UNIVERSITY OF CALIFORNIA SAN DIEGO

Statistical and Algorithmic Methods to Analyze Genome Sequencing Data

A dissertation submitted in partial satisfaction of the
requirements for the degree Doctor of Philosophy

in

Electrical Engineering (Communication Theory and Systems)

by

Shahab Sarmashghi

Committee in charge:

Professor Vineet Bafna, Chair
Professor Siavash Mirarab, Co-Chair
Professor Alon Orlitsky
Professor Brian Palenik
Professor Pavel Pevzner

2021

The Dissertation of Shahab Sarmashghi is approved, and it is acceptable in quality and form for publication on microfilm and electronically.

University of California San Diego

2021

DEDICATION

To my beloved mom and dad.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGEMENTS

I would like to thank my advisor, Professor Vineet Bafna, for his constant support and guidance. He taught me the fundamentals of scientific research, helped me to become a computer scientist, and patiently explained basic biology concepts to me. He truly believes in investing in his students training for their future success in their career and personal lives. I was beyond fortunate to have such an amazing scientist and person as my PhD advisor.

I would like to thank my co-advisor, Professor Siavash Mirarab, who is an amazing computer scientist that I learned a lot from him during the course of my PhD. His advising was crucial for the success of my PhD work. I also would like to express my gratitude for the guidance and support of other committee members Professor Alon Orlitsky, Professor Brian Palenik, and Professor Pavel Pevzner, who are all inspiring scientists.

I should take this opportunity and thank everyone in our lab who helped me throughout my studies. In particular, I want to thank Dr. Ali Akbari, Dr. Arya Iranmehr, Dr. Viraj Deshpande, Dr. Peter Edge, Dr. Doruk Beyter, Dr. Seong Won Cha, Dr. Mehrdad Bakhtiari, Jens Luebeck, and Miin Lin, who all took their valuable time to help me whenever I needed their expertise. I also want to thank my great friends and roommates Dr. Erfan Sayyari and Dr. Moein Falahatgar who helped me a lot in my scientific work and personal life.

At the end, I would like to thank my family, especially my mother and father. Everything that I have achieved in my life, I owe it to the sacrifices they did for me and our family.

And here is the formal acknowledgment to the collaborators and co-authors of the papers that I published and used to write this dissertation:

Chapter 2, in full, is a reprint of the material as it appears in Cell systems 8, no. 6 (2019): 523-529. “Computing the statistical significance of overlap between genome annotations with iStat”. Shahab Sarmashghi and Vineet Bafna. The dissertation author was the primary investigator and author of this paper.

Chapter 3, in full, is a reprint of the material as it appears in Genome biology 20, no. 1 (2019): 1-20. “Skmer: assembly-free and alignment-free sample identification using genome

skims". Shahab Sarmashghi, Kristine Bohmann, M. Thomas P. Gilbert, Vineet Bafna, and Siavash Mirarab. The dissertation author was the primary investigator and author of this paper.

Chapter 4, in full, is a reprint of the material as it appears in PLOS Computational Biology 17(11): e1009449. "Estimating repeat spectra and genome length from low-coverage genome skims with RESPECT". Shahab Sarmashghi, Metin Balaban , Eleonora Rachtman, Behrouz Touri, Siavash Mirarab, and Vineet Bafna. The dissertation author was the primary investigator and author of this paper.

VITA

2013	B.Sc. in Electrical Engineering, Sharif University of Technology, Iran
2015	M.Sc. in Electrical Engineering, Sharif University of Technology, Iran
2016–2021	Graduate Research Assistant, University of California San Diego
2021	Ph.D. in Electrical Engineering (Communication Theory and Systems), University of California San Diego

PUBLICATIONS

Shahab Sarmashghi, Metin Balaban , Eleonora Rachtman, Behrouz Touri, Siavash Mirarab, and Vineet Bafna. (2021) "Estimating repeat spectra and genome length from low-coverage genome skims with RESPECT". *PLOS Computational Biology* 17(11): e1009449.

Metin Balaban, Shahab Sarmashghi, and Siavash Mirarab. "APPLES: scalable distance-based phylogenetic placement with or without alignments". *Systematic Biology* 69, no. 3 (2020): 566-578.

Shahab Sarmashghi, and Vineet Bafna. "Computing the statistical significance of overlap between genome annotations with iStat". *Cell systems* 8, no. 6 (2019): 523-529.

Shahab Sarmashghi, Kristine Bohmann, M. Thomas P. Gilbert, Vineet Bafna, and Siavash Mirarab. "Skmer: assembly-free and alignment-free sample identification using genome skims". *Genome biology* 20, no. 1 (2019): 1-20.

ABSTRACT OF THE DISSERTATION

Statistical and Algorithmic Methods to Analyze Genome Sequencing Data

by

Shahab Sarmashghi

Doctor of Philosophy in Electrical Engineering (Communication Theory and Systems)

University of California San Diego, 2021

Professor Vineet Bafna, Chair
Professor Siavash Mirarab, Co-Chair

With continuing reductions in the cost of genome sequencing, and the advent of new sequencing technologies, it has become a routine process to sequence genomes in experiments across different fields of biology in order to study the foundations of life. Collecting multi-omics data is now an integral part of studying the human health and underlying genetic cause of diseases. Genome sequencing is extensively used to study the evolution of life and how different species are genetically related, and it is becoming an important tool to monitor the health of ecosystems and study the dynamics of biodiversity in this era of rapid climate change. As large datasets of genomic data become available through worldwide consortia and collaborative efforts, an

important challenge is processing and interpreting these massive datasets. In this dissertation, I present a collection of statistical and algorithmic methods to address different computational problems faced in using genome sequencing data to study the function and properties of genome and its variation across species. In the first part of this dissertation, I describe the method we developed to address the problem of statistical significance of overlap between genome annotations–the assignment of function to specific genomic regions, which is a foundational effort of modern biology. To the best of our knowledge, the p-value computation for sets of overlapping intervals has been limited either to permutation tests that do not scale to computation of small p-values or simple parametric tests such as hypergeometric or binomial tests that are based on simplifying assumptions about the length and structure of intervals. Our method, however, formulates a null model where the size of intervals and their relative arrangement are considered when the significance of overlap is evaluated. In the second part, I introduce the idea of using whole genome sequencing reads at low coverage–genome skims–without requiring any genome assembly or alignment. We have developed methods to compute genomic distances between genome skims to use them for sample identification and phylogenetic placement, and to estimate genomic parameters such as genome length and repeat content of the genome to lay the foundation for accurate assessment of genetic biodiversity.

Chapter 1

Introduction

1.1 Genome annotation

Annotating the genome is a central problem in biology. Subsequent to the sequencing and assembly of the human genome, and the development of deep sequencing technologies, researchers have employed creative ideas to develop better insight into the regions that support genome structure and function.

Examples of annotation include repeat elements [1], protein coding genes [2], non-coding RNA [3], regulatory regions [4], sites with specific epigenetic modifications [5], transcription start sites [4], ribosome initiation sites [6, 7]. Annotation may also involve structural features, such as the regions with a change in copy number and other structural variation [8, 9]. These regions can be dynamic and change depending upon tissue type and experimental conditions (e.g., histone methylation marks, regions with high gene expression, etc.), or relatively static (e.g., location of protein coding genes).

In all of these annotations, we can work with an abstract representation by considering the genome as a line segment, and any annotation as a collection of non-overlapping intervals on that line. Having a pair of annotations modeled by two sets of intervals, enables us to evaluate their overlap which is very useful in uncovering biological principles, and widely used by scientists. It has been a standard practice to formulate the problems as a hypothesis test and compute the p-value to substantiate the statistical significance of the observed overlap. Randomization tests

are known to be exact significance tests when the space of all possible random samples can be enumerated. Random annotations can be generated by randomizing the position of intervals while preserving the coherence of each region. However, in many real-life examples including the above studies, the sample space is enormous, and naive sampling-based methods cannot achieve adequate resolution to distinguish between rare events in feasible running times. On the other hand, while parametric tests used in the literature are computationally efficient, they oversimplify the problem by casting intervals as points and ignoring the dimension of annotated regions on the genome, which often lead to more significant p-values. In Chapter 2, we propose an algorithm which efficiently enumerates over all possible randomized samples to find the exact null distribution, under the assumption that the order of intervals is preserved when randomizing their position. Using simulated data, we show that the impact of our assumption on p-value calculation is limited. We also provide a fast approximate solution based on Poