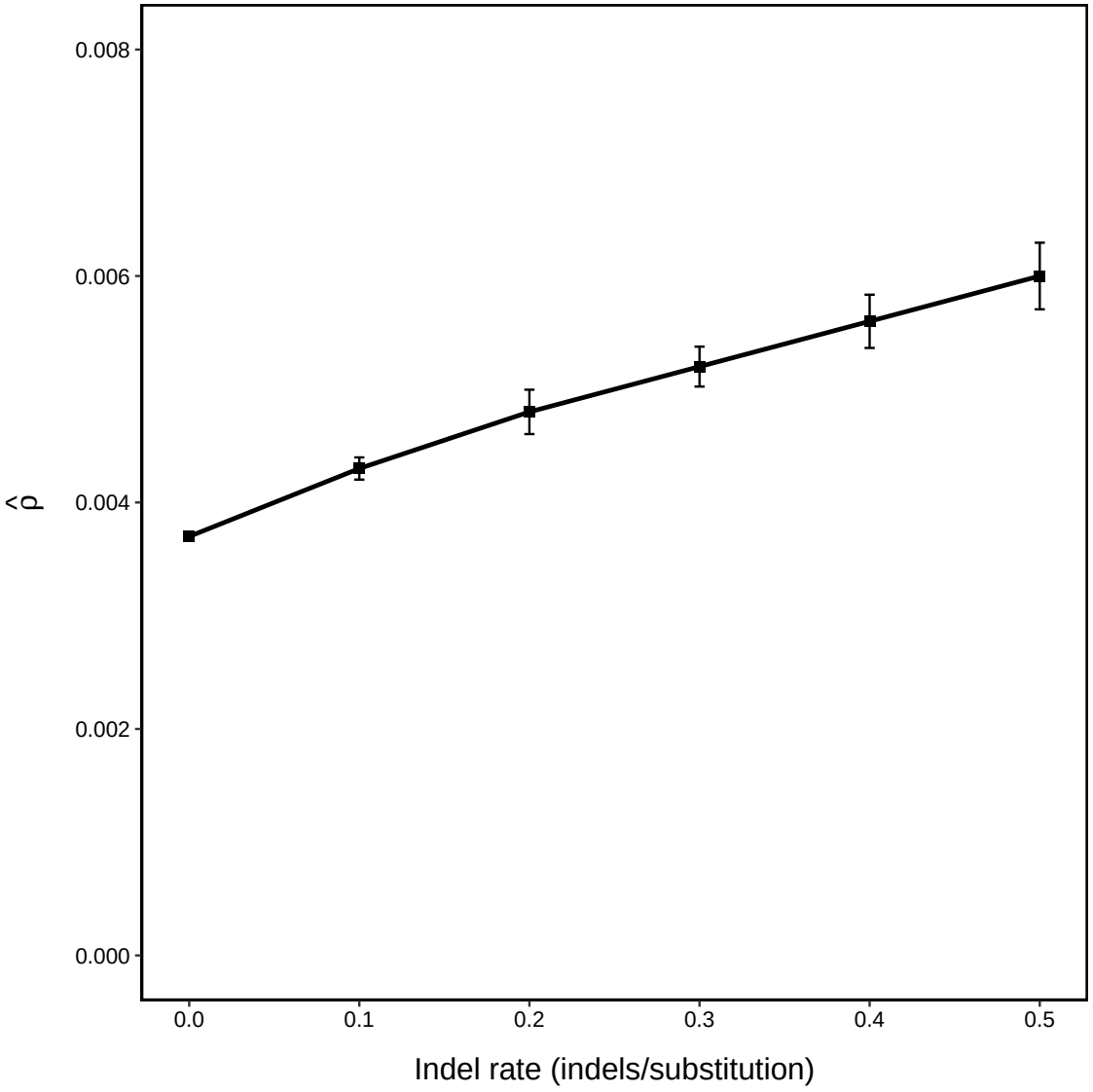


FIGURE 3.40: **Estimated ρ (and 95% CI) for varying indel rate.** There is a moderate increase in $\hat{\rho}$ as indel rate increases. This is unsurprising, as some of indel events are mistaken for recombinations, distorting the inference of the recombination rate.

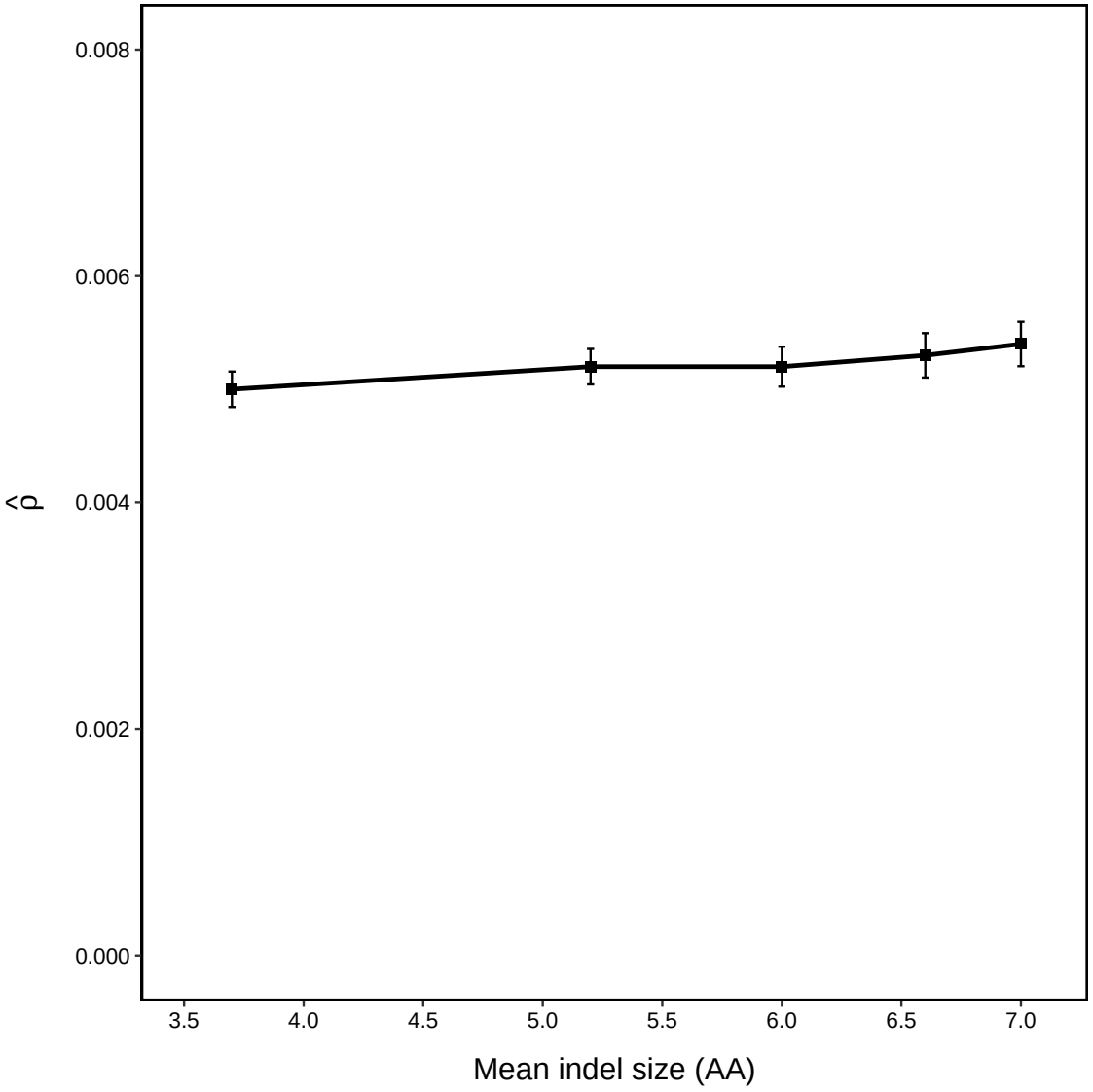


FIGURE 3.41: **Estimated ρ (and 95% CI) for varying indel size.** Indel size (but constant indel rate) does not appear to have a drastic effect on the estimated ρ.

Chapter 4

An improved JHMM for recombination detection

4.1 Introduction

Plasmodium falciparum is the deadliest parasite species, attributed to over 90% of the malaria death toll in the world [217]. The protein PfEMP1 (*Plasmodium falciparum* erythrocyte membrane protein 1) is expressed at the surface of infected red blood cells. It acts as both an antigen and adhesion protein, playing a key role in the high virulence and immune evasion of *P. falciparum* [40, 218, 219]. The PfEMP1 is encoded by the highly diverse *var* gene family. One of the primary mechanisms for maintaining *var* diversity is recombination. Therefore, identifying the recombination of *var* genes has been of major interest to biologists. Due to the complex composition of full-length *var* genes, a lot of studies [62, 69–75] have focused on the immunogenic DBLα domain of *var* genes. However, the DBLα domain itself is still highly variable. As a result, the lack of a reliable alignment precludes the use of most standard methods for recombination detection.

The jumping hidden Markov model (JHMM) proposed by Zilversmit et al. [54] is the first systematic model to detect recombination of *var* gene DBLα domains. It maps each sequence to its closest sequences from the source database, and the resulting mosaic representations enable the identification of recombination events.

In terms of the ability to identify recombinants, we propose an algorithm in the last chapter to detect recombinants if all input sequences are collected simultaneously.

We show that this algorithm requires additional steps after applying the JHMM. A notable advantage of applying the JHMM to time series data is that it allows us to determine the recombinants directly from the JHMM output. We need only designate the old sequences as sources and more recent sequences as targets. This time direction allows us to impute the recombination's parents and child. Furthermore, although it is possible that both old and recent sequences could be collected simultaneously, we think the possibility is rather low for frequently recombining DBLα tags. Specifically, on the order of 10^4 to 10^8 new mosaic *var* genes can be generated through recombination every two days in a single infected people [42].

Despite this, there are two major limitations of the JHMM for modelling DBLα recombination. Firstly, the JHMM allows recombination between any two points in a pair of sequences. Through our analysis of DBLα recombination patterns at Section A.2.2.4, recombination happens at the roughly same normalized locations in a pair of sequences. Unfortunately, the JHMM designs the recombination to occur between any two positions, and it gives equal probability to rare events as to frequent events. Thus the JHMM is likely to be less accurate than giving lower probability to rare events. Moreover, its implementation is computationally expensive when applying to the large dataset. We show the time and memory cost (Table 3.5) when applying it to over 17,000 DBLα tags. The DBLα tag is a short segment (100-500 base pairs) of DBLα domain. This application was done with the aid of cluster and parallelization techniques. In reality, there are far more sequences at each time point in a longitudinal dataset. This requires much more resources, and they are beyond modern techniques' general capacity. Consequently, detecting DBLα recombination from the large dataset is impractical.

This chapter introduces an improved JHMM that addresses the above two issues. We constrain recombination to only act between nearby positions. We also propose a sampling scheme for making the parameter estimation tractable. By doing so, our model allows recombination to occur at homologous location between sequences and is more efficient than the JHMM. We demonstrate this model's accuracy and efficiency through simulations. We also apply our model to the large longitudinal dataset of DBLα tags introduced in Section 1.2. Our results reaffirm the discoveries in the previous chapter using the pilot dataset, we also find the recombination patterns maintain with time.

4.2 Structure of the improved JHMM

The JHMM allows recombination between any two positions in a pair of sequences. This is done by switching the source sequences in the hidden states. That is to say, the JHMM permits the 'jump' between any two characters in a pair of source sequences representing recombination. However, we find that the jump distance in our dataset is usually very short (Section A.2.2.4), so we add constraints on the jump destinations. Specifically, we restrict the jump destination to a narrow interval. See the top two panels of Figure 4.1 as an illustration. By doing so, there are two main differences between the JHMM and our model, (1) the number of hidden states, (2) some (but not all) transition probabilities. We explain them in detail below.

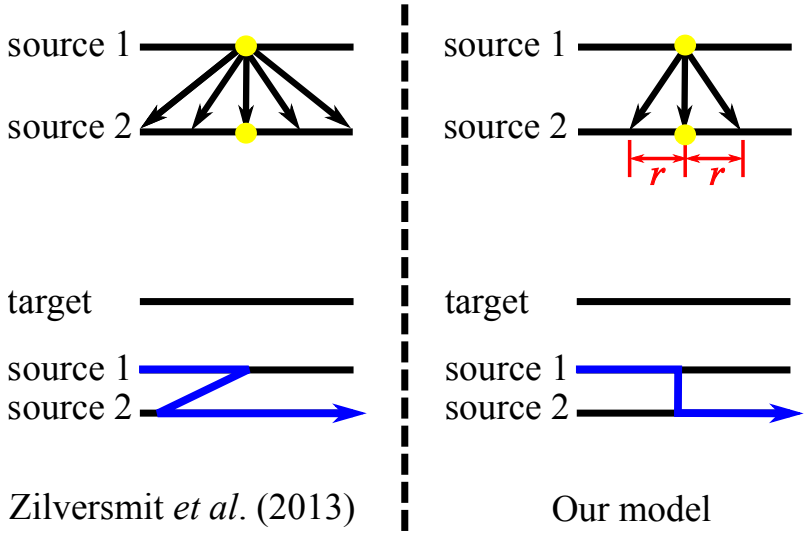


FIGURE 4.1: **Difference between the JHMM [54] and our model.** The JHMM allows jumps (represented by arrowed lines) between any two positions in a pair of source sequences representing recombination, while we restrict the jump destination to an interval colored by red. Blue line for each model illustrates a possible resulting jump path. Overall, our model avoids the scenario on the left generated by the JHMM.

For a character in the target sequence, its hidden state of the JHMM is the position of the character from the source sequence. Technically, following previous notation in Table 2.2, in a source dataset Y consisting of n sequences, all the possible hidden states at character $x(i)$ of target sequence x are s_k^j, $s \in \{M, I, D\}, k \in \{1, ..., n\}, j \in \{1, ..., \max_k l_k\}$. l_k is the length of kth source sequence y_k. For our model, s and k are the same as the JHMM, while for the range of source position j, it changes.

For each position in the target, we define its 'relative position' in the source sequence. For the ith character in the target sequence, its relative position at source sequence

y_k is $r_{k,i} = int(\frac{i \times l_k}{m})$, where m is the length of the target, int refers to the closest integer. Since the jumping distance is not strictly 0, we use an interval. As shown in the Figure 4.1, the center of this interval is the relative position (highlighted by yellow dots), before and after $r_{k,i}$ are both r characters. We name r the radius. However, not every position in this interval exists in y_k, we define set $\omega_{k,i}$ as a collection of existing positions, $\omega_{k,i} := \big[\max\{r_{k,i} - r, 1\}, \min\{r_{k,i} + r, l_k\}\big]$. In our model, $j \in \omega_{k,i}$. Apparently the range of j is smaller compared with the JHMM, therefore, our model reduces the number of hidden states.

The decrease in the number of jump destinations requires us to change the transition probabilities accordingly. For the hidden state of the first character of the target sequence, instead of making this starting point uniformly from all positions of all source sequences in the JHMM, we choose it uniformly from $\bigcup_{k=1}^{n} \omega_{k,1}$. Suppose $y_k(j)$ for $x(i)$ is selected with the hidden state M_k^j, I_k^j or D_k^j, and $j \in \omega_{k,i}$ (we keep the alignment not "too far" even without a jump), the hidden state of next step and corresponding transition probabilities are shown in Table 4.1. In this table, $M_{k'}^p$ ($I_{k'}^p$, $D_{k'}^p$ respectively) refers to a hidden state representing recombination, $k' \in \{1, ..., n\} \setminus \{k\}$, $p \in \omega_{k',i+1}$. Our model has the same parameters as the JHMM, gap opening probability (δ), gap extension probability (ϵ) and the probability of recombination (ρ). π_M, π_I and τ are the same default values with the JHMM.

TABLE 4.1: **Transition probabilities from a given hidden state (rows) at a current time step to all possible hidden states (columns) at next time step.** T refers to termination.

	M_k^{j+1}	I_k^j	D_k^{j+1}	$M_{k'}^p$	$I_{k'}^p$	$D_{k'}^p$	T
M_k^j	1-2δ-ρ-τ	δ	δ	$\rho\pi_M/\kappa$	$\rho\pi_I/\kappa$	0	τ
I_k^j	1-ϵ-ρ-τ	ϵ	0	$\rho\pi_M/\kappa$	$\rho\pi_I/\kappa$	0	τ
D_k^j	1-ϵ	0	ϵ	0	0	0	0

Compared with the transition probabilities of the JHMM (Table 2.3), our model has several modifications. Firstly, the JHMM allows the jumping to the same source sequence, which is not biologically realistic. Our model does not permit this by making $k' \neq k$. Secondly, we denote $\kappa = |\bigcup_{s=1,s\neq k}^{n} \omega_{s,i+1}|$ as the overall number of positions our model could jump to. While in the JHMM, the number of jump positions is $L = \sum_{s=1}^{n} l_s$. Therefore, although it seems only a number changes (L to κ) from Table 2.3 to Table 4.1 (highlighted by grey), the number of columns representing

the recombination in our model is much less than the JHMM, the summation of each row is still 1. Overall, the transition probabilities involving recombination are modified. Other transition probabilities that do not involve recombination remain the same.

For training our model, we follow a similar approach to that used in [54]. We first estimate δ and ϵ with the Baum-Welch algorithm, and then estimate ρ over the interval [0,0.1] using the forward algorithm with $\hat{\delta}$ and $\hat{\epsilon}$. Since the composite likelihood with ρ is a unimodal function (Figure A.2), here we adopt the golden section search method to speed up the estimation of ρ. Importantly, under our new HMM framework, we need to revise related algorithms, i.e., forward and backward algorithms, respectively. Moreover, after estimating parameters, the computation of Viterbi paths requires a modified Viterbi algorithm. We show these three algorithms' derivations in Supplementary Section 4.6.1.

4.3 Simulation results

We conducted simulations to evaluate the effectiveness and efficiency of our model. We followed a similar simulation pipeline described in Section 3.3.1:

- Step 1, we simulated an arbitrary tree under the coalescent (without recombination) using `msprime` [199].
- Step 2, we then simulated amino acid sequences from this tree under the WAG model [212]. We simulated both equal-length (without indels) and unequal-length (with indels) sequences as input. For simulating equal-length sequences, we used `Pyvolve` [200], and for unequal-length sequences, we used `INDELible` [201].
- Step 3, we next generated recombinants by randomly selecting the parental sequences with uniformly distributed breakpoints. The parents were then removed. For equal-length sequences, the breakpoint position is identical among the parental sequences. For unequal-length sequences, we firstly aligned the parental sequences (with MAFFT [197]) and chose the breakpoints randomly from the alignment.

There are a wide variety of parameters that might affect the model performance. Here we considered the proportion of recombinant sequences, average number of recombinations per recombinant sequence, number of sequences in the data, sequence length and mutation rate. The values of each parameter were shown in Table 4.2. For tractable simulations, we varied only one parameter in turn and kept the other parameters fixed at default values.

An important advantage of the JHMM and our model is that sequences of unequal length are accepted as input. To explore how variations in sequence length affect our model, we varied the indel rate while maintaining a constant mean indel size, we also varied the mean indel size with a constant indel rate. A greater indel rate/size results in a greater diversity of sequence lengths. The values of indel rate and size are in Table 4.3. In summary, there are 47 parameter combinations in our simulation. Specifically, for equal-length sequences, there are 38 simulation settings in total; for unequal-length sequences, there are 9 simulation settings. For each parameter combination, we simulated 100 datasets and applied our model and the JHMM.

TABLE 4.2: **General simulation parameters (no indels).** We vary each parameter in turn while holding the others fixed at the default values (in bold).

Parameter	Values
① Proportion of recombinant sequences (%)	10, 20, 30, 40, **50**, 60, 70, 80, 90
② Average number of recombinations per recombinant sequence	1.0, 1.1, 1.2, 1.3, 1.4, **1.5**, 1.6, 1.7, 1.8, 1.9, 2.0
③ Dataset size (sequences)	20, 40, 60, 80, **100**, 120
④ Sequence length (AA)	75, **100**, 125, 150, 175, 200
⑤ Mutation rate (substitutions/site/coalescent unit)	0.1, 0.2, 0.3, 0.4, **0.5**, 0.6, 0.7, 0.8, 0.9, 1.0

TABLE 4.3: **Indel simulation parameters (default values in bold).** Insertions and deletions are simulated at the same rate, with lengths according to a negative binomial distribution with variance 10 and specified mean.

Parameter	Values
⑥ Indel rate (expected number of indels/substitution)	0.1, 0.2, **0.3**, 0.4, 0.5
⑦ Mean indel size (AA)	3.7, 5.2, **6.0**, 6.6, 7.0

Choosing a suitable radius r is important for our model. The mosaic representations from the empirical data could calibrate it. We calculated the difference between actual jump positions and the expected positions using the mosaic representations from the Ghana and global datasets [62]. Supplementary Figures 4.21 and 4.22 show the resulting distributions of these distance values. We see that half of the distance in each dataset is less than 1 amino acid, and 90% of the distance is less than 5 amino acids. The proportion of distance values captured by radii 10, 15 and 20 is shown in Supplementary Table 4.10. In the following simulation, we chose radii 10, 15, and 20 for our model.

We trained the JHMM and our model with various radii for each dataset in turn, followed by the generation of mosaic representations using the Viterbi algorithm. In the following sections, we firstly examined the accuracy of improved JHMM, after that, we compared the estimated parameters, mosaic representations and execution time between these two models.

4.3.1 Accuracy of the improved JHMM

We aim to test the accuracy of our improved JHMM. Specifically, we aim to see whether the improved JHMM can correctly identify the recombinant sequences and non-recombinant sequences (in the context of sequences from different time points), and the level of accuracy in identifying breakpoints and finding parental sequences.

To do so, we used the model to search target sequences against the parental sequences of recombinants. Here we focused on two default parameter settings with multiple replicates, one dataset consisting of equal-length sequences and another for unequal-length sequences (see Table 4.2 and 4.3). One can test the accuracy using all 47 parameter combinations, but we envision the similar conclusions would hold.

We used sensitivity and specificity to measure the accuracy of our model in identifying recombinants and non-recombinants. Sensitivity (or specificity) is the proportion of recombinants (or non-recombinants) that are correctly identified. We recorded the breakpoints of each recombinant and aimed to compute the difference between the identified breakpoint positions and actual positions. When the inferred number of breakpoints is different from the truth (for instance, 4 vs. 2), it is unclear how to unambiguously match the true and inferred breakpoints, so we only focused on

the recombinants with a correctly inferred number of breakpoints. The percentage of such recombinants is shown in Supplementary Table 4.11.

For each correctly identified recombinant sequence, we defined its accuracy in finding parental sequences as the proportion of amino acids from the parental sequences that were correctly inferred. From the example mosaic representation (Figure 2.3), the target sequence (same as its parental sequence) is `AGTCKDIMMMF`, while the inferred parental sequence is `AGTTKDMMMKF`. We excluded the deletion (`-` vs. `K`) and treated the mutation (`C` vs. `T`) and insertion (`I` vs. `-`) as incorrect inferences. As a result, the accuracy for finding parents in this example is 9/11.

Overall, the improved JHMM is accurate across all measures (see Figure 4.2). We found the mean sensitivity, specificity and accuracy values in finding parental sequences are all over 80%, regardless if the lengths of input sequences are equal or unequal. The inferred breakpoint locations are also very similar to the true locations. We noticed that the specificity and accuracy in finding parental sequences are slightly lower in the unequal-length scenario than the equal-length scenario. This is not surprising as sequences with indels increase the difficulty in finding parental sequences, as a consequence, more breakpoints are inferred. Moreover, a shorter radius will also worsen such inference (see panel (d) of Figure 4.2 and Supplementary Figure 4.14). Unfortunately, this also increases the chance of introducing mutations and indels into the mosaic representation, thus influencing the accuracy of finding parents for recombinants.

In summary, the improved JHMM has almost the same accuracy as the JHMM, when the radius is large. The improved model even enjoys higher sensitivity in finding recombinant sequences than the JHMM. However, if a short radius is chosen, it might impact the accuracy of finding parental sequences and introduce more false positives (misidentified non-recombinants), especially when input sequences are of unequal length.

When applying the JHMM of Zilversmit et al. [54] to a dataset of DBLα tags from a pilot study in Ghana, we found that there is a large number of length-one segments in the generated mosaic representations (See Figure A.7). We think this is the artifact of the JHMM, as this model simply picks out an amino acid that is not truly homologous. Our improved JHMM has been designed to overcome this issue.

To validate this, we firstly checked whether this phenomenon appears in our simulations or not. We got the segment length distribution from unequal-length and

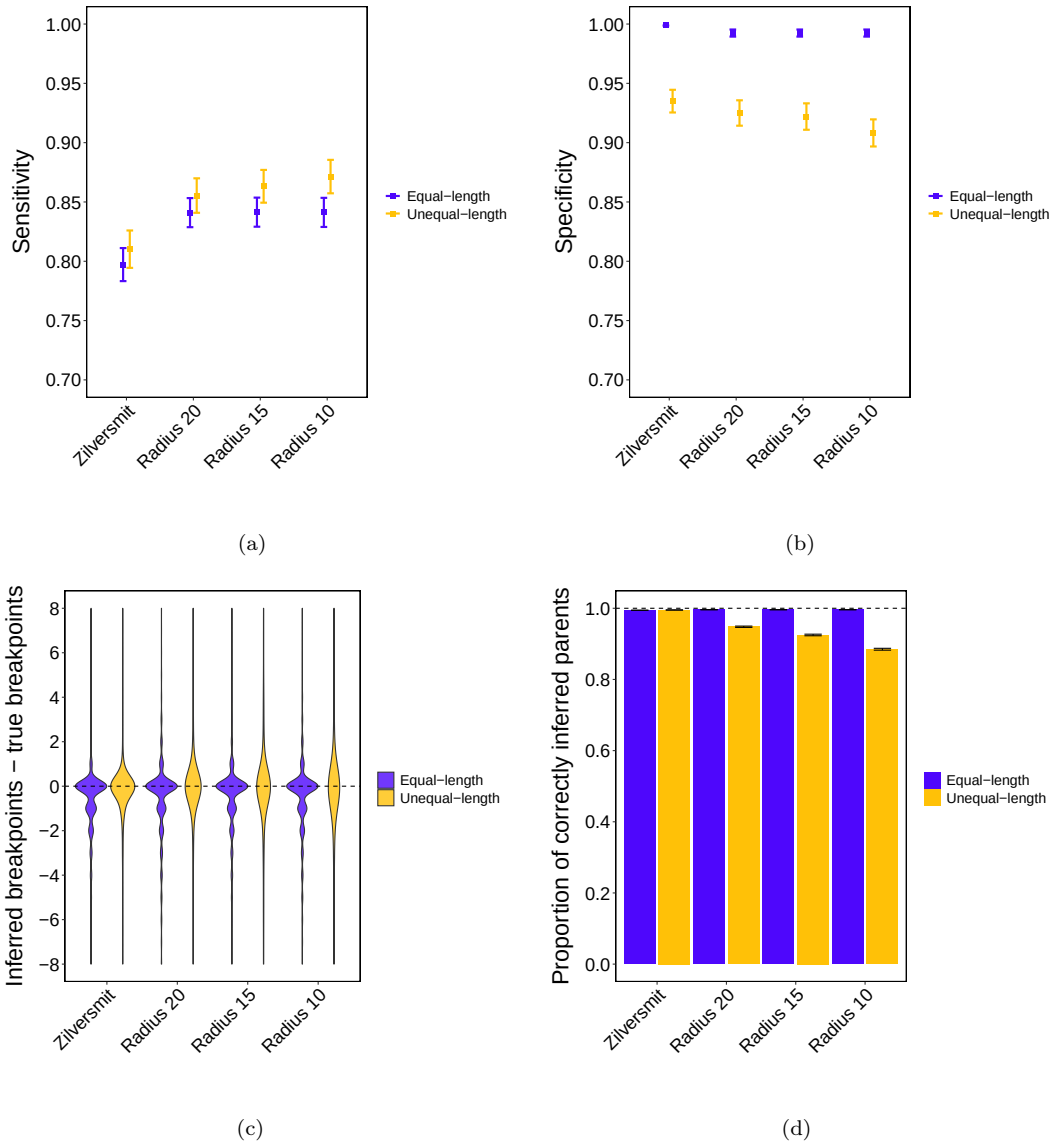


FIGURE 4.2: **Accuracy of the improved JHMM.** Mean (with 95% confidence intervals) of (a) sensitivity, (b) specificity using the JHMM and improved JHMM with various radii. (c) shows the distribution of bias in inferring breakpoints. (d) shows the accuracy in finding parental sequences, each error bar represents the mean accuracy +/− standard error.

equal-length scenario separately (shown in Supplementary Figures 4.23 and 4.24). From these distributions, we found that there is not a tendency to find many artificial (extremely short) segments. We further applied our improved JHMM to the Ghana dataset of DBLα tags, and found that the excessive length-one segments disappear (see Figure 4.3). Therefore, although the surfeit of extremely short segments is not present in the simulations, our improved JHMM helps to reduce artificial segments from the JHMM in the real data.

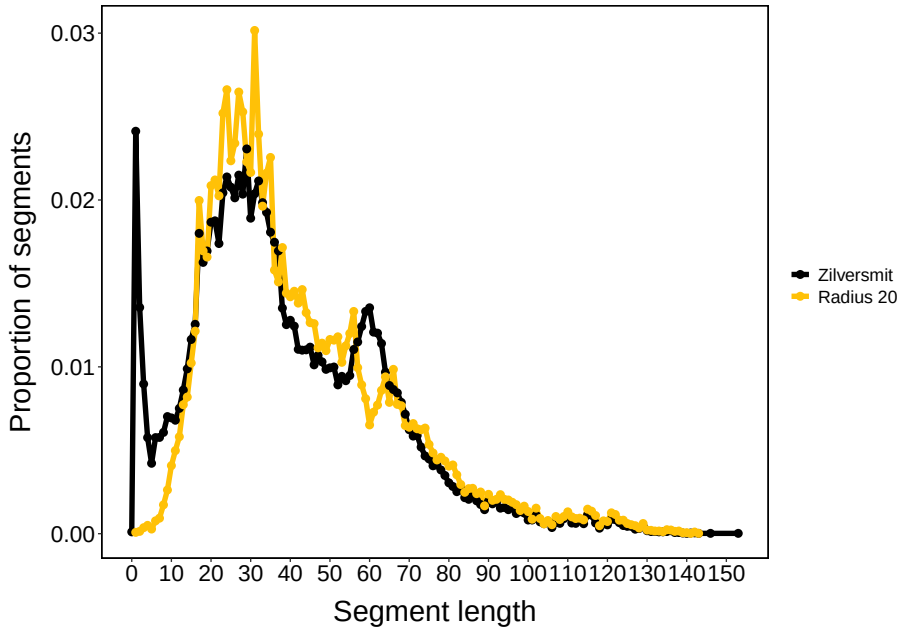


FIGURE 4.3: **Source segment length from the mosaic representations using the dataset of Ghana pilot study with the JHMM and improved JHMM.** For computational purposes, we used the estimated parameters from the JHMM when implementing the improved JHMM with radius 20.

4.3.2 Performance comparison between the JHMM and improved JHMM

4.3.2.1 Estimated parameters

There are three parameters (gap opening probability δ, gap extension probability ϵ and the probability of recombination ρ) to estimate for training the models. As the radius increases, the jump destinations are broader. Therefore our model becomes more similar to the JHMM. We expect that our model's estimated parameters would approach those using Zilversmit's method.

The estimated parameters using unequal-length sequences are summarised in Figures 4.4 and 4.5, and Supplementary Figures 4.25–4.29 show the estimation results using equal-length sequences. From these figures, we see that as the radius of improved JHMM increases, the estimated values approach those of the JHMM for all model parameters and simulation parameters. This is in line with our expectation.

The trends of estimated parameters per simulation parameter are also reasonable. In particular, we find that the $\hat{\delta}$ increases (with stable $\hat{\epsilon}$) with indel rate and $\hat{\epsilon}$ increases (with stable $\hat{\delta}$) with indel size. They are expected as the gap opening

probability represents the indel rate and gap extension probability represents the indel size. $\hat{\rho}$ grows with the proportion of recombinant sequences, average number of recombinations per recombinant sequence and mutation rate. A possible explanation is that higher number of substitutions makes the model difficult for finding the breakpoints, and thus increases the number of recombinations.

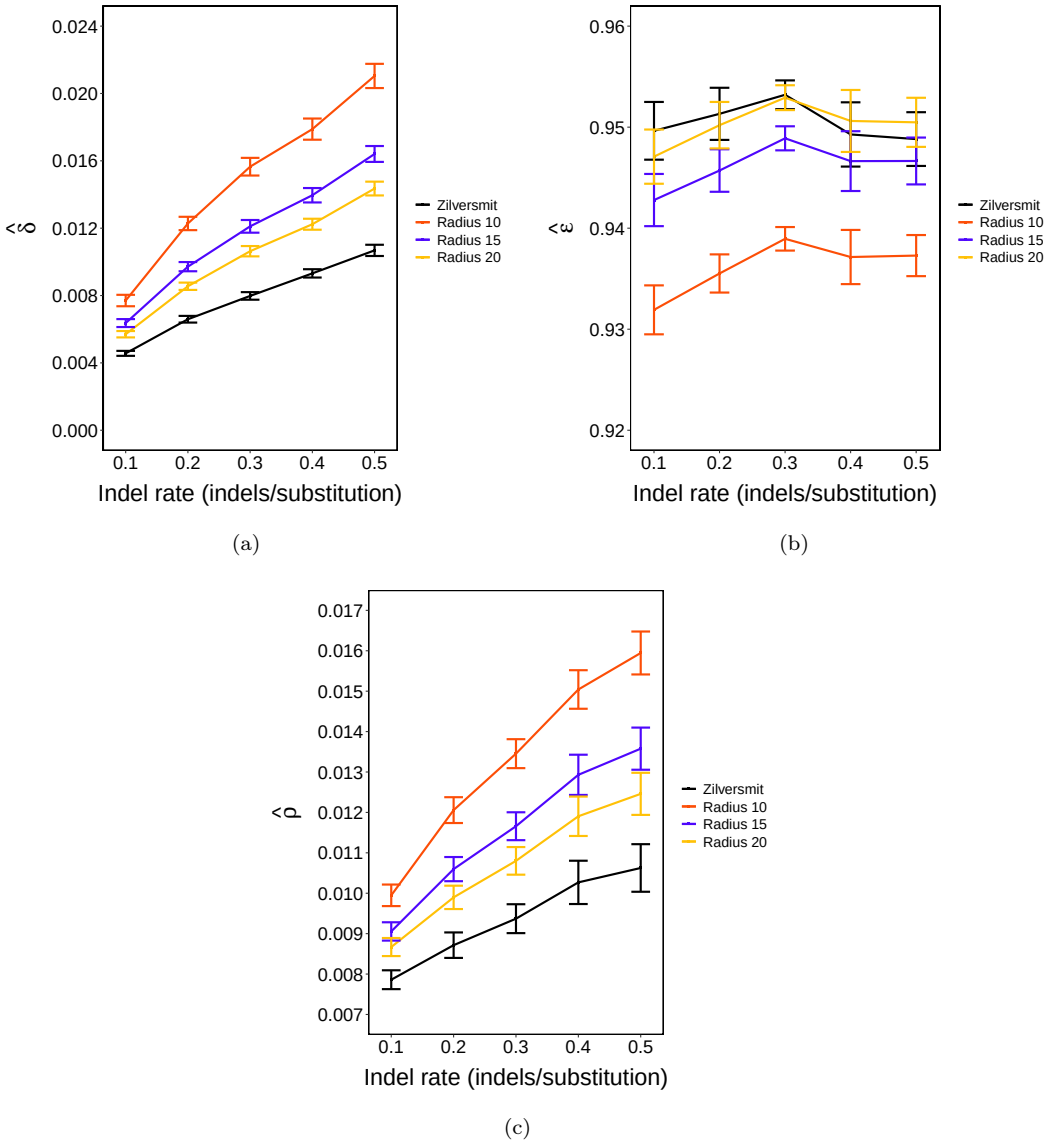


FIGURE 4.4: **Mean (with 95% confidence intervals) of each estimated parameter for varying indel rate using the JHMM and improved JHMM.**

4.3.2.2 Mosaic representations

Four Viterbi paths (mosaic representations) exist for each sequence: three from our model with three different radii and one from the JHMM. We compared the Viterbi

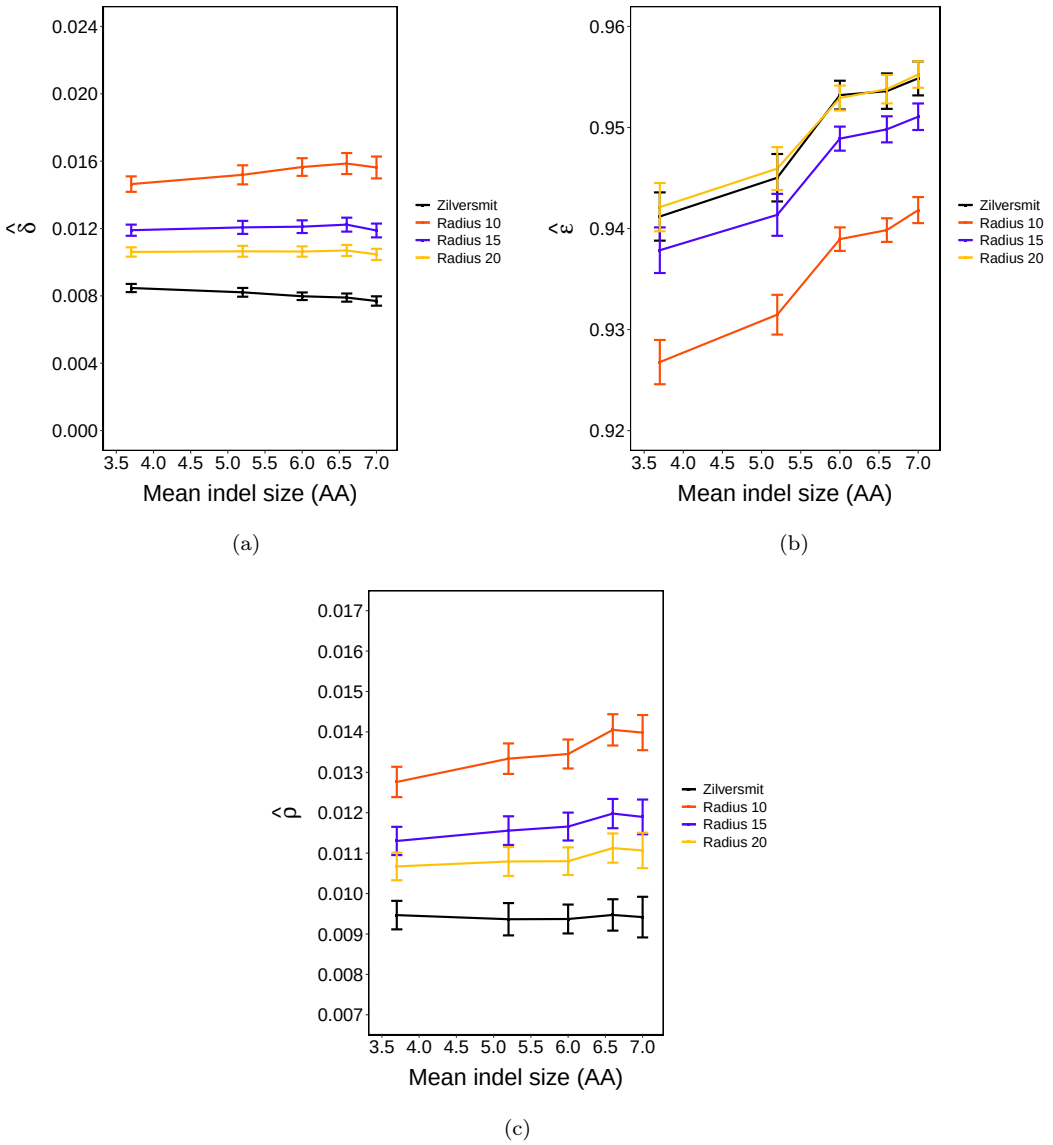


FIGURE 4.5: **Mean (with 95% confidence intervals) of each estimated parameter for varying indel size using the JHMM and improved JHMM.**

paths between our model and the JHMM for each radius. For a detailed comparison, we divided the resulting paths from the JHMM into five categories.

1. Ambiguous. The source segment occurs more than once in its full sequence. The primary reason is that the segment is too short. Therefore it is not able to be located uniquely.

2. Not allowed in our model. We can obtain the source segment's true position, but it jumps further than our maximum allowable distance using our model.

3. Dissimilar. The paths do not have the same source sequence composition or the same number of source sequences as our model.

4. Similar. The paths have the same source sequence composition and the same number of source sequences as our model. They only differ at the breakpoints, indel locations or source identities.

5. Same. Sequences' paths are the same as our model.

For the case of equal-length sequences, we showed the frequency of these five categories when varying the proportions of recombinant sequences in Figure 4.6. See Supplementary Figures 4.30–4.33 for the results of other parameters. We observed that the proportion of sequences with the same and similar path between the JHMM and our model decreases with increasing proportion of recombinant sequences and sequence length. But even so, this proportion maintains over 80% across all simulation parameters. We also found the frequency profile (each panel of the figure) remains among various radii per simulation parameter, this is expected as the equal-length sequences could be treated as aligned input and breakpoints are much easier to be inferred. Importantly, this case is favourable for the original JHMM, since we restrict horizontal jumps, but there are no horizontal jumps to restrict here.

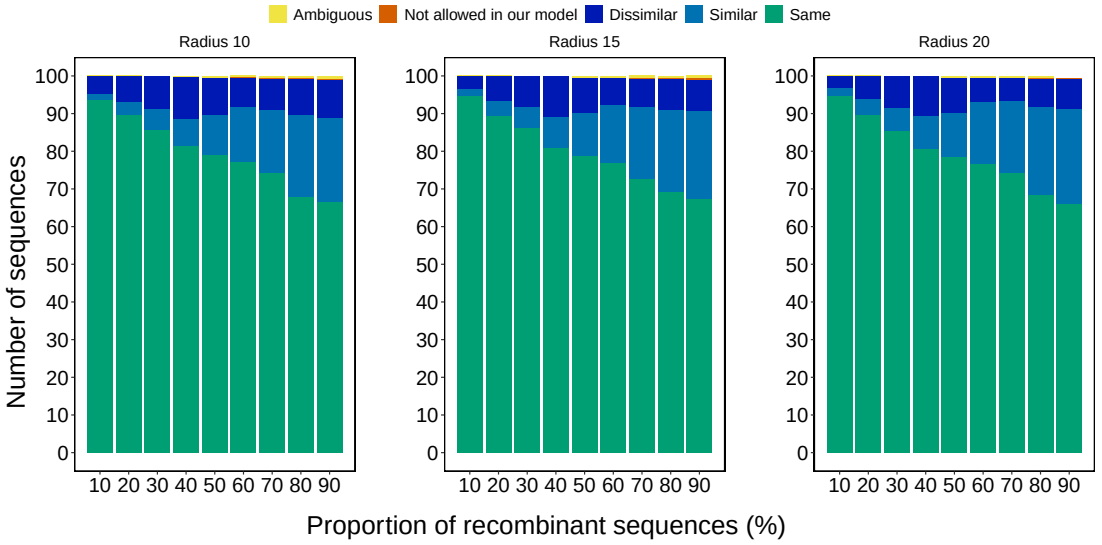


FIGURE 4.6: **Viterbi path comparison between the JHMM and improved JHMM when varying proportions of recombinant sequences.**

However, once we included the indels in the sequence generation process, the frequency profile of five categories changed substantially compared with previous ones. We noticed that this is a much larger number of sequences whose paths from the

JHMM are not allowed in our model, and this quantity increases with the indel rate (Figure 4.7) and indel size (Supplementary Figure 4.34). In the meantime, the sequences with identical or nearly identical (similar) paths with our model are declining. This is not surprising as higher diversity of sequences makes the model difficult to find breakpoints and it is more likely to jump further than our specified maximum allowable jump distance. In addition, by comparing the three panels of each figure, we found more sequences whose paths estimated from the JHMM are not allowed in our model, when the radius decreases from 20 to 10. This is reasonable, as decreasing the radius imposes stricter constraints and eliminates more possible paths.

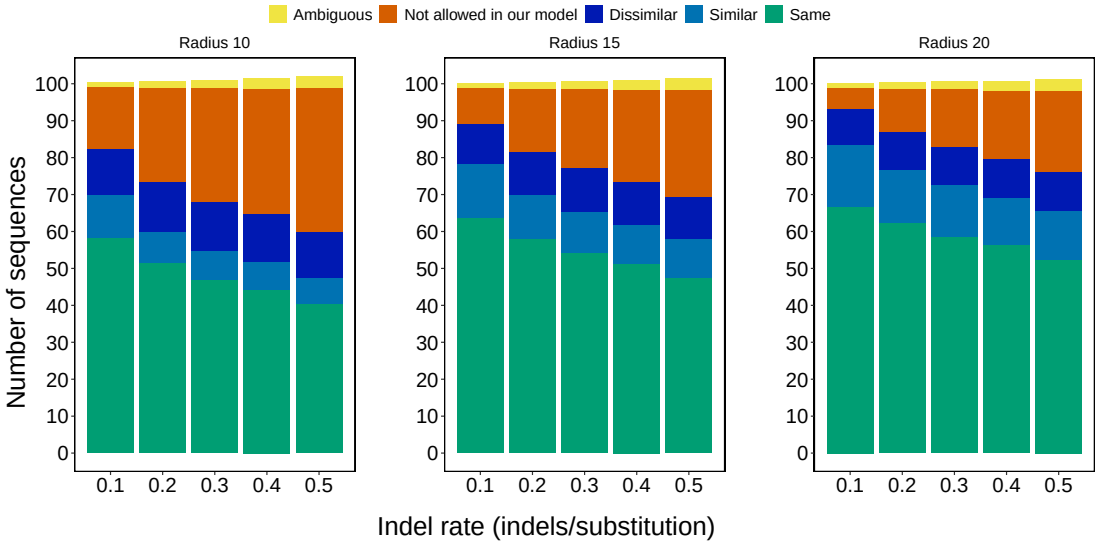


FIGURE 4.7: **Viterbi path comparison between the JHMM and improved JHMM when varying indel rate.**

In summary, the improved JHMM has almost the same mosaic representations with the JHMM when the input sequences are of equal length. However, when the input sequences are of unequal length, there are more unreasonable mosaic representations generated from the JHMM, especially when the indel rate or size is large. Our improved JHMM avoids this consequence.

4.3.2.3 Execution time

We also examined the execution time of each model. For each position of each target sequence, we search $O(nr)$ (n is the number of sequences) positions using the methods in our improved JHMM. However, it takes $O(nl)$ (l is the average sequence

length) in the JHMM. Therefore, the time complexity of methods in our model is $O(n^2lr)$ where $r \ll l$, while the time complexity of methods in the JHMM is $O(n^2l^2)$. That is to say, ours is faster by a factor of $O(l/r)$.

In the simulation, we calculated the overall running time for varying dataset size and sequence length. As expected, the results (Figure 4.8) suggest that the running time of our model appears to grow quadratically with the number of sequences and grow linearly with sequence length and radius. Moreover, our model is quicker than the JHMM as expected, suggesting that our model can be applied to datasets containing more or longer sequences than the JHMM.

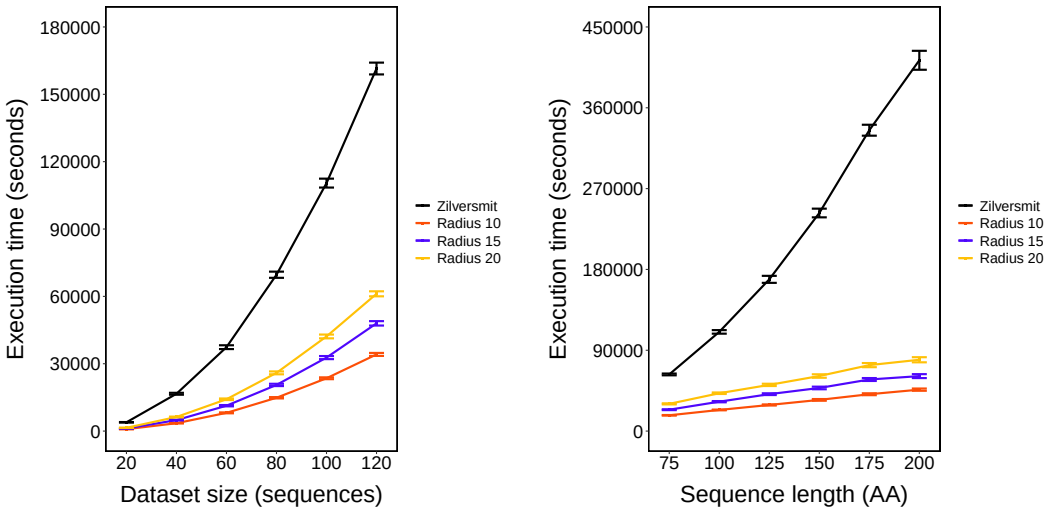


FIGURE 4.8: **Time (with 95% confidence intervals) taken by each model for varying dataset size and sequence length.**

There are three steps in both models, (1) estimating δ and ϵ with the Baum-Welch algorithm, (2) estimating ρ with the forward algorithm and (3) inferring the path with Viterbi algorithm. We show the lower running time of each step using improved JHMM than the JHMM in Supplementary Section 4.6.2.

Because the JHMM is a crucial component of our recombinant detection method described in the preceding chapter, we substituted it with our improved JHMM. We then studied the effects on the accuracy of identified recombinants. Overall, it maintains the accuracy of the JHMM, and we do not see any significant improvements (Supplementary Section 4.6.3).

4.4 Analysis of DBLα tags from a longitudinal study of Ghana

We applied our model to detect recombinants in a large longitudinal dataset from Ghana. As introduced in Section 1.3.2, we focused on the dataset which were collected at the end of wet seasons, Survey 1 (S1), Survey 4 (S4) and Survey 5 (S5). The IRS intervention was performed during S4, S1 and S5 represent the pre and post IRS respectively (see Table 1.1). For the initial data pre-processing, we repeated what we did for the pilot dataset (Section 3.5.3.1), i.e., we followed the existing pipeline [62, 74] and clustered raw tags into DBLα types. We were interested in the recombinants among all these types. Except this, we were also interested in detecting recombinant types belonging to the upsA group only. This is because upsA *var* genes are more conserved than other ups groups [71] and are expected to recombine mostly with themselves. In Table 4.4, we show the resulting number of DBLα types and basic information of the whole and upsA group.

TABLE 4.4: **Data summary.**

	# isolates	# DBLα types	Average length (sd)	UpsA	Average length (sd)
S1	918	35,095	125 (8.3)	2,200	115 (3.9)
S4	393	15,876	124 (8.1)	1,453	115 (3.4)
S5	510	18,668	124 (8.4)	1,615	115 (3.6)

Our model requires the specification of the target and source sequences and a radius. There are two time periods among these three surveys (S1-S4, S4-S5). For each period, we assigned the sequences from the more recent survey as the target and the another as the source. For instance, S4 is the target while older S1 is the source in the first period. We determined a sequence as a recombinant if there are at least two contributing source segments in its mosaic representation. Regarding the radius, we found that a bigger radius resulted in more accurate estimation in our simulation results, so we set the radius as 20.

We notice the parameter estimation is too slow using this large longitudinal dataset. We ran the analysis on the high performance computing (HPC) system at the University of Melbourne (72 Intel(R) Xeon(R) Gold 6254 CPU cores @ 3.10GHz, 768GB RAM). When the number of target and source sequences is both $1,000$, estimating δ and ϵ took over 55 days. When estimating ρ, if we follow the strategy of [62] by

only sampling 1,000 targets and keeping all sources, it finally ends up running over 10^4 jobs on HPC. Although each job cost around one day and all jobs could run in parallel theoretically, the practical running time is largely affected by the availability of HPC. Taken together, training the full dataset is time prohibited.

We dealt with this issue by randomly sampling from the dataset and using the sampled dataset to estimate parameters. We studied the effects of sampling and proposed to sample 50 target sequences and 500 source sequences (see more details in Supplementary Section 4.6.4) to estimate all parameters. Table 4.5 shows the resulting inferred parameters, and we see that the upsA group seems to have smaller $\hat{\delta}$ and $\hat{\rho}$ than the parameters from all sequences, which is reasonable.

TABLE 4.5: **Estimated parameters on the longitudinal dataset of Ghana.** S1-S4 and S4-S5 represent two time periods, Survey 1 to Survey 4 and Survey 4 to Survey 5. When the types belonging to upsA group only were considered, we denote with a bracket.

	δ	ϵ	ρ
S1-S4	0.015	0.833	0.045
S4-S5	0.015	0.806	0.043
S1 (upsA)-S4 (upsA)	0.003	0.877	0.014
S4 (upsA)-S5 (upsA)	0.003	0.897	0.012

With estimated parameters, we computed Viterbi paths in parallel. We divided all targets into subsets, and each time 30 sequences (against all source sequences in the dataset) were calculated on one core. In Table 4.6, we show the average running time across all subsets, and the time for estimating all parameters. From the resulting mosaic representations (see examples in Supplementary Figure 4.42), we finally identified 4,241 and 10,125 recombinants in each period using the whole data, and 199 and 494 recombinants using the upsA group only. As shown in Figure 4.9, the proportion of identified recombinants increases with time. This might be associated with IRS intervention and needs further exploration. Moreover, the proportions of recombinants from the upsA group are generally smaller than the proportions from the whole data. To explore the reasons for this difference, we conducted simulations and found that this difference is due to the high similarities and low recombination rate among upsA DBLα tags (see Supplementary Section 4.6.5). We also note that from this supplementary section, we see a high similarity between our simulated sequences and empirical DBLα tags.

TABLE 4.6: **Time consumption (hours) of the algorithm on the longitudinal dataset.** S1-S4 and S4-S5 represent two time periods, Survey 1 to Survey 4 and Survey 4 to Survey 5. When the types belonging to upsA group only were considered, we denote with a bracket.

	Estimation of δ and ϵ	Estimation of ρ	Viterbi path (# subsets)
S1-S4	33.5	10.9	41.2 (530)
S4-S5	32.4	11.0	14.0 (623)
S1 (upsA)-S4 (upsA)	23.5	9.5	1.6 (49)
S4 (upsA)-S5 (upsA)	29.3	9.5	1.0 (54)

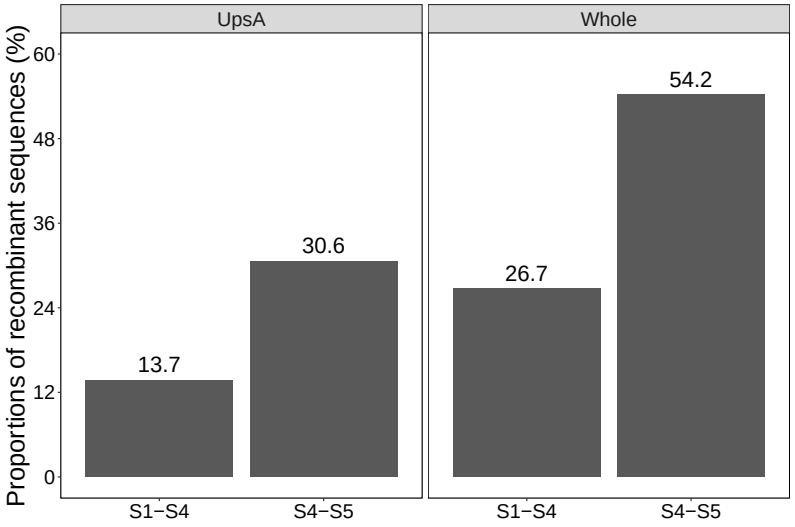


FIGURE 4.9: **The proportion of identified recombinants from our model using the sequences from upsA group only and the whole data.** S1-S4 and S4-S5 in x-axis represent two time periods, Survey 1 to Survey 4 and Survey 4 to Survey 5.

To explore the recombination patterns in this longitudinal dataset, we followed the analysis described in Section 3.3.2. We discuss the results as below.

4.4.1 Recombinations from the same ups group or domain subclass

In Section 3.3.2.1, we used the pilot dataset to test the hypothesis that *var* recombination occurs more frequently within the same ups group [61]. We first calculated the proportion of recombination triples in which one parent and the child, two parents and both parents and child (family) belong to the same ups group or DBLα

subclass. We also examined two distinct ups group classification schemes: one classifies each sequence to upsA and upsB/C using the existing pipeline [74], and the other classifies each sequence into upsA, upsB and upsC using the BLASTP [220] search and the reference database of Rask et al. [46]. We used the BLASTP to match each sequence to the closest reference sequence, then classified that sequence to the ups group of the closest reference sequence. We found that recombination happens more often in both the same ups group and domain subclass significantly.

Using our time series data, we again tested this pattern. It is worth mentioning that later we proposed an accurate method for classifying sequences into upsA, upsB and upsC three groups, as described in the following Chapter 5. Therefore, we used this method instead of the previous BLASTP.

We show the results in Table 4.7. It indicates that regardless of the dataset used, the proportions of parent-child and two parents belonging to the same ups group are significantly higher than the expected proportions, here the expected proportions were computed based on the independence of parents' ups groups and the ups group sharing between child and one of the parents. We used the one-sample t-test at 5% significance level for comparing proportions statistically. In addition, we see that recombination occurs preferentially within the same DBLα subclass with only one exception. These reaffirm our previous findings. When comparing Tables 3.1 and 4.7, we find the proportions of all three patterns (parent-child, parents and family) belonging to the same ups group or domain subclass are roughly the same over time. Although we used different methods for identifying recombinants and applied them to datasets from different time points, this particular recombination pattern remains.

TABLE 4.7: **Inferred proportions of recombinants from the same ups groups or DBLα subclasses.** Expected proportions are given in brackets. All p-values are highly significant ($< 2.2 \times 10^{-16}$) except for the entries marked in red.

		Parent-child	Parents	Family
S1-S4	UpsA vs. upsB/C	100.0% (94.1%)	99.2% (88.2%)	99.2% (88.2%)
	UpsA, B and C	86.7% (73.1%)	57.8% (46.2%)	44.9% (46.2%)
	DBLα	60.1% (54.2%)	22.2% (8.4%)	16.3% (8.4%)
S4-S5	UpsA vs. upsB/C	99.9% (91.7%)	98.9% (83.4%)	98.8% (83.4%)
	UpsA, B and C	84.8% (72.5%)	56.2% (45.0%)	41.7% (45.0%)
	DBLα	53.5% (53.6%)	20.3% (7.2%)	13.6% (7.2%)

4.4.2 Recombination proportions among ups groups or domain subclasses

We found different proportions of recombinants among the ups groups using the pilot dataset (Section 3.3.2.1). To explore whether this pattern maintains, we also studied it using our time series data. The proportions of recombinants in each ups group for all the datasets are shown in Table 4.8. We found that regardless of the dataset used, there was a significant association between recombination and ups groups (both $p < 2.2 \times 10^{-16}$ from a χ^2 contingency table test), with upsA having the smallest proportion of recombinants, and upsC having the largest. This ranking conforms to previous findings using the pilot dataset (Section 3.3.2.1). Next, when comparing the proportions of recombinants over time, we see a significant drop followed by a moderate increase for each ups group from pilot to S5. We suggest this should be investigated further by accounting for biological factors like IRS intervention.

TABLE 4.8: **Proportions of recombinants among ups groups.** The overall proportion of recombinants in each dataset is shown in brackets.

	UpsA	UpsB	UpsC
Pilot (85.4%)	82.3%	84.9%	87.6%
S1-S4 (26.7%)	13.8%	26.9%	30.5%
S4-S5 (54.2%)	31.7%	54.1%	60.6%

Using the pilot dataset, we also identified seven DBLα subclasses whose proportions of recombinants differed significantly from the average under the Bonferroni correction, as shown in Figure 3.4. DBLα0.1, 0.5 and 0.11 were too high, DBLα0.3, 0.8, 0.9 and 0.23 were too low. Using this time series data, we also calculated the proportions of recombinants among DBLα subclasses. We show the results in Figures 4.10 and 4.11. Importantly, we find all the previously identified seven DBLα subclasses differed again from the average significantly in this longitudinal dataset. Moreover, their directions (too high or too low) remain the same. Apart from these seven DBLα subclasses, we identified seven additional ones in both periods. DBLα0.4, 0.18, 1.2, 1.4, 1.8 and 2 were too low, while DBLα0.14 was too high.

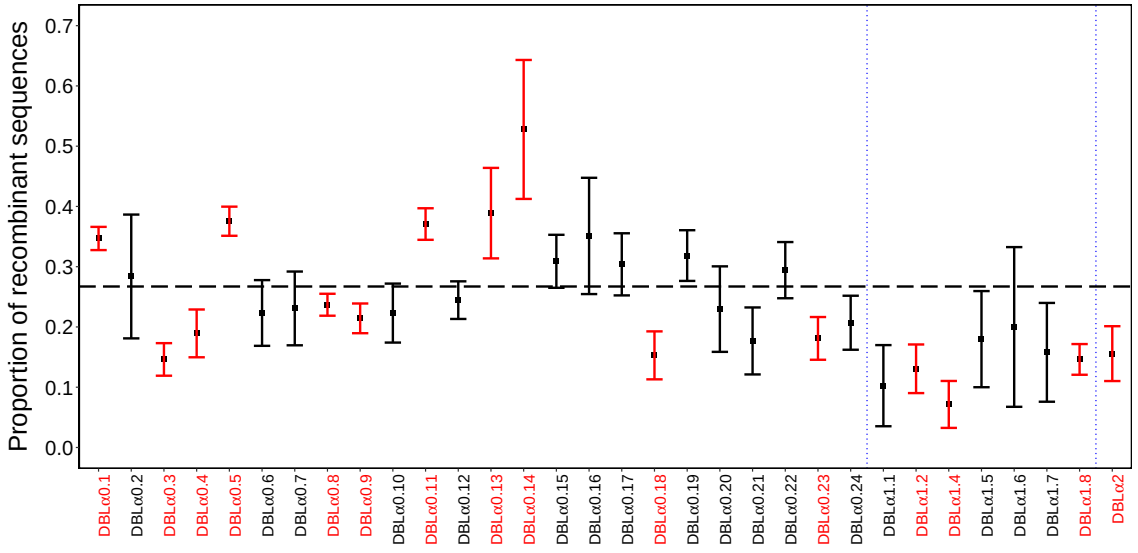


FIGURE 4.10: **Proportions (and 95% confidence intervals) of recombinants for each DBLα subclass using the S1-S4 data.** Subclasses which are significantly different from the overall average (under a correction for multiple testing) are highlighted in red. The horizontal dashed line displays the overall proportion of recombinant sequences in the entire dataset.

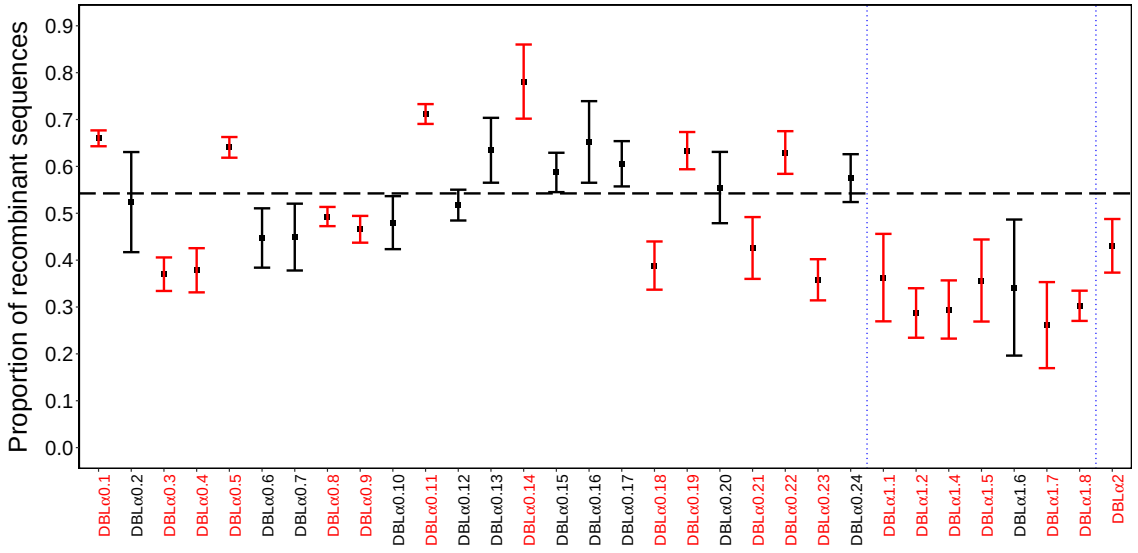


FIGURE 4.11: **Proportions (and 95% confidence intervals) of recombinants for each DBLα subclass using the S4-S5 data.** Subclasses which are significantly different from the overall average (under a correction for multiple testing) are highlighted in red. The horizontal dashed line displays the overall proportion of recombinant sequences in the entire dataset.

4.4.3 Conservation level between recombinant and non-recombinant types

We next studied the frequencies of DBLα types. Based on the results (Section 3.3.2.3) using the pilot dataset, we found that the non-recombinant types were more conserved than the recombinants. We want to evaluate whether this pattern still exists in our time series data. It turned out that both two time periods datasets demonstrated that non-recombinants occurred significantly more often than recombinants (average 4.2 vs. 3.1 using S1-S4 data; average 5.0 vs. 2.7 using S4-S5 data. Both $p < 2.2 \times 10^{-16}$ from Wilcoxon rank sum tests). In addition, we see an increase in the mean type occurrences among non-recombinants and a decrease in the mean type occurrences among recombinants with time (Supplementary Figure 4.43).

We subsequently examined the frequency distributions of DBLα types in isolates. We discovered that their distributions are highly right-skewed (Supplementary Figures 4.44 and 4.45), and they are similar to Figure 3.26 from the pilot dataset. Repeating the analysis in Section 3.3.2.3, we compared the proportion of frequent DBLα types between non-recombinant and recombinant groups. The results (Table 4.9) likewise indicated a greater proportion of frequent types in the non-recombinants than the recombinants; this effect is statistically significant in all datasets when the threshold is less than 10. Moreover, the proportion of frequent types in recombinants declines from pilot to S5, regardless of the threshold used. In contrast, the proportion of frequent types in non-recombinants grows in the same time period (Supplementary Figure 4.46). This suggests that recombinant types tend to be less stable over time than non-recombinants.

TABLE 4.9: **Proportions of frequent (larger than the threshold) recombinant and non-recombinant DBLα types.**

	Threshold	5	10	15	20
S1-S4	Recombinants	7.2%	2.1%	1.6%	1.2%
	Non-recombinants	21.5%	6.2%	2.8%	1.8%
	P-value (χ^2 test)	2.961×10^{-14}	3.027×10^{-4}	0.112	0.368
S4-S5	Recombinants	5.4%	0.8%	0.4%	0.2%
	Non-recombinants	28.2%	8.8%	4.6%	2.6%
	P-value (χ^2 test)	$< 2.2 \times 10^{-16}$	$< 2.2 \times 10^{-16}$	$< 2.2 \times 10^{-16}$	2.685×10^{-11}

4.4.4 Breakpoint distribution

We analysed the distribution of breakpoints in each dataset. We show the results in Figure 4.12. Similar to the breakpoint distribution of the pilot dataset (Figure 3.5), we found the recombination rate is also not constant in the entire dataset or upsA group data. All the breakpoint distributions display recombination peaks at some positions of the DBLα sequence. We discovered frequent recombinations at 25% and 85% positions of sequence, regardless of the dataset used. Intriguingly, we see a distribution peak in the middle of the sequence, no matter using the upsA group sequences in time series data or pilot dataset. However, this sole peak is substituted by two peaks before and after the sequence center using the whole time series data. We suggest that more biological factors should be considered for reasonable interpretations.

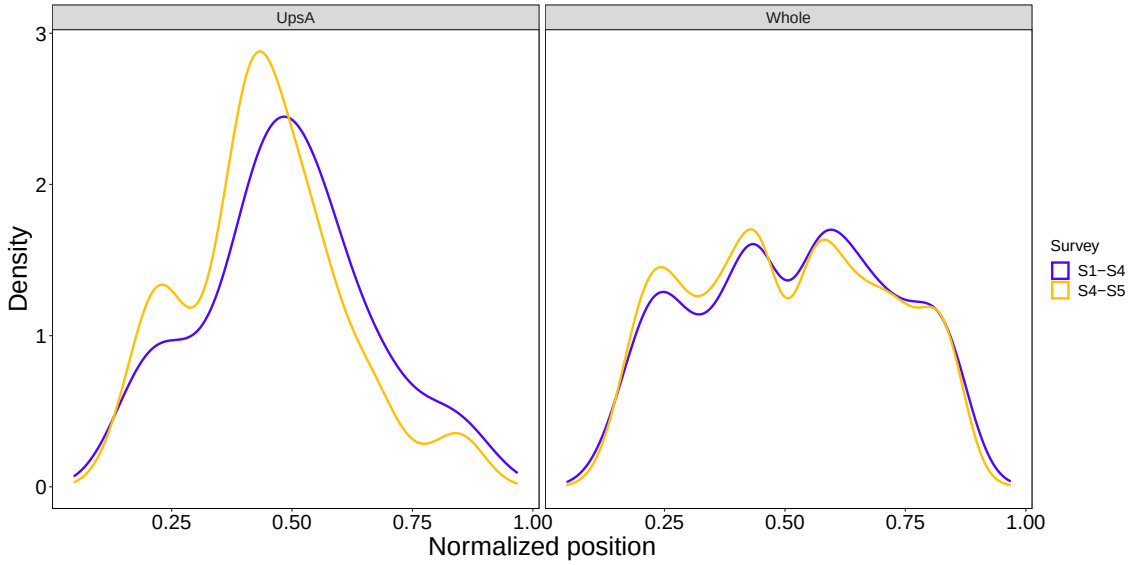


FIGURE 4.12: **Positions of recombination breakpoints.** Left panel was from the upsA group sequences, while right panel was obtained using the whole data.

4.5 Discussion

In this chapter, we have proposed an improved JHMM for detecting recombination from unaligned sequences. This model adds constraints on the jump destinations, based on the underlying reality in the uncalculated alignment of homologous sequences. Extensive simulations demonstrate the model's accuracy and efficiency. With a bigger radius, the estimated parameters of improved JHMM are closer to

the JHMM parameters, and two models have similar accuracy in identifying recombination. Importantly, the improved JHMM greatly reduces the artificial short segments from the JHMM in the real dataset of DBLα tags. This would be beneficial for finding the breakpoint positions more accurately. The efficiency improvement allows us to train the model on a dataset of over 10,000 tags, which only could have been done before with Viterbi training. Finally, we apply our model to detect recombinants between two time-separated datasets in Ghana. We find the recombination patterns within these datasets similar to the ones using the pilot dataset, and most patterns maintain with time.

Determining a suitable radius is important for the improved JHMM, and our study suggests that it depends on the user's preferences. The mosaic representations from empirical datasets could be used to calibrate the radius. A larger radius leads to a higher accuracy in finding parental sequences and less misidentified non-recombinants when the input sequences are of unequal length, in the meantime, using a larger radius increases the computational time linearly. Therefore, choosing the radius is a trade-off between the model accuracy and efficiency. We note that an implicit assumption of this model is that jumps longer than the radius are impossible. This is a strong and perhaps unrealistic assumption since we constrain the jump distance within an explicit interval. If the recombination occurs between homologous segments, the jump distance is zero with some obfuscating due to indels. Therefore, the jump distance from the empirical data is usually presented by a distribution rather than an interval.

Our model naturally inherits several advantages of the JHMM for detecting recombinations. Our model (1) is not limited to DBLα tags, and is applicable to any biological sequences (DNA or protein) involving recombination; (2) does not require a multiple sequence alignment or a reference panel of known non-recombinants as input; (3) accepts equal-length or unequal-length sequences; (4) can also detect recombinants when applying to time series data. One caveat is that the jump distances should be small so that using an interval as the constraint is relatively sensible.

Although our model is better than the original JHMM, it still has possible further improvements. In our model, the maximum jump distance is designed to be the same across all sequence positions. Therefore, exploring the jump distance distributions at different positions is useful. If they show a different pattern, setting various radii instead of a sole one would be more sensible. Moreover, estimating parameters δ and ϵ for large datasets is still time-consuming, and we suggest an acceleration of

the Baum-Welch algorithm (like the methods described in [221]) would be helpful. Lastly, from the resulting mosaic representations of real datasets, we notice that some target sequences are fully matched to a single source sequence without any mutation, indels or recombination. This is because there are duplicated DBLα types between the target and source databases. We believe excluding the duplication would speed up the Viterbi path calculation.

There are also several future works of simulations. In our current simulation pipeline, we simulated all sequences from the same time point. Each sequence was the target, and all the remaining sequences were the source. However, we applied our model to time-separated datasets in real data analysis. Therefore, the simulation does not fully simulate the case in the real world, and we suggest this simulation protocol could be improved by simulating datasets from different time points. For instance, we could manually specify the parents from a previous time point and simulate the recombinant child from a recent time. In addition, when applying to the time series data, the JHMM identifies recombinants' putative parents from ancestral sequences which most resemble their parents. We could mimic this by sampling datasets from the simulated datasets at each time point and examining their impacts on the inference. This improvement is also helpful for exploring the effects of various time intervals and sampling coverage on the performance of our model. Another thing is that we sampled datasets for estimating parameters in real data analysis since the computational burden from large data size. We could examine the effects of such sampling strategy using simulations as well.

Finally, further analysis of DBLα tags is needed in the future. There are a few unsolved questions in our current work. For instance, the proportion of recombinants increases from S1 to S5, and the reasons are still uncovered. Although we suspect this might be associated with IRS intervention, more quantitative analysis is required to support this hypothesis. We also believe more recombination patterns, such as meiotic and mitotic recombination, should be investigated. Importantly, all the recombination patterns we obtained are based on the datasets from a single district of Ghana; the patterns are compared across only three datasets (the pilot, S1-S4 and S4-S5). Studying the distribution of recombination patterns would be informative if we apply our model to datasets from different geographical locations (like the datasets from ten countries in [62]). To explore statistically significant recombination patterns, it is also imperative to detect recombination in datasets from more time points.

Codes and data availability

The DBLα tags from the Ghana study at Survey 1, Survey 4 and Survey 5 are publicly available (GenBank BioProject Number: PRJNA396962). Source codes are freely available at the Github repository (`https://github.com/qianfeng2/impJHMM`) or Zenodo (`https://zenodo.org/records/13826732`).

4.6 Supplementary materials

4.6.1 Algorithm derivation

4.6.1.1 Viterbi algorithm

For a target sequence x and source sequence y_k, we compute three $m+2$ by l_k+2 Viterbi matrices. $\forall i, i \in \{1, ..., m\}$ and $\forall j, j \in \{1, ..., l_k\}$, $v(i, M_k^j)$ is the probability of the most probable alignment, which ends by $(x(i), y_k(j))$ with hidden state M_k^j. Similarly, $v(i, I_k^j)$ and $v(i, D_k^j)$ are the probabilities of the most probable alignment which ends $(x(i), -)$ or $(-, y_k(j))$ with hidden state I_k^j or D_k^j respectively.

Initialization For computational purpose, we set the margin of each matrix as 0.

$$v(0, M_k^j) = v(0, I_k^j) = v(0, D_k^j) = v(m+1, M_k^j) = v(m+1, I_k^j) = v(m+1, D_k^j) = 0$$

$$v(i, M_k^0) = v(i, I_k^0) = v(i, D_k^0) = v(i, M_k^{l_k+1}) = v(i, I_k^{l_k+1}) = v(i, D_k^{l_k+1}) = 0$$

With probability π_M, the path starts with a match state.

$$v(1, M_k^j) = \begin{cases} \frac{\pi_M}{\left|\bigcup_{k=1}^{n} \omega_{k,1}\right|} e(x(1) \mid y_k(j)) & j \in \omega_{k,1} \\ 0 & \text{otherwise} \end{cases}$$

With probability $\pi_I = 1 - \pi_M$, the path starts with an insert state.

$$v(1, I_k^j) = \begin{cases} \frac{\pi_I}{\left|\bigcup_{k=1}^{n} \omega_{k,1}\right|} e(x(1)) & j \in \omega_{k,1} \\ 0 & \text{otherwise} \end{cases}$$

Recurrence We define $Z \in \{M, I\}$. For $i = 1, ..., m, j = 1, ..., l_k$,

$$v(i, M_k^j) = \begin{cases} e(x(i) \mid y_k(j)) \cdot \max \begin{cases} (1-2\delta-\rho-\tau)v(i-1, M_k^{j-1}) \\ (1-\epsilon-\rho-\tau)v(i-1, I_k^{j-1}) \\ (1-\epsilon)v(i-1, D_k^{j-1}) \\ \frac{\rho\pi_M}{\left|\bigcup_{k'=1,k'\neq k}^{n} \omega_{k',i}\right|} \max_{Z,k',j^*,j^* \in \omega_{k',i-1}} v(i-1, Z_{k'}^{j^*}) \end{cases} & j \in \omega_{k,i} \\ 0 & \text{otherwise} \end{cases}$$

$$v(i, I_k^j) = \begin{cases} e(x(i)) \cdot \max \begin{cases} \delta v(i-1, M_k^j) \\ \epsilon v(i-1, I_k^j) \\ \frac{\rho\pi_I}{\left|\bigcup_{k'=1,k'\neq k}^{n} \omega_{k',i}\right|} \max_{Z,k',j^*,j^* \in \omega_{k',i-1}} v(i-1, Z_{k'}^{j^*}) \end{cases} & j \in \omega_{k,i} \\ 0 & \text{otherwise} \end{cases}$$

$$v(i, D_k^j) \begin{cases} \max \begin{cases} \delta v(i, M_k^{j-1}) \\ \epsilon v(i, D_k^{j-1}) \end{cases} & j \in \omega_{k,i} \\ 0 & \text{otherwise} \end{cases}$$

We do not let two sequences get "too far" out of alignment even without a jump, so we make the corresponding entries of matrices be 0 when $j \notin \omega_{k,i}$. With recurrence, three matrices are computed row by row (the first index i).

Termination $v^E = \tau \max_{Z,k,j,j \in \omega_{k,m}} v(m, Z_k^j)$.

We keep pointers and trace back for finding the best alignment.

4.6.1.2 Forward algorithm

For a target sequence x and source sequence y_k, we compute three $m+2$ by l_k+2 forward matrices. $\forall i, i \in \{1, ..., m\}$ and $\forall j, j \in \{1, ..., l_k\}$, $f(i, M_k^j)$ is the probability of observing the partial alignment from the beginning to $(x(i), y_k(j))$, which ends with the hidden state M_k^j. Similarly, $f(i, I_k^j)$ and $f(i, D_k^j)$ are the probabilities of

observing the partial alignment from the beginning to $(x(i), -)$ or $(-, y_k(j))$, which ends with states I_k^j or D_k^j.

Initialization For computational purpose, we set the margin of each matrix as 0.

$$f(0, M_k^j) = f(0, I_k^j) = f(0, D_k^j) = f(m+1, M_k^j) = f(m+1, I_k^j) = f(m+1, D_k^j) = 0$$

$$f(i, M_k^0) = f(i, I_k^0) = f(i, D_k^0) = f(i, M_k^{l_k+1}) = f(i, I_k^{l_k+1}) = f(i, D_k^{l_k+1}) = 0$$

and

$$f(1, M_k^j) = \begin{cases} \frac{\pi_M}{\left| \bigcup_{k=1}^{n} \omega_{k,1} \right|} e(x(1) \mid y_k(j)) & j \in \omega_{k,1} \\ 0 & \text{otherwise} \end{cases}$$

$$f(1, I_k^j) = \begin{cases} \frac{\pi_I}{\left| \bigcup_{k=1}^{n} \omega_{k,1} \right|} e(x(1)) & j \in \omega_{k,1} \\ 0 & \text{otherwise} \end{cases}$$

Recurrence For $i = 1, ..., m, j = 1, ..., l_k$,

$$f(i, M_k^j) = \begin{cases} e(x(i) \mid y_k(j)) \cdot \Big[& (1 - 2\delta - \rho - \tau) f(i-1, M_k^{j-1}) + \\ & (1 - \epsilon - \rho - \tau) f(i-1, I_k^{j-1}) + \\ & (1 - \epsilon) f(i-1, D_k^{j-1}) + \\ & \frac{\rho \pi_M}{\left| \bigcup_{k'=1, k' \neq k}^{n} \omega_{k',i} \right|} \sum_{Z, k', j^*, j^* \in \omega_{k', i-1}} f(i-1, Z_{k'}^{j^*}) \Big] & j \in \omega_{k,i} \\ 0 & & \text{otherwise} \end{cases}$$

$$f(i, I_k^j) = \begin{cases} e(x(i)) \cdot \Big[& \delta f(i-1, M_k^j) + \\ & \epsilon f(i-1, I_k^j) + \\ & \frac{\rho \pi_I}{\left| \bigcup_{k'=1, k' \neq k}^{n} \omega_{k',i} \right|} \sum_{Z, k', j^*, j^* \in \omega_{k', i-1}} f(i-1, Z_{k'}^{j^*}) \Big] & j \in \omega_{k,i} \\ 0 & & \text{otherwise} \end{cases}$$

$$f(i, D_k^j) = \begin{cases} \delta f(i, M_k^{j-1}) + \epsilon f(i, D_k^{j-1}) & j \in \omega_{k,i} \\ 0 & \text{otherwise} \end{cases}$$

With recurrence, three matrices are computed row by row (first index i).

Termination $f^E = \tau \sum_{Z,k,j,j\in\omega_{k,m}} f(m, Z_k^j)$.

4.6.1.3 Backward algorithm

For a target sequence x and source sequence y_k, we compute three $m+2$ by l_k+2 backward matrices. $\forall i, i \in \{1, ..., m\}$ and $\forall j, j \in \{1, ..., l_k\}$, $b(i, M_k^j)$ is the probability of observing the partial alignment from $(x(i), y(j))$ to the end, given the hidden state M_k^j for $(x(i), y(j))$. Similarly, $b(i, I_k^j)$ and $b(i, D_k^j)$ are the probabilities of observing the partial alignment from the $(x(i), -)$ or $(-, y_k(j))$ to the end, given the hidden state I_k^j for observing $(x(i), -)$ or the hidden state D_k^j for $(-, y_k(j))$.

Initialization For computational purpose, we set the margin of each matrix as 0.

$$b(0, M_k^j) = b(0, I_k^j) = b(0, D_k^j) = b(m+1, M_k^j) = b(m+1, I_k^j) = b(m+1, D_k^j) = 0$$

$$b(i, M_k^0) = b(i, I_k^0) = b(i, D_k^0) = b(i, M_k^{l_k+1}) = b(i, I_k^{l_k+1}) = b(i, D_k^{l_k+1}) = 0$$

With the probability of termination τ,

$$b(m, M_k^j) = \begin{cases} \tau & j \in \omega_{k,m} \\ 0 & \text{otherwise} \end{cases}$$

$$b(m, I_k^j) = \begin{cases} \tau & j \in \omega_{k,m} \\ 0 & \text{otherwise} \end{cases}$$

Recurrence For $i = m, ..., 1, j = l_k, ..., 1$,

$$b(i, M_k^j) = \begin{cases} (1-2\delta-\rho-\tau)e(x(i+1) \mid y_k(j+1))b(i+1, M_k^{j+1}) \\ +\delta e(x(i+1))b(i+1, I_k^j) + \delta b(i, D_k^{j+1}) \\ +\frac{\rho\pi_M}{|\bigcup_{k'=1,k'\neq k}^{n} \omega_{k',i+1}|} \sum_{k'} \sum_{p,p\in\omega_{k',i+1}} b(i+1, M_{k'}^p)e(x(i+1) \mid y_{k'}(p)) \\ +\frac{\rho\pi_I}{|\bigcup_{k'=1,k'\neq k}^{n} \omega_{k',i+1}|} \sum_{k'} \sum_{p,p\in\omega_{k',i+1}} b(i+1, I_{k'}^p)e(x(i+1)) & j \in \omega_{k,i} \\ 0 & \text{otherwise} \end{cases}$$

$$b(i, I_k^j) = \begin{cases} (1-\epsilon-\rho-\tau)e(x(i+1) \mid y_k(j+1))b(i+1, M_k^{j+1}) \\ +\epsilon e(x(i+1))b(i+1, I_k^j) \\ +\frac{\rho\pi_M}{|\bigcup_{k'=1,k'\neq k}^{n} \omega_{k',i+1}|} \sum_{k'} \sum_{p,p\in\omega_{k',i+1}} b(i+1, M_{k'}^p)e(x(i+1) \mid y_{k'}(p)) \\ +\frac{\rho\pi_I}{|\bigcup_{k'=1,k'\neq k}^{n} \omega_{k',i+1}|} \sum_{k'} \sum_{p,p\in\omega_{k',i+1}} b(i+1, I_{k'}^p)e(x(i+1)) & j \in \omega_{k,i} \\ 0 & \text{otherwise} \end{cases}$$

$$b(i, D_k^j) = \begin{cases} (1-\epsilon)e(x(i+1) \mid y_k(j+1))b(i+1, M_k^{j+1}) + \epsilon b(i, D_k^{j+1}) & j \in \omega_{k,i} \\ 0 & \text{otherwise} \end{cases}$$

With recurrence, three matrices are computed row by row (the first index i).

Termination $b = \sum_k \sum_{j,j\in\omega_{k,1}} \{b(1, M_k^j)e(x(1) \mid y_k(j))\pi_M/\iota + b(1, I_k^j)e(x(1))\pi_I/\iota\}$, where $\iota = |\bigcup_{k=1}^{n} \omega_{k,1}|$.

4.6.2 Running time for each step of the improved JHMM

Since the algorithm complexity of each step is the same, we expect the running time of each step using our improved JHMM is lower than the JHMM. We therefore calculated the running time of both models for each step from the simulation, and found the results (Supplementary Figures 4.35–4.41) agree with this expectation. In addition, since the parameters that affect the running time of our model are

only dataset size, sequence length and radius, we found from the simulation results that the running time of our improved JHMM is stable when varying other parameters (proportions of recombinant sequences, average number of recombinations per recombinant sequence, indel rate and indel size). This matches our expectations. Interestingly, we noticed that there is a slight running time decrease in estimating ρ when the mutation rate increases from 0.1 to 0.5 (Supplementary Figure 4.39), this is because the upper bound of ρ is $\min\{1-2\delta-\tau, 1-\epsilon-\tau\}$, striking change of estimated $\hat{\epsilon}$ (Supplementary Figure 4.29) reduces the range of ρ (the input of golden section search), thus decreasing the running time of this step.

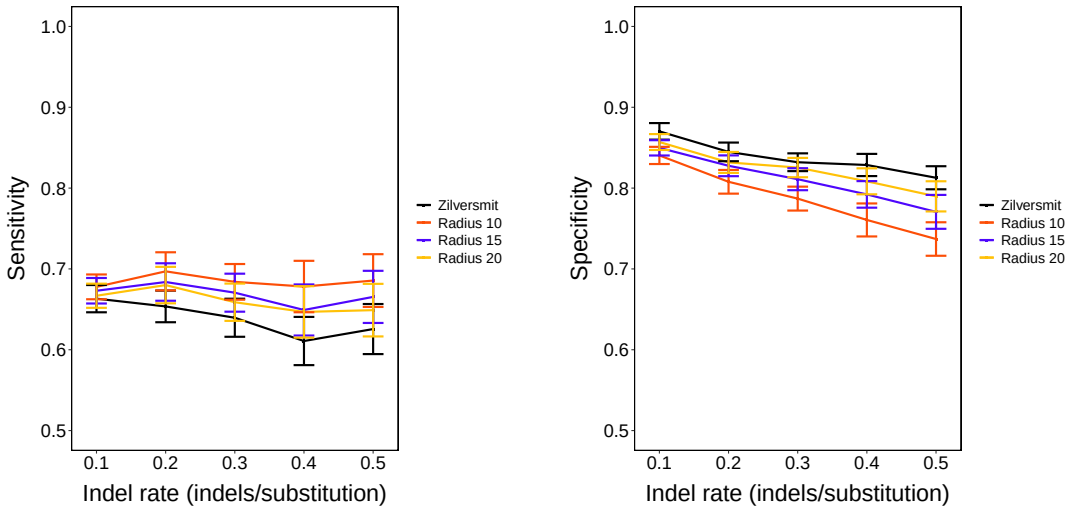
4.6.3 Effects of improved JHMM on the accuracy of identified recombinants

We used our improved JHMM as the first step in the previous recombinant detection algorithm (Chapter 3) and applied it to the simulated datasets. We can see (Figure 4.13) that our improved JHMM has slightly higher sensitivity and lower specificity than the conventional JHMM. This can be explained by more triples identified by our model, as shown in Figure 4.14. By setting a radius, our model eliminates more paths and results in more recombination triples. As a result, our model has higher true positive and false positive rates. Moreover, increasing the radius makes our model similar to the JHMM. Therefore, the resulting sensitivity and specificity get closer to the ones using the JHMM. This satisfies our expectations too. Overall, there is no significant accuracy improvement using this improved JHMM for identifying recent recombinants with our algorithm in Chapter 3.

4.6.4 Sampling effects on the estimated parameters

4.6.4.1 Sampling strategy

When estimating parameters, we found that the implementation of the Baum-Welch algorithm for estimating δ and ϵ was rather slow. Therefore, we decided to randomly sample without replacement from the whole dataset, and use the sampled data for estimating parameters. In this step, the probability of recombination ρ was set to 0, so there is no recombination, and we implement the Baum-Welch algorithm under a pair HMM. Based on the nature of this algorithm, except sampling sequences from



(a) (b)

FIGURE 4.13: **Sensitivity and specificity (and 95% CIs) for varying indel rate.**

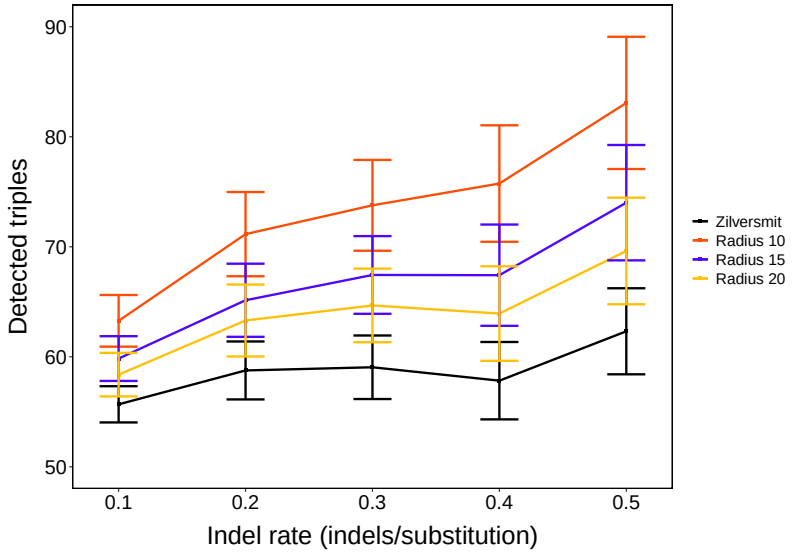


FIGURE 4.14: **Number of identified recombination triples (with 95% CIs) using our model with various radii and the JHMM.**

the datasets, we can also sample pairs, where each pair comprises a single target and a single source sequence. In total, we explored three strategies below.

I. Sampling sequences from the target and source datasets before the Baum-Welch algorithm.

II. Sampling pairs from the target and source datasets before the Baum-Welch algorithm.

III. Sampling sequences from the target and source datasets for each iteration within the Baum-Welch algorithm.

To evaluate these three options, we created a 'temporary' population by randomly sampling 100 sequences from both S1 and S4. The 100 sampled sequences of S4 formed a new target dataset, and the 100 sampled sequences from S1 formed a new source dataset. This new dataset allowed us to apply the Baum-Welch algorithm[1] directly and get estimated non-jump parameters (δ and ϵ). We then subsampled from this reference population using the above three strategies and evaluated them.

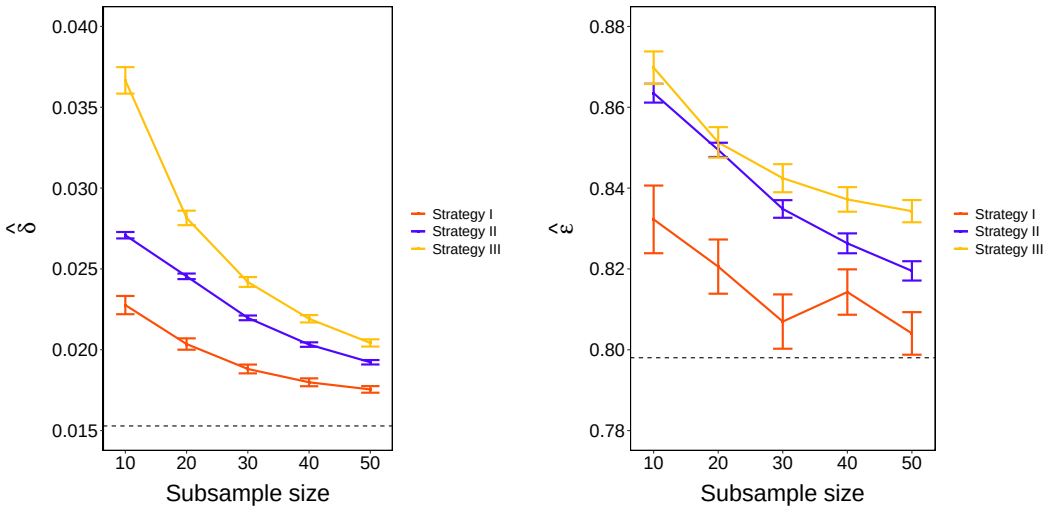
When sampling sequences with strategies I and III, we sampled from targets and sources without replacement with a range of sizes (from 10 to 50). The number of sampled targets is the same as the number of sampled sources. We resampled 100 datasets from the temporary population for each size and estimated parameters for each dataset in turn. When sampling pairs with strategy II, we sampled from 100×100 pairs without replacement. The number of sampled pairs ranged from 10^2 to 50^2. Same with other strategies, 100 datasets were generated again.

We show the results of these three strategies in Figure 4.15. Overall, the more data is used, the closer the estimated parameters are to the ones using the temporary population. The strategy I shows a significant advantage over strategies II and III, since the estimated parameters are the closest to the results using the temporary population.

We firstly investigated why strategy I performed better than strategy II. As introduced in Section 2.2.1.3, the most important component of Baum-Welch algorithm is to calculate the expected transitions and emissions. From these expectations, we are able to update the parameters for each iteration. Following the same notation of Section 4.2, we denote $x(i)$ as the ith character in target sequence x, and $y_k(j)$ as the jth character in source sequence y_k. For calculating the expected transitions from hidden state s to s' (s and s' could be the same or different), the quantity to compute is $\Pr(\pi(\diamond_1, \diamond_2) = s, \pi(i,j) = s' \mid x, Y)$, where π refers to the hidden state. Indices $\diamond_1, \diamond_2$ depend on the state of s',

$$(\diamond_1, \diamond_2) = \begin{cases} (i-1, j-1) & s' = M \\ (i-1, j) & s' = I \\ (i, j-1) & s' = D \end{cases}$$

[1]We corrected errors in the source code of Zilversmit *et al.* (2013).



(a) (b)

FIGURE 4.15: **Mean $\hat{\delta}$ and $\hat{\epsilon}$ (with 95% CIs) using three strategies for varying subsample size.** Dashed horizontal lines corresponds to the results using the temporary population.

$$\begin{aligned}
\Pr(\pi(\diamond_1, \diamond_2) = s, \pi(i,j) = s' \mid x, Y) &= \frac{\Pr(\pi(\diamond_1, \diamond_2) = s, \pi(i,j) = s', x \mid Y)}{\Pr(x \mid Y)} \\
&= \frac{\sum_{k=1}^{n} \Pr(\pi(\diamond_1, \diamond_2) = s, \pi(i,j) = s', x \mid Y, c = k) \Pr(c = k)}{\sum_{k=1}^{n} \Pr(x \mid Y, c = k) \Pr(c = k)} \\
&= \frac{\sum_{k=1}^{n} \Pr(\pi(\diamond_1, \diamond_2) = s, \pi(i,j) = s', x \mid y_k)}{\Pr(x \mid y_1) + \Pr(x \mid y_2) + ... + \Pr(x \mid y_n)} \\
&= \frac{\sum_{k=1}^{n} \Pr(\pi(\diamond_1, \diamond_2) = s, \pi(i,j) = s' \mid x, y_k) \Pr(x \mid y_k)}{\Pr(x \mid y_1) + \Pr(x \mid y_2) + ... + \Pr(x \mid y_n)}
\end{aligned}$$

$\Pr(\pi(\diamond_1, \diamond_2) = s, \pi(i,j) = s' \mid x, y_k)$ is the quantity for calculating expected transitions if we consider sequence pair as input of the algorithm. Likewise, for the expected emissions,

$$\Pr(\pi(i,j) = s \mid x, Y) = \frac{\sum_{k=1}^{n} \Pr(\pi(i,j) = s \mid x, y_k) \Pr(x \mid y_k)}{\Pr(x \mid y_1) + \Pr(x \mid y_2) + ... + \Pr(x \mid y_n)}$$

No matter the expected transitions or emissions, strategy I is always a weighted version of strategy II. The weight is $\Pr(x \mid y_k)$. Therefore, strategy I performed better.

Sampling sequences for each iteration in strategy III seems to use more data for estimating parameters. However, it breaks the structure of the Baum-Welch algorithm and cannot guarantee the likelihood to increase until converge. In fact, we observed that the likelihood of the data increased at the first few iterations and then kept fluctuating. This requires a well-designed stopping rule and beyond the scope of this thesis. Last but not least, one might propose to sample pairs for each algorithm iteration. With our current results, we expect this will not perform better than strategy I and finally decide not to do so.

4.6.4.2 Sampling targets vs. sampling sources

Once we were determined to sample sequences, the naturally arising problem was how many target sequences we needed to sample and how many source sequences to sample. The above section kept the number of sampled target and source sequences equal. We then studied another two scenarios, (1) only sample the target and fix the source sequences and (2) only sample the source and fix the target sequences. The results (Figure 4.16) suggested that sampling targets only made the estimated parameters much closer to the reference results, compared with sampling sources only or sampling targets and source (Figure 4.15). Therefore, we should sample a small number of target sequences and a larger number of source sequences.

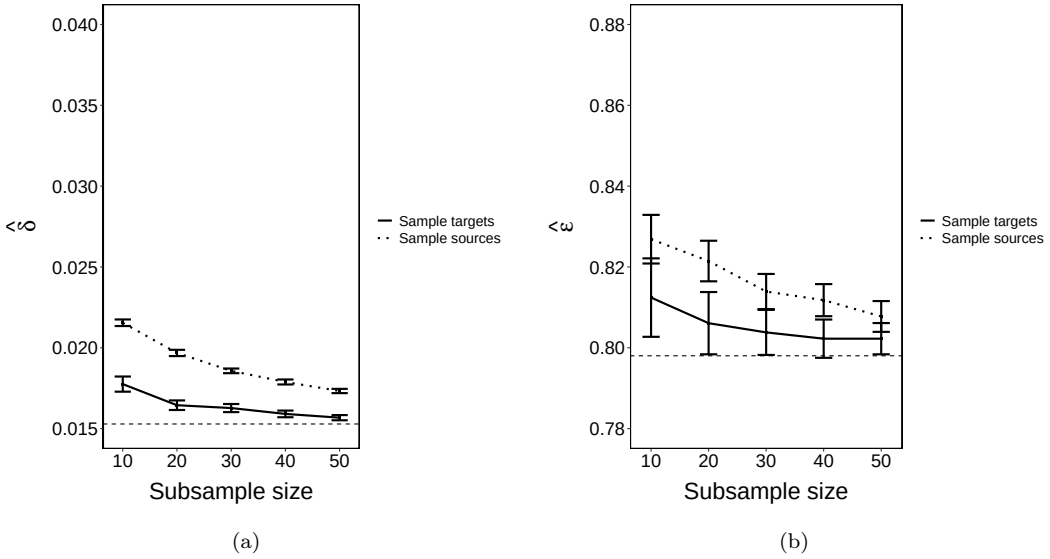


FIGURE 4.16: **Mean $\hat{\delta}$ and $\hat{\epsilon}$ (with 95% CIs) for varying subsample size.** Dashed horizontal lines corresponds to the results using the temporary population.

4.6.4.3 The number of sampled target and source sequences

We next focused on the ratio between the number of sampled targets and the number of sampled sources, while fixing their product which keeps the number of pairs fixed. We increased the ratio from 0 to 1 and the product from 1,000 to 20,000. For each ratio and product, we computed the number of target sequences and the number of source sequences we needed to sample. In order to investigate the scenarios of small ratios like 0.05 and 0.1, we increased our temporary population size from 100 to 200. For each subsample size, we resampled 100 datasets from the temporary population for estimating parameters. We show the estimated δ, ϵ and ρ in Figures 4.17, 4.18 and 4.19 separately.

Overall, increasing the product returned a more accurate estimation. This is reasonable as more sequences were used for training the model. Importantly, we noticed a growing trend for δ and ρ, indicating a higher ratio would lead to less accurate results; meanwhile, a too-small ratio resulted in a large variance of each parameter. We therefore set the ratio to 0.1. We note that this ratio is a user-chosen parameter. The product of the number of sampled target and source sequences affects the running time. Larger product results in longer computational time linearly. Based on time availability, we set this product to 25,000.

Therefore, we finally sampled 50 target and 500 source sequences for estimating parameters. Compared with the approximate time for estimating parameters from over 10,000 targets and 30,000 sources in real data, our sampling strategy has saved $\sim$554 months for estimating δ and ϵ, and $\sim$188 months for estimating ρ.

4.6.5 Lower proportion of recombinants in upsA DBLα tags

We found (from Figure 4.9) that the proportion of identified recombinants using the sequences from the upsA group is smaller than the whole data. On the one hand, the conserved upsA sequences could lead to a smaller number of recombinants detected, as sequences are more similar to each other. This biology indicates a lower mutation rate of upsA sequences. From our current simulation result (Figure 4.29), we indeed found that a decreased mutation rate can be confounded with a decreased recombination rate. On the other hand, the lower recombination rate of upsA sequences than the non-upsA sequences would also lead to fewer recombinants (as shown in Table 4.5).

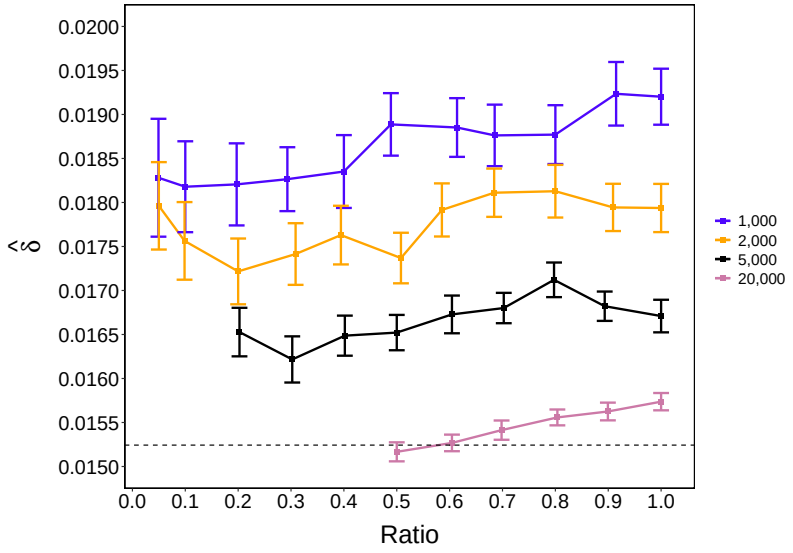


FIGURE 4.17: **Mean $\hat{\delta}$ (with 95% CIs) for varying ratio.** The dashed horizontal line represents the $\hat{\delta}$ estimated from the temporary population.

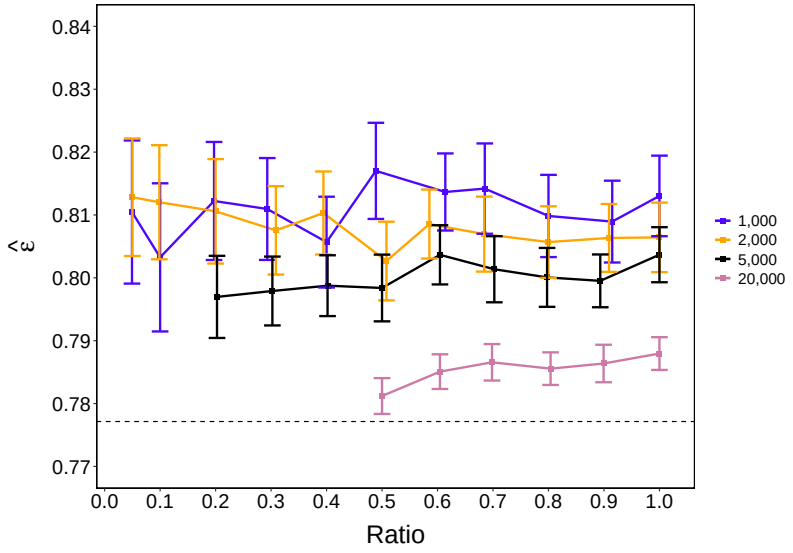


FIGURE 4.18: **Mean $\hat{\epsilon}$ (with 95% CIs) for varying ratio.** The dashed horizontal line represents the $\hat{\epsilon}$ estimated from the temporary population.

To investigate the causes of this difference, we focused on the simulation with varying mutation rate where the recombination rate is fixed. The simulation parameters are shown in bold texts of Table 4.2. In order to get the difference in mutation rate between sequences from upsA and whole data, we took the pairwise sequences' distance as a proxy for mutation rate. This also helps us compare the mutation rate between the real and simulated data. Here, we used the previously adopted [62] FFP method [222] to calculate DBLα pairwise distance. We calculated the distances using all upsA DBLα sequences and randomly sampled (for computational purposes)

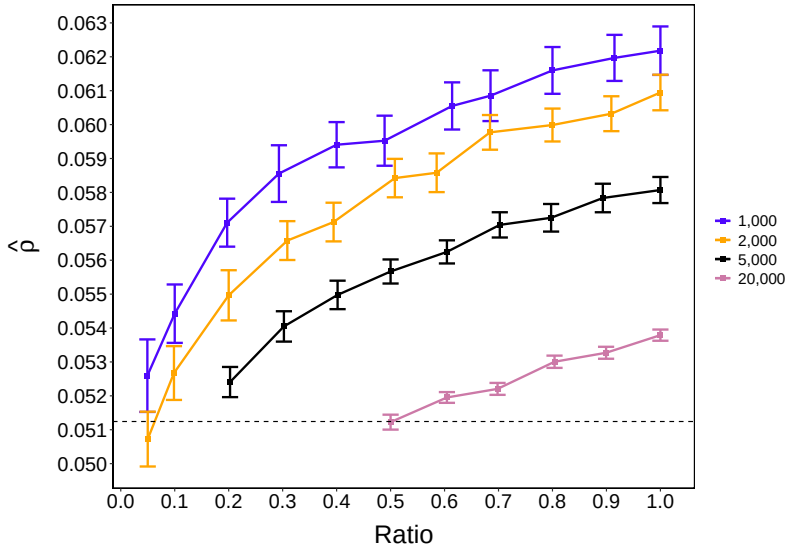


FIGURE 4.19: **Mean $\hat{\rho}$ (with 95% CIs) for varying ratio.** The dashed horizontal line represents the $\hat{\rho}$ estimated from the temporary population.

20K sequences from whole group. For a fair comparison, we also calculated the FFP distances using all simulated data with various mutation rates (from 0.1 to 0.5).

The results are shown in Figure 4.20. We found the mean similarities of real data matched to the simulated data with a mutation rate between 0.1 and 0.2. Based on Figure 4.29, this corresponds approximately to a difference of 0.002 for the estimated recombination rate. Compared with the difference in estimated recombination rate from our model (Table 4.5), this difference is small. In conclusion, the differences in the proportion of identified recombinants from the upsA group and whole data are not solely due to mutation rate but also recombination rate differences.

4.6.6 Supplementary figures and tables

TABLE 4.10: **The proportion of jump distance from empirical datasets captured by various radii.**

	Radius 10	Radius 15	Radius 20
Ghana data	99.5%	99.9%	99.9%
Global data [62]	98.2%	99.5%	99.8%

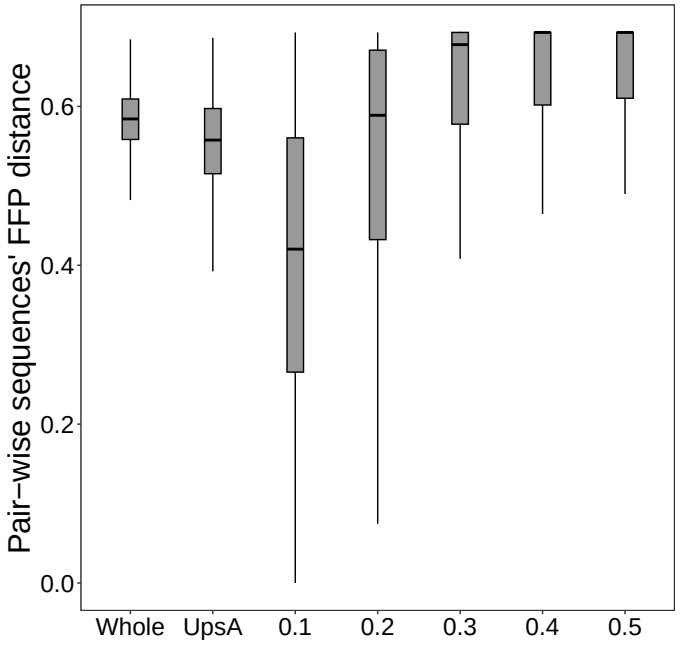


FIGURE 4.20: **Distributions of pair-wise distance using DBLα sequences (from upsA group only and whole data) and simulated datasets.**

TABLE 4.11: **The average proportion of recombinants with a correctly inferred number of breakpoints.** The standard deviation is shown in the bracket.

	Zilversmit et al. [54]	Radius 10	Radius 15	Radius 20
Equal-length sequences	55.0% (8.5%)	61.2% (7.6%)	61.2% (7.7%)	61.2% (7.7%)
Unequal-length sequences	54.9% (10.5%)	43.8% (10.0%)	49.5% (9.2%)	52.5% (9.2%)

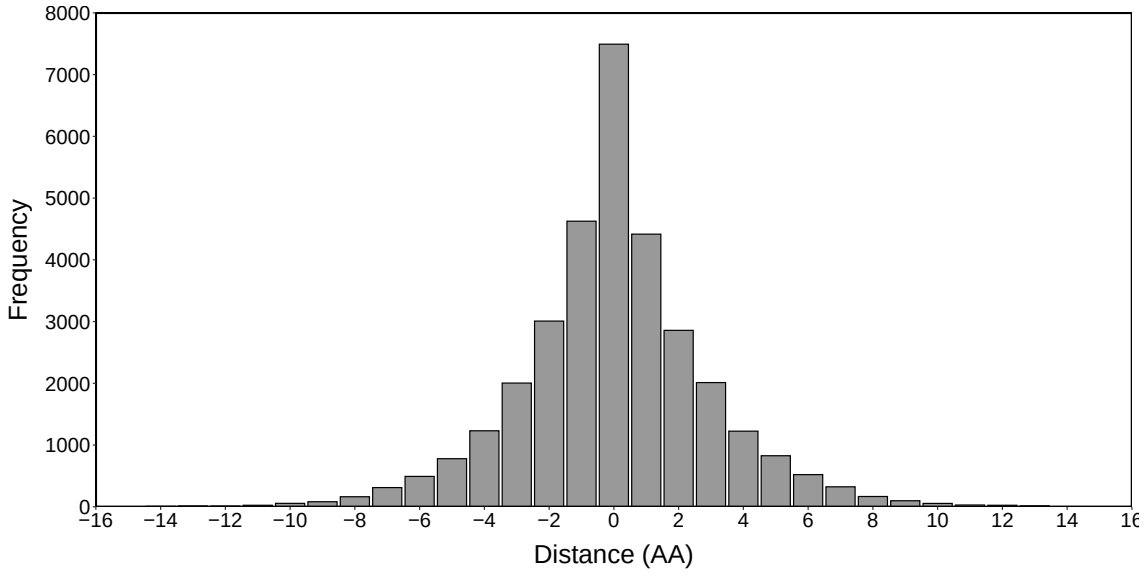


FIGURE 4.21: **Histogram of the distance calculated from mosaic representations using the Ghana pilot dataset.** The distance is the difference value between real and expected jump position for each recombination.

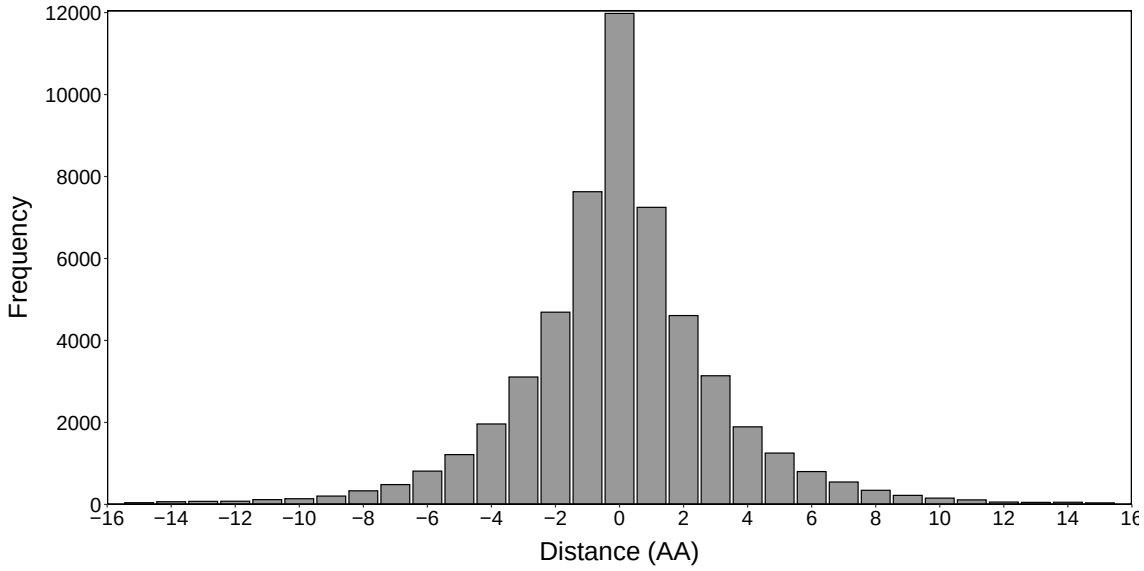


FIGURE 4.22: **Histogram of the distance calculated from mosaic representations using the global dataset.** The distance is the difference value between real and expected jump position for each recombination.

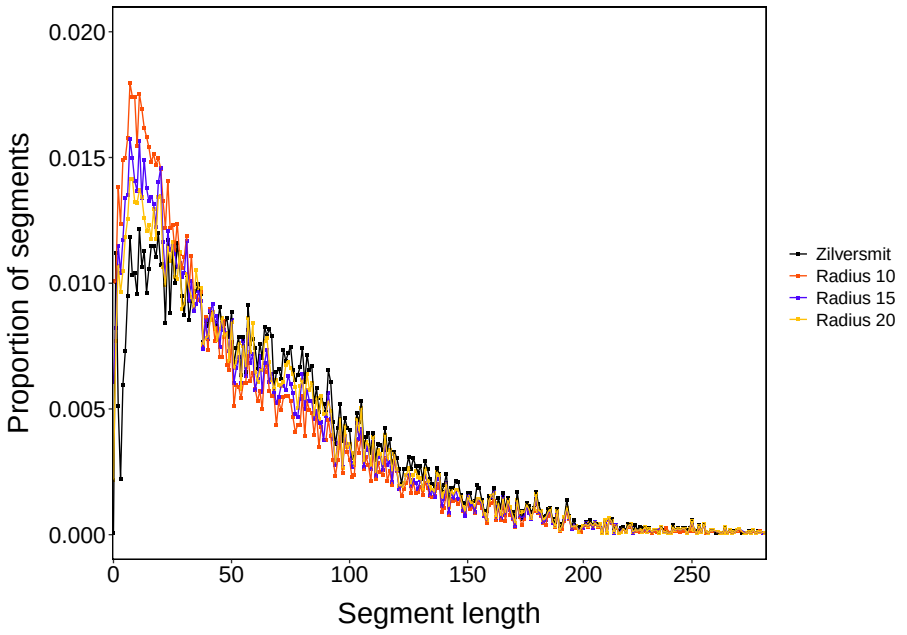


FIGURE 4.23: **Source segment length from the mosaic representations using simulated unequal-length sequences with the JHMM and improved JHMM.**

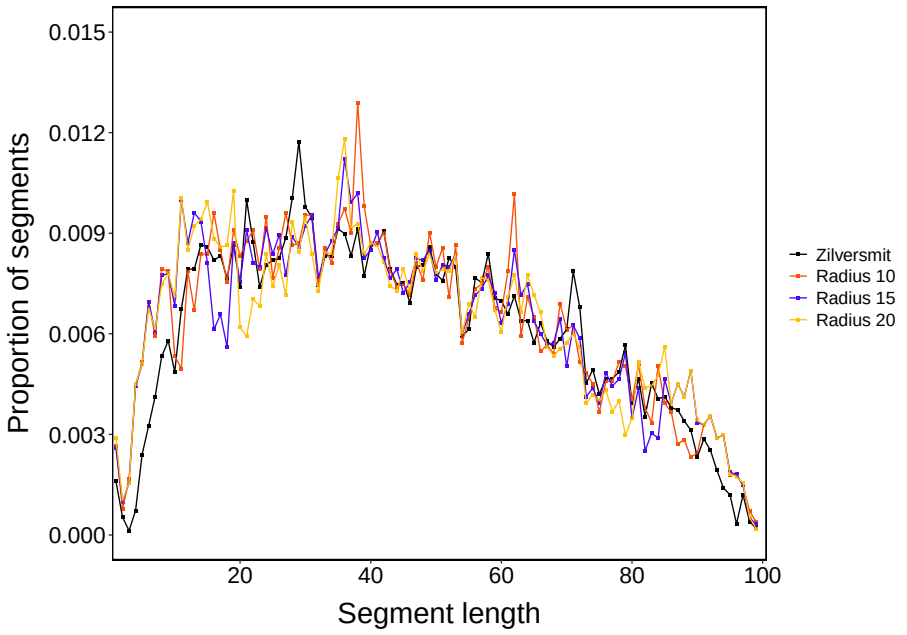


FIGURE 4.24: **Source segment length from the mosaic representations using simulated equal-length sequences with the JHMM and improved JHMM.**

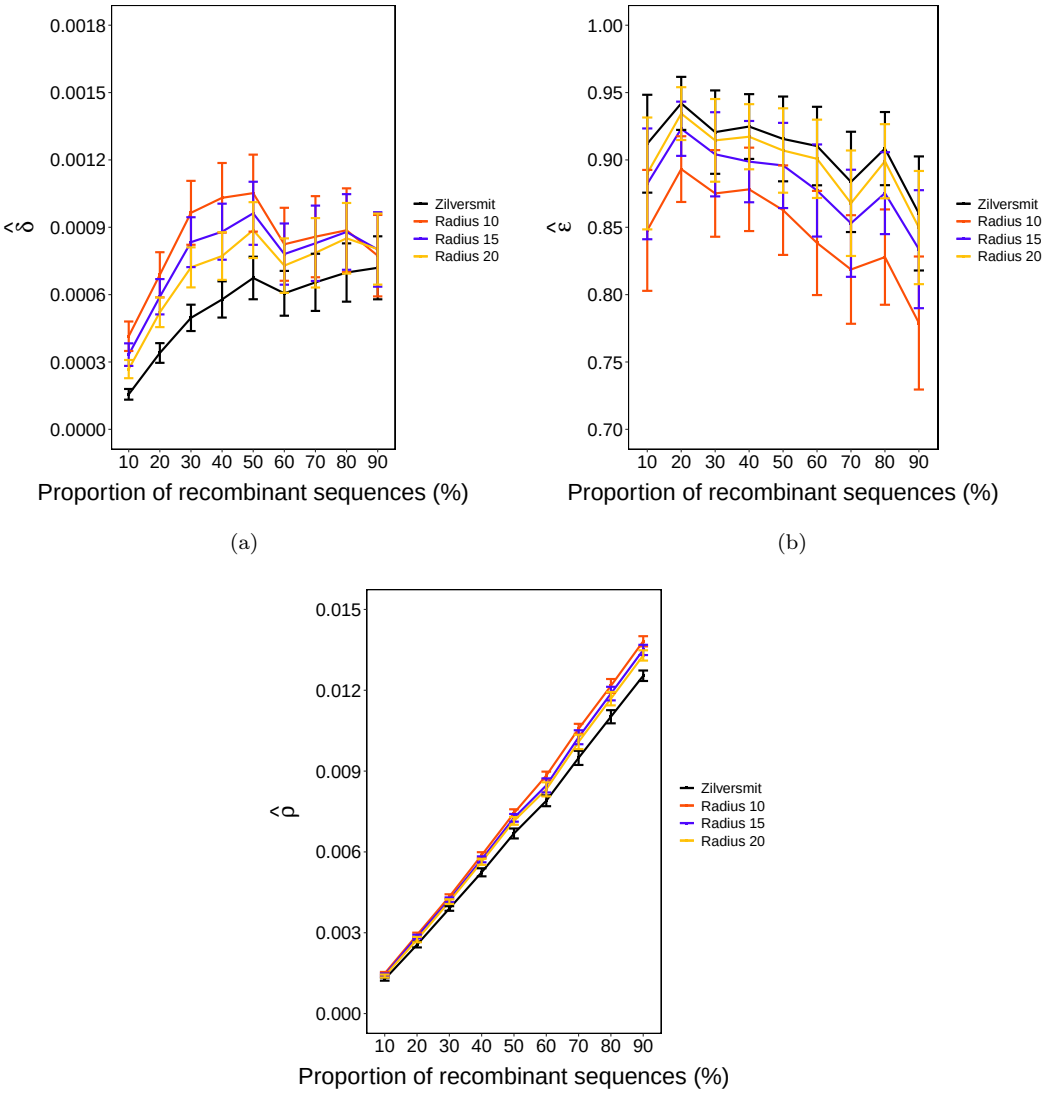


FIGURE 4.25: **Mean (with 95% confidence intervals) of each estimated parameter for varying proportions of recombinant sequences using the JHMM and improved JHMM.**

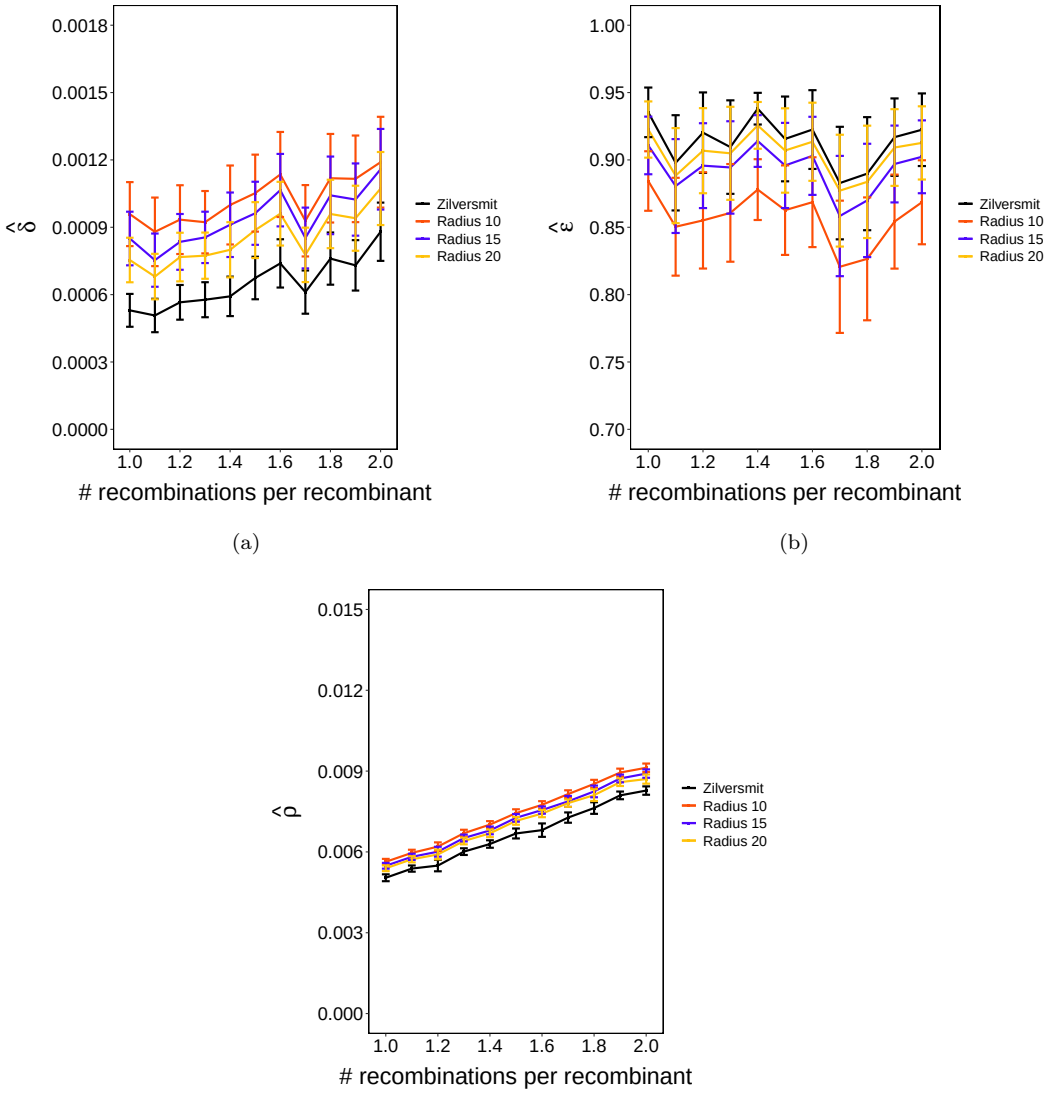


FIGURE 4.26: **Mean (with 95% confidence intervals) of each estimated parameter for varying number of recombinations per recombinant sequence using the JHMM and improved JHMM.**

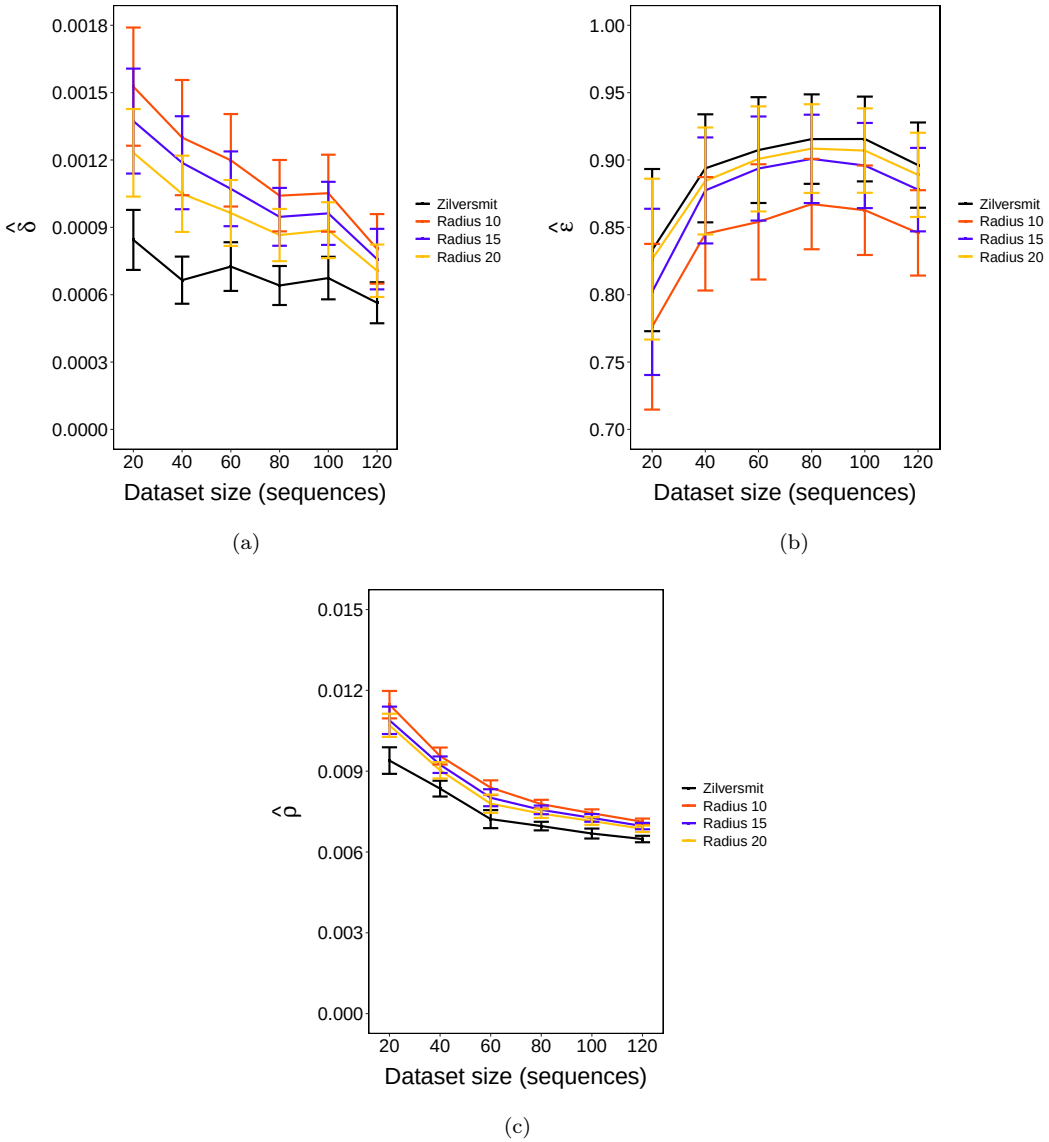


FIGURE 4.27: **Mean (with 95% confidence intervals) of each estimated parameter for varying dataset size using the JHMM and improved JHMM.**

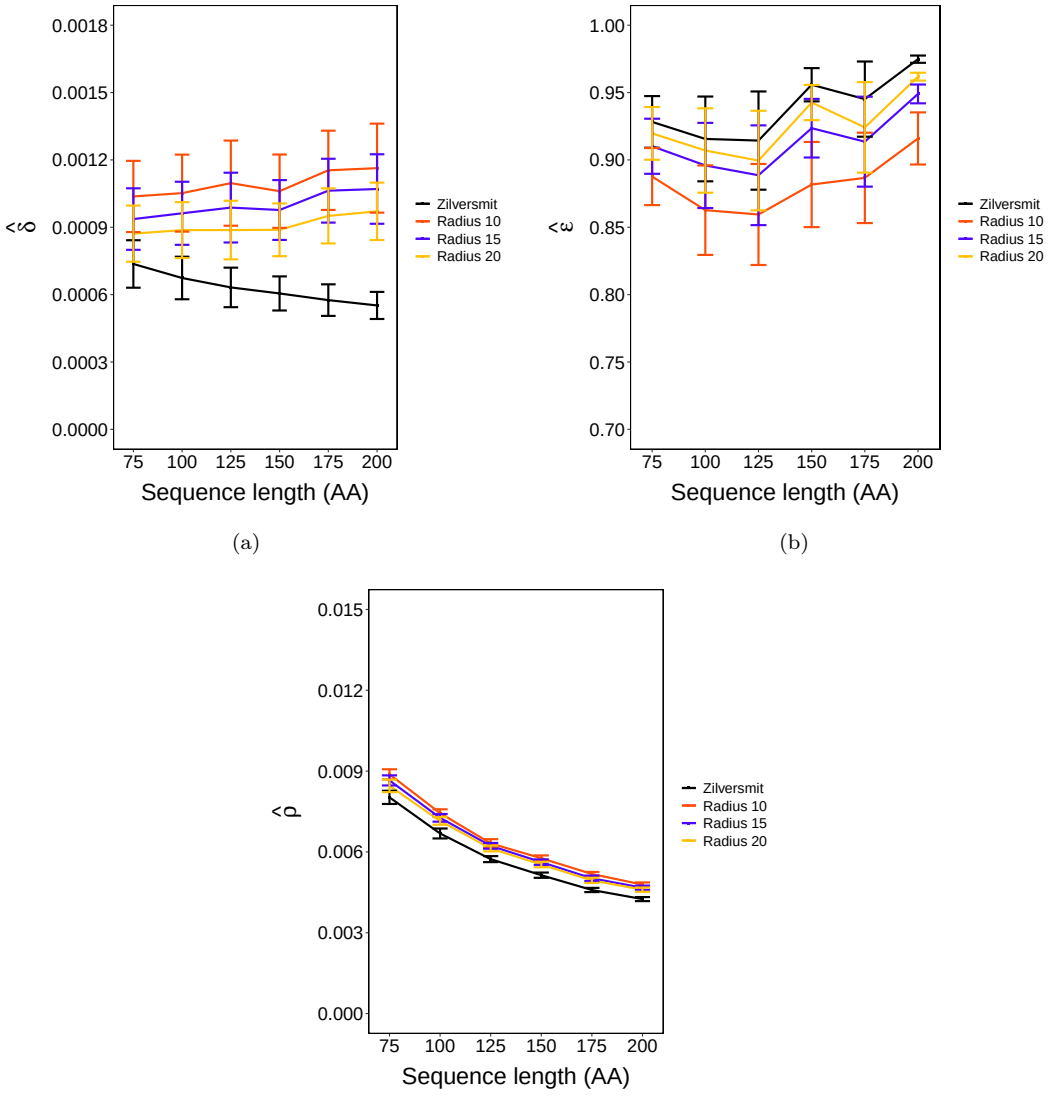
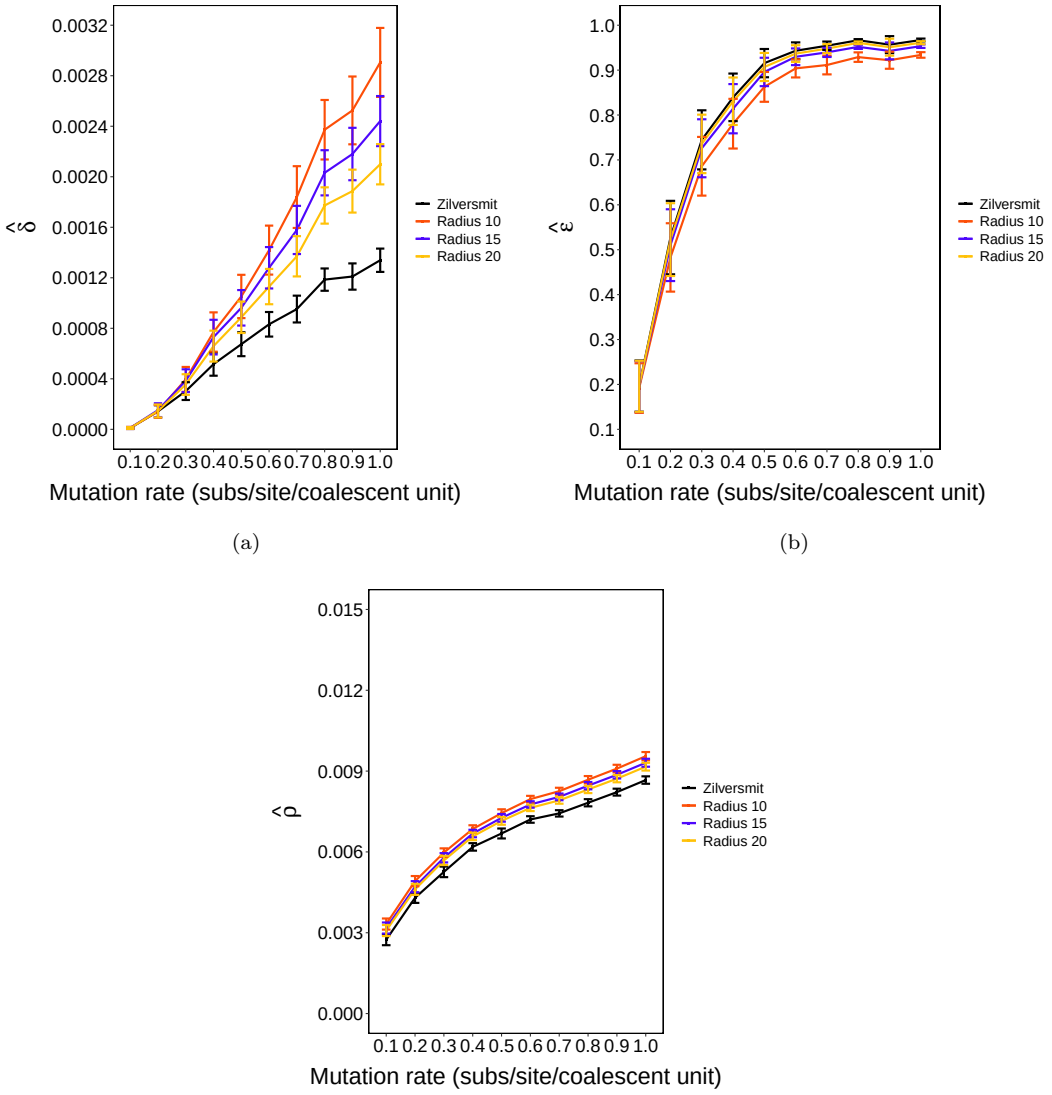


FIGURE 4.28: **Mean (with 95% confidence intervals) of each estimated parameter for varying sequence length using the JHMM and improved JHMM.**



(a)

(b)

(c)

FIGURE 4.29: **Mean (with 95% confidence intervals) of each estimated parameter for varying mutation rate using the JHMM and improved JHMM.**

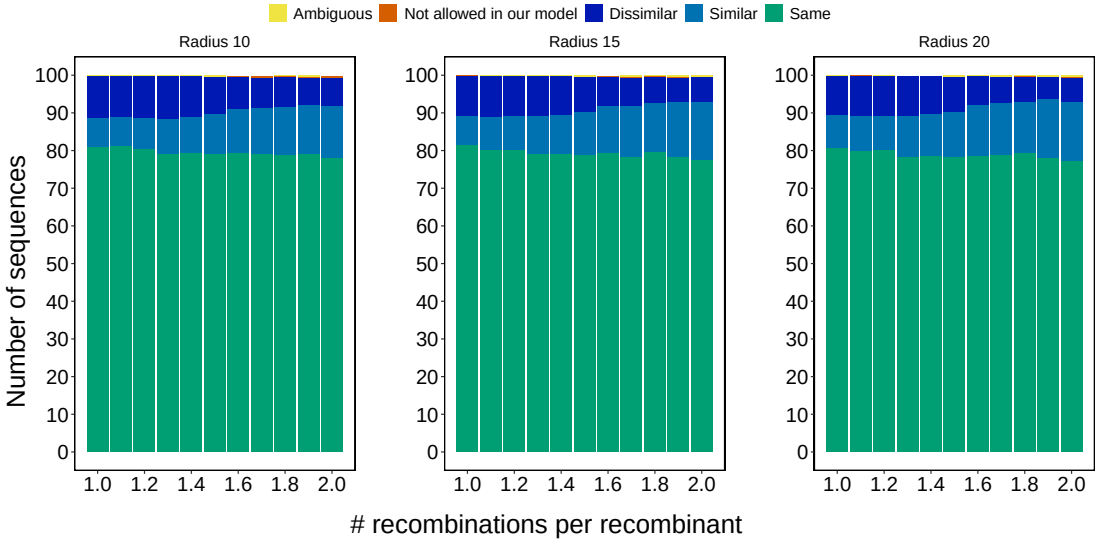


FIGURE 4.30: **Viterbi path comparison between the JHMM and improved JHMM when varying average number of recombinations per recombinant sequence.**

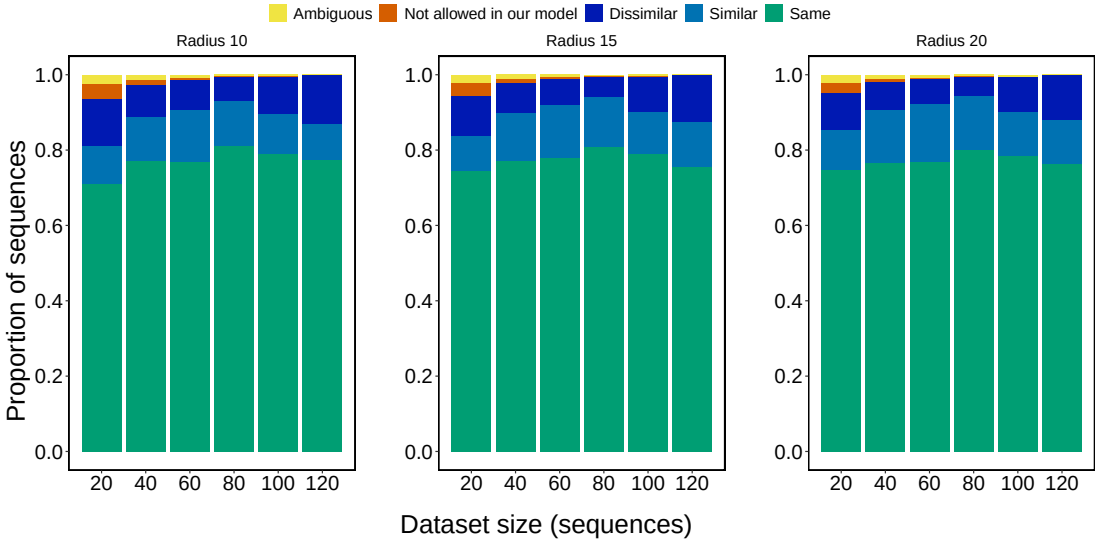


FIGURE 4.31: **Viterbi path comparison between the JHMM and improved JHMM when varying dataset size.**

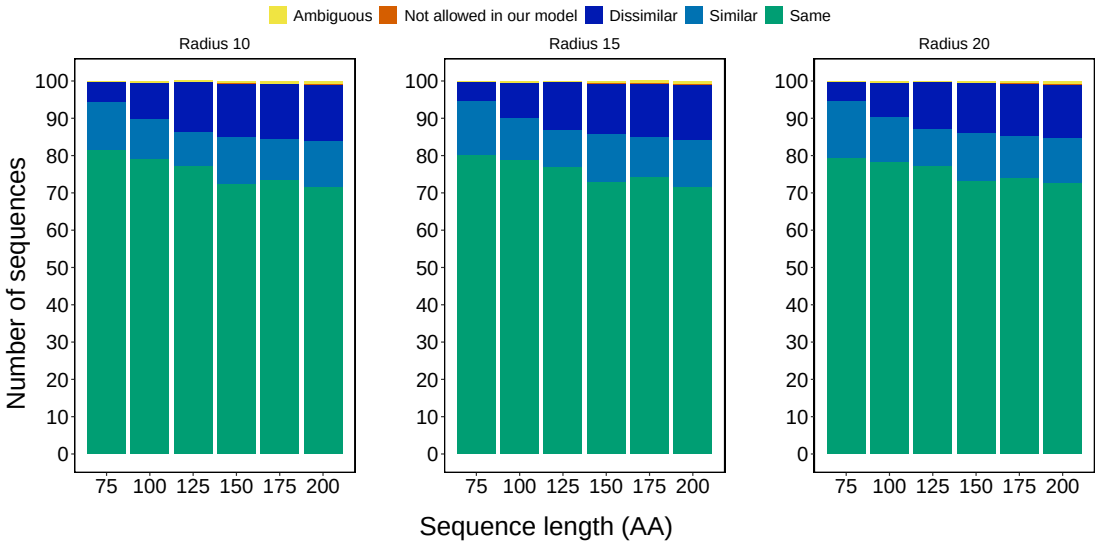


FIGURE 4.32: **Viterbi path comparison between the JHMM and improved JHMM when varying sequence length.**

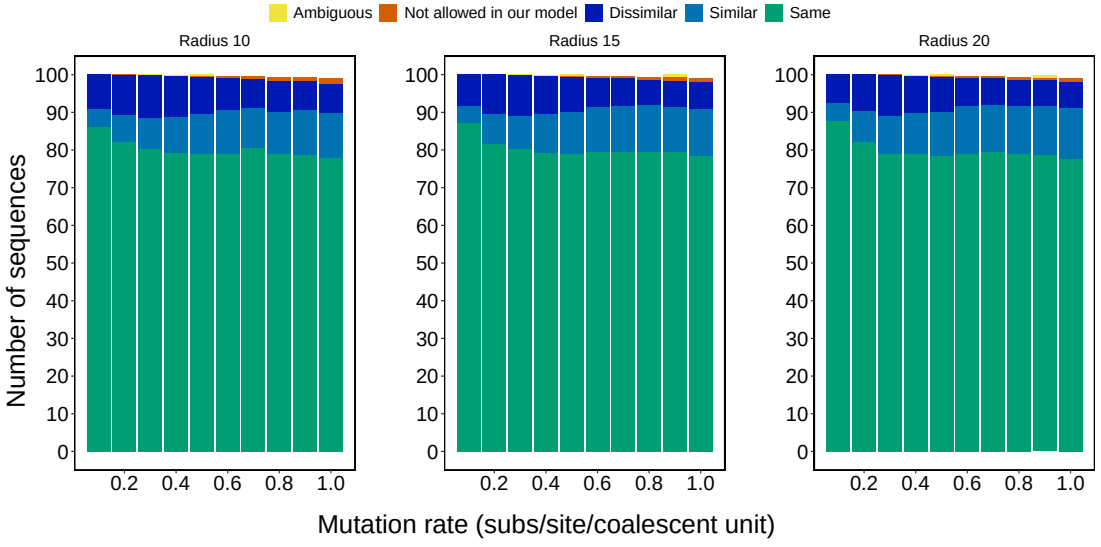


FIGURE 4.33: **Viterbi path comparison between the JHMM and improved JHMM when varying mutation rate.**

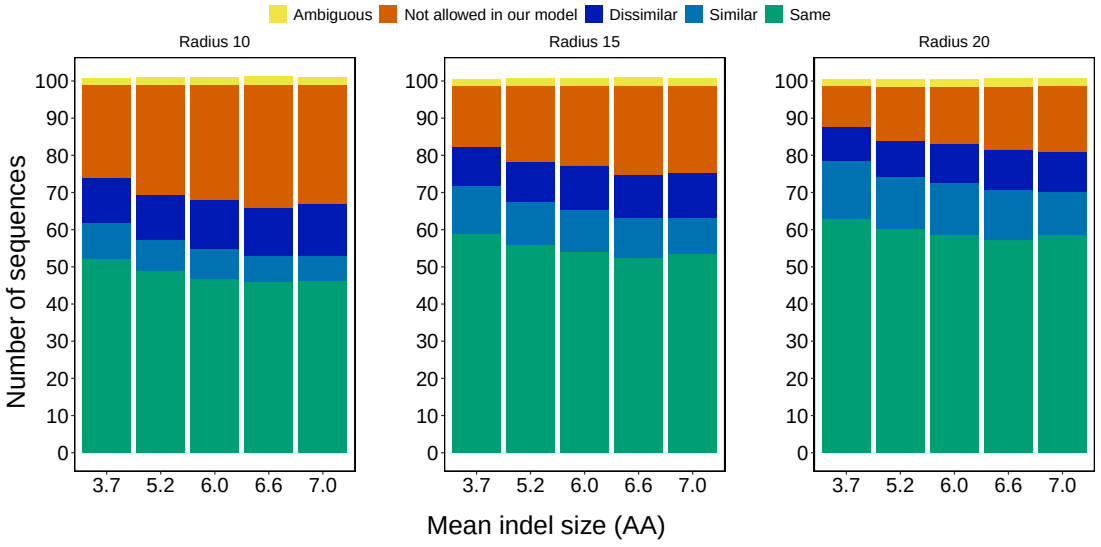


FIGURE 4.34: **Viterbi path comparison between the JHMM and improved JHMM when varying indel size.**

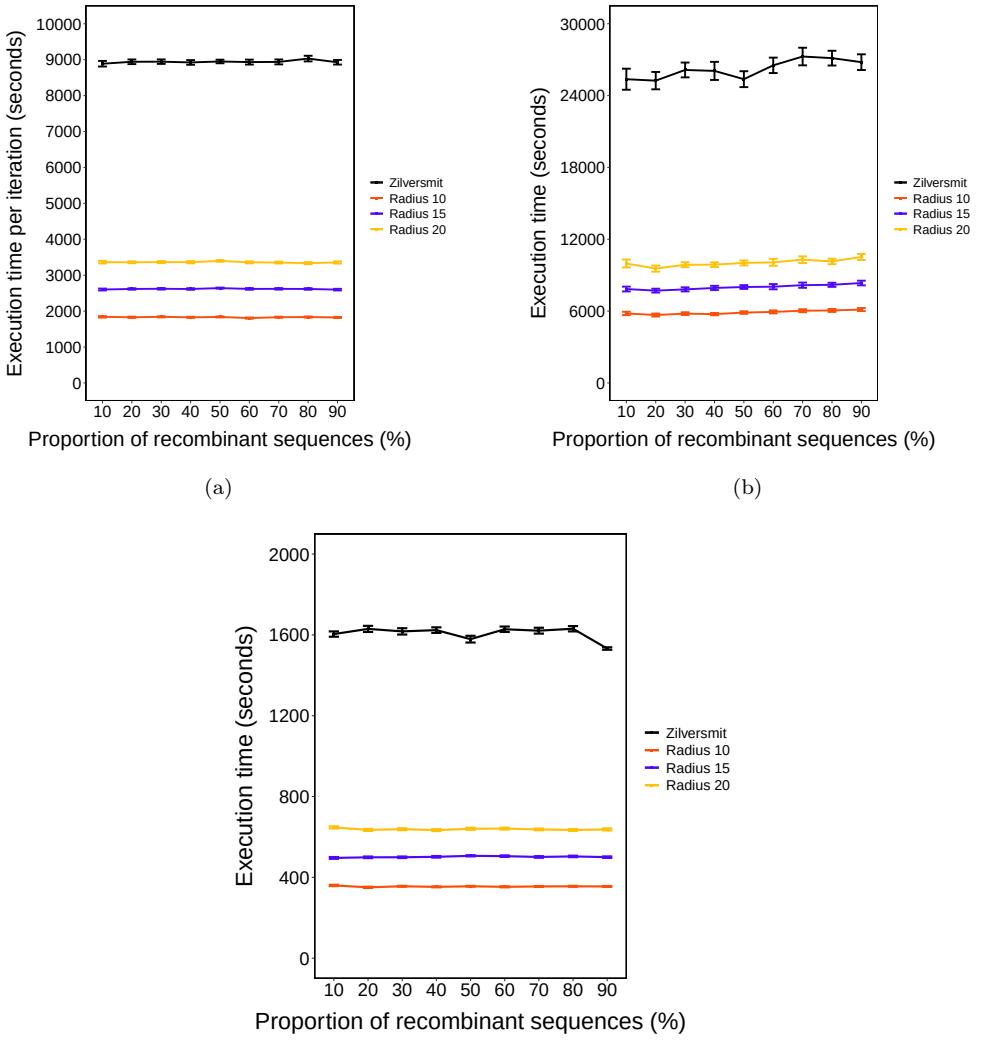


FIGURE 4.35: **Time taken by each model for (a) Baum-Welch algorithm, (b) estimating ρ, and (c) generating Viterbi paths when varying proportions of recombinant sequence.**

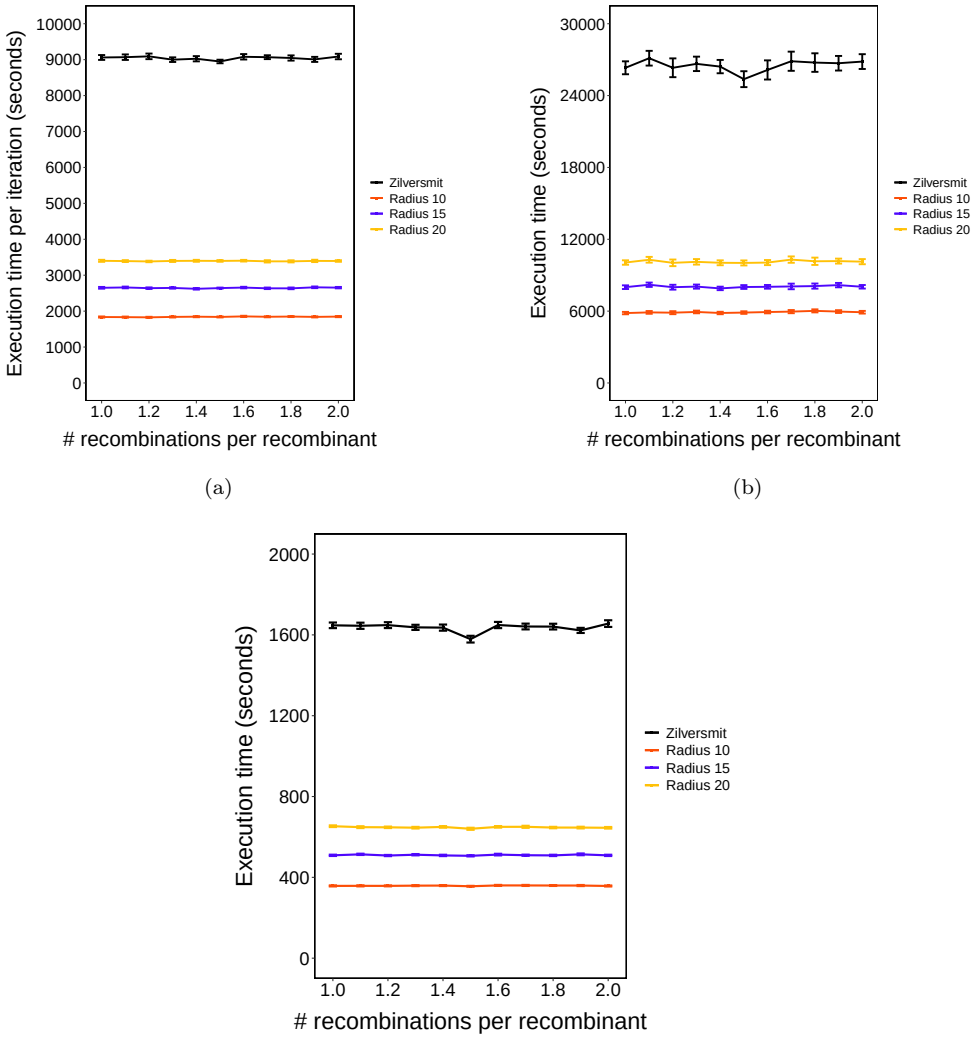


FIGURE 4.36: **Time taken by each model for (a) Baum-Welch algorithm, (b) estimating ρ, and (c) generating Viterbi paths when varying average number of recombinations per recombinant sequence.**

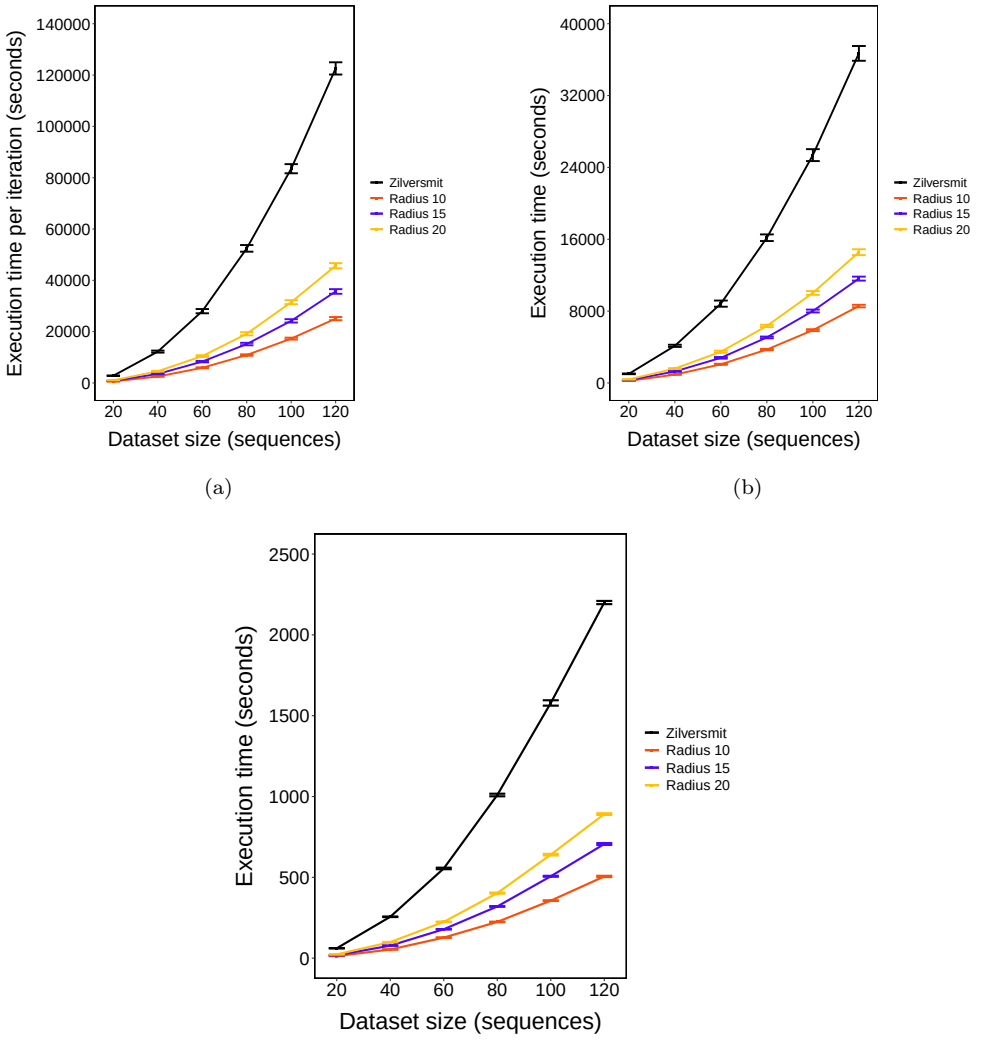


FIGURE 4.37: **Time taken by each model for (a) Baum-Welch algorithm, (b) estimating ρ, and (c) generating Viterbi paths when varying dataset size.**

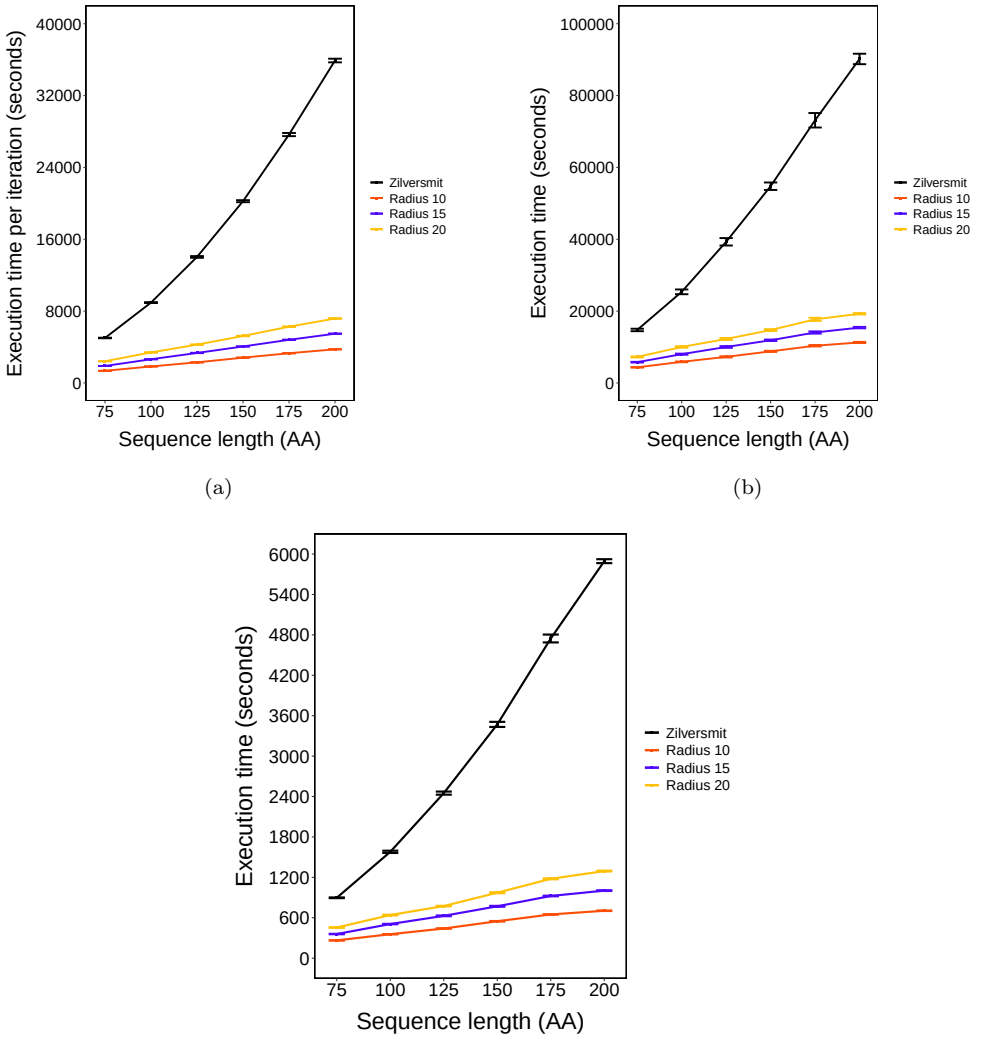


FIGURE 4.38: **Time taken by each model for (a) Baum-Welch algorithm, (b) estimating ρ, and (c) generating Viterbi paths when varying sequence length.**

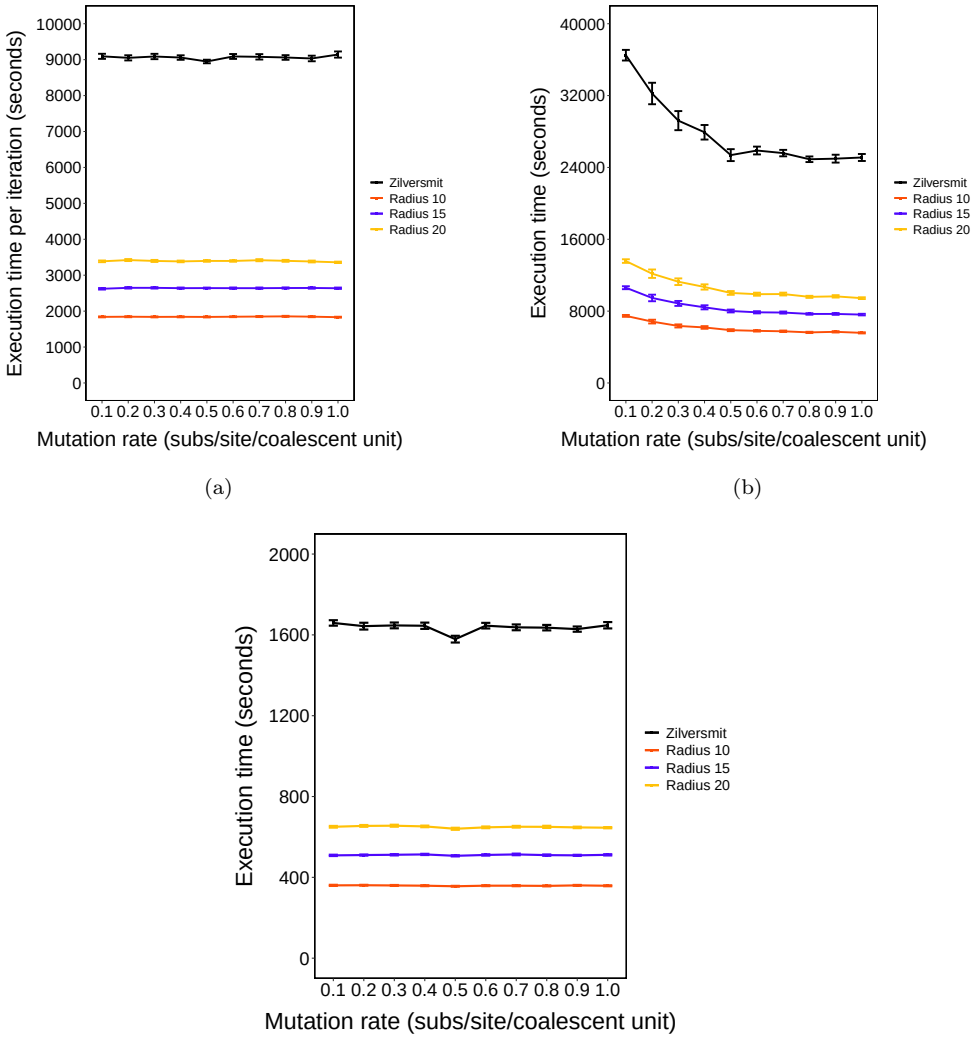


FIGURE 4.39: **Time taken by each model for (a) Baum-Welch algorithm, (b) estimating ρ, and (c) generating Viterbi paths when varying mutation rate.**

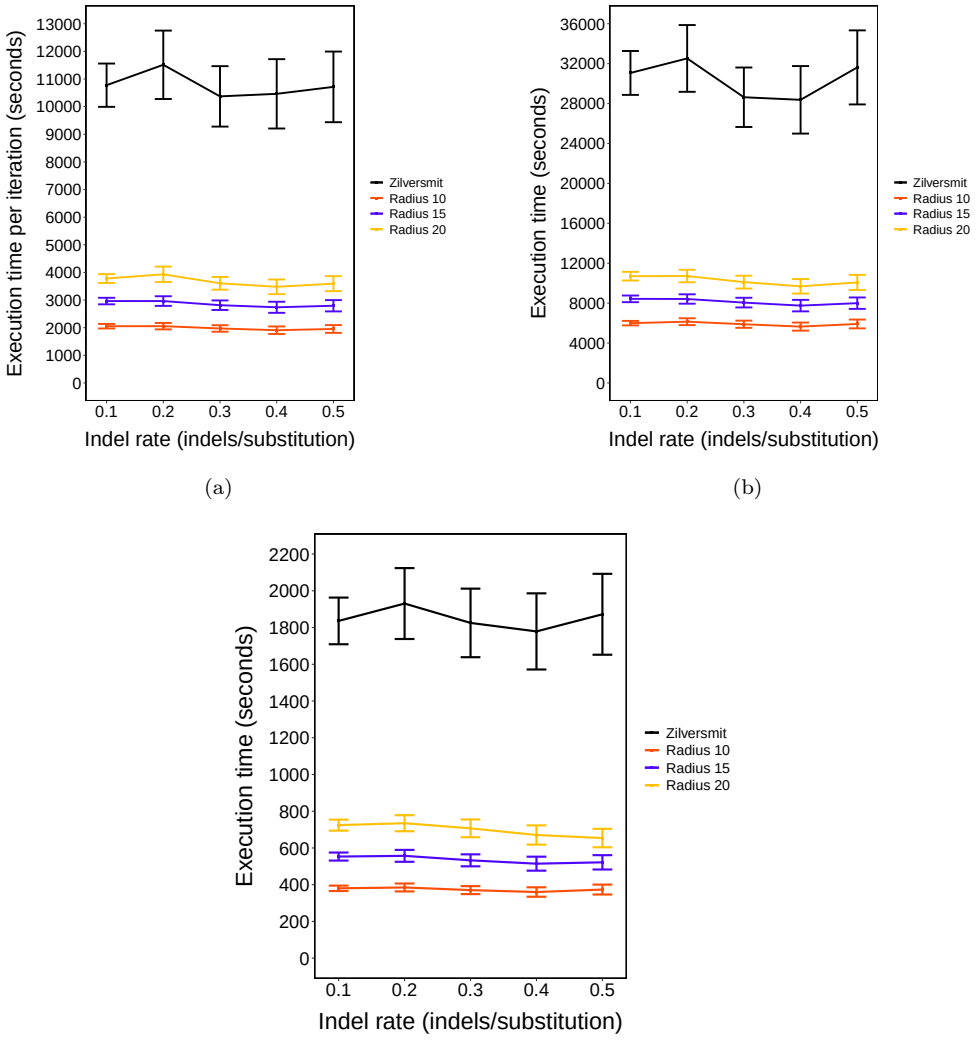


FIGURE 4.40: **Time taken by each model for (a) Baum-Welch algorithm, (b) estimating ρ, and (c) generating Viterbi paths when varying indel rate.**

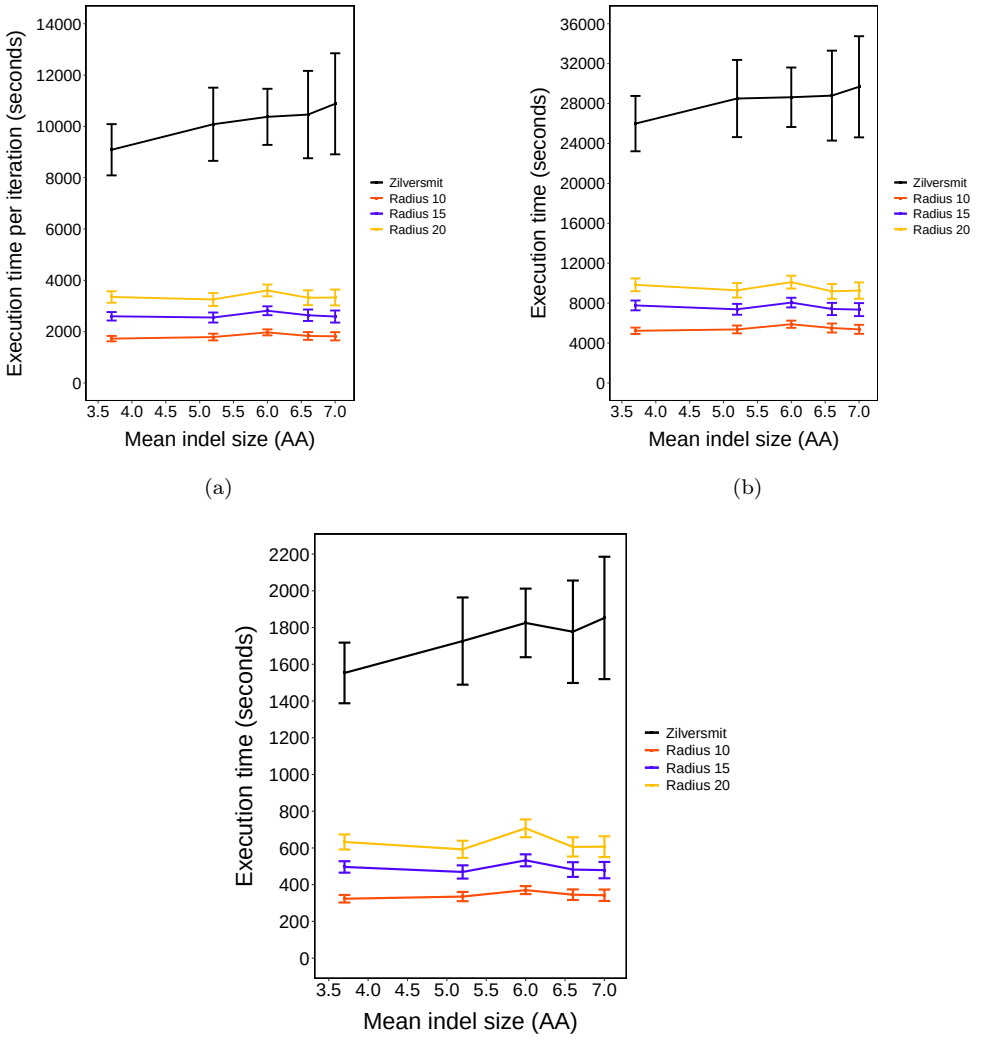


FIGURE 4.41: **Time taken by each model for (a) Baum-Welch algorithm, (b) estimating ρ, and (c) generating Viterbi paths when varying mean indel size.**

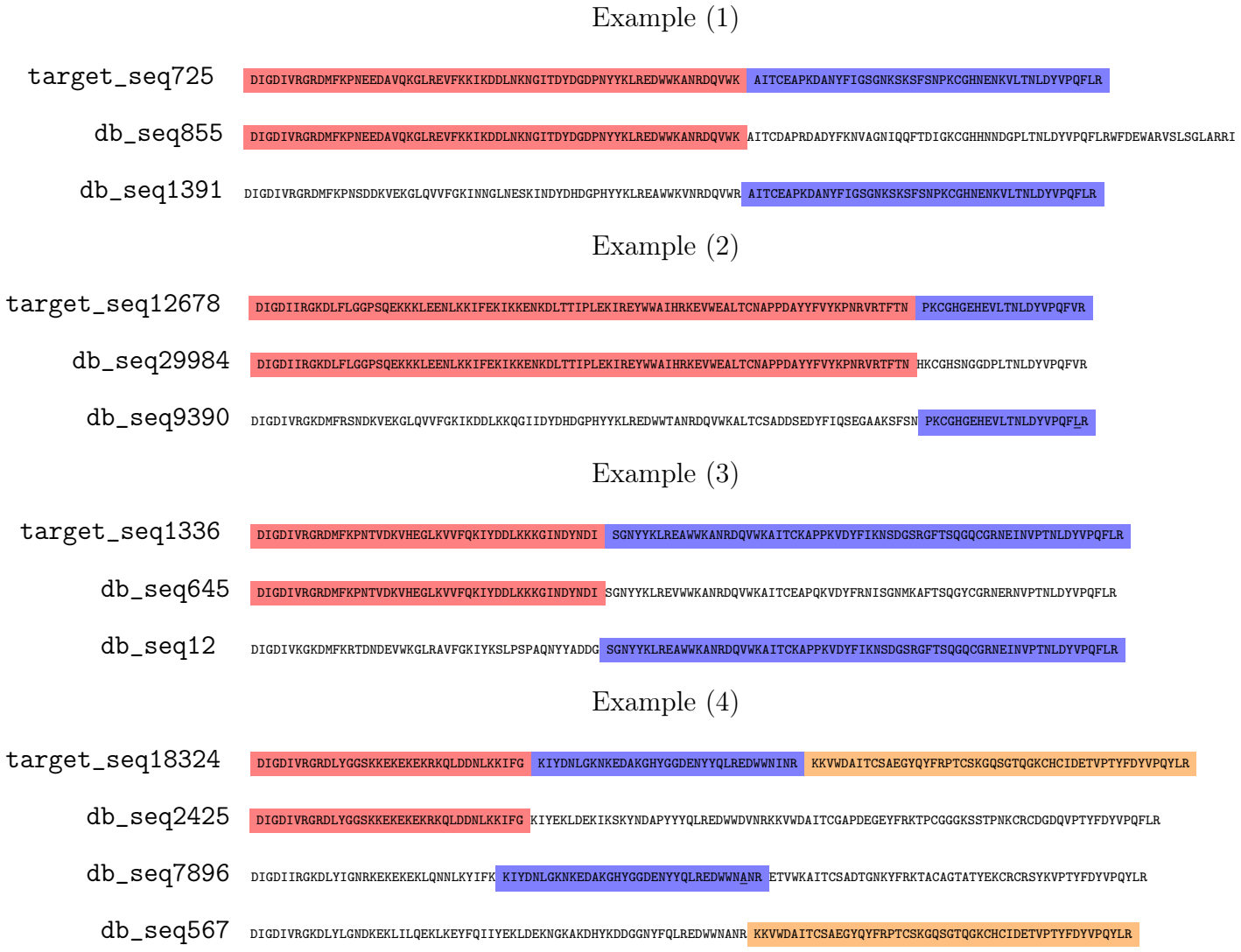


FIGURE 4.42: **Four example mosaic representations of improved JHMM using real datasets.** These four mosaic representations are arbitrarily selected from the results using S1 (upsA)-S4 (upsA), S1-S4, S4 (upsA)-S5 (upsA) and S4-S5 datasets in order. In each example, the first row is the inferred recombinant, and subsequent rows are the inferred parental sequences. The contributing segments are highlighted by colors. When the amino acid differs from the parent and the child, we underline it in the parent.

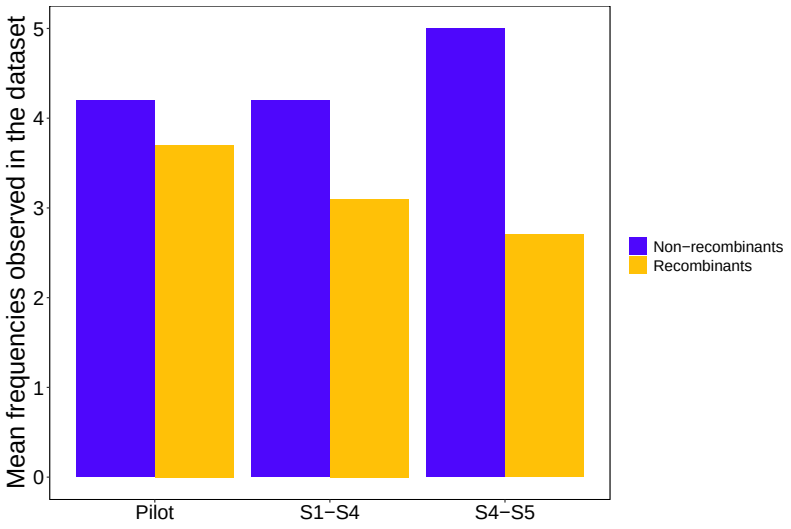


FIGURE 4.43: **Mean conservation level of recombinant and non-recombinant DBLα types from different datasets.** Conservation level is the number of isolates that the DBLα type is present. The recombinants and non-recombinants of each dataset were identified from the mosaic representations separately.

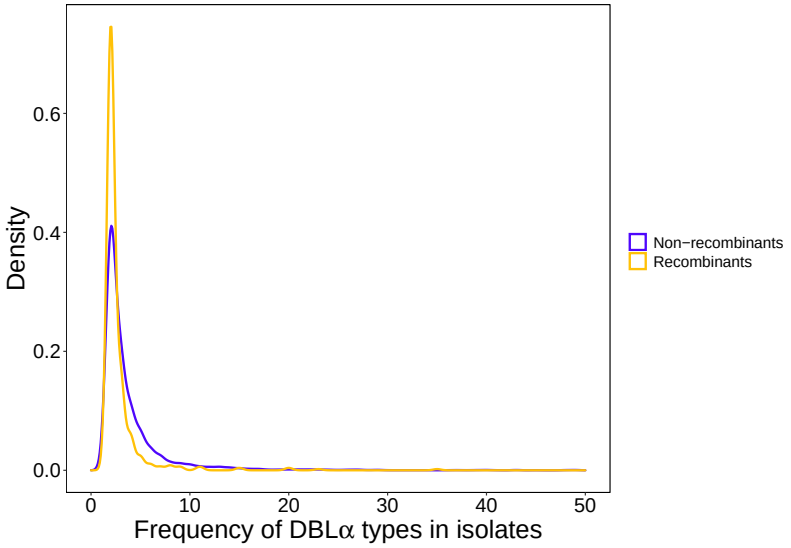


FIGURE 4.44: **Frequency of DBLα types in the isolates of S4.** The recombinants and non-recombinants of S4 were identified from the mosaic representations using S1-S4 data.

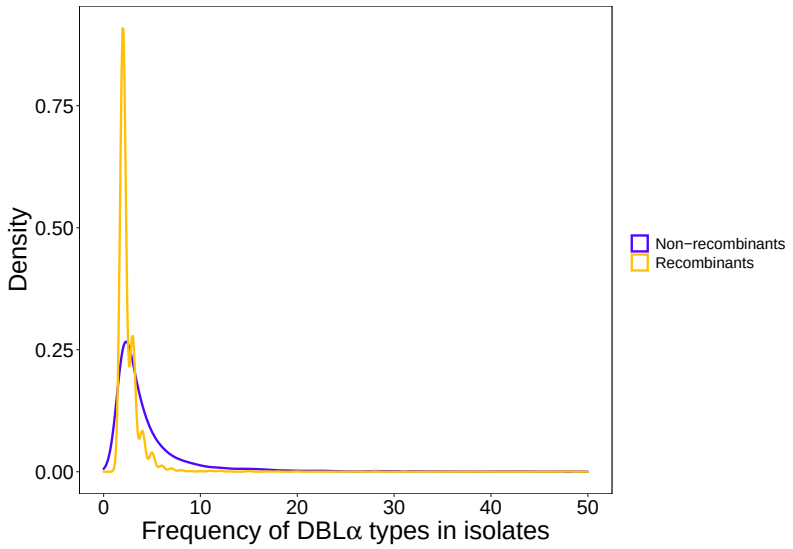


FIGURE 4.45: **Frequency of DBLα types in the isolates of S5.** The recombinants and non-recombinants of S5 were identified from the mosaic representations using S4-S5 data.

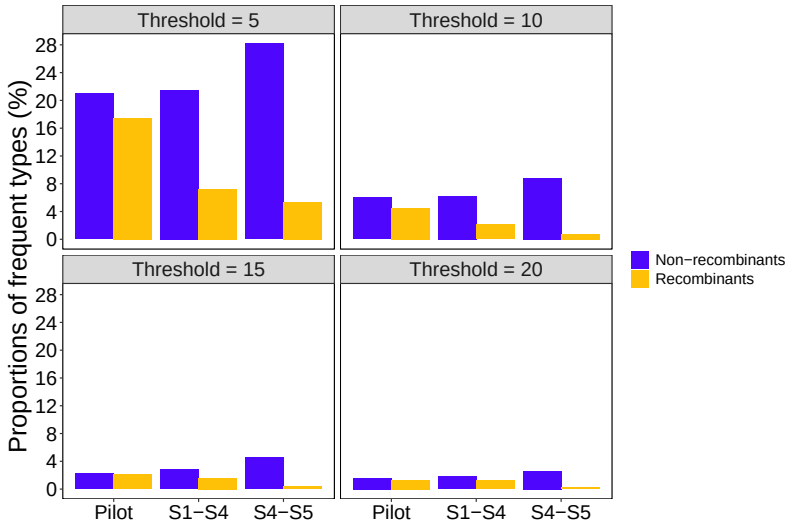


FIGURE 4.46: **Proportions of frequent DBLα types in the recombinants and non-recombinants among various datasets.** Frequent DBLα types were defined by thresholding the number of isolates. The recombinants and non-recombinants of each dataset were identified from the mosaic representations separately.

Chapter 5

Classifying the malaria parasite *var* genes into ups groups

5.1 Background

Plasmodium falciparum [27–29] is the deadliest human malaria parasite [5] and has developed resistance to nearly all available anti-malaria drugs [37]. This is at least in part because of its key virulence factor, the protein [223] called *Plasmodium falciparum* erythrocyte membrane protein 1 (PfEMP1). PfEMP1 acts as both an adhesion and antigen protein. It is expressed on the membrane surface of red blood cells. *P. falciparum*-infected erythrocytes (IEs) could bind to other uninfected red blood cells and cause the dysfunction of vital organs like the brain or placenta [224]. PfEMP1 is encoded by hyper-variable *var* genes, and each parasite genome contains around 60 *var* genes. Mutually exclusive expression of these genes [225] aids the parasites in evading the detection of human immune system [226] and may lead to waves of parasitemia. Therefore, studying the *var* genes has always been of major interest especially for malariologists.

Based on the similarity of upstream sequences (ups) and chromosomal positions, *var* genes are classified into four major groups (upsA, upsB, upsC and upsD) [45, 117, 118] and an upsE group consisting of *Var2CSA* gene (distantly related to other *var* genes) with unique domain structure [50]. UpsD has been grouped into upsA later due to the high similarity observed in a phylogenetic study [61].

Ups groups are strongly associated with various chromosome locations, transcription orientations [118], and importantly, disease severities. Specifically, upsA *var* genes are found in subtelomeric regions and transcribed towards the telomere. They are preferentially involved in the pathogenesis of severe and cerebral malaria in children [227–229]. UpsB *var* genes are located in subtelomeric or central chromosome regions but transcribed in the opposite orientation to upsA *var* genes. A large proportion of upsB transcripts are associated with clinically mild (symptomatic) and severe malaria cases [230]. All upsC *var* genes are in central chromosomal areas, and predominant in African children with asymptomatic malaria infections [230]. Classifying the *var* genes to ups groups is thus crucial for the prevention and diagnosis of malaria.

To develop a method for classifying *var* genes into ups groups, we focus on the conserved region in DBLα domain of the *var* gene called DBLα tag. Each DBLα tag is predominantly found in a unique *var* gene [231]. While there are limited ups sequences which have been published [44], a large number of DBLα tags are publicly available [46, 71, 74, 84, 85] (even from a global scale [44, 62]).

In the past two decades, the DBLα domain has been a popular marker for population genetic studies of *var* genes. Nearly all *var* genes (except *var2CSA* [49–51]) encode a DBLα domain, moreover, this domain has been found to be immunogenic [67] and a potential target for vaccination [68]. Rask et al. [46] retrieved *var* genes from seven genomes and classified the DBLα domain of *var* genes into DBLα0, DBLα1 and DBLα2 three groups. Based on the tree analysis, sequences from DBLα0 were further grouped into 24 subclasses (DBLα0.1–24) and sequences from DBLα1 were grouped into 8 subclasses (DBLα1.1–8). Taken altogether, Rask et al. [46] identified 33 DBLα subclasses.

The only existing method for classifying ups groups was developed by Ruybal-Pesántez et al. [74]. They classified each DBLα tag to its most probable DBLα subclass using the reference from [46]. If a sequence was assigned to DBLα1, the method [74] classified it to upsA group; otherwise, it was assigned to DBLα0 or DBLα2, then upsB/C group was classified. The limitation of this approach is that it can only classify tags into two categories, upsA and non-upsA. Their results do not show a clear separation of upsB vs upsC based on DBLα subclasses.

In this chapter, we develop two algorithms to classify each DBLα tag into upsA, upsB and upsC three groups, with the help of the most up-to-date and the largest

reference database from Otto et al. [44]. Our first algorithm searches each query DBLα tag against the reference database with known ups groups using BLASTP [220]. Each query tag is then classified (or assigned) to the ups group of its closest reference tag. However, one of its drawbacks is that it classifies the tag into an ups group explicitly. To overcome this, we propose another algorithm which uses the profile hidden Markov model (HMM) [156]. This approach provides probabilities of membership to three ups groups, so it helps making informed decisions of inference or setting a threshold.

We show our two algorithms perform well through cross-validation. We also compare these two with the existing method [74]. Results indicate that the overall accuracy is similar between our two algorithms, and they are more accurate than the existing method. Moreover, for our profile HMM-based method, setting a threshold on the inferred probabilities further improves the accuracy but at the cost of less classified tags. In the following, we introduce our algorithms.

5.2 Methods

We firstly use BLASTP to search each DBLα tag to the reference database of DBLα tags (with known ups groups). We classify each tag to the ups group of its closest reference tag. This BLASTP-based method provides an explicit classification for each query DBLα tag. Since this method is straightforward, we introduce our profile HMM-based method as below.

We propose a probabilistic method to classify the *var* genes into ups groups using the DBLα tags. It takes as input a reference database of DBLα tags with known ups groups and a set of DBLα tags to be classified, and outputs the probabilities of membership to all three ups groups per tag. Compared with the existing method, the main improvements in our method are (1) the ability to classify a tag to all three ups groups (upsA, upsB and upsC) rather than traditional either upsA or non-upsA, (2) the inferred probabilities to three ups groups.

The method consists of the following three steps:

1. Each DBLα tag belongs to one of 33 DBLα subclasses and one of 3 ups groups. We divide the reference DBLα tags into categories, where tags in each category have the same subclass and ups group.

2. We fit a profile hidden Markov model (HMM) [156] as a representation of the tags in each category. We then calculate the likelihood of a new tag (to be classified) for each category conditional on the estimated HMM parameters.

3. With Bayes' theorem, we infer the posterior probabilities, i.e., the new tag's assignment probabilities to the three ups groups.

We discuss each step in detail in the following sections.

5.2.1 Dividing reference database into categories

Since each reference tag belongs to one DBLα subclass and one ups group, we divide all reference tags into categories, where each category refers to a set of reference tags belonging to one combination of the 33 DBLα subclasses and 3 ups groups (Figure 5.1).

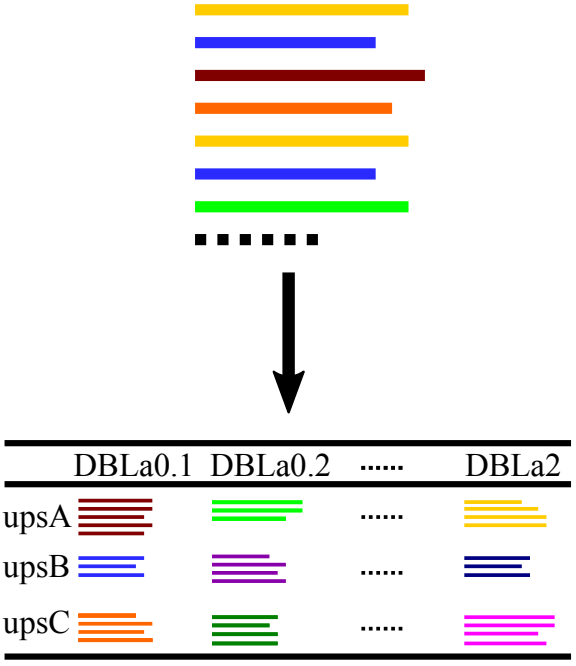


FIGURE 5.1: **An overview of dividing reference tags into categories.** From an input set of reference tags (represented by colored horizontal lines), we divide them into categories, where tags in each category belong to both one of the DBLα subclasses and one of the ups groups.

5.2.2 Fitting a profile HMM for each category

The profile hidden Markov model (profile HMM) [156] is a hidden Markov model well suited for modelling multiple sequences. It takes as input of a multiple alignment of a set of sequences belonging to a sequence family (with similar functions), and represents the consensus sequence in this family [148] instead of any particular sequence in it. This model has been widely used for measuring how likely a sequence is generated from a sequence family and become very popular in current molecular biology [232–239]. In this step, we fit the hidden Markov model for each category of reference database. Its rationale is that, using the existing forward algorithm [148] of HMM, we can calculate the likelihood of a new sequence to be classified to any category. These likelihoods are used for the next step.

The profile HMM provides the position-specific information of the input alignment. Sequences in a family are subject to variation in the symbols (amino acid for protein sequences; nucleotide for DNA sequences) at each position (column) of alignment and gaps within the alignment. To model this, the profile HMM uses a probabilistic way to describe which symbols are likely to be observed and how frequently insertions/deletions occur at each column of the sequence alignment.

Specifically, the profile HMM uses a repetitive structure of hidden states, see Figure 5.2 for an example architecture. For each consensus column (where most sequences have symbols) of the alignment, we have match (M), insert (I) and delete (D) three hidden states. In a length L profile HMM,

- the lth match state M_l emits a symbol from the lth position's emission probability distribution over all symbols;
- the lth insert state I_l emits a symbol using this position's background emission probability distribution;
- the lth delete state D_l emits nothing, it represents a symbol at this position is skipped.

These hidden states are connected by transitions. In practice, the transition between the insert and delete states are usually prohibited [240], as shown in Figure 5.2. The possible transition from a hidden state to another is described by the transition probabilities of that position.

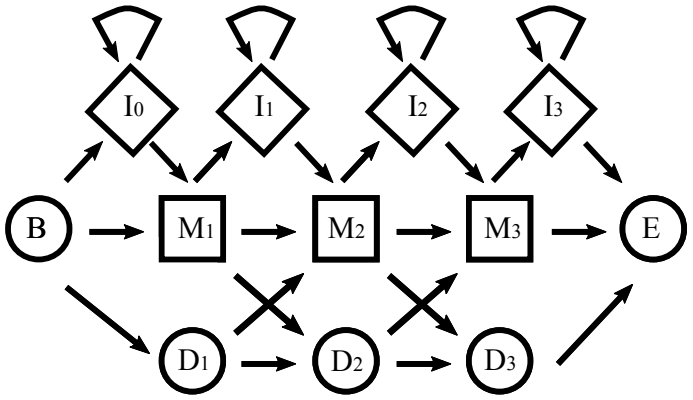


FIGURE 5.2: **An example structure of profile HMM.** This figure is a minor modification of Figure 5.2 from [148]. This length 3 profile HMM starts from the begin (B) and terminates at end (E) state. Inside the model, we denote match state (M), insert state (I) and delete state (D) as squares, diamonds and circles separately. Arrows represent the possible transitions among hidden states.

Building a profile HMM would require a suitable model length L and all the probabilities (emission probabilities from each match and insert state; transition probabilities). The procedure of training the profile HMM is to maximize the likelihood of the input data given all these parameters. We briefly introduce how we estimated parameters below, see [148] for an in-depth discussion.

The key to determining the model length L is to detect which alignment columns refer to M state, since the number of match states is defined as L. A straightforward way is introduced in Durbin et al. [148], i.e., columns where the alignment contains less than half gaps are assigned to match states, and the remaining columns are inserts. Regarding the probabilities, we took the maximum likelihood estimation of parameters by counting the number of times of each state transition or symbol emission and computed the relative empirical frequencies. In addition, due to the absence of some transitions and emissions in the alignment, we added the number 1 as the *pseudocounts* to the actual counts to avoid zero probabilities.

Here Clustal Omega v1.2.4 [241, 242] was used for aligning reference DBLα tags. To build a profile HMM for each category of the reference database, we next ran the `hmmbuild` command in HMMER v3.2.1 [240] with default settings. We finally used a custom script to implement the forward algorithm, and computed the likelihood of each tag to be classified under the profile HMM parameters for each category.

5.2.3 Calculating the assignment probabilities of each sequence

In this step, we calculate the posterior (assignment) distribution of ups groups for a given DBLα tag. Consider a new DBLα tag X to be classified, and let G_p ($p = 1, 2, 3$) denote one of 3 ups groups, D_q ($q = 1, 2, ..., 33$) denote one of 33 DBLα subclasses. With Bayes' theorem, we estimate the posterior probability of DBLα tag X to the ups group G_p:

$$\begin{aligned} \Pr(G_p \mid X) &= \frac{\sum_{q=1}^{33} \Pr(X \mid G_p, D_q) \Pr(G_p, D_q)}{\Pr(X)} \\ &\propto \sum_{q=1}^{33} \Pr(X \mid G_p, D_q) Pr(D_q) \Pr(G_p \mid D_q) \end{aligned} \tag{5.1}$$

Since the summation of posterior probabilities to three ups groups is 1, we only need to compute the $\Pr(X \mid G_p, D_q)$, $\Pr(D_q)$ and $\Pr(G_p \mid D_q)$. $\Pr(X \mid G_p, D_q)$ is the likelihood of X given the profile HMM of category G_p, D_q, which we have calculated in the previous step; for $\Pr(D_q)$ and $\Pr(G_p \mid D_q)$, we use empirical distributions of reference database to compute this prior.

There are two ways to classify sequences using these probabilities. A direct method is classifying each sequence to the ups group with the highest posterior probability. We get the classified ups group for each sequence. An alternative method is to establish assignment probability thresholds. A sequence's ups group is annotated as 'known' if its maximum probability is greater than a specified threshold; otherwise, its ups group is annotated as 'unknown'. In this way, we focus on the sequences with more confident classification results.

5.2.4 Measuring algorithm accuracy

To evaluate the accuracy of our BLASTP-based and profile HMM-based algorithms, we adopted leave-one-out cross-validation on the reference database. Our reference database consists of 846 DBLα tags with known ups group for each tag (148 upsA, 502 upsB and 196 upsC). See Supplementary Section 5.5.1 for details about this reference database. We used two standard statistical measures to assess the accuracy of methods, sensitivity and precision. Sensitivity is the proportion of correctly

predicted tags, and precision is the proportion of correctly classified tags among predictions.

We first classified each tag to the ups group with the highest posterior probability. By comparing the predicted and actual results, we obtained a confusion matrix. We presented the results of accuracy measures using this matrix. We also established the probability thresholds, and demonstrated how various threshold values affected the accuracy of our profile HMM-based method.

Comparison with Ruybal-Pesántez's method For comparing our two algorithms with the previous method [74], we converted the prediction result into a probability vector for each sequence to have a uniform output format across methods. For instance, if Ruybal-Pesántez's method [74] classified a sequence as non-upsA, we set the output vector as $(0, 0.5, 0.5)$, where the three values represent the probabilities of belonging to the upsA, upsB, and upsC groups, in that order. Likewise, if the BLASTP-based method classified this sequence as upsC, we used the vector $(0, 0, 1)$.

To evaluate the accuracy of classifiers with the above output, we used measures of overall accuracy oa, the κ and root mean square error ($RMSE$). Larger oa or κ values and smaller $RMSE$ indicate greater accuracy. All these assessment indices were calculated from the extended traditional confusion matrix called sub-pixel confusion matrix (SCM) [243]. See Supplementary Section 5.5.2 for details about how to calculate the SCM and related equations of indices.

5.3 Results

5.3.1 Classifying the tags with the highest probability

We show the confusion matrix of BLASTP-based and profile HMM-based methods in Table 5.1 and 5.2, and the accuracy measures of these methods are presented in Table 5.3. Overall, our methods perform very well with an accuracy (weighted mean in the table) of 76.7% for the BLASTP-based method and 75.2% for the profile HMM-based method. Furthermore, both two methods show that the upsA group has the highest sensitivity, followed by the upsB or upsC group; the precision is over

80% in the upsA and upsB groups and slightly lower in the upsC group. In general, the accuracy of our two methods is similar.

TABLE 5.1: **Confusion matrix using BLASTP-based method.** The rows represent the predicted ups groups, and columns represent the real ups groups.

		Real			
		UpsA	UpsB	UpsC	Total
Predicted	UpsA	**145**	3	0	148
	UpsB	3	**382**	74	459
	UpsC	0	117	**122**	239
	Total	148	502	196	846

TABLE 5.2: **Confusion matrix using profile HMM-based method.** The rows represent the predicted ups groups, and columns represent the real ups groups.

		Real			
		UpsA	UpsB	UpsC	Total
Predicted	UpsA	**145**	1	1	147
	UpsB	3	**350**	54	407
	UpsC	0	151	**141**	292
	Total	148	502	196	846

TABLE 5.3: **Accuracy of BLASTP-based and profile HMM-based methods.** The weighted mean is the mean weighted by the real ups group sizes for sensitivity and by the predicted ups group sizes for precision.

Method	Measure	UpsA	UpsB	UpsC	Weighted mean
BLASTP-based	Sensitivity	98.0%	76.1%	62.2%	76.7%
	Precision	98.0%	83.2%	51.0%	76.7%
Profile HMM-based	Sensitivity	98.0%	69.7%	71.9%	75.2%
	Precision	98.6%	86.0%	48.3%	75.2%

5.3.2 Setting a threshold on the probabilities

We found the distribution of highest posterior probabilities of tags is highly left-skewed and most values are distributed in $[0.95, 1]$ (Supplementary Figure 5.5). To study how our method's accuracy varies with threshold, we vary the threshold to $1 - \exp(-x)$, the minimum, increment and maximum value of x is 0, 0.5 and 25. For each ups group, we presented the proportion of classified tags and calculated the sensitivity and precision using different threshold values.

As anticipated, the proportion of overall classified DBLα tags decreased with increasing thresholds (Figure 5.3). The upsA group experienced the slowest decline in the proportion of classified tags, while the upsB group experienced the greatest decline. This indicated that setting a too-high threshold would result in a relatively small number of classified tags, particularly non-upsA DBLα tags.

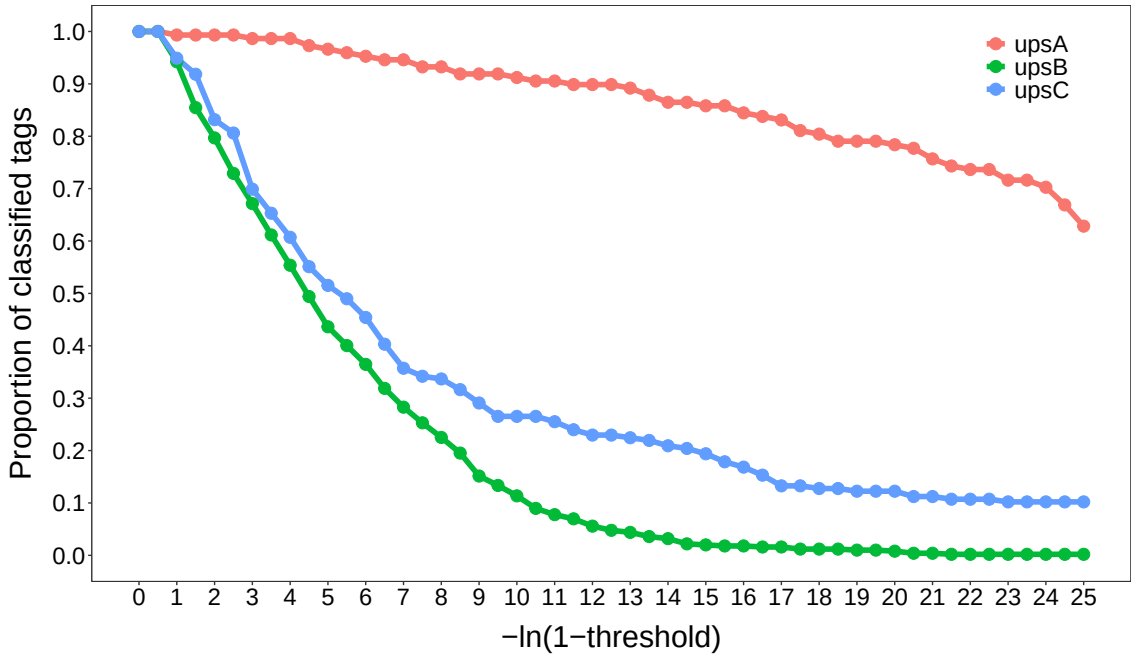


FIGURE 5.3: **Proportion of classified DBLα tags with threshold.**

We then calculated the sensitivity and precision for three ups groups at each threshold value (Figure 5.4). The sensitivity and precision were both the highest (over 97.0%) for the upsA group at nearly every threshold. This suggested that it is easier to distinguish upsA and non-upsA groups than upsA, upsB, upsC three groups. In the upsC group, sensitivity and specificity increased steadily with minor fluctuations before reaching the peak (100%). However, setting a higher threshold cannot make the detection of each ups group more accurate. Notably, the upsB group showed an initial increase in both accuracy measures until the threshold was around $1 - \exp(-8)$, after which there was a rapid decline.

In conclusion, setting a threshold on inferred probabilities would affect the number of tags classified and the accuracy of each ups group. Selecting a too-high threshold is advantageous for the upsA and upsC groups' accuracy, but not for upsB and the total number of classified tags. Conversely, setting a too-low threshold would reduce the accuracy of upsB and upsC groups, particularly the precision of upsC, while increasing the number of classified upsB and upsC tags. An example threshold we

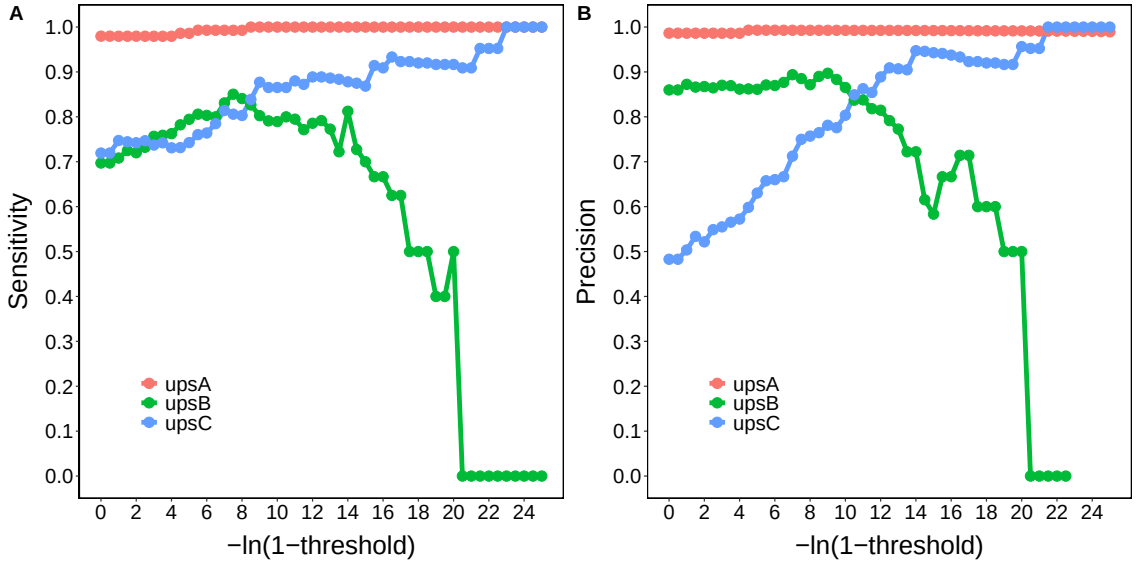


FIGURE 5.4: **Accuracy of profile HMM-based method with threshold.** Note that the precision of upsB group is not computed when threshold is larger than $1 - \exp(-22.5)$, this is due to no classified sequences since then.

recommend is $1 - \exp(-8)$, where the upsB group accuracy is nearly the highest and the upsC group accuracy is over 75.0%. This results in 37.5% of classified tags (92.6%, 22.5%, 33.7% of classified tags for upsA, B and C group separately). Supplementary Table 5.9 shows the number of classified tags in each ups group with this threshold. Moreover, this study suggests that choosing an appropriate threshold depends on the user's preferences.

Finally, in terms of comparing our methods with the existing method, Table 5.4 displays the results for each method employing our three accuracy measures (*oa*, κ, *RMSE*). The results suggest a similar accuracy between our BLASTP-based and profile HMM-based method, and this supports the discovery in Table 5.3 of Section 5.3.1. Moreover, the *oa*, κ of our methods are both higher than Ruybal-Pesántez's method.

To further illustrate the performance of our profile HMM-based method with threshold, we used our previously recommended threshold, $1 - \exp(-8)$ and calculated each accuracy measure. We found with a threshold, our profile HMM-based method exhibited the highest *oa*, κ and the lowest *RMSE* than any other method, indicating the best performance in classifying tags, especially for distinguishing upsB and upsC. In the meantime, we stress that the consequence of setting a threshold is that this method could not classify all tags, and the corresponding accuracy values were computed using a small set of tags.

TABLE 5.4: **Accuracy measures of each method.** We calculated the accuracy values of each method using the output of all tags. Once we set a threshold for the profile HMM-based method, we calculated the accuracy using the classified tags only. The best performance among methods for each measure is in bold.

Method	*Oa*	κ	*RMSE*
Ruybal-Pesántez's method	0.582	0.335	0.375
BLASTP-based	0.767	0.600	0.394
Profile HMM-based	0.743	0.576	0.377
Profile HMM-based method with threshold	**0.899**	**0.843**	**0.259**

Instead of DBLα tags, we also studied the DBLα types. We were interested in whether the frequent DBLα types have a higher classification accuracy than the infrequent DBLα types. Here we did not detect any significant difference (see Supplementary section 5.5.3).

5.4 Discussion

In this chapter, we have developed two algorithms (BLASTP-based and profile HMM-based) for classifying DBLα tags into ups groups. We demonstrate that both methods can distinguish between upsB and upsC group tags, while the existing method cannot. Our cross-validation results indicate that the accuracy of these two methods is similar across a series of statistical measures, and each method has a considerable classification accuracy for every ups group. Moreover, compared with the BLASTP-based method, our profile HMM-based method even quantifies group membership probability. These are the primary improvements over the current method.

While the performances of our two methods are similar, setting a threshold on the inferred probabilities using our profile HMM-based method significantly increases the accuracy. However, it reduces the number of classified tags. The crucial caveat for users is that there is a trade-off between the number of tags classified with a threshold and the resulting classification accuracy.

We grouped the tags into categories based on all DBLα subclasses and three ups groups. To determine whether we over-divided the reference tags, we clustered the DBLα subclasses to reduce the number of categories. However, it turned out that

there was no further classification accuracy improvement. Furthermore, one can either group reference tags by the subclass alone or ups group alone, but we do not use them since this way makes extra assumptions on data. For instance, if we divide the reference database into three ups groups regardless of tags' DBLα subclass, this implicitly assumes the distribution of DBLα tags does not depend on the DBLα subclass if the ups group is known, and the ups group solely is fully capable of deciding tag's profile composition (used for calculating the likelihood). By contrast, there is no such assumption in our approach. Consequently, we maintain the current data division scheme.

Although our reference database has been the largest so far, there are still some issues. Our reference tags come from *Plasmodium falciparum* genomes of 18 isolates, and these genomes contain over 1,000 *var* genes. Compared with the large number of *var* genes from natural parasite populations introduced at Section 3.3.2 and 4.4, whether the current reference database is representative or not is still an unsolved issue. After all, we don't have the ups group information for these datasets and thus they can not be added into the reference database of our method. In addition, we notice that there is no tag in several categories (Figure 5.1), and it is undetermined if this happens by chance. Therefore, higher sampling coverage is required. Lastly, the number of tags across the three ups groups is different. It is thus useful to explore whether more balanced data helps increase the accuracy of our methods.

For fitting a profile HMM, we aligned the reference tags for each category. Note that the reference database consists of DBLα tags of *var* genes. In general, the extreme diversity of DBLα tags affects the alignment quality; also there are always lots of available tags (over 10,000). As a consequence, DBLα tags are not suitable to align. However, the average number of DBLα tags per category in our reference is less than ten, and we aligned them with Clustal as the existing method [74] did for small datasets. Here we think the quality of the resulting alignment is unclear, and using more advanced alignment tools (like PRANK [244] and AQUA [245]) is a future direction. Moreover, fitting a profile HMM for diverse sequences like *var* genes is the subject of future work.

Regarding our BLASTP-based algorithm, we used the best BLAST hit to assign the ups group. However, this might be distorted by tags that just happen to be close to each other. A potential solution is to use a set of top BLAST hits. The subsequent problems are how to determine the number of top hits and how to extract the information from top hits. Existing studies have selected top hits by determining a

percent identity [246] or bit-score [247] threshold and chosen the one with the largest Max Score [246] or consensus [248, 249] from top hits. We believe this idea is doable and itself is a good research project.

Finally, although the application of our methods goes beyond the scope of this chapter's work, there are numerous research directions.

1. After applying our methods to the longitudinal data (e.g. date introduced in Section 1.2) or the global data from [62], we can explore whether the proportions of tags belonging to each ups group change with time or locations.

2. As mentioned in the Background section, ups groups are associated with various disease severities. Therefore, we hypothesise that ups groups' proportions differ among symptomatic and asymptomatic malaria cases. Our methods could quantify this hypothesis statistically.

3. We can compare the biological functions among three ups groups. For instance, we can study whether the DBLα types of the upsA group occur more frequently in isolates than the upsC group, as we have found that the proportion of recombinants in the upsA group is always lower than the proportion in the upsC group and recombinants are less conserved than the non-recombinants.

4. In the past, we had limited information on ups sequences, but now we have over 1,000 ups sequences. Exploring this data is useful, such as the phylogenetic analysis, recombinant detection using our methods and the relationship between ups and DBLα tag.

Source code and data availability

The reference database and scripts of proposed algorithms are available from the Github repository (`https://github.com/qianfeng2/cUps`) or Zenodo (`https://zenodo.org/records/13729367`).

5.5 Supplementary materials

5.5.1 Reference database

Our biologist collaborators Tan et al. [250] preprocessed the reference DBLα tags. Here we briefly introduce their pipeline.

Tan et al. [250] firstly assembled 18 *P. falciparum* isolate genomes from Otto et al. [44] and NCBI, and then extracted the 2kb upstream promoter sequences from the whole *var* genes. This resulted in 1,092 *var* genes with full ups sequences in total. Table 5.5 displays the source of isolates and the number of *var* genes per genome. To our knowledge, they form the largest *var* gene database with known ups group information in the literature. In order to infer the ups group of each *var* gene, Tan et al. [250] employed the methods from Rask et al. [46] and constructed phylogenetic trees using two different approaches — Markov clustering (MCL) [251, 252] and neighbor joining clustering (NJ). This yielded two trees which were then compared to find the congruent clades. Each ups sequence has two inferred ups groups. One is from MCL, and another is from NJ.

We next conducted a series of filtering steps before formulating our reference database. First, we excluded the *var* genes without the DBLα tag or the genes from the mixed infections isolates (ML01 and TG01 in Table 5.5). Next, we removed the tags whose inferred ups groups are inconsistent by MCL and NJ methods. These steps finally resulted in 846 reference tags, with upsB having the highest number of tags and upsA the lowest (17.5%, 59.3% and 23.2% from A, B and C, respectively).

5.5.2 Computation of accuracy measures

Significant efforts have been made in the literature [243, 253–256] to evaluate the accuracy of classifiers with a probability vector as the output. The traditional confusion matrix was extended to the sub-pixel confusion matrix (SCM) [243], initially used to describe the partial membership of a raster map pixel to different classes. The SCM is similar to the confusion matrix. Each pixel has a SCM, averaging the SCM across all pixels results in a final SCM. Accuracy measures are derived from the final SCM.

TABLE 5.5: **Details of reference database.** Genomes of RAJ116, IGH-CR14 were from NCBI. The remaining genomes were from Otto et al. [44] using Pacific Biosciences SMRT technology.

Isolate	Source	Number of *var* genes
3D7	lab	61
7G8	lab	41
Dd2	lab	44
GB4	lab	64
HB3	lab	42
IT	lab	55
RAJ116	lab	48
IGH-CR14	lab	54
KH01	Cambodia	58
KH02	Cambodia	52
CD01	Congo	63
GA01	Gabon	57
GN01	Guinea	75
KE01	Kenya	49
SN01	Senegal	66
SD01	Sudan	49
ML01	Mali	85
TG01	Togo	129

For the nth tag to be classified in our study, we denote r_{ni} ($i \in \{1, 2, 3\}$) to indicate whether it belongs to ith ups group (yes for 1 or no for 0), c_{nj} ($j \in \{1, 2, 3\}$) as the inferred probability of belonging to jth ups group. Table 5.6 depicts the SCM structure of this tag. p_{nij} of this table is the entry at the ith row and jth column. Constructing the SCM for each tag requires us to calculate the p_{nij} given the real and predicted vectors. Here we adopted the most practical operator—composite operator [243, 255]. Specifically, as shown in Equation 5.2, for the diagonal elements of SCM, we used the minimum rule to demonstrate the agreement of each ups group; for off-diagonal elements, we calculated the disagreement utilizing the distribution of $r_{nj} - p_{njj}$ (called omission proportion) across three groups.

TABLE 5.6: **Structure of the sub-pixel confusion matrix of nth tag.**

		Real			
		UpsA	UpsB	UpsC	Total
Predicted	UpsA	p_{n11}	p_{n12}	p_{n13}	c_{n1}
	UpsB	p_{n21}	p_{n22}	p_{n23}	c_{n2}
	UpsC	p_{n31}	p_{n32}	p_{n33}	c_{n3}
	Total	r_{n1}	r_{n2}	r_{n3}	1

$$p_{nij} = \begin{cases} \min(r_{ni},\, c_{nj}) & i = j \\ \frac{(c_{ni} - p_{nii})(r_{nj} - p_{njj})}{\sum_{j=1}^{k}(r_{nj} - p_{njj})} & i \neq j \end{cases} \tag{5.2}$$

After computing the SCM for each tag, the final SCM is the average of the SCMs for all tags; see Table 5.7 for its final structure. In this table, $p_{ij} = \sum_{n=1}^{N} p_{nij}/N$, where N is the overall number of tags. $r_j = \sum_{i=1}^{3} p_{ij}$ represents if it belongs to jth ups group, and $c_i = \sum_{j=1}^{3} p_{ij}$ is the inferred probability of belonging to ith ups group.

TABLE 5.7: **Final structure of the sub-pixel confusion matrix of a classifier.**

		Real			
		UpsA	UpsB	UpsC	Total
Predicted	UpsA	p_{11}	p_{12}	p_{13}	c_1
	UpsB	p_{21}	p_{22}	p_{23}	c_2
	UpsC	p_{31}	p_{32}	p_{33}	c_3
	Total	r_1	r_2	r_3	1

The two most popular assessment indices [255] are proposed using the final SCM: overall accuracy (*oa*, which differs from the accuracy previously calculated from the ordinary confusion matrix) and the kappa coefficient (κ). *oa* is the sum of the diagonal elements of the SCM ($\sum_{j=1}^{3} p_{jj}$), which represents the overall proportion of agreement for all three ups groups; κ is the proportion of agreement after the chance agreement is removed (see Equation 5.3) [257]. The range of *oa* and κ are both $[0, 1]$. Larger *oa* and κ values indicate greater accuracy.

$$\kappa = \frac{\sum_{j=1}^{3} p_{jj} - \sum_{j=1}^{3}(r_j \times c_j)}{1 - \sum_{j=1}^{3}(r_j \times c_j)} \tag{5.3}$$

Another frequently used accuracy measure is root mean square error (*RMSE*). This combines the difference between each tag's classification and the true ups group into a single value (Equation 5.4). A smaller *RMSE* denotes higher accuracy.

$$RMSE = \sqrt{\frac{\sum_{n=1}^{N}\sum_{j=1}^{3}(c_{nj} - r_{nj})^2}{3N}} \tag{5.4}$$

5.5.3 Comparison between frequent and infrequent DBLα types

We followed with standard pipeline [62, 74] and firstly clustered all reference DBLα tags with 96% sequence similarity cutoff. This resulted in 747 DBLα types in total (128 upsA, 453 upsB and 166 upsC), of which 709 types were seen in only an isolate, while remaining types were found in more than two isolates. Here we defined the 709 types as infrequent DBLα types, the remaining 38 types as frequent ones.

We conducted leave-one-out cross-validation using these 747 DBLα types. Using the highest posterior probability, we calculated the sensitivity and precision; using the probability vectors, we also calculated the *oa*, κ and *RMSE*. The results (Table 5.8) indicate that the accuracy between these two groups is similar. We also tested if the *RMSE* was statistically different, and it turned out there was no significant difference here ($p = 0.998$ from a two-sample Wilcoxon test).

TABLE 5.8: **Accuracy measures for frequent and infrequent DBLα types.** Sensitivity and precision are the weighted mean of each ups group.

Group	Sensitivity	Precision	*Oa*	κ	*RMSE*
Frequent DBLα types	65.8%	65.8%	0.672	0.497	0.423
Infrequent DBLα types	70.9%	70.9%	0.700	0.498	0.407

5.5.4 Supplementary figures and tables

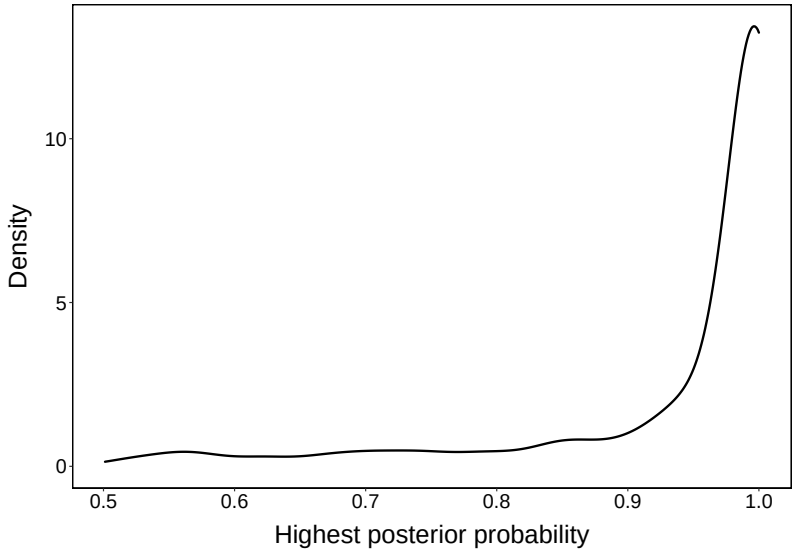


FIGURE 5.5: **The density distribution of the highest posterior probability.** These highest posterior probabilities were obtained from cross validations using the profile HMM-based method.

TABLE 5.9: **Number of classified tags in each ups group with the threshold** $1 - \exp(-8)$**.** The total number of tags in each ups group is in bracket.

	Number of classified tags	Proportion of classified tags
UpsA	138 (148)	92.6%
UpsB	113 (502)	22.5%
UpsC	66 (196)	33.7%
Total	317 (846)	37.5%

Chapter 6

Conclusions and future work

The antigen PfEMP1 plays a central role in the pathogenicity and immune evasion of the deadliest malaria parasite — *Plasmodium falciparum*. It is encoded by hyper-variable and rapidly evolving *var* genes. A key driver in the diversification of the *var* genes is recombination, and detecting recombinant *var* genes has been of great interest to scientists with implications for malaria interventions. Although the architecture of *var* genes is highly variable, almost all *var* genes (except *var2CSA*) encode the DBLα domain, which has been found to be immunogenic. In this thesis, we focus on the extensively studied DBLα tag in the DBLα domain and aim to understand the evolutionary processes behind DBLα evolution. We specifically focus on detecting the recombination and recombinant tags. There are various methods for recombination and recombinant identification, such as distance-based, phylogenetic, substitution distribution-based and compatibility methods. In the context of *var* genes, there are also a few methods (like BLAST used in Otto et al. [59], 'clone tree' used in Claessens et al. [42]) for identifying *var* recombination. However, no suitable method for detecting recombination and recombinants from a large number of unaligned sequences like DBLα exists. This is the main driving force of my work.

6.1 Summary and final remarks

Zilversmit et al. [54] conducted the first systematic attempt to quantify the recombination of DBLα domains. They hypothesized that each sequence is reconstructed by a mosaic of source sequences and propose a jumping hidden Markov model to map each sequence to its nearest source sequences. The JHMM was then used to detect

the recombination of DBLα domains from three isolates of different *Plasmodium* species. The ancient evolution of the *var* gene family was thus uncovered. In Appendix A, we apply this JHMM to a large dataset of DBLα tags from a Ghana pilot study. In addition, the JHMM has been applied to a dataset of DBLα tags collected from ten countries worldwide [62]. We analyze the recombination patterns using the mosaic representations from these two datasets and then conduct extensive comparisons. We find consistent results regardless of the dataset employed, such as comparable HMM parameters, analogous distributions of recombination breakpoints, and similarly short horizontal jump distances. Moreover, the uneven distribution of breakpoints is similar to the finding in Zilversmit et al. [54].

This work is the first time that the recombination patterns of DBLα tags are systematically analyzed and uncovered from large datasets (over 100,000 tags). Therefore, these findings offer us a fundamental understanding of *var* gene DBLα tags. Biological realism facilitates more related discoveries and our subsequent methodological advancement. For instance, the three peaks in the breakpoint distributions motivate us to explore further, leading to a new finding related to the positions of frequent homology blocks (in Chapter 3). The short horizontal jumps motivate us to develop an improved JHMM described in Chapter 4.

6.1.1 Detecting recombinants from unaligned sequences

Utilizing information extracted from the mosaic representations, we introduce a statistical method for detecting recent recombinants when the input sequences are collected simultaneously (Chapter 3). Our algorithm has multiple advantages:

- Our method can process thousands of relatively short sequences (see our discussion about running time in Section 3.5.2.1) without requiring a multiple sequence alignment or a reference panel.
- Our method is not restricted to DBLα tags. It has the potential to be applied to other unaligned sequences that are recombinant, although this is not tested and beyond the scope of this thesis (this thesis specifically focuses on DBLα tags).
- Our method accepts as input both DNA and protein sequences.

- In the presence of a high recombination rate, long sequences, a large dataset, and even indels, our method maintains high sensitivity and specificity. Notable is the fact that existing methods typically fail when datasets include indels.
- For every detected recombinant sequence, our algorithm provides a statistical support value for measuring the identification uncertainty.

This work contributes to this field in two ways. Firstly, there is no method for detecting recombinants from a large number of unaligned sequences, and our algorithm fills this gap. The advantages of our algorithm, such as the generalization and scalability, help scientists identify recombinants from more living organisms. This is a big algorithmic advancement. The application of our algorithm also increases the knowledge of many aspects, such as the properties of recombinants and non-recombinants, the frequency of recombinants and the evolution of more species.

Secondly, applying our algorithm to Ghana's DBLα tags yields many biologically significant results. For the first time, our data quantitatively support the hypothesis that recombination occurs preferentially within the same ups group. For the first time, we compare relevant patterns between recombinant and non-recombinant DBLα types. This allows us to derive multiple new results. For instance, the proportions of recombinants differ among DBLα domain subclasses and ups groups, and the non-recombinant types are more conserved than the recombinant ones. These findings shed light on a better understanding of DBLα tags' evolutionary history.

6.1.2 Detecting recombination with an improved JHMM

A primary shortcoming of the JHMM is that it allows recombination between any two points in a pair of sequences, which lacks biological realism. This is also the main cause of the JHMM's slow performance, especially when processing lengthy sequences. In Chapter 4, we present an improved JHMM that constrains recombination to only act between nearby positions in an unaligned set of sequences.

It is the first time that the JHMM has been improved algorithmically. In addition to inheriting the benefits of the JHMM, our model possesses two additional advantages. First, our model can largely reduce the artificial segments from the JHMM. It also enjoys the similar accuracy in identifying recombinants and locating breakpoints as the JHMM. Second, our model is faster than the JHMM, and its scalability allows

it to be applied to larger datasets. These advantages facilitate broader applications and more discoveries than the original JHMM.

Its application to the longitudinal datasets of Ghana expands our knowledge of malaria DBLα tags. By identifying recombinants and analyzing recombination patterns using two time-separated datasets, we obtain similar results regardless of the dataset used. Moreover, both are consistent with the pilot dataset results, reaffirming our previous findings. Additionally, these results obtained from various time points allow us to explore how the patterns change with time, further uncovering new findings. For instance, the proportions of recombinants in fourteen DBLα subclasses are found to be significantly different from the average in both time-period data, and half of subclasses has been confirmed using the pilot dataset. There is also an increasing number of DBLα subclasses with time (from pilot to S5) whose proportions of recombinants are significantly different from the average. We suggest that this should be explored further for reasonable interpretations. Overall, although more questions are proposed (and even unsolved) during our exploration, it is indispensable for understanding the evolutionary history of malaria DBLα tags.

6.1.3 Classifying *var* genes into ups groups

We focus on classifying *var* genes into ups groups in Chapter 5. There are three major ups groups, upsA, upsB and upsC. The current method uses the DBLα subclass of each DBLα tag and classifies the tag into only two ups groups, upsA and non-upsA (upsB/upsC). We introduce two methods (BLASTP-based and profile HMM-based), which can classify each DBLα tag into three groups. The cross-validation results demonstrate the accuracy of these two methods, which are both more accurate than the existing method.

Importantly, our profile HMM-based method quantifies group membership probability. Although the accuracy of the BLASTP-based and profile HMM-based methods is similar, we show that setting a threshold on probabilities using our profile HMM-based method could improve the classification accuracy. However, the cost is that there are fewer classified tags.

This work is the first time in the scientific literature that non-upsA group tags can be accurately separated using our methods. This work enables scientists to directly compare the DBLα tags and even the whole *var* genes of the upsB and upsC groups

and investigate the patterns of tags belonging to each ups group. This greatly improves studying ups groups of *var* genes. Additionally, while these two methods are explicitly designed for classifying malaria DBLα tags, they could be extended to a similar classification for other sequences. We note that regardless of the method used, a reference database is required.

6.2 Future directions

We discussed the limitations for each proposed algorithm/model and future research at Discussion section 3.4, 4.5 and 5.4. Because they are provided under specific contexts, we outline several future directions from a macro perspective below.

6.2.1 Recombination patterns with time and locations

Finding reliable biological knowledge requires datasets with spatial and temporal characteristics. Firstly, the results could be more evident when applying our algorithm (Chapter 3) to multiple locations' datasets. All the biological findings in this dissertation result from a study conducted in a single Ghanaian district. It is worthwhile to determine whether our current findings apply to all regions.

Since it needs to collect sequences from various locations, the number of sequences in some locations is likely large. It is thus possible that our current method cannot run the dataset within a practical time. As a result, large data size stops detecting recombinants. A possible solution we suggest is to replace the original JHMM with our improved JHMM (Chapter 4) in our recombinant detection algorithm (Chapter 3). This reintegrated method applies to larger datasets.

Many questions could be studied in the future. It is interesting to see whether the recombination rates or breakpoint distributions are similar across different countries/continents. We see the largest number of malaria cases in Africa. Whether it is associated with a higher recombination rate or a particular recombination pattern is a meaningful topic. Moreover, we see that the non-recombinants are more conserved than the recombinants, and the proportions of recombinants in several DBLα domain subclasses are different from the average. Whether these patterns remain across various locations is still an unsolved problem.

Secondly, although we have applied our improved JHMM to a longitudinal dataset from Ghana, only three time points are used. Once tags are collected from follow-up surveys, applying our method (Chapter 4) enables us to obtain more statistically significant results. Likewise, several problems could be explored further. The frequency of non-recombinants is increasing from pilot to S5, and the number of DBLα subclasses whose proportions of recombinants are significantly different from the average is also increasing. Using more time-separated datasets helps examine such trends. It's interesting to see how and why the non-recombinants become more conserved with time while the recombinants are being less conserved. In addition, one could also explore other patterns like meiotic and mitotic recombinations, and study their prevalence in wet and dry seasons respectively.

6.2.2 Constructing a phylogenetic network of DBLα tags

Another research direction is to study the evolution of DBLα tags through a phylogenetic network. Such a phylogenetic network facilitates a good understanding of various evolutionary processes, and it also helps the prediction of the gene sequences' evolution. Eventually, this facilitates the development of more quantitative strategies to control the frequent recombination of DBLα tags.

Currently, there is no method for inferring the phylogenetic network with extremely diverse sequences like DBLα tags. However, our methods manage to detect recombinants from the DBLα tags, and the tags are separated into recombinants and non-recombinants. We can infer the phylogenetic tree using the non-recombinant tags. Although this tree is only a subset of the network, it partially uncovers the DBLα tags' evolution.

In our view, there are two possible methods for constructing this network. The first method is to divide the tags into small regions based on biological properties. Shorter sequences are easier to infer the evolutionary history. Another method is to use the trees of each triplet obtained from Chapter 3 and then join them together. In addition, the tags in our longitudinal study could calibrate the inference. While the design of such methods goes beyond the scope of this thesis's work, constructing a phylogenetic network is essential for studying the evolution of DBLα tags.

Last but not least, although our work described in this thesis contribute a better understanding of DBLα evolution, there are still lots of unsolved questions for implications in practical malaria intervention and control. From our work, we have noticed that the proportion of recombinant DBLα tags experiences a sharp decrease from pre-IRS to IRS and a slow increase from IRS to post-IRS. However there is no evidence that this IRS intervention is the cause of the change in recombination frequency; moreover, it's unclear how much the frequent recombination in a tag of $\sim$370nt length affects the overall virulence of parasites and continuously emerging drug resistance; we also found different proportions of recombinants in three ups groups of *var* genes. Since different ups groups are associated with disease severity, it is unknown whether recombination plays a role in the disease severity, even if it does, how to manipulate the recombination to minimise disease impacts is unclear.

In conclusion, there is a huge gap between our current knowledge like DBLα recombination and conclusive implications in malaria. In my view, scientific measures to prevent and control malaria require much more research in the future among scientists (e.g. mathematician, biologist, geneticist, immunologist) from various fields. Nevertheless, our work in this thesis is a stepping stone toward this future research.

Appendix A

Analysis of DBLα tags from a cross-sectional study in Ghana

A.1 Introduction

Plasmodium falciparum [27–29] is the most virulent and predominant parasite species [217] that causes human malaria. It has caused 300,000 deaths and 200 million clinical cases each year [26] and imposed a huge health and economic burden [5]. The antigen *Plasmodium falciparum* erythrocyte membrane protein 1 (PfEMP1) plays a key role in the immune evasion of *P. falciparum* [40]. It helps parasites to evade the detection of the human immune system, as a consequence, the parasites could stay in the human blood for a long period of time and cause waves of parasitemia [258]. PfEMP1 is encoded by hyper-diverse *var* genes, and recombination is one of the main mechanisms for maintaining *var* diversity. Therefore, studying the recombination of *var* genes has been of great interest to scientists.

A lot of studies have specifically focused on the DBLα tag of *var* genes [62, 69–76]. Recently Tonkin-Hill et al. [62] combined previously published DBLα tags from ten countries and applied the jumping hidden Markov model (JHMM) [54] to explore the DBLα structure globally. With the generated mosaic representations from the JHMM, one could study many aspects of the recombination patterns, for

instance, the number of parental sequences for each recombinant, the length of parental segments. However, their analysis about these studies is very limited owing to different research purposes.

To fill in this gap, we applied the JHMM to a dataset of DBLα tags from the Ghana pilot study (introduced in Section 1.2) and performed a comprehensive analysis of the mosaic representations. We studied the breakpoint distribution and frequencies, the length of parental segments and the characteristics of recombination positions between parental sequences. We also repeated the analysis for the generated mosaic representations using the global data [62] and subsequently made comparisons using these two datasets. Overall, we aim to understand the underlying recombination patterns in these datasets.

In addition, Tonkin-Hill et al. [62] found the DBLα population structure using the global sequences. Specifically, they used the DBLα tags with less than 96% sequence identity and found the DBLα population structure was stratified by countries. Although DBLα tags from the pilot study (Section 1.2) were only from a district of Ghana, the information about the specific catchment area for most isolates was available, so we also wanted to explore if there was any similar population structure stratified by catchment areas. Therefore, we applied their approaches to this dataset, although we did not find any population structure at our side eventually.

We emphasize that above work is exploratory analysis. However we do believe these basic analysis would help readers to better understand our data. This is also beneficial to better understand Chapter 3 and 4. After all, it is vital to understand the structure of the data before attempting to develop methods for it. Finally, even though these results are in the appendix, we do provide enough details as below.

A.2 Results

A.2.1 Data summary

The dataset from Ghana pilot study consists of 35,591 DBLα tags with an average length of 370nt (s.d. 25.6nt). They are distributed among a total of 161 isolates, noting that only 133 isolates (82.6%) have information about the corresponding catchment areas (51 isolates from Soe; 82 isolates from Vea/Gowrie), and the remaining ones are unknown. The dataset in Tonkin-Hill et al. [62] consists of 98,322 DBLα tags collected from 1,248 *P. falciparum* infected isolates with an average length of 373nt (s.d. 35.5nt).

We note that a subset of isolates in the Ghana dataset of [62] also makes up the dataset in the pilot study (Section 1.2). Though both DBLα datasets were generated through targeted amplicon sequencing, the former was sequenced on the Roche 454 platform while the latter was generated using the Illumina MiSeq platform. Compared to the Roche 454 platform, the Illumina platform generates higher throughput of data, thus resulting in higher sequence coverage, with a lower sequencing error rate [259, 260]. We obtained 35,591 tags from the pilot study and 20,643 tags in Ghana from the global study [62], and these two datasets are largely the same. Although the global dataset and the Ghana pilot data differ in sequencing technology and scope, our goal is to see whether the recombination patterns of DBLα tags are similar among these two datasets.

A.2.2 Comparison of the JHMM output between Ghana and global datasets

A.2.2.1 Estimated parameters

As mentioned in Section 2.2.2.2, there are three parameters to be estimated in the JHMM: gap opening probability δ, gap extension probability ϵ and probability of recombination ρ. Zilversmit et al. [54] estimated these parameters in two steps. In

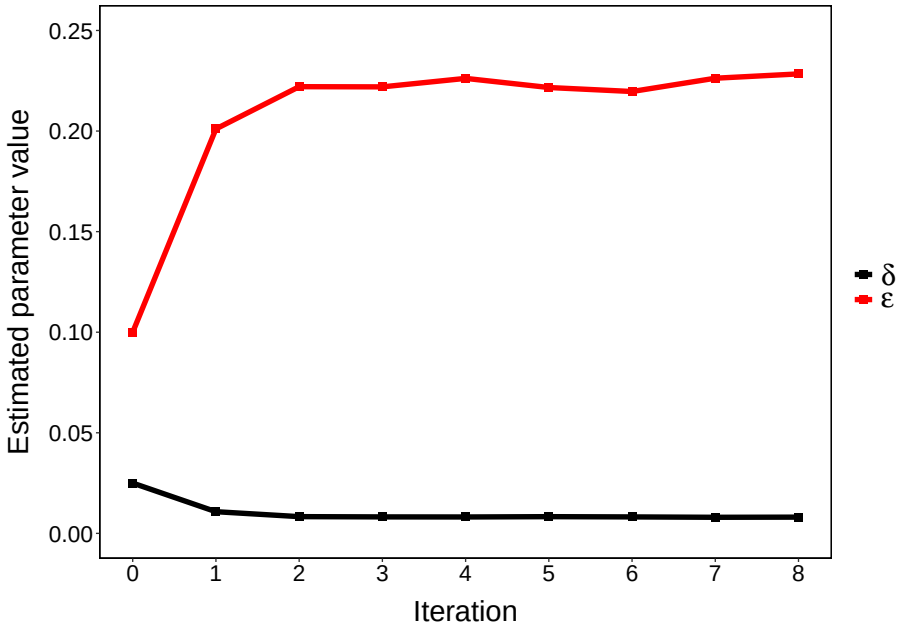


FIGURE A.1: **Convergence of non-jump parameters.** This figure shows the estimated δ and ϵ (using the Viterbi training algorithm) with iterations on our Ghana pilot dataset.

the first step, they estimated the non-jump parameters by employing the Baum-Welch algorithm with ρ fixed at 0; in the second step, a grid of ρ from 0 to 0.1 was used to compute the composite likelihoods for all sequences. Finally, $\hat{\rho}$ was determined by the maximum composite likelihood.

Tonkin-Hill et al. [62] modified the above steps since a practical computation is infeasible owing to their large number of sequences. They replaced the Baum-Welch algorithm with the faster Viterbi training algorithm (see Section 2.2.1.3) for estimating δ and ϵ; they randomly sampled $1,000$ sequences from the target set to compute the likelihood when estimating ρ.

In order to achieve a realistic computational time for the dataset in Ghana pilot study, we followed this modified pipeline. As shown in Figure A.1, the δ and ϵ converged (parameter difference is less than 1% between consecutive iterations) after 8 iterations. We generated a likelihood surface (Figure A.2) by summing $1,000$ likelihood values and determined 0.015 as the optimal value of ρ.

Table A.1 shows the estimated parameters for Ghana and global datasets. The gap opening probability is slightly lower in the Ghana data than in the global data,

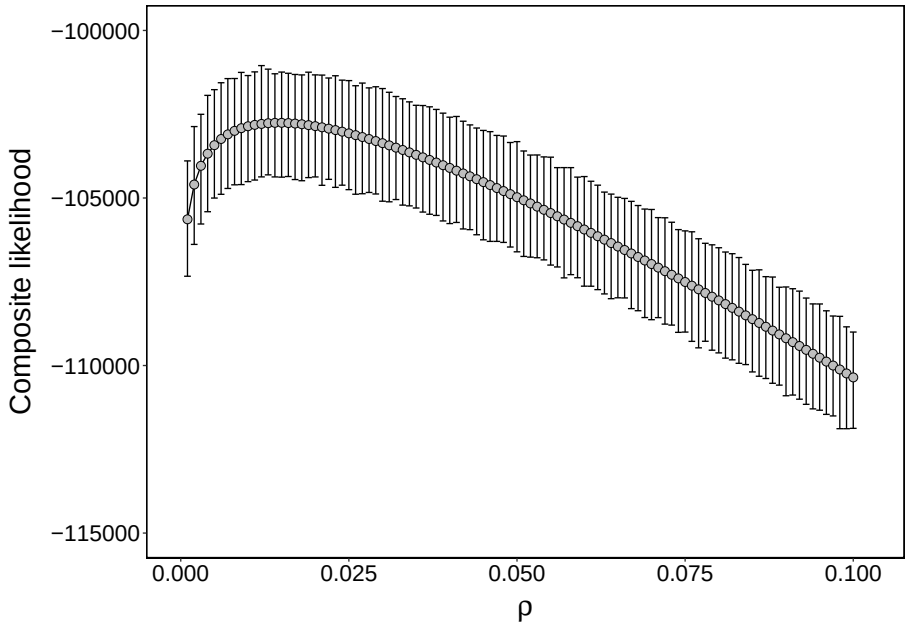


FIGURE A.2: **Composite likelihood surface for a grid of ρ.** Each bar is the 95% confidence interval based on 1,000 bootstrap replicates on the likelihoods.

while the probabilities of gap extension and recombination are similar. Overall, the differences in estimated parameters between these two datasets are small, as it should be.

TABLE A.1: **Estimated JHMM parameters using the Ghana and global datasets.**

Parameters	Ghana data	Global data [62]
δ	0.008	0.017
ϵ	0.228	0.273
ρ	0.015	0.014

A.2.2.2 Distribution of breakpoints

Mosaic representations from the JHMM provide us with the breakpoint positions of recombinations. We computed the relative positions of breakpoints from the Ghana data to explore how the breakpoints are distributed along the DBLα tags. We show a few mosaic representations using this data in Figure A.3. The result (Figure A.4) indicates the uneven distribution of breakpoints. In particular, recombination occurs more frequently in three regions. We identified conserved regions in the DBLα

FIGURE A.3: **Few mosaic representations from the JHMM using the Ghana data.**

tags, called homology blocks (HB) [46]. The close relationship between these three regions and locations of common HBs are discussed in Section 3.3.2.4. Furthermore, although the frequency of breakpoints at both ends of the tag seems low, this does not mean recombination there is rare. More likely it is because the sequences there are very similar, and it is difficult/impossible to detect recombination.

To compare the breakpoint distributions using the Ghana pilot data and global data, we also computed the breakpoints' relative positions from the JHMM output of the global dataset. The global data shows three peaks in the breakpoint distribution, which overlay the peaks using the Ghana pilot data. Figure A.4 also shows that the bar heights of two histograms are almost the same across intervals. Overall, the breakpoint distributions are similar in both two datasets.

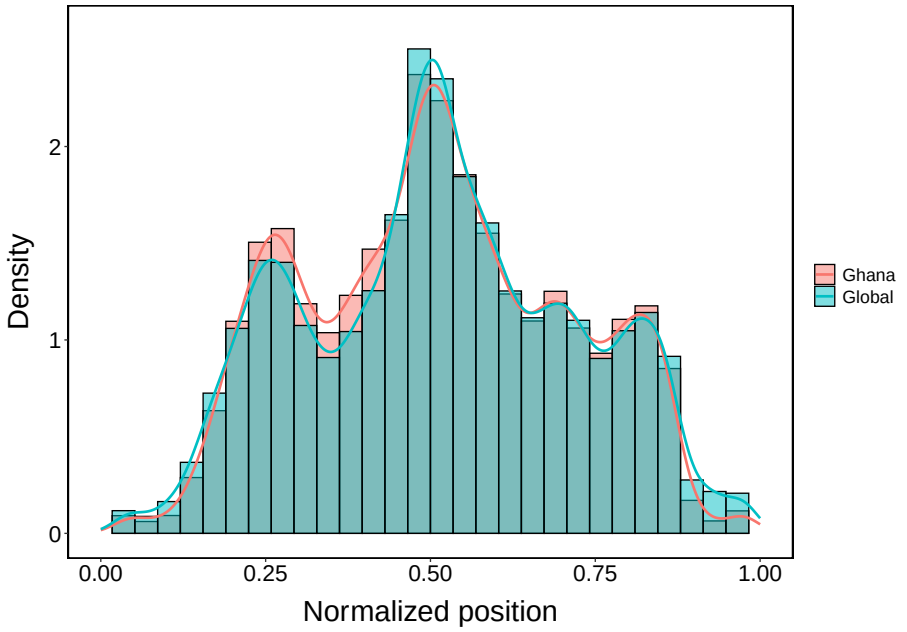


FIGURE A.4: **Kernel density curves and histograms for the breakpoint distribution along DBLα tag from the global and Ghana datasets.** Bar height indicates the normalized breakpoint frequency at each location interval.

A.2.2.3 Number of breakpoints per tag

To explore the frequency of recombination, we counted the number of constituting source segments per DBLα tag from the JHMM output. Figure A.5 shows the distribution of the number of source segments. We can see that the two distributions from the global and Ghana data are almost the same. The most common scenario for both two datasets is that there is only a single breakpoint (44.0% of tags in global data, 33.7% of tags for Ghana data). As the number of breakpoints increases, the frequency decreases. This matches our expectations since it is unlikely that many recombination breakpoints exist in a single tag. When the breakpoints reaches 3 at two datasets, both take up more than 85.0% of their separate sequence sets.

We noticed that some source fragments consist of only one amino acid (see the third source at Figure A.6 as an example) in the JHMM output. This indicates the JHMM is not finding the biological truth. In order to see the proportion of these cases, we generated the distribution of segment length using the global and Ghana data. The corresponding result is shown in Figure A.7. We found that the proportion of length-one segments is the largest in each dataset. There are two

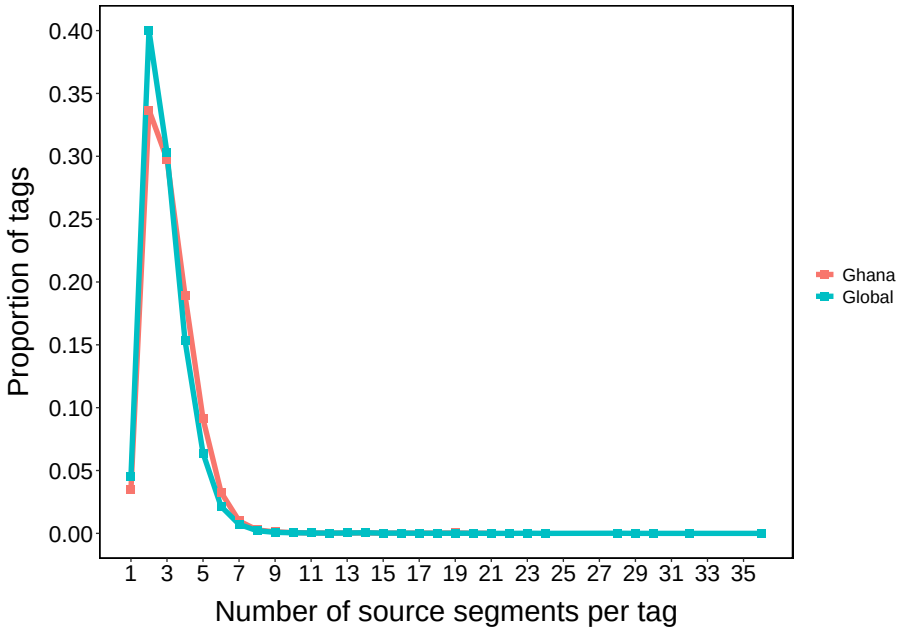


FIGURE A.5: **Distribution of source segment counts from the JHMM output in the global and Ghana data.**

```
Target: target_seq156   Length: 121     Llk: -90.213
target_seq156   DIGDIIRGKDLYLGNNKKDKLQIKLKEYFKKIYEQLDRSIRSKYDETSENYFQLREDWWTANRETVWKAITCGTHDGDTYFRATCSDERGGAQARNQCRCPNGKNADQVPTYFDYVPQFLR
                ||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||| ||
db_seq1984      DIGDIIRGKDLYLGNNKKDKLQIKLKEYFKKIYEQLDRSIRSKYDETSENYFQLREDWWTANRETVW
db_seq7334                                                                      KAITCGTHDGDTYFRATCSD
db_seq13755                                                                                        E
db_seq1984                                                                                          RGGAQARNQCRCPNGKNADQVPTYFDYVPQYLR
```

FIGURE A.6: **An example of the mosaic representation from the Ghana data that contains a single character as the contributing source.**

possible reasons for this. Firstly, the JHMM allows recombination between any two points in a pair of sequences. This unconstrained pattern makes the model to simply pick out an amino acid that is not truly homologous, and we think this is the artifact of the JHMM. Our improved JHMM described in Chapter 4 is designed to address this issue. Another potential reason is that the estimated recombination parameter is too high. A high recombination rate might lead to many short source fragments.

We studied the second hypothesis and found that it's highly unlikely. We note that the recombination parameter is estimated by maximizing the likelihood of data given the gap opening probability and gap extension probability. We are not sure whether reducing the estimated ρ manually is appropriate or not, but we still used a lower ρ (0.01) to regenerate the mosaic representations using both the Ghana and global data. Results (not shown) indicate that there is a decrease in the proportion of length-one source fragments in Ghana data, but the difference is small (2.4% to

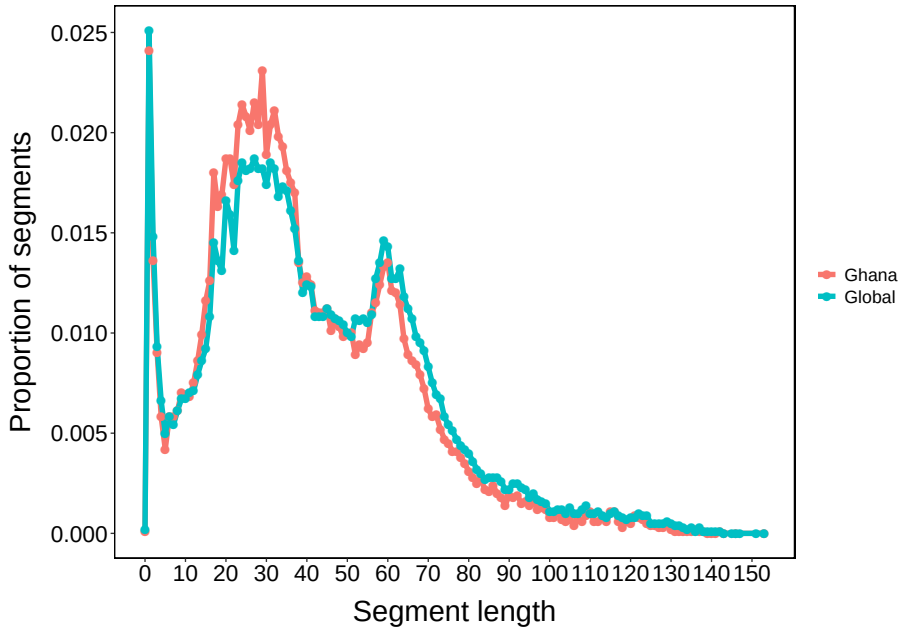


FIGURE A.7: **Source segment length from the JHMM output using the global and Ghana data.**

2.2%), while there is no change of the proportion in global data (2.5% before and after).

Finally, we found that the length distributions (Figure A.7) using the two data sets both show multimodality. One peak of segment length distribution is at 1/2 of the tag, and another peak is at 1/4 of the tag (a slight rise is around 3/4 of the tag). These peaks are reasonable as the breakpoint distributions contain three frequent recombination positions (1/2, 1/4 and 3/4 of the tag).

A.2.2.4 Horizontal jumps

Although the DBLα tags' lengths are diverse, they are homologous sequences. The alignment of these sequences is expected to keep the relative positions of each sequence roughly the same per character (column). Therefore, we hypothesized the recombination happens at homologous positions. Specifically, when a source sequence "jumps" to another source in a mosaic representation, we would like to see it does not jump very far. For instance, if a breakpoint is at the center of the first source sequence, we aim to explore whether it jumps to the center of the second

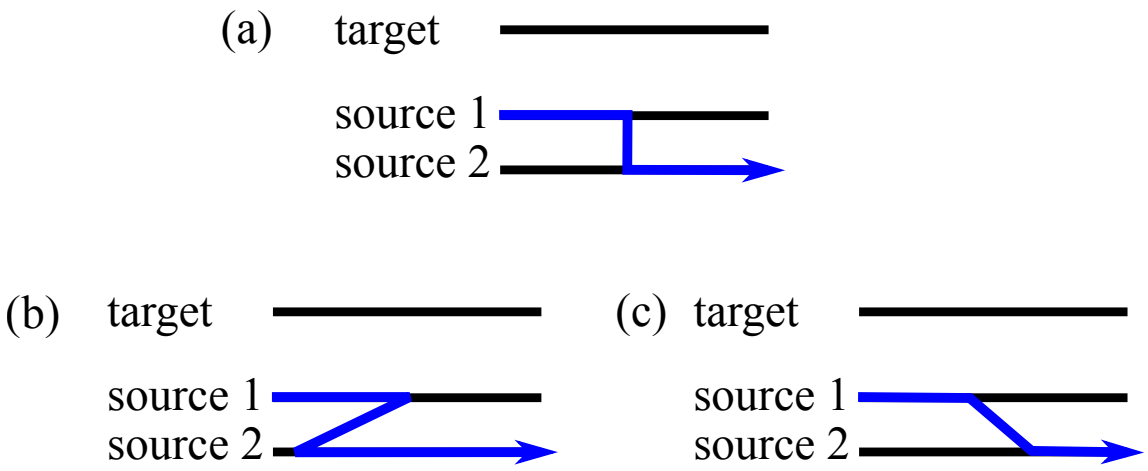


FIGURE A.8: **Three types of jump in the mosaic representation.** (a) shows the recombination happens at homologous position of source sequences. In contrast, examples of a backward and forward jump are shown in panel (b) and (c) respectively, indicating the recombination happens at non-homologous position.

source as well. If the recombination does not happen at homologous positions, we show such examples in the panel (b) and (c) of Figure A.8, in these scenarios, there are horizontal jumps between sequences.

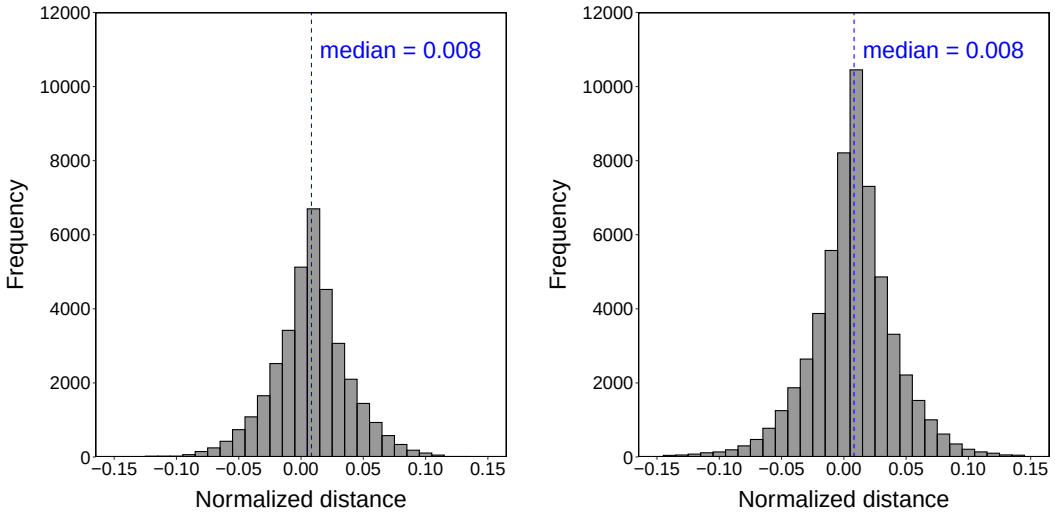
When a mosaic representation contained breakpoints, we compared the relative positions on either side of a recombination. This resulted in the distance between its consecutive source segments (the start position of second source segment minus the end position of first source segment). If most distance values are small, it suggests that the DBLα recombination happens at homologous positions.

For calculating the above distance values, we computed the normalized position range for each contributing source segment, and then calculated the distance between each pair of consecutive intervals. See Figure A.9 as an example, the distance values in this mosaic representation are (1) $0.14 - 0.13 = 0.01$, (2) $0.66 - 0.67 = -0.01$. Single-site segments were excluded since they are difficult to locate accurately.

Figure A.10 shows the distribution of distances using Ghana and global data. This graph suggests that most distance values fall in the interval $[-0.025, 0.025]$ of tag in both datasets. 38.3% of distances from the Ghana data and 40.4% of distances from the global data are outside of this interval. The mean distance is 0.006 for Ghana and 0.005 for the global dataset. Although the means are not statistically equal,

Target	A G T C K D I M M M - F	
Source_1	A G T T	$[0.01, 0.13]$
Source_2	K D - M	$[0.14, 0.67]$
Source_3	M M K F	$[0.66, 1.00]$

FIGURE A.9: **An example of the JHMM output for calculating the distance between consecutive source segments.** The normalized position range of each contributing source is shown by the interval.



(a) (b)

FIGURE A.10: **Histogram of the difference between consecutive source segments from the JHMM output using our Ghana (left) and global (right) datasets. The dashed line indicates the median.**

they both correspond to one amino acid by accounting for the tag length (∼124aa) and this is in line with our expectation. In summary, these results do not support the presence of horizontal jumps in the recombination of DBLα tags.

A.2.3 Local DBLα population structure

Tonkin-Hill et al. [62] used phylogenetic trees to assess the population structure of DBLα tags. Each tip of the tree represents an isolate (individual), and the structure of the tree shows the relatedness of isolates. They found the isolates in their constructed trees were clearly separated by country of origin and therefore

uncovered the global population structure. Here we followed their methods and explored the local population structure of DBLα tags from the Ghana pilot dataset.

Two different methods for generating DBLα phylogenetic trees were used in [62]. The first method used the binary presence/absence matrix after clustering DBLα tags into DBLα "types" based on 96% similarity. This matrix shows which DBLα type (row) is present in each isolate (column). With this matrix, RAxML [261] was then used to generate a tree by treating each isolate's presence/absence vector as a binary sequence. The second method used individual tags (rather than DBLα types to avoid information loss) as input and generated a tree with an alignment-free algorithm named feature frequency profile (FFP) [222]. The FFP approach determined a distance matrix between sequences with a sliding-window method and the phylogenetic tree was then inferred by FastME [262] based on balanced minimum evolution [263]. We show the details for how to generate the binary presence/absence matrix and related FFP analysis in the Methods section.

We followed these two methods and the resulting trees are in Figure A.11 and A.12. These two trees show different topologies, since we used different information of the data. We colored the tree tips by the catchment areas of isolates (Vea/Gowrie, Soe, unknown location). We note that both trees are unrooted, and the roots shown in these figures were chosen arbitrarily for the display. From these trees, we can see the isolates were not stratified by catchment areas. To test this statistically, we used the δ statistic [264] to measure the phylogenetic signal between a phylogeny and a categorical trait. Since there are few isolates with unknown catchment areas, we removed those tips and only considered the extracted phylogeny so that each tip has the location information. We found that there is no evidence for the phylogenetic signal from both two trees generated from (a) RAxML, (b) FFP and FastME (p values are 0.84 and 0.18 respectively). Therefore, there is no clear separation of geographical locations from both trees. A potential reason is that the distance between catchment areas is very close in the relatively flat Ghana Bongo District.

In addition, we see both two trees show long branch lengths for tips and short branch lengths for internal nodes. There are also few branches with support values

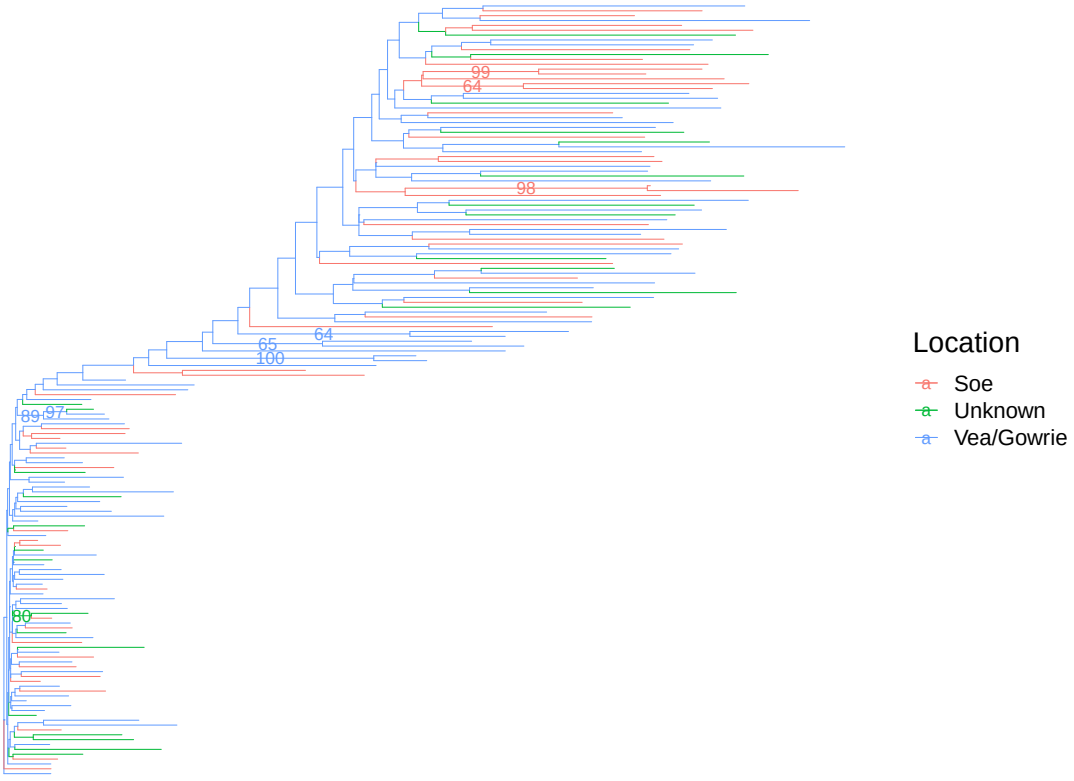


FIGURE A.11: **A phylogenetic tree of isolates generated by RAxML v8.2.12 [261] with BINCAT model.** This tree was built using each isolate's binary sequence from the presence/absence matrix of DBLα types. We set the number of bootstrap replicates to 100, and the branches with support value over 60 were annotated.

over 60 in the tree in Figure A.11. All of them indicate the resulting trees may not contain sufficient phylogenetic information. We note that this is not a meaningless analysis. In fact, it is the opposite! The reasons that makes these trees with limited phylogenetic signals is the existence of frequent recombination. Therefore, we need to develop recombination-aware methods to address it. This is also the motivation for majority of our work described in this thesis (Chapter 3 and 4). In addition, our results further suggest new ways of phylogenetic analysis for DBLα tags are needed in the future.

The binary presence/absence matrix (Figure A.13) provided us with additional details of the data. We found most DBLα types were rare, occurring in less than 20 isolates (Figure A.14). Soe has the largest median number of DBLα types per isolate (Figure A.15), although the mean difference in the number of DBLα types between the two catchment areas is not significant (two-samples Wilcoxon test, $p = 0.109$

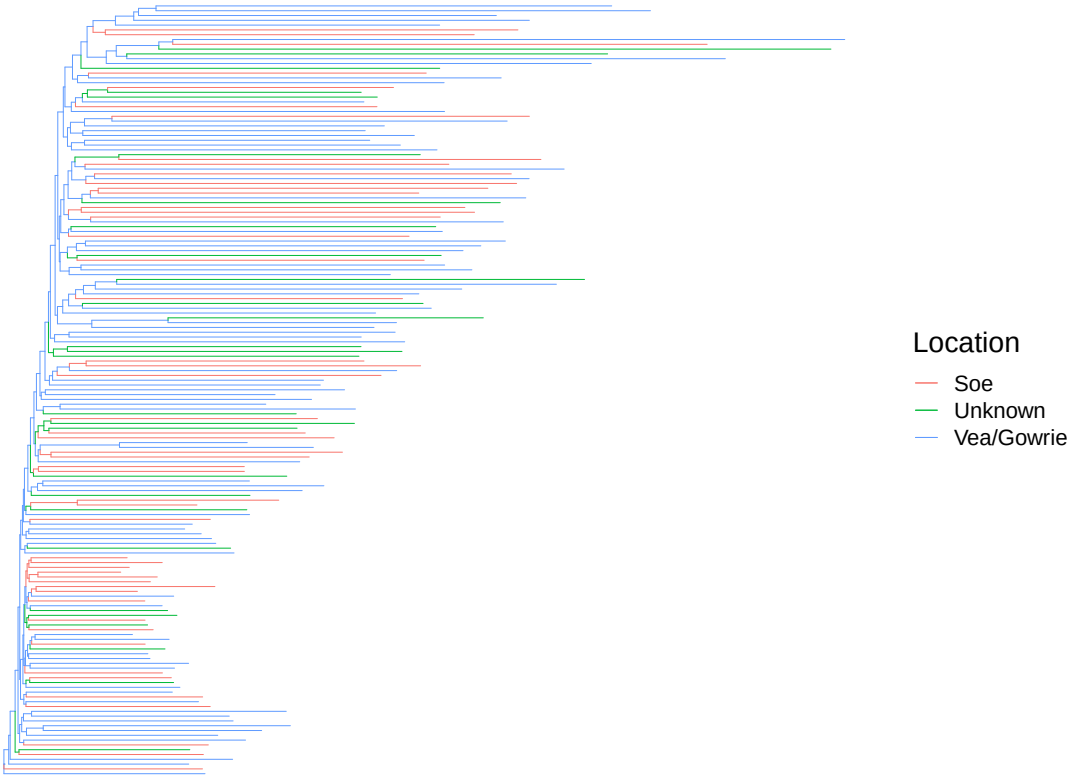


FIGURE A.12: **A phylogenetic tree of isolates produced by FFP v**3.19 **[222] and FastME** *v*2.0 **[262].** This tree was built using the distance matrix from all DBLα sequences with the FFP method. We could not do bootstrapping as in Figure A.11, since FastME cannot perform bootstrapping when the input data for inferring the phylogeny is a distance matrix, see its online execution [265] and README from downloads.

for Soe and Vea/Gowrie). In addition, Tonkin-Hill et al. [62] conducted a principal component analysis (PCA) on this binary matrix and discovered a clear separation between South American isolates and other regions' isolates in their PCA plots. Following this strategy, we generated the plots using our matrix. The result (Figure A.16) suggests that the catchment area is not a highly variable feature that PCA could identify. Again, the geographical distances of isolates might explain this. After all, the isolates in [62] are from ten countries across Africa, Asia and South America. In contrast, the isolates from the pilot study are only from a district in Ghana.

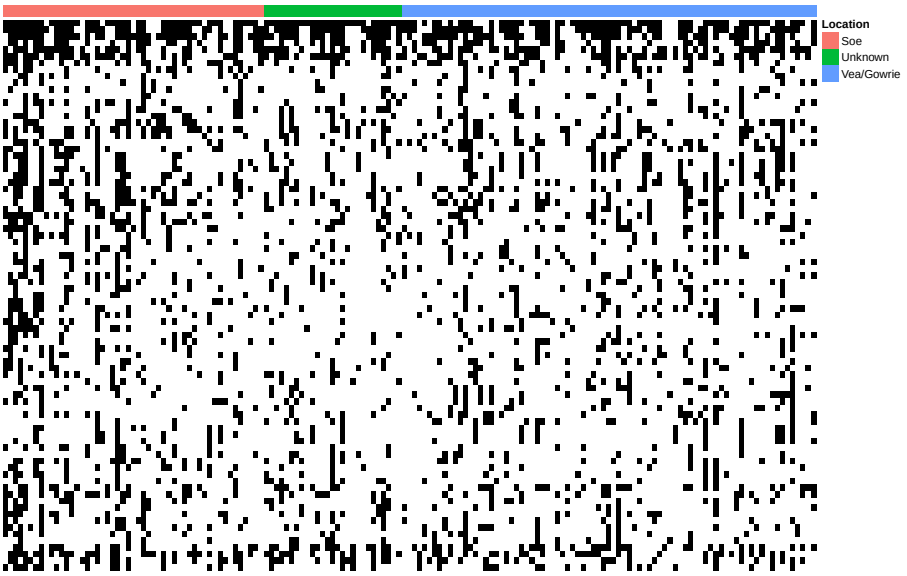


FIGURE A.13: **Binary presence/absence matrix of conserved DBLα types (occurring in over 20 isolates).** Each entry of this matrix is 1 (black) or 0 (white), representing the presence or absence of a DBLα type (row) in the isolate (column). Isolates were arranged by catchment areas.

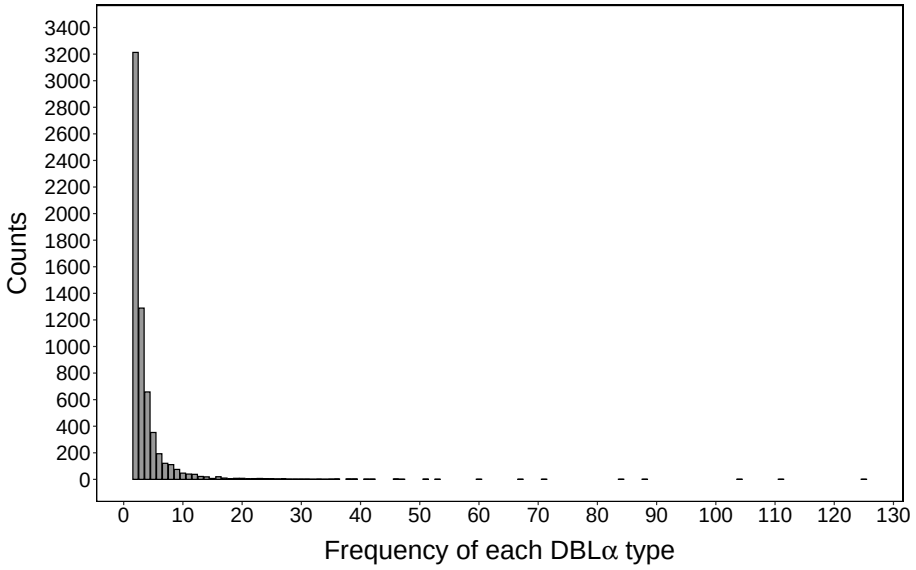


FIGURE A.14: **Histogram of DBLα type frequency.** We counted the frequency of each DBLα type present in isolates using the trimmed binary presence/absence matrix.

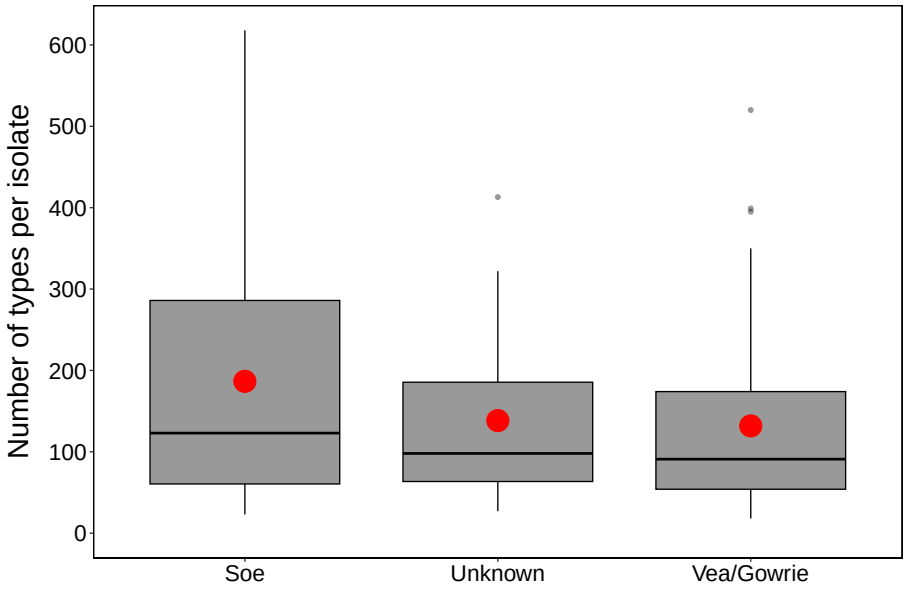


FIGURE A.15: **Number of DBLα types per isolate in Ghana.** We counted the number of DBLα types per isolate from the trimmed binary presence/absence matrix and used a boxplot to describe the corresponding distribution for each catchment area. Mean values were highlighted by red dots.

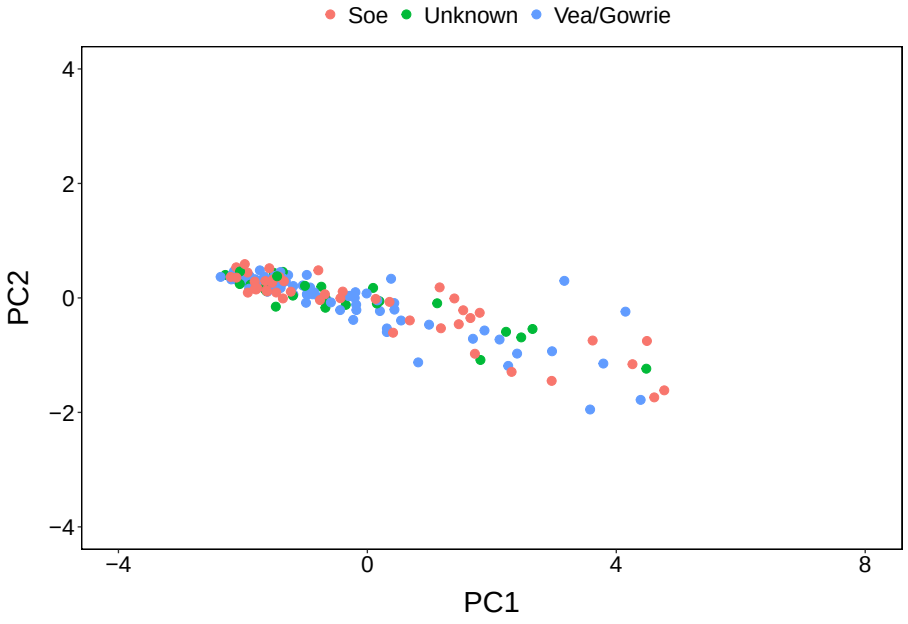


FIGURE A.16: **PCA plot from the trimmed binary presence/absence matrix of DBLα types.** Each dot represents an isolate, colored by locations. We found there is no clear separation between catchment areas from the first six components including first two components.

A.3 Conclusion

In this appendix, we have applied the JHMM [54] and the methods of Tonkin-Hill et al. [62] to a new dataset of DBLα tags to understand the underlying recombination patterns in the data and how the population structure observed in a worldwide dataset translates to a more localized dataset.

We systematically analyzed the patterns of DBLα recombination through two different datasets. We found uneven breakpoint distribution along the DBLα tag, with more recombination in HB regions. This is consistent with the findings in Tonkin-Hill et al. [62]. We also found the number of parental sequences varied among recombinants, with most recombinants having two parents only. The lack of horizontal jumps indicates the recombination occurs between homologous locations. Moreover, although these two datasets differed in scope and sequencing technology, we found the recombination patterns were similar across dataset. These are all novel findings. Importantly, these results have inspired us to take further investigations of DBLα recombination, as shown in Chapter 3 and 4.

We also explored the local DBLα population structure using the same methods in Tonkin-Hill et al. [62] with minor adjustments. Although we did not find the generated phylogenetic trees stratified by catchment areas of isolates, the Ghana data we used might be under-representative for exploring the local structure. One interesting direction would be collecting DBLα tags from more districts within a country, we think the results might provide a finer resolution compared with the existing global DBLα population structure.

A.4 Methods

The generation of DBLα tags in this study is described elsewhere [70, 74, 84]. Briefly, after obtaining the informed consent from each participant, finger-prick blood samples were collected as dried blood spots for laboratory molecular analyses and the identification of parasite species. They studied the individuals with asymptomatic

malaria and positive *P. falciparum* infections tested by microscopy or *18S rRNA* polymerase chain reactions (PCRs) [74, 84]. From the dried blood spots of these individuals, genomic DNA was extracted using QIAamp^{TM} DNA Mini Kit (Qiagen, Valencia and CA). The DBLα domains of *var* genes from genomic DNA were then ready for PCR amplification. The universal degenerate primer sequences [47, 266] were used for this process. The amplicons were then pooled and sequenced with Illumina MiSeq sequencer [70], and finally corresponding DBLα tags were identified from Illumina paired end data with a custom python pipeline [267]. We accessed 35,591 DBLα tags from the Ghana pilot study, and these tags are publicly available (see the Data availability section).

We note that all the methods for generating phylogenetic trees and the pre-process step of the data for implementing the JHMM are all from Tonkin-Hill et al. [62]. Therefore, we did not introduce new methods in this appendix. Below we summarize these methods.

A.4.1 Binary type analysis

For generating the binary presence/absence matrix, we followed the standard pipeline used in [62, 74] by clustering DBLα tags to representative DBLα tags ('types'). This was done by a series of commands of USEARCH $v8.1.1861\_i86osx32$ [216] software suite. Specifically, the tags were clustered into centroids based on 96% similarity using the `cluster_fast` command after removing the duplicates with the `derep_prefix` command. The threshold of 96% identity is a standard part of the DBLα preprocessing pipeline [204] and has been used in many studies ([67, 78, 195, 196, 202]). Here "removing the duplicates" means that a sequence A is discarded if it is a prefix of another sequence B in the set. This filtered out both the fragments and full-length duplicated DBLα tags [268]. Afterwards, by searching the original unfiltered tags against the centroids using the `usearch_global` command, a binary presence/absence matrix was generated. Each row of this matrix refers to a DBLα type, and each column represents an isolate. Every entry in this matrix describes whether the DBLα type is present (1) or absent (0) in the isolate.

This binary matrix was trimmed prior to the downstream analysis. Specifically, 2 isolates (1.2% of entire isolates) with less than 20 DBLα types were excluded due to the limited number of types per isolate, and singleton types (occurring in only one isolate, 64.8% of entire DBLα types) were also removed as we were more interested in the relationships between isolates. Eventually, we obtained a binary presence/absence matrix containing $6,311$ DBLα types distributed among 159 isolates.

A.4.2 Feature frequency profile analysis

For determining the window size and calculating the distance matrix of isolates, we followed the feature frequency profile (FFP) method [222]. We first chose the 3D7 isolate sequences as the reference. The whole genome sequences of *P. falciparum* clone 3D7 have been investigated for years [269–274] and are trustworthy [45]. To calculate the optimal window size k based on the reference sequences, we computed its lower and upper bounds using methods in [222].

1. Lower bound is determined by the value which corresponds to the largest number of unique vocabulary features (features that occur at least twice in the sequence). A k below this lower bound yields unreliable tree topologies.

2. Upper bound is determined by cumulative relative entropy (CRE), measured by the difference between the expected feature frequency and the observed one. When the CRE converges to zero, the expected feature frequency is sufficiently accurate and larger k is redundant.

In Figure A.17, the left panel determines the lower bound of k, which is 17, and the right panel shows the upper bound of k, i.e., 22, so the range of k is between 17 and 22. Therefore, $k = 20$ is a sensible choice.

We then calculated the distance matrix with the determined k. First, concatenating the sequences of each isolate, we counted all the possible length-k segments by the sliding-window technique. These raw frequency counts were then normalized to a probability distribution vector (or FFP) per isolate. Next, we computed the distance

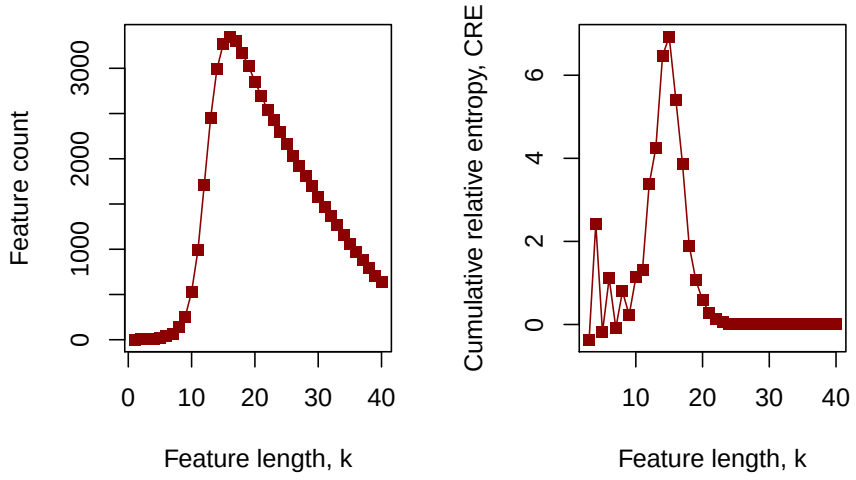


FIGURE A.17: **Determining the window size k.** The lower bound of k corresponds to the largest feature count, while the upper bound of k is the value when the CRE starts to converge to 0.

between any two isolates via the Jensen-Shannon (JS) Divergence [275] between the corresponding FFPs. Although there are multiple distance measures, the advantage of JS divergence is that the resulting distance matrix is symmetric, and each value in the matrix is positive, which is essential for tree construction using distance-based methods. Finally, we used FastME $v2.0$ [262] with default settings and the R package `ggtree` [276] for generating and visualizing the resultant tree respectively, which is shown in Figure A.12.

A.4.3 Data pre-processing for the JHMM

There are several pre-processing steps before implementing the JHMM. Following the standard pipeline [62], we first removed the non-translatable tags, i.e, tags containing a stop codon. Next, we clustered the DNA tags at 96% similarity threshold (see above Section A.4.1) to obtain DBLα types. Finally, we extracted the protein sequences that correspond to the DBLα types, resulting in $17,335$ representative protein sequences as the input to the JHMM [54]. In contrast, there were $31,946$ representative protein sequences generated from the global data. The reason we

chose the amino acid instead of DNA sequences as input to the JHMM is for computational purpose (as protein sequences are shorter). The ratio of tags to types in the Ghana data is lower than that of global data (2.05 vs. 3.08), suggesting greater overlap among global DBLα tags than the tags from Ghana locally.

Data availability

The DBLα tags from Ghana pilot study are publicly available (GenBank BioProject Number: PRJNA396962). Scripts and data are available on Github (`https://github.com/qianfeng2/Ghana_data_analysis`) or Zenodo (`https://zenodo.org/doi/10.5281/zenodo.13764593`).

Bibliography

[1] Kathryn N. Suh, Kevin C. Kain, and Jay S. Keystone. Malaria. *CMAJ*, 170 (11):1693–1702, 2004.

[2] Jasminka Talapko, Ivana Škrlec, Tamara Alebić, Melita Jukić, and Aleksandar Včev. Malaria: the past and the present. *Microorganisms*, 7(6):179, 2019.

[3] Durgaprasad Subramanian, Kenneth J Moise Jr, and A Clinton White Jr. Imported malaria in pregnancy: report of four cases and review of management. *Clin. Infect. Dis.*, 15(3):408–413, 1992.

[4] Nazanin Ghazanfari, Scott N Mueller, and William R Heath. Cerebral malaria in mouse and man. *Front. Immunol.*, 9:2016, 2018.

[5] World Health Organization et al. World malaria report 2023. 2023.

[6] S David Shahbodaghi and Nicholas A Rathjen. Malaria: Prevention, diagnosis, and treatment. *Am Fam Physician*, 106(3):270–278, 2022.

[7] Julianna Schantz-Dunn and Nawal M Nour. Malaria and pregnancy: a global health perspective. *Rev Obstet Gynecol.*, 2(3):186, 2009.

[8] Whitney E Harrington, Sami B Kanaan, Atis Muehlenbachs, Robert Morrison, Philip Stevenson, Michal Fried, Patrick E Duffy, and J Lee Nelson. Maternal microchimerism predicts increased infection but decreased disease due to *Plasmodium falciparum* during early childhood. *J. Infect. Dis.*, 215(9):1445–1451, 2017.

[9] Cdc.gov. (2021). CDC - malaria - about malaria - malaria's impact worldwide, 2021. URL `https://www.cdc.gov/malaria/malaria_worldwide/impact.html`. Accessed: 2023-09-06.

[10] World Health Organization. *Global technical strategy for malaria 2016-2030*. WHO, 2015.

[11] World Health Organization et al. Zeroing in on malaria elimination: final report of the E-2020 initiative. 2021.

[12] Liping Gao, Qi Shi, Zhiguo Liu, Zhenjun Li, and Xiaoping Dong. Impact of the COVID-19 pandemic on malaria control in Africa: a preliminary analysis. *Trop. Med. Infect. Dis.*, 8(1):67, 2023.

[13] Stephen J Rogerson, James G Beeson, Moses Laman, Jeanne Rini Poespoprodjo, Timothy William, Julie A Simpson, and Ric N Price. Identifying and combating the impacts of COVID-19 on malaria. *BMC Med.*, 18(1):1–7, 2020.

[14] Anna-Katharina Heuschen, Guangyu Lu, Oliver Razum, Alhassan Abdul-Mumin, Osman Sankoh, Lorenz Von Seidlein, Umberto D'Alessandro, and Olaf Müller. Public health-relevant consequences of the COVID-19 pandemic on malaria in sub-Saharan Africa: a scoping review. *Malar. J.*, 20:1–16, 2021.

[15] Jiwook Park, Seungwoo Kang, Dayoung Seok, Yae Jee Baek, Se Young An, Junga Lee, Alina Jun, and Sun-Young Kim. Barriers against and strategies for malaria control during the COVID-19 pandemic in low-and middle-income countries: a systematic review. *Malar. J.*, 22(1):41, 2023.

[16] Daniel J Weiss, Amelia Bertozzi-Villa, Susan F Rumisha, Punam Amratia, Rohan Arambepola, Katherine E Battle, Ewan Cameron, Elisabeth Chestnutt, Harry S Gibson, Joseph Harris, et al. Indirect effects of the COVID-19 pandemic on malaria intervention coverage, morbidity, and mortality in Africa: a geospatial modelling analysis. *Lancet Infect. Dis.*, 21(1):59–69, 2021.

[17] Mogahed Ismail Hassan Hussein, Ahmed Abdalazim Dafallah Albashir, Omer Ali Mohamed Ahmed Elawad, and Anmar Homeida. Malaria and COVID-19: unmasking their ties. *Malar. J.*, 19(1):1–10, 2020.

[18] Cleveland Clinic, 2022. URL `https://my.clevelandclinic.org/health/diseases/15014-malaria`. Accessed: 2023-09-07.

[19] Elizabeth A Winzeler and Micah J Manary. Drug resistance genomics of the antimalarial drug artemisinin. *Genome Biol.*, 15(11):1–12, 2014.

[20] Melissa D Conrad and Philip J Rosenthal. Antimalarial drug resistance in Africa: the calm before the storm? *Lancet Infect. Dis.*, 19(10):e338–e351, 2019.

[21] Matthew M Ippolito, Kara A Moser, Jean-Bertin Bukasa Kabuya, Clark Cunningham, and Jonathan J Juliano. Antimalarial drug resistance and implications for the WHO global technical strategy. *Curr. Epidemiol. Rep.*, 8(2): 46–62, 2021.

[22] Shigeharu Sato. Plasmodium—a brief introduction to the parasites causing human malaria and their basic biology. *J. Physiol. Anthropol.*, 40(1):1–13, 2021.

[23] Doudou M Yobi, Nadine K Kayiba, Dieudonné M Mvumbi, Raphael Boreux, Pius Z Kabututu, Pierre Z Akilimali, Hippolyte NT Situakibanza, Patrick De Mol, Niko Speybroeck, Georges L Mvumbi, et al. Biennial surveillance of *Plasmodium falciparum* anti-malarial drug resistance markers in democratic republic of congo, 2017 and 2019. *BMC Infect. Dis.*, 22(1):1–9, 2022.

[24] Cdc.gov. (2020). CDC - malaria - about malaria - how to reduce malaria's impact, 2020. URL `https://www.cdc.gov/malaria/malaria_worldwide/reduction/drug_resistance.html`. Accessed: 2023-09-07.

[25] Patrícia Feitosa Souza, Diego Ricardo Xavier, Martha Cecilia Suarez Mutis, Jurema Corrêa da Mota, Paulo Cesar Peiter, Vanderlei Pascoal de Matos, Mônica de Avelar Figueiredo Mafra Magalhães, and Christovam Barcellos. Spatial spread of malaria and economic frontier expansion in the Brazilian Amazon. *PLoS One*, 14(6):e0217615, 2019.

[26] Cdc.gov. (2020). CDC - malaria - about malaria - biology - malaria parasites., 2020. URL `https://www.cdc.gov/malaria/about/biology/#tabs-1-6#`. Accessed: 2023-09-07.

[27] Chris Lambros and Jerome P Vanderberg. Synchronization of *Plasmodium falciparum* erythrocytic stages in culture. *J. Parasitol. Res.*, pages 418–420, 1979.

[28] Sunetra Gupta, AV Hill, Dominic Kwiatkowski, Alice M Greenwood, Brian M Greenwood, and Karen P Day. Parasite virulence and disease patterns in *Plasmodium falciparum* malaria. *Proc. Natl. Acad. Sci.*, 91(9):3715–3719, 1994.

[29] Deirdre A Joy, Xiaorong Feng, Jianbing Mu, Tetsuya Furuya, Kesinee Chotivanich, Antoniana U Krettli, May Ho, Alex Wang, Nicholas J White, Edward Suh, et al. Early origin and recent expansion of *Plasmodium falciparum*. *Science*, 300(5617):318–321, 2003.

[30] John H Adams and Ivo Mueller. The biology of *Plasmodium vivax*. *Cold Spring Harb. perspect. med.*, 7(9), 2017.

[31] William E Collins and Geoffrey M Jeffery. *Plasmodium ovale*: parasite and disease. *Clin. Microbiol. Rev.*, 18(3):570–581, 2005.

[32] Colin J Sutherland, Naowarat Tanomsing, Debbie Nolder, Mary Oguike, Charlie Jennison, Sasithon Pukrittayakamee, Christiane Dolecek, Tran Tinh Hien, Virgilio E Do Rosário, Ana Paula Arez, et al. Two nonrecombining sympatric forms of the human malaria parasite *Plasmodium ovale* occur globally. *J. Infect. Dis.*, 201(10):1544–1550, 2010.

[33] William E Collins and Geoffrey M Jeffery. *Plasmodium malariae*: parasite and disease. *Clin. Microbiol. Rev.*, 20(4):579–592, 2007.

[34] NJ White. *Plasmodium knowlesi*: the fifth human malaria parasite. *Clin. Infect. Dis.*, 46(2):172–173, 2008.

[35] Weimin Liu, Yingying Li, Gerald H Learn, Rebecca S Rudicell, Joel D Robertson, Brandon F Keele, Jean-Bosco N Ndjango, Crickette M Sanz, David B Morgan, Sabrina Locatelli, et al. Origin of the human malaria parasite *Plasmodium falciparum* in gorillas. *Nature*, 467(7314):420–425, 2010.

[36] Thomas D Otto, Aude Gilabert, Thomas Crellen, Ulrike Böhme, Céline Arnathau, Mandy Sanders, Samuel O Oyola, Alain Prince Okouga, Larson Boundenga, Eric Willaume, et al. Genomes of all known members of a *Plasmodium* subgenus reveal paths to virulent human malaria. *Nat. Microbiol.*, 3 (6):687–697, 2018.

[37] Cdc.gov. (2018). CDC - malaria - about malaria - biology - how to reduce malaria's impact., 2018. URL `https://www.cdc.gov/malaria/malaria_worldwide/reduction/drug_resistance.html`. Accessed: 2023-09-08.

[38] Matthew B Laurens. RTS,S/AS01 vaccine (mosquirix™): an overview. *Hum. Vaccines Immunother.*, 16(3):480–489, 2020.

[39] Mehreen S Datoo, Hamtandi Magloire Natama, Athanase Somé, Duncan Bellamy, Ousmane Traoré, Toussaint Rouamba, Marc Christian Tahita, N Félix André Ido, Prisca Yameogo, Daniel Valia, et al. Efficacy and immunogenicity of R21/Matrix-M vaccine against clinical malaria after 2 years'

follow-up in children in Burkina Faso: a phase 1/2b randomised controlled trial. *Lancet Infect. Dis.*, 22(12):1728–1736, 2022.

[40] Noa D Pasternak and Ron Dzikowski. PfEMP1: an antigen that plays a key role in the pathogenicity and immune evasion of the malaria parasite *Plasmodium falciparum*. *Int. J. Biochem. Cell Biol.*, 41(7):1463–1466, 2009.

[41] Richard Idro, Kevin Marsh, Chandy C John, and Charles RJ Newton. Cerebral malaria: mechanisms of brain injury and strategies for improved neurocognitive outcome. *Pediatr. Res.*, 68(4):267–274, 2010.

[42] Antoine Claessens, William L Hamilton, Mihir Kekre, Thomas D Otto, Adnan Faizullabhoy, Julian C Rayner, and Dominic Kwiatkowski. Generation of antigenic diversity in *Plasmodium falciparum* by structured rearrangement of var genes during mitosis. *PLoS Genet.*, 10(12):e1004812, 2014.

[43] Yourgenome. Malaria: the master of disguise, 2021. URL `https://www.yourgenome.org/stories/malaria-the-master-of-disguise/`. Accessed: 2023-09-08.

[44] Thomas D Otto, Ulrike Böhme, Mandy Sanders, Adam Reid, Ellen I Bruske, Craig W Duffy, Pete C Bull, Richard D Pearson, Abdirahman Abdi, Sandra Dimonte, et al. Long read assemblies of geographically dispersed *Plasmodium falciparum* isolates reveal highly structured subtelomeres. *Wellcome Open Res.*, 3, 2018.

[45] Malcolm J Gardner, Neil Hall, Eula Fung, Owen White, Matthew Berriman, Richard W Hyman, Jane M Carlton, Arnab Pain, Karen E Nelson, Sharen Bowman, et al. Genome sequence of the human malaria parasite *Plasmodium falciparum*. *Nature*, 419(6906):498, 2002.

[46] Thomas S Rask, Daniel A Hansen, Thor G Theander, Anders Gorm Pedersen, and Thomas Lavstsen. *Plasmodium falciparum* erythrocyte membrane protein 1 diversity in seven genomes–divide and conquer. *PLoS Comput. Biol.*, 6(9): e1000933, 2010.

[47] Helen M Taylor, Susan A Kyes, David Harris, Neline Kriek, and Christopher I Newbold. A study of var gene transcription in vitro using universal var gene primers. *Mol. Biochem. Parasitol.*, 105(1):13–23, 2000.

[48] Joseph D Smith, Gangadharan Subramanian, Benoit Gamain, Dror I Baruch, and Louis H Miller. Classification of adhesive domains in the *Plasmodium falciparum* erythrocyte membrane protein 1 family. *Mol. Biochem. Parasitol.*, 110(2):293–310, 2000.

[49] Anand Srivastava, Stéphane Gangnard, Adam Round, Sébastien Dechavanne, Alexandre Juillerat, Bertrand Raynal, Grazyna Faure, Bruno Baron, Stéphanie Ramboarina, Saurabh Kumar Singh, et al. Full-length extracellular region of the var2CSA variant of PfEMP1 is required for specific, high-affinity binding to CSA. *Proc. Natl. Acad. Sci.*, 107(11):4884–4889, 2010.

[50] Antoine Dara, Mark A Travassos, Matthew Adams, Sarah Schaffer DeRoo, Elliott F Drábek, Sonia Agrawal, Miriam K Laufer, Christopher V Plowe, and Joana C Silva. A new method for sequencing the hypervariable *Plasmodium falciparum* gene var2csa from clinical samples. *Malar. J.*, 16(1):1–9, 2017.

[51] Ernest Diez Benavente, Damilola R Oresegun, Paola Florez de Sessions, Eloise M Walker, Cally Roper, Jamille G Dombrowski, Rodrigo M de Souza, Claudio RF Marinho, Colin J Sutherland, Martin L Hibberd, et al. Global genetic diversity of var2csa in *Plasmodium falciparum* with implications for malaria in pregnancy and vaccine development. *Sci. Rep.*, 8(1):15429, 2018.

[52] Selina ER Bopp, Micah J Manary, A Taylor Bright, Geoffrey L Johnston, Neekesh V Dharia, Fabio L Luna, Susan McCormack, David Plouffe, Case W McNamara, John R Walker, et al. Mitotic evolution of *Plasmodium falciparum* shows a stable core genome but recombination in antigen families. *PLoS Genet.*, 9(2):e1003293, 2013.

[53] Lúcio H Freitas-Junior, Emmanuel Bottius, Lindsay A Pirrit, Kirk W Deitsch, Christine Scheidig, Francoise Guinet, Ulf Nehrbass, Thomas E Wellems, and Artur Scherf. Frequent ectopic recombination of virulence factor genes in telomeric chromosome clusters of *P. falciparum*. *Nature*, 407(6807):1018–1022, 2000.

[54] Martine M Zilversmit, Ella K Chase, Donald S Chen, Philip Awadalla, Karen P Day, and Gil McVean. Hypervariable antigen genes in malaria have ancient roots. *BMC Evol. Biol.*, 13(1):110, 2013.

[55] Sarah C Gilbert, Magdalena Plebanski, Sunetra Gupta, Joanne Morris, Martin Cox, Michael Aidoo, Dominic Kwiatkowski, Brian M Greenwood, Hilton C

Whittle, and Adrian VS Hill. Association of malaria parasite population structure, HLA, and immunological antagonism. *Science*, 279(5354):1173–1177, 1998.

[56] Helen M Taylor, Susan A Kyes, Christopher I Newbold, et al. Var gene diversity in *Plasmodium falciparum* is generated by frequent recombination events. *Mol. Biochem. Parasitol.*, 110(2):391–397, 2000.

[57] Michael F Duffy, Timothy J Byrne, Celine Carret, Alasdair Ivens, and Graham V Brown. Ectopic recombination of a malaria var gene during mitosis associated with an altered var switch rate. *J. Mol. Biol.*, 389(3):453–469, 2009.

[58] Eliana Real, Flore Nardella, Artur Scherf, and Liliana Mancio-Silva. Repurposing of *Plasmodium falciparum* var genes beyond the blood stage. *COMICR*, 70:102207, 2022.

[59] Thomas D Otto, Sammy A Assefa, Ulrike Böhme, Mandy J Sanders, Dominic Kwiatkowski, et al. Evolutionary analysis of the most polymorphic gene family in falciparum malaria. *Wellcome Open Res.*, 4, 2019.

[60] Xu Zhang, Noah Alexander, Irina Leonardi, Christopher Mason, Laura A Kirkman, and Kirk W Deitsch. Rapid antigen diversification through mitotic recombination in the human malaria parasite *Plasmodium falciparum*. *PLoS Biol.*, 17(5):e3000271, 2019.

[61] Susan M Kraemer, Sue A Kyes, Gautam Aggarwal, Amy L Springer, Siri O Nelson, Zoe Christodoulou, Leia M Smith, Wendy Wang, Emily Levin, Christopher I Newbold, et al. Patterns of gene recombination shape var gene repertoires in *Plasmodium falciparum*: comparisons of geographically diverse isolates. *BMC Genom.*, 8(1):45, 2007.

[62] Gerry Tonkin-Hill, Shazia Ruybal-Pesántez, Kathryn E Tiedje, Virginie Rougeron, Michael F Duffy, Sedigheh Zakeri, Tepanata Pumpaibool, Pongchai Harnyuttanakorn, OraLee H Branch, Lastenia Ruiz-Mesía, et al. Evolutionary analyses of the major variant surface antigen-encoding genes reveal population structure of *Plasmodium falciparum* within and between continents. *PLoS Genet.*, 17(2):e1009269, 2021.

[63] David A Morrison. Phylogenetic networks: a review of methods to display evolutionary history. *Annu. Res. Rev. Biol.*, 4(10):1518–1543, 2014.

[64] Louxin Zhang, Niloufar Abhari, Caroline Colijn, and Yufeng Wu. A fast and scalable method for inferring phylogenetic networks from trees by aligning lineage taxon strings. *Genome Res.*, 33(7):1053–1060, 2023.

[65] Vincent Berry, Celine Scornavacca, and Mathias Weller. Scanning phylogenetic networks is NP-hard. In *SOFSEM 2020: Theory and Practice of Computer Science: 46th International Conference on Current Trends in Theory and Practice of Informatics, SOFSEM 2020, Limassol, Cyprus, January 20–24, 2020, Proceedings 46*, pages 519–530. Springer, 2020.

[66] Samuel Ayalew Assefa. *De novo assembly of the var multi-gene family in clinical samples of Plasmodium falciparum*. PhD thesis, University of Cambridge, 2013.

[67] Sofonias K Tessema, Rie Nakajima, Algis Jasinskas, Stephanie L Monk, Lea Lekieffre, Enmoore Lin, Benson Kiniboro, Carla Proietti, Peter Siba, Philip L Felgner, et al. Protective immunity against severe malaria in children is associated with a limited repertoire of antibodies to conserved PfEMP1 variants. *Cell Host Microbe*, 26(5):579–590, 2019.

[68] Irwin Sherman. *Reflections on a century of malaria biochemistry*, volume 67. Academic Press, 2011.

[69] George Githinji and Peter C Bull. A re-assessment of gene-tag classification approaches for describing var gene expression patterns during human *Plasmodium falciparum* malaria parasite infections. *Wellcome Open Res.*, 2, 2017.

[70] Qixin He, Shai Pilosof, Kathryn E Tiedje, Shazia Ruybal-Pesántez, Yael Artzy-Randrup, Edward B Baskerville, Karen P Day, and Mercedes Pascual. Networks of genetic similarity reveal non-neutral processes shape strain structure in *Plasmodium falciparum*. *Nat. Commun.*, 9(1):1–12, 2018.

[71] Peter C Bull, Caroline O Buckee, Sue Kyes, Moses M Kortok, Vandana Thathy, Bernard Guyah, José A Stoute, Chris I Newbold, and Kevin Marsh. *Plasmodium falciparum* antigenic variation. Mapping mosaic var gene sequences onto a network of shared, highly polymorphic sequence blocks. *Mol. Microbiol.*, 68 (6):1519–1534, 2008.

[72] Kivisi C Andisi and Abdirahman I Abdi. Analysis of var gene transcription pattern using DBLα tags. In *Malaria Immunology: Targeting the Surface of Infected Erythrocytes*, pages 173–184. Springer, 2022.

[73] Mary M Rorick, Yael Artzy-Randrup, Shazia Ruybal-Pesántez, Kathryn E Tiedje, Thomas S Rask, Abraham Oduro, Anita Ghansah, Kwadwo Koram, Karen P Day, and Mercedes Pascual. Signatures of competition and strain structure within the major blood-stage antigen of *Plasmodium falciparum* in a local community in ghana. *Ecol. Evol.*, 8(7):3574–3588, 2018.

[74] Shazia Ruybal-Pesántez, Kathryn E Tiedje, Gerry Tonkin-Hill, Thomas S Rask, Moses R Kamya, Bryan Greenhouse, Grant Dorsey, Michael F Duffy, and Karen P Day. Population genomics of virulence genes of *Plasmodium falciparum* in clinical isolates from Uganda. *Sci. Rep.*, 7(1):11810, 2017.

[75] Shazia Ruybal-Pesántez, Kathryn E Tiedje, Shai Pilosof, Gerry Tonkin-Hill, Qixin He, Thomas S Rask, Lucas Amenga-Etego, Abraham R Oduro, Kwadwo A Koram, Mercedes Pascual, et al. Age-specific patterns of DBLα var diversity can explain why residents of high malaria transmission areas remain susceptible to *Plasmodium falciparum* blood stage infection throughout life. *Int. J. Parasitol.*, 52(11):721–731, 2022.

[76] Kathryn E Tiedje, Qi Zhan, Shazia Ruybal-Pésantez, Gerry Tonkin-Hill, Qixin He, Mun Hua Tan, Dionne C Argyropoulos, Samantha L Deed, Anita Ghansah, Oscar Bangre, et al. Measuring changes in *Plasmodium falciparum* census population size in response to sequential malaria control interventions. *Elife*, 12, 2023.

[77] Gavin Mackenzie, Rasmus W Jensen, Thomas Lavstsen, and Thomas D Otto. Varia: a tool for prediction, analysis and visualisation of variable genes. *BMC Bioinform.*, 23(1):52, 2022.

[78] Mun Hua Tan, Heejung Shim, Yao-ban Chan, and Karen P Day. Unravelling var complexity: relationship between DBLα types and var genes in *Plasmodium falciparum*. *Front. Parasitol.*, 1:1006341, 2023.

[79] Mark A Larkin, Gordon Blackshields, Nigel P Brown, R Chenna, Paul A McGettigan, Hamish McWilliam, Franck Valentin, Iain M Wallace, Andreas Wilm, Rodrigo Lopez, et al. Clustal W and Clustal X version 2.0. *Bioinformatics*, 23(21):2947–2948, 2007.

[80] C Nelson Hayes, Diego Diez, Nicolas Joannin, Wataru Honda, Minoru Kanehisa, Mats Wahlgren, Craig E Wheelock, and Susumu Goto. varDB: a

pathogen-specific sequence database of protein families involved in antigenic variation. *Bioinformatics*, 24(21):2564–2565, 2008.

[81] Daniel B Larremore, Sesh A Sundararaman, Weimin Liu, William R Proto, Aaron Clauset, Dorothy E Loy, Sheri Speede, Lindsey J Plenderleith, Paul M Sharp, Beatrice H Hahn, et al. Ape parasite origins of human malaria virulence genes. *Nat. Commun.*, 6(1):8368, 2015.

[82] Mary M Rorick, Thomas S Rask, Edward B Baskerville, Karen P Day, and Mercedes Pascual. Homology blocks of *Plasmodium falciparum* var genes and clinically distinct forms of severe malaria in a local population. *BMC Microbiol.*, 13(1):244, 2013.

[83] Shazia Ruybal-Pesántez, Kathryn E Tiedje, Mary M Rorick, Lucas Amenga-Etego, Anita Ghansah, Abraham R Oduro, Kwadwo A Koram, and Karen P Day. Lack of geospatial population structure yet significant linkage disequilibrium in the reservoir of *Plasmodium falciparum* in Bongo District, Ghana. *Am. J. Trop. Med. Hyg.*, 97(4):1180–1189, 2017.

[84] Kathryn E Tiedje, Abraham R Oduro, Godfred Agongo, Thomas Anyorigiya, Daniel Azongo, Timothy Awine, Anita Ghansah, Mercedes Pascual, Kwadwo A Koram, and Karen P Day. Seasonal variation in the epidemiology of asymptomatic *Plasmodium falciparum* infections across two catchment areas in Bongo District, Ghana. *Am. J. Trop. Med. Hyg.*, 97(1):199–212, 2017.

[85] Shai Pilosof, Qixin He, Kathryn E Tiedje, Shazia Ruybal-Pesántez, Karen P Day, and Mercedes Pascual. Competition for hosts modulates vast antigenic diversity to generate persistent strain structure in *Plasmodium falciparum*. *PLoS Biol.*, 17(6):e3000336, 2019.

[86] B Ampadu, BA Akurugu, MS Zango, SK Abanyie, S Ampofo, et al. Assessing the impact of a dam on the livelihood of surrounding communities: a case study of Vea Dam in the Upper East Region of Ghana. *J. Environ. Earth Sci.*, 5(4):20–25, 2015.

[87] Graham J Etherington, Jo Dicks, and Ian N Roberts. Recombination Analysis Tool (RAT): a program for the high-throughput detection of recombination. *Bioinformatics*, 21(3):278–281, 2005.

[88] Kavita S Lole, Robert C Bollinger, Ramesh S Paranjape, Deepak Gadkari, Smita S Kulkarni, Nicole G Novak, Roxann Ingersoll, Haynes W Sheppard,

and Stuart C Ray. Full-length human immunodeficiency virus type 1 genomes from subtype C-infected seroconverters in India, with evidence of intersubtype recombination. *J. Virol.*, 73(1):152–160, 1999.

[89] Adam C Siepel, Aaron L Halpern, Catherine Macken, and Bette TM Korber. A computer program designed to screen rapidly for HIV type 1 intersubtype recombinant sequences. *AIDS Res. Hum. Retrovir.*, 11(11):1413–1416, 1995.

[90] Wah-Heng Lee and Wing-Kin Sung. RB-finder: an improved distance-based sliding window method to detect recombination breakpoints. *J. Comput. Biol.*, 15(7):881–898, 2008.

[91] Mika O Salminen, Jean K Carr, Donald S Burke, and Francine E McCutchan. Identification of breakpoints in intergenotypic recombinants of HIV type 1 by bootscanning. *AIDS Res. Hum. Retrovir.*, 11(11):1423–1425, 1995.

[92] DP Martin, D Posada, KA Crandall, and C Williamson. A modified bootscan algorithm for automated identification of recombinant sequences and recombination breakpoints. *J. Virol.*, 95(18), 2005.

[93] Dirk Husmeier and Gráinne McGuire. Detecting recombination in 4-taxa DNA sequence alignments with Bayesian hidden Markov models and Markov chain Monte Carlo. *Mol. Biol. Evol.*, 20(3):315–337, 2003.

[94] Xavier Didelot and Daniel J Wilson. ClonalFrameML: efficient inference of recombination in whole bacterial genomes. *PLoS Comput. Biol.*, 11(2):e1004041, 2015.

[95] Yatish Turakhia, Bryan Thornlow, Angie Hinrichs, Jakob McBroome, Nicolas Ayala, Cheng Ye, Kyle Smith, Nicola De Maio, David Haussler, Robert Lanfear, et al. Pandemic-Scale Phylogenomics Reveals The SARS-CoV-2 Recombination Landscape. *Nature*, pages 1–2, 2022.

[96] John Maynard Smith. Analyzing the mosaic structure of genes. *J. Mol. Evol.*, 34(2):126–129, 1992.

[97] David Posada and Keith A Crandall. Evaluation of methods for detecting recombination from DNA sequences: computer simulations. *Proc. Natl. Acad. Sci.*, 98(24):13757–13762, 2001.

[98] Georg F Weiller. Phylogenetic profiles: a graphical method for detecting genetic recombinations in homologous sequences. *Mol. Biol. Evol.*, 15(3):326–335, 1998.

[99] Maciej F Boni, David Posada, and Marcus W Feldman. An exact nonparametric method for inferring mosaic structure in sequence triplets. *Genetics*, 176(2):1035–1047, 2007.

[100] Ha Minh Lam, Oliver Ratmann, and Maciej F Boni. Improved algorithmic complexity for the 3SEQ recombination detection algorithm. *Mol. Biol. Evol.*, 35(1):247–251, 2018.

[101] Nicholas J Croucher, Andrew J Page, Thomas R Connor, Aidan J Delaney, Jacqueline A Keane, Stephen D Bentley, Julian Parkhill, and Simon R Harris. Rapid phylogenetic analysis of large samples of recombinant bacterial whole genome sequences using Gubbins. *Nucleic Acids Res.*, 43(3):e15–e15, 2015.

[102] Darren P Martin, Ben Murrell, Michael Golden, Arjun Khoosal, and Brejnev Muhire. RDP4: Detection and analysis of recombination patterns in virus genomes. *Virus Evol.*, 1(1), 2015.

[103] Ingrid B Jakobsen and Simon Easteal. A program for calculating and displaying compatibility matrices as an aid in determining reticulate evolution in molecular sequences. *Bioinformatics*, 12(4):291–295, 1996.

[104] Ingrid B Jakobsen, Susan R Wilson, and Simon Easteal. The partition matrix: exploring variable phylogenetic signals along nucleotide sequence alignments. *Mol. Biol. Evol.*, 14(5):474–484, 1997.

[105] Yi-Pin Lai and Thomas R Ioerger. A compatibility approach to identify recombination breakpoints in bacterial and viral genomes. In *Proceedings of the 8th ACM International Conference on Bioinformatics, Computational Biology, and Health Informatics*, pages 11–20, 2017.

[106] Yi-Pin Lai and Thomas R Ioerger. A statistical method to identify recombination in bacterial genomes based on SNP incompatibility. *BMC Bioinform.*, 19(1):1–15, 2018.

[107] Mark J Gibbs, John S Armstrong, and Adrian J Gibbs. Sister-Scanning: a Monte Carlo procedure for assessing signals in recombinant sequences. *Bioinformatics*, 16(7):573–582, 2000.

[108] Stanley Sawyer. Statistical tests for detecting gene conversion. *Mol. Biol. Evol.*, 6(5):526–538, 1989.

[109] Darren Martin and Ed Rybicki. RDP: detection of recombination amongst aligned sequences. *Bioinformatics*, 16(6):562–563, 2000.

[110] Darren P Martin, Arvind Varsani, Philippe Roumagnac, Gerrit Botha, Suresh Maslamoney, Tiana Schwab, Zena Kelz, Venkatesh Kumar, and Ben Murrell. RDP5: A computer program for analysing recombination in, and removing signals of recombination from, nucleotide sequence datasets. *Virus Evol.*, 2020.

[111] Kristoffer Forslund, Daniel H Huson, and Vincent Moulton. VisRD—visual recombination detection. *Bioinformatics*, 20(18):3654–3655, 2004.

[112] Korbinian Strimmer, Kristoffer Forslund, Barbara Holland, and Vincent Moulton. A novel exploratory method for visual recombination detection. *Genome Biol.*, 4(5):1–12, 2003.

[113] Philippe Lemey, Martin Lott, Darren P Martin, and Vincent Moulton. Identifying recombinants in human and primate immunodeficiency virus sequence alignments using quartet scanning. *BMC Bioinform.*, 10(1):1–18, 2009.

[114] Andrew F Neuwald and Stephen F Altschul. Bayesian top-down protein sequence alignment with inferred position-specific gap penalties. *PLoS Comput. Biol.*, 12(5):e1004936, 2016.

[115] André Lin Ouédraogo, Bronner P Gonçalves, Awa Gnémé, Edward A Wenger, Moussa W Guelbeogo, Amathe Ouédraogo, Jaline Gerardin, Caitlin A Bever, Hil Lyons, Xavier Pitroipa, et al. Dynamics of the human infectious reservoir for malaria determined by mosquito feeding assays and ultrasensitive malaria diagnosis in burkina faso. *J. Infect. Dis.*, 213(1):90–99, 2016.

[116] Bronner P Gonçalves, Melissa C Kapulu, Patrick Sawa, Wamdaogo M Guelbéogo, Alfred B Tiono, Lynn Grignard, Will Stone, Joel Hellewell, Kjerstin Lanke, Guido JH Bastiaens, et al. Examining the human infectious reservoir for *Plasmodium falciparum* malaria in areas of differing transmission intensity. *Nat. Commun.*, 8(1):1133, 2017.

[117] Susan M Kraemer and Joseph D Smith. Evidence for the importance of genetic structuring to the structural and functional specialization of the *Plasmodium falciparum* var gene family. *Mol. Microbiol.*, 50(5):1527–1538, 2003.

[118] Susan M Kraemer and Joseph D Smith. A family affair: var genes, PfEMP1 binding, and malaria disease. *Curr. Opin. Microbiol.*, 9(4):374–380, 2006.

[119] Till S Voss, Mirjam Kaestli, Denise Vogel, Selina Bopp, and Hans-Peter Beck. Identification of nuclear proteins that interact differentially with *Plasmodium falciparum* var gene promoters. *Mol. Microbiol.*, 48(6):1593–1607, 2003.

[120] Na Li and Matthew Stephens. Modeling linkage disequilibrium and identifying recombination hotspots using single-nucleotide polymorphism data. *Genetics*, 165(4):2213–2233, 2003.

[121] Philippe Lemey, Marco Salemi, and Anne-Mieke Vandamme. *The phylogenetic handbook: a practical approach to phylogenetic analysis and hypothesis testing*. Cambridge University Press, 2009.

[122] Anton E Shikov, Yury V Malovichko, Anton A Nizhnikov, and Kirill S Antonets. Current methods for recombination detection in bacteria. *Int. J. Mol. Sci.*, 23(11):6257, 2022.

[123] Sergei L Kosakovsky Pond, David Posada, Michael B Gravenor, Christopher H Woelk, and Simon DW Frost. GARD: a genetic algorithm for recombination detection. *Bioinformatics*, 22(24):3096–3098, 2006.

[124] Jakob McBroome, Bryan Thornlow, Angie S Hinrichs, Alexander Kramer, Nicola De Maio, Nick Goldman, David Haussler, Russell Corbett-Detig, and Yatish Turakhia. A daily-updated database and tools for comprehensive SARS-CoV-2 mutation-annotated trees. *Mol. Biol. Evol.*, 38(12):5819–5824, 2021.

[125] Carsten Wiuf, Thomas Christensen, and Jotun Hein. A simulation study of the reliability of recombination detection methods. *Mol. Biol. Evol.*, 18(10): 1929–1939, 2001.

[126] Celeste J Brown, Ethan C Garner, A Keith Dunker, and Paul Joyce. The power to detect recombination using the coalescent. *Mol. Biol. Evol.*, 18(7): 1421–1424, 2001.

[127] Trevor C Bruen, Hervé Philippe, and David Bryant. A simple and robust statistical test for detecting the presence of recombination. *Genetics*, 172(4): 2665–2681, 2006.

[128] Gil McVean, Philip Awadalla, and Paul Fearnhead. A coalescent-based method for detecting and estimating recombination from gene sequences. *Genetics*, 160 (3):1231–1241, 2002.

[129] Walter J LeQuesne. Further studies based on the uniquely derived character concept. *Syst. Zool.*, 21(3):281–288, 1972.

[130] George F Estabrook and FR McMorris. When is one estimate of evolutionary relationships a refinement of another? *J. Math. Biol.*, 1980.

[131] Christopher A Meacham and George F Estabrook. Compatibility methods in systematics. *Annu Rev Ecol Evol Syst*, pages 431–446, 1985.

[132] Adam F Allred, Hilary Renshaw, Scott Weaver, Robert B Tesh, and David Wang. VIPR HMM: a hidden Markov model for detecting recombination with microbial detection microarrays. *Bioinformatics*, 28(22):2922–2929, 2012.

[133] Anne-Kathrin Schultz, Ming Zhang, Thomas Leitner, Carla Kuiken, Bette Korber, Burkhard Morgenstern, and Mario Stanke. A jumping profile Hidden Markov Model and applications to recombination sites in HIV and HCV genomes. *BMC Bioinform.*, 7(1):1–15, 2006.

[134] Anne-Kathrin Schultz, Ingo Bulla, Mariama Abdou-Chekaraou, Emmanuel Gordien, Burkhard Morgenstern, Fabien Zoulim, Paul Deny, and Mario Stanke. jpHMM: recombination analysis in viruses with circular genomes such as the hepatitis B virus. *Nucleic Acids Res.*, 40(W1):W193–W198, 2012.

[135] Meijun Gao and Kevin J Liu. Statistical analysis of GC-biased gene conversion and recombination hotspots in eukaryotic genomes: a phylogenetic hidden Markov model-based approach. In *Proceedings of the 12th ACM Conference on Bioinformatics, Computational Biology, and Health Informatics*, pages 1–24, 2021.

[136] Leonard E Baum and Ted Petrie. Statistical inference for probabilistic functions of finite state Markov chains. *Ann. Math. Stat.*, 37(6):1554–1563, 1966.

[137] Leonard E Baum and John Alonzo Eagon. An inequality with applications to statistical estimation for probabilistic functions of Markov processes and to a model for ecology. *Bull. Am. Math. Soc.*, 73(3):360–363, 1967.

[138] Leonard E Baum. Growth functions for trasformations on manifolds. *Pac. J. Math.*, 27(2):211–227, 1968.

[139] Leonard E Baum, Ted Petrie, George Soules, and Norman Weiss. A maximization technique occurring in the statistical analysis of probabilistic functions of Markov chains. *Ann. Math. Stat.*, 41(1):164–171, 1970.

[140] Leonard E Baum et al. An inequality and associated maximization technique in statistical estimation for probabilistic functions of Markov processes. *Inequal.*, 3(1):1–8, 1972.

[141] James Baker. The DRAGON system–An overview. *IEEE Trans. Signal Process*, 23(1):24–29, 1975.

[142] L Bahl and Frederick Jelinek. Decoding for channels with insertions, deletions, and substitutions with applications to speech recognition. *IEEE Trans. Inf. Theory*, 21(4):404–411, 1975.

[143] Frederick Jelinek, Lalit Bahl, and Robert Mercer. Design of a linguistic statistical decoder for the recognition of continuous speech. *IEEE Trans. Inf. Theory*, 21(3):250–256, 1975.

[144] Frederick Jelinek. Continuous speech recognition by statistical methods. *Proc. IEEE*, 64(4):532–556, 1976.

[145] Raimo Bakis. Continuous speech recognition via centisecond acoustic states. *J. Acoust. Soc. Am.*, 59(S1):S97–S97, 1976.

[146] Lawrence R Rabiner. A tutorial on hidden Markov models and selected applications in speech recognition. *Proc. IEEE.*, 77(2):257–286, 1989.

[147] MJ Bishop and Elizabeth A Thompson. Maximum likelihood alignment of DNA sequences. *J. Mol. Biol.*, 190(2):159–165, 1986.

[148] Richard Durbin, Sean R Eddy, Anders Krogh, and Graeme Mitchison. *Biological sequence analysis: probabilistic models of proteins and nucleic acids*. Cambridge university press, 1998.

[149] Andrew Viterbi. Error bounds for convolutional codes and an asymptotically optimum decoding algorithm. *IEEE Trans. Inf. Theory*, 13(2):260–269, 1967.

[150] Arthur P Dempster, Nan M Laird, and Donald B Rubin. Maximum likelihood from incomplete data via the EM algorithm. *J R Stat Soc Series B Stat Methodol*, 39(1):1–22, 1977.

[151] B-H Juang and Lawrence R Rabiner. The segmental K-means algorithm for estimating parameters of hidden Markov models. *IEEE Trans. Signal Process*, 38(9):1639–1641, 1990.

[152] Luis Javier Rodríguez and Inés Torres. Comparative study of the baum-welch and viterbi training algorithms applied to read and spontaneous speech recognition. In *Iberian Conference on Pattern Recognition and Image Analysis*, pages 847–857. Springer, 2003.

[153] Armen Allahverdyan and Aram Galstyan. Comparative analysis of viterbi training and maximum likelihood estimation for hmms. In *Advances in Neural Information Processing Systems*, pages 1674–1682, 2011.

[154] Dirk Husmeier and Frank Wright. Detection of recombination in DNA multiple alignments with hidden Markov models. *J. Comput. Biol.*, 8(4):401–427, 2001.

[155] Bastien Boussau, Laurent Guéguen, and Manolo Gouy. A mixture model and a hidden markov model to simultaneously detect recombination breakpoints and reconstruct phylogenies. *Evol. Bioinform.*, 5:EBO–S2242, 2009.

[156] Anders Krogh, Michael Brown, I Saira Mian, Kimmen Sjölander, and David Haussler. Hidden Markov models in computational biology: Applications to protein modeling. *J. Mol. Biol.*, 235(5):1501–1531, 1994.

[157] Rasmus Wernersson and Anders Gorm Pedersen. RevTrans: multiple alignment of coding DNA from aligned amino acid sequences. *Nucleic Acids Res.*, 31(13):3537–3539, 2003.

[158] Lisa Mullan. Pairwise sequence alignment—it's all about us! *Brief. Bioinform.*, 7(1):113–115, 2006.

[159] W David. Mount. 2004. bioinformatics: Sequence and genome analysis. *Gold Spring Harbor Laboratory press, New York*, pages 1–18, 2003.

[160] Chuong B Do, Mahathi SP Mahabhashyam, Michael Brudno, and Serafim Batzoglou. ProbCons: Probabilistic consistency-based multiple sequence alignment. *Genome Res.*, 15(2):330–340, 2005.

[161] Ari Löytynoja and Michel C Milinkovitch. A hidden Markov model for progressive multiple alignment. *Bioinformatics*, 19(12):1505–1513, 2003.

[162] Ari Löytynoja and Nick Goldman. An algorithm for progressive multiple alignment of sequences with insertions. *Proc. Natl. Acad. Sci.*, 102(30):10557–10562, 2005.

[163] Yongchao Liu, Bertil Schmidt, and Douglas L Maskell. MSAProbs: multiple sequence alignment based on pair hidden Markov models and partition function posterior probabilities. *Bioinformatics*, 26(16):1958–1964, 2010.

[164] Lior Pachter, Marina Alexandersson, and Simon Cawley. Applications of generalized pair hidden markov models to alignment and gene finding problems. In *Proceedings of the fifth annual international conference on Computational biology*, pages 241–248, 2001.

[165] Marina Alexandersson, Simon Cawley, and Lior Pachter. SLAM: cross-species gene finding and alignment with a generalized pair hidden Markov model. *Genome Res.*, 13(3):496–502, 2003.

[166] Irmtraud M Meyer and Richard Durbin. Comparative ab initio prediction of gene structures using pair HMMs. *Bioinformatics*, 18(10):1309–1318, 2002.

[167] William H Majoros, Mihaela Pertea, and Steven L Salzberg. Efficient implementation of a generalized pair hidden Markov model for comparative gene finding. *Bioinformatics*, 21(9):1782–1788, 2005.

[168] Manimozhiyan Arumugam, Chaochun Wei, Randall H Brown, and Michael R Brent. Pairagon+ N-SCAN_EST: a model-based gene annotation pipeline. *Genome Biol.*, 7(1):1–10, 2006.

[169] Sitao Huang, Gowthami Jayashri Manikandan, Anand Ramachandran, Kyle Rupnow, Wen-mei W Hwu, and Deming Chen. Hardware acceleration of the pair-HMM algorithm for DNA variant calling. In *Proceedings of the 2017 ACM/SIGDA International Symposium on Field-Programmable Gate Arrays*, pages 275–284, 2017.

[170] Subho S Banerjee, Mohamed El-Hadedy, Ching Y Tan, Zbigniew T Kalbarczyk, Steve Lumetta, and Ravishankar K Iyer. On accelerating pair-HMM computations in programmable hardware. In *2017 27th International Conference on Field Programmable Logic and Applications (FPL)*, pages 1–8. IEEE, 2017.

[171] Enliang Li, Subho S Banerjee, Sitao Huang, Ravishankar K Iyer, and Deming Chen. Improved GPU Implementations of the Pair-HMM Forward Algorithm for DNA Sequence Alignment. In *2021 IEEE 39th International Conference on Computer Design (ICCD)*, pages 299–306. IEEE, 2021.

[172] Pengfei Wang, Yuanwu Lei, and Yong Dou. Comparative Analysis of FPGA-Based Pair-HMM Accelerator Structures. *Electron.*, 8(9):965, 2019.

[173] Alkes L Price, Arti Tandon, Nick Patterson, Kathleen C Barnes, Nicholas Rafaels, Ingo Ruczinski, Terri H Beaty, Rasika Mathias, David Reich, and Simon Myers. Sensitive detection of chromosomal segments of distinct ancestry in admixed populations. *PLoS Genet.*, 5(6):e1000519, 2009.

[174] Michael Salter-Townshend and Simon Myers. Fine-scale inference of ancestry segments without prior knowledge of admixing groups. *Genetics*, 212(3):869–889, 2019.

[175] Matthew D Rasmussen, Melissa J Hubisz, Ilan Gronau, and Adam Siepel. Genome-wide inference of ancestral recombination graphs. *PLoS Genet.*, 10 (5):e1004342, 2014.

[176] Jerome Kelleher, Yan Wong, Anthony W Wohns, Chaimaa Fadil, Patrick K Albers, and Gil McVean. Inferring whole-genome histories in large population datasets. *Nat. Genet.*, 51(9):1330–1338, 2019.

[177] Olivier Delaneau, Jean-François Zagury, Matthew R Robinson, Jonathan L Marchini, and Emmanouil T Dermitzakis. Accurate, scalable and integrative haplotype estimation. *Nat. Commun.*, 10(1):1–10, 2019.

[178] Leo Speidel, Marie Forest, Sinan Shi, and Simon R Myers. A method for genome-wide genealogy estimation for thousands of samples. *Nat. Genet.*, 51 (9):1321–1329, 2019.

[179] Brian L Browning, Xiaowen Tian, Ying Zhou, and Sharon R Browning. Fast two-stage phasing of large-scale sequence data. *Am. J. Hum. Genet.*, 108(10): 1880–1890, 2021.

[180] Yun Li, Cristen J Willer, Jun Ding, Paul Scheet, and Gonçalo R Abecasis. MaCH: using sequence and genotype data to estimate haplotypes and unobserved genotypes. *Genet. Epidemiol.*, 34(8):816–834, 2010.

[181] Robert W Davies, Marek Kucka, Dingwen Su, Sinan Shi, Maeve Flanagan, Christopher M Cunniff, Yingguang Frank Chan, and Simon Myers. Rapid genotype imputation from sequence with reference panels. *Nat. Genet.*, 53(7): 1104–1111, 2021.

[182] J Gay, Simon Myers, and Gilean McVean. Estimating meiotic gene conversion rates from population genetic data. *Genetics*, 177(2):881–894, 2007.

[183] Junming Yin, Michael I Jordan, and Yun S Song. Joint estimation of gene conversion rates and mean conversion tract lengths from population SNP data. *Bioinformatics*, 25(12):i231–i239, 2009.

[184] Kiran V Garimella, Zamin Iqbal, Michael A Krause, Susana Campino, Mihir Kekre, Eleanor Drury, Dominic Kwiatkowski, Juliana M Sá, Thomas E Wellems, and Gil McVean. Detection of simple and complex de novo mutations with multiple reference sequences. *Genome Res.*, 30(8):1154–1169, 2020.

[185] Connie M Drysdale, Dennis W McGraw, Catharine B Stack, J Claiborne Stephens, Richard S Judson, Krishnan Nandabalan, Kevin Arnold, Gualberto Ruano, and Stephen B Liggett. Complex promoter and coding region β2-adrenergic receptor haplotypes alter receptor expression and predict in *vivo* responsiveness. *Proc. Natl. Acad. Sci.*, 97(19):10483–10488, 2000.

[186] David L Robertson, Beatrice H Hahn, and Paul M Sharp. Recombination in AIDS viruses. *J. Mol. Evol.*, 40(3):249–259, 1995.

[187] Edward C Holmes, Michael Worobey, and Andrew Rambaut. Phylogenetic evidence for recombination in dengue virus. *Mol. Biol. Evol.*, 16(3):405–409, 1999.

[188] Mark J Gibbs, John S Armstrong, and Adrian J Gibbs. Recombination in the hemagglutinin gene of the 1918 "Spanish flu". *Science*, 293(5536):1842–1845, 2001.

[189] Hongying Jiang, Na Li, Vivek Gopalan, Martine M Zilversmit, Sudhir Varma, Vijayaraj Nagarajan, Jian Li, Jianbing Mu, Karen Hayton, Bruce Henschen, et al. High recombination rates and hotspots in a *Plasmodium falciparum* genetic cross. *Genome Biol.*, 12(4):R33, 2011.

[190] D Brent Weatherly, Duo Peng, and Rick L Tarleton. Recombination-driven generation of the largest pathogen repository of antigen variants in the protozoan *Trypanosoma cruzi*. *BMC Genom.*, 17(1):729, 2016.

[191] Adam Auton and Gil McVean. Recombination rate estimation in the presence of hotspots. *Genome Res.*, 17(8):1219–1227, 2007.

[192] Thomas Huber, Geoffrey Faulkner, and Philip Hugenholtz. Bellerophon: a program to detect chimeric sequences in multiple sequence alignments. *Bioinformatics*, 20(14):2317–2319, 2004.

[193] Patricia Buendia and Giri Narasimhan. Sliding MinPD: building evolutionary networks of serial samples via an automated recombination detection approach. *Bioinformatics*, 23(22):2993–3000, 2007.

[194] Jotun Hein. Reconstructing evolution of sequences subject to recombination using parsimony. *Math. Biosci.*, 98(2):185–200, 1990.

[195] Donald S Chen, Alyssa E Barry, Aleksandra Leliwa-Sytek, Terry-Ann Smith, Ingrid Peterson, Stuart M Brown, Florence Migot-Nabias, Philippe Deloron, Moses M Kortok, Kevin Marsh, et al. A molecular epidemiological study of var gene diversity to characterize the reservoir of *Plasmodium falciparum* in humans in Africa. *PLoS One*, 6(2):e16629, 2011.

[196] Karen P Day, Yael Artzy-Randrup, Kathryn E Tiedje, Virginie Rougeron, Donald S Chen, Thomas S Rask, Mary M Rorick, Florence Migot-Nabias, Philippe Deloron, Adrian JF Luty, et al. Evidence of strain structure in *Plasmodium falciparum* var gene repertoires in children from Gabon, West Africa. *Proc. Natl. Acad. Sci.*, 114(20):E4103–E4111, 2017.

[197] Kazutaka Katoh and Martin C Frith. Adding unaligned sequences into an existing alignment using MAFFT and LAST. *Bioinformatics*, 28(23):3144–3146, 2012.

[198] Steven Henikoff and Jorja G Henikoff. Amino acid substitution matrices from protein blocks. *Proc. Natl. Acad. Sci.*, 89(22):10915–10919, 1992.

[199] Jerome Kelleher, Alison M Etheridge, and Gilean McVean. Efficient coalescent simulation and genealogical analysis for large sample sizes. *PLoS Comput. Biol.*, 12(5):e1004842, 2016.

[200] Stephanie J Spielman and Claus O Wilke. Pyvolve: a flexible python module for simulating sequences along phylogenies. *PLoS One*, 10(9):e0139047, 2015.

[201] William Fletcher and Ziheng Yang. INDELible: a flexible simulator of biological sequence evolution. *Mol. Biol. Evol.*, 26(8):1879–1888, 2009.

[202] Virginie Rougeron, Kathryn E Tiedje, Donald S Chen, Thomas S Rask, Dionicia Gamboa, Amanda Maestre, Lise Musset, Eric Legrand, Oscar Noya, Erhan Yalcindag, et al. Evolutionary structure of *Plasmodium falciparum* major variant surface antigen genes in South America: Implications for epidemic transmission and surveillance. *Ecol. Evol.*, 7(22):9376–9390, 2017.

[203] Adam F. Sander, Thomas Lavstsen, Thomas S. Rask, Michael Lisby, Ali Salanti, Sarah L. Fordyce, Jakob S. Jespersen, Richard Carter, Kirk W. Deitsch, Thor G. Theander, Anders Gorm Pedersen, and David E. Arnot. DNA secondary structures are associated with recombination in major *Plasmodium falciparum* variable surface antigen gene families. *Nucleic Acids Res.*, 42(4):2270–2281, 2013.

[204] Alyssa E Barry, Aleksandra Leliwa-Sytek, Livingston Tavul, Heather Imrie, Florence Migot-Nabias, Stuart M Brown, Gilean AV McVean, and Karen P Day. Population genomics of the immune evasion (var) genes of *Plasmodium falciparum*. *PLOS Pathog.*, 3(3):e34, 2007.

[205] Daniel H Huson, Regula Rupp, and Celine Scornavacca. *Phylogenetic networks: concepts, algorithms and applications*. Cambridge University Press, 2010.

[206] Fabio Pardi and Celine Scornavacca. Reconstructible phylogenetic networks: do not distinguish the indistinguishable. *PLoS Comput. Biol.*, 11(4), 2015.

[207] Alexander Mirsky, Linda Kazandjian, and Maria Anisimova. Antibody-specific model of amino acid substitution for immunological inferences from alignments of antibody sequences. *Mol. Biol. Evol.*, 32(3):806–819, 2015.

[208] M Dayhoff, R Schwartz, and B Orcutt. Chapter 22: A model of evolutionary change in proteins. In *Atlas of Protein Sequence and Structure*, volume 5, pages 345–352. National Biomedical Research Foundation Silver Spring MD, 1978.

[209] DT Jones, WR Taylor, and JM Thornton. A mutation data matrix for transmembrane proteins. *FEBS Letters*, 339(3):269–275, 1994.

[210] Si Quang Le and Olivier Gascuel. An improved general amino acid replacement matrix. *Mol. Biol. Evol.*, 25(7):1307–1320, 2008.

[211] Ziheng Yang, Rasmus Nielsen, and Masami Hasegawa. Models of amino acid substitution and applications to mitochondrial protein evolution. *Mol. Biol. Evol.*, 15(12):1600–1611, 1998.

[212] John FC Kingman. On the genealogy of large populations. *J. Appl. Probab.*, 19(A):27–43, 1982.

[213] Malla Padidam, Stanley Sawyer, and Claude M Fauquet. Possible emergence of new geminiviruses by frequent recombination. *Virol. J.*, 265(2):218–225, 1999.

[214] Joseph Felsenstein. Evolutionary trees from DNA sequences: a maximum likelihood approach. *J. Mol. Evol.*, 17(6):368–376, 1981.

[215] Robert C Edgar. MUSCLE: a multiple sequence alignment method with reduced time and space complexity. *BMC Bioinform.*, 5(1):1–19, 2004.

[216] Robert C Edgar. Search and clustering orders of magnitude faster than BLAST. *Bioinformatics*, 26(19):2460–2461, 2010.

[217] Robert W Snow. Global malaria eradication and the importance of *Plasmodium falciparum* epidemiology in africa. *BMC Med.*, 13(1):1–3, 2015.

[218] Joseph D Smith, Chetan E Chitnis, Alistar G Craig, David J Roberts, Diana E Hudson-Taylor, David S Peterson, Robert Pinches, Chris I Newbold, and Louis H Miller. Switches in expression of *Plasmodium falciparum* var genes correlate with changes in antigenic and cytoadherent phenotypes of infected erythrocytes. *Cell*, 82(1):101–110, 1995.

[219] Kirsten Moll, Mia Palmkvist, Junhong Ch'ng, Mpungu Steven Kiwuwa, and Mats Wahlgren. Evasion of immunity to *Plasmodium falciparum*: rosettes of blood group A impair recognition of PfEMP1. *PLoS One*, 10(12):e0145120, 2015.

[220] Christiam Camacho, George Coulouris, Vahram Avagyan, Ning Ma, Jason Papadopoulos, Kevin Bealer, and Thomas L Madden. BLAST+: architecture and applications. *BMC Bioinform.*, 10(1):421, 2009.

[221] M Pietras and P Klęsk. FPGA implementation of logarithmic versions of Baum-Welch and Viterbi algorithms for reduced precision hidden Markov models. *Bull. Pol. Acad. Sci.: Tech. Sci.*, 65(6):935–946, 2017.

[222] Gregory E Sims, Se-Ran Jun, Guohong A Wu, and Sung-Hou Kim. Alignment-free genome comparison with feature frequency profiles (FFP) and optimal resolutions. *Proc. Natl. Acad. Sci.*, 106(8):2677–2682, 2009.

[223] Estela Shabani, Benjamin Hanisch, Robert O Opoka, Thomas Lavstsen, and Chandy C John. *Plasmodium falciparum* EPCR-binding PfEMP1 expression increases with malaria disease severity and is elevated in retinopathy negative cerebral malaria. *BMC Med.*, 15(1):1–14, 2017.

[224] Louis H Miller, Dror I Baruch, Kevin Marsh, and Ogobara K Doumbo. The pathogenic basis of malaria. *Nature*, 415(6872):673–679, 2002.

[225] A Scherf, R Hernandez-Rivas, P Buffet, E Bottius, C Benatar, B Pouvelle, J Gysin, and M Lanzer. Antigenic variation in malaria: in situ switching, relaxed and mutually exclusive transcription of var genes during intra-erythrocytic development in *Plasmodium falciparum*. *The EMBO Journal*, 17 (18):5418–5426, 1998.

[226] Abdirahman I Abdi, Susanne H Hodgson, Michelle K Muthui, Cheryl A Kivisi, Gathoni Kamuyu, Domtila Kimani, Stephen L Hoffman, Elizabeth Juma, Bernhards Ogutu, Simon J Draper, et al. *Plasmodium falciparum* malaria parasite var gene expression is modified by host antibodies: longitudinal evidence from controlled infections of Kenyan adults with varying natural exposure. *BMC Infect. Dis.*, 17:1–11, 2017.

[227] Anja TR Jensen, Pamela Magistrado, Sarah Sharp, Louise Joergensen, Thomas Lavstsen, Antonella Chiucchiuini, Ali Salanti, Lasse S Vestergaard, John P Lusingu, Rob Hermsen, et al. *Plasmodium falciparum* associated with severe childhood malaria preferentially expresses PfEMP1 encoded by group A var genes. *J. Exp. Med.*, 199(9):1179–1190, 2004.

[228] Joseph D Smith, J Alexandra Rowe, Matthew K Higgins, and Thomas Lavstsen. Malaria's deadly grip: cytoadhesion of *Plasmodium falciparum*-infected erythrocytes. *Cell. Microbiol.*, 15(12):1976–1983, 2013.

[229] Helen M Kyriacou, Graham N Stone, Richard J Challis, Ahmed Raza, Kirsten E Lyke, Mahamadou A Thera, Abdoulaye K Koné, Ogobara K Doumbo, Christopher V Plowe, and J Alexandra Rowe. Differential var gene transcription in *Plasmodium falciparum* isolates from patients with cerebral malaria compared to hyperparasitaemia. *Mol. Biochem. Parasitol.*, 150(2): 211–218, 2006.

[230] Mirjam Kaestli, Ian A Cockburn, Alfred Cortés, Kay Baea, J Alexandra Rowe, and Hans-Peter Beck. Virulence of malaria is associated with differential expression of *Plasmodium falciparum* var gene subgroups in a case-control study. *J. Infect. Dis.*, 193(11):1567–1574, 2006.

[231] Mun Hua Tan, Heejung Shim, Yao-ban Chan, and Karen Day. Unravelling var complexity: Relationship between DBLα types and var genes in *Plasmodium falciparum*. *Front. Parasitol.*, 1:12, 2023.

[232] Sean R Eddy. Hidden markov models. *Curr. Opin. Struct. Biol.*, 6(3):361–365, 1996.

[233] Takuya Aramaki, Romain Blanc-Mathieu, Hisashi Endo, Koichi Ohkubo, Minoru Kanehisa, Susumu Goto, and Hiroyuki Ogata. KofamKOALA: KEGG Ortholog assignment based on profile HMM and adaptive score threshold. *Bioinformatics*, 36(7):2251–2252, 2020.

[234] Johannes Söding. Protein homology detection by HMM–HMM comparison. *Bioinformatics*, 21(7):951–960, 2005.

[235] Travis J Wheeler and Sean R Eddy. nhmmer: DNA homology search with profile HMMs. *Bioinformatics*, 29(19):2487–2489, 2013.

[236] Yanni Sun and Jeremy Buhler. Designing patterns for profile HMM search. *Bioinformatics*, 23(2):e36–e43, 2007.

[237] Can Firtina, Ziv Bar-Joseph, Can Alkan, and A Ercument Cicek. Hercules: a profile HMM-based hybrid error correction algorithm for long reads. *Nucleic Acids Res.*, 46(21):e125–e125, 2018.

[238] Martin Madera and Julian Gough. A comparison of profile hidden Markov model procedures for remote homology detection. *Nucleic Acids Res.*, 30(19): 4321–4328, 2002.

[239] Frédéric Lemoine, Luc Blassel, Jakub Voznica, and Olivier Gascuel. COVID-Align: Accurate online alignment of hCoV-19 genomes using a profile HMM. *Bioinformatics*, 37(12):1761–1762, 2021.

[240] Sean R Eddy. Accelerated profile HMM searches. *PLoS Comput. Biol.*, 7(10): e1002195, 2011.

[241] Fabian Sievers, Andreas Wilm, David Dineen, Toby J Gibson, Kevin Karplus, Weizhong Li, Rodrigo Lopez, Hamish McWilliam, Michael Remmert, Johannes Söding, et al. Fast, scalable generation of high-quality protein multiple sequence alignments using Clustal Omega. *Mol. Syst. Biol.*, 7(1):539, 2011.

[242] Fabian Sievers and Desmond G Higgins. Clustal Omega for making accurate alignments of many protein sequences. *Protein Sci.*, 27(1):135–145, 2018.

[243] Robert Gilmore Pontius Jr and Mang Lung Cheuk. A generalized cross-tabulation matrix to compare soft-classified maps at multiple resolutions. *Int. J. Geogr. Inf. Syst.*, 20(1):1–30, 2006.

[244] Ari Löytynoja. Phylogeny-aware alignment with PRANK. *Multiple sequence alignment methods*, pages 155–170, 2014.

[245] Jean Muller, Christopher J Creevey, Julie D Thompson, Detlev Arendt, and Peer Bork. AQUA: automated quality improvement for multiple sequence alignments. *Bioinformatics*, 26(2):263–265, 2010.

[246] Raúl Y Tito, Simone Macmil, Graham Wiley, Fares Najar, Lauren Cleeland, Chunmei Qu, Ping Wang, Frederic Romagne, Sylvain Leonard, Agustín Jiménez Ruiz, et al. Phylotyping and functional analysis of two ancient human microbiomes. *PLoS One*, 3(11):e3703, 2008.

[247] Daniel H Huson, Alexander F Auch, Ji Qi, and Stephan C Schuster. MEGAN analysis of metagenomic data. *Genome Res.*, 17(3):377–386, 2007.

[248] Evan Bolyen, Jai Ram Rideout, Matthew R Dillon, Nicholas A Bokulich, Christian C Abnet, Gabriel A Al-Ghalith, Harriet Alexander, Eric J Alm,

Manimozhiyan Arumugam, Francesco Asnicar, et al. Reproducible, interactive, scalable and extensible microbiome data science using QIIME 2. *Nat. Biotechnol.*, 37(8):852–857, 2019.

[249] Nicholas A Bokulich, Benjamin D Kaehler, Jai Ram Rideout, Matthew Dillon, Evan Bolyen, Rob Knight, Gavin A Huttley, and J Gregory Caporaso. Optimizing taxonomic classification of marker-gene amplicon sequences with QIIME 2's q2-feature-classifier plugin. *Microbiome*, 6:1–17, 2018.

[250] Mun Hua Tan, Kathryn E Tiedje, Qian Feng, Qi Zhan, Mercedes Pascual, Heejung S Shim, Yao-ban Chan, and Karen P Day. A paradoxical population structure of var DBLα types in Africa. *bioRxiv*, 2023–11, 2023.

[251] Stijn vanDongen. A cluster algorithm for graphs. *Inf. Syst. [INS]*, (R 0010), 2000.

[252] Anton J Enright, Stijn Van Dongen, and Christos A Ouzounis. An efficient algorithm for large-scale detection of protein families. *Nucleic Acids Res.*, 30 (7):1575–1584, 2002.

[253] Elisabetta Binaghi, Pietro A Brivio, Paolo Ghezzi, and Anna Rampini. A fuzzy set-based accuracy assessment of soft classification. *Pattern Recognit. Lett.*, 20(9):935–948, 1999.

[254] José Luis Silván-Cárdenas and Lizhu Wang. Sub-pixel confusion–uncertainty matrix for assessing soft classifications. *Remote Sens. Environ.*, 112(3):1081–1095, 2008.

[255] Jin Chen, Xiaolin Zhu, Hidefumi Imura, and Xuehong Chen. Consistency of accuracy assessment indices for soft classification: Simulation analysis. *ISPRS J. Photogramm. Remote Sens.*, 65(2):156–164, 2010.

[256] Stien Heremans and Jos Van Orshoven. A new way of calculating the subpixel confusion matrix: a comparative evaluation using an artificial dataset. In *Proceedings of the Tenth International Symposium on Spatial Accuracy Assessment in Natural Resources and Environmental Sciences*, pages 31–36, 2012.

[257] Jacob Cohen. A coefficient of agreement for nominal scales. *Educ Psychol Meas*, 20(1):37–46, 1960.

[258] Lea Barfod, Michael B Dalgaard, Suzan T Pleman, Michael F Ofori, Richard J Pleass, and Lars Hviid. Evasion of immunity to *Plasmodium falciparum* malaria by IgM masking of protective IgG epitopes in infected erythrocyte surface-exposed PfEMP1. *PNAS*, 108(30):12485–12490, 2011.

[259] Jacob E Crawford, Wamdaogo M Guelbeogo, Antoine Sanou, Alphonse Traoré, Kenneth D Vernick, N'Fale Sagnon, and Brian P Lazzaro. De novo transcriptome sequencing in Anopheles funestus using Illumina RNA-seq technology. *PLoS One*, 5(12):e14202, 2010.

[260] Nicholas J Loman, Raju V Misra, Timothy J Dallman, Chrystala Constantinidou, Saheer E Gharbia, John Wain, and Mark J Pallen. Performance comparison of benchtop high-throughput sequencing platforms. *Nat. Biotechnol*, 30(5):434–439, 2012.

[261] Alexandros Stamatakis. RAxML version 8: a tool for phylogenetic analysis and post-analysis of large phylogenies. *Bioinformatics*, 30(9):1312–1313, 2014.

[262] Vincent Lefort, Richard Desper, and Olivier Gascuel. FastME 2.0: a comprehensive, accurate, and fast distance-based phylogeny inference program. *Mol. Biol. Evol.*, 32(10):2798–2800, 2015.

[263] Richard Desper and Olivier Gascuel. Fast and accurate phylogeny reconstruction algorithms based on the minimum-evolution principle. *J. Comput. Biol.*, 9(5):687–705, 2002.

[264] Rui Borges, João Paulo Machado, Cidália Gomes, Ana Paula Rocha, and Agostinho Antunes. Measuring phylogenetic signal between categorical traits and phylogenies. *Bioinformatics*, 35(11):1862–1869, 2019.

[265] Vincent Lefort, Richard Desper, and Olivier Gascuel. FastME 2.0: a comprehensive, accurate and fast distance-based phylogeny inference program, 2015. URL `http://www.atgc-montpellier.fr/fastme/`. Accessed: 2024-09-15.

[266] Peter C Bull, Matthew Berriman, Sue Kyes, Michael A Quail, Neil Hall, Moses M Kortok, Kevin Marsh, and Chris I Newbold. *Plasmodium falciparum* variant surface antigen expression patterns during malaria. *PLoS Pathog.*, 1 (3):e26, 2005.

[267] Kathryn E Tiedje and Mun Hua Tan. DBLaCleaner - A pipeline for demultiplexing and identifying VAR DBLa tags in Illumina paired end data,

2016. URL `https://github.com/UniMelb-Day-Lab/DBLaCleaner`. Accessed: 2024-09-11.

[268] Robert Edgar. USEARCH manual - algorithms - dereplication, 2018. URL `https://www.drive5.com/usearch/manual6/dereplication.html`. Accessed: 2023-12-24.

[269] Tim-W Gilberger, R Heiner Schirmer, Rolf D Walter, and Sylke Müller. Deletion of the parasite-specific insertions and mutation of the catalytic triad in glutathione reductase from chloroquine-sensitive *Plasmodium falciparum* 3D7. *Mol. Biochem. Parasitol*, 107(2):169–179, 2000.

[270] Mary Lopez-Perez, Elizabeth Villasis, Ricardo LD Machado, Marinete M Póvoa, Joseph M Vinetz, Silvia Blair, Dionicia Gamboa, and Sara Lustigman. *Plasmodium falciparum* field isolates from South America use an atypical red blood cell invasion pathway associated with invasion ligand polymorphisms. *PloS One*, 7(10):e47913, 2012.

[271] Pooja Agarwal, AR Anvikar, CR Pillai, and Kumkum Srivastava. In vitro susceptibility of indian *Plasmodium falciparum* isolates to different antimalarial drugs & antibiotics. *Indian J. Med. Res.*, 146(5):622, 2017.

[272] Kara A Moser, Elliott F Drábek, Ankit Dwivedi, Emily M Stucke, Jonathan Crabtree, Antoine Dara, Zalak Shah, Matthew Adams, Tao Li, Priscila T Rodrigues, et al. Strains used in whole organism *Plasmodium falciparum* vaccine trials differ in genome structure, sequence, and immunogenic potential. *Genome Med.*, 12(1):1–17, 2020.

[273] Stephen D Woolley, Melissa Fernandez, Maria Rebelo, Stacey A Llewellyn, Louise Marquart, Fiona H Amante, Helen E Jennings, Rebecca Webster, Katharine Trenholme, Stephan Chalon, et al. Development and evaluation of a new *Plasmodium falciparum* 3D7 blood stage malaria cell bank for use in malaria volunteer infection studies. *Malar. J.*, 20(1):1–9, 2021.

[274] Yemback Kenneth Pierre, Eyong Kenneth Oben, Noella M Efange, Michael HK Kamdem, Derek T Ndinteh, Patricia O Odumosu, Gabriel N Folefoc, Lawrence Ayong, and Thomas Werner. Glyceraldehyde-3-phosphate dehydrogenase (Pf-GAPDH): Design, Isolation, Synthesis and preliminary anti-malarial activity against *Plasmodium falciparum* 3D7 strains. 2022.

[275] Jianhua Lin. Divergence measures based on the shannon entropy. *IEEE Trans. Inf. Theory*, 37(1):145–151, 1991.

[276] Guangchuang Yu, David K Smith, Huachen Zhu, Yi Guan, and Tommy Tsan-Yuk Lam. ggtree: an R package for visualization and annotation of phylogenetic trees with their covariates and other associated data. *Methods Ecol. Evol.*, 8(1):28–36, 2017.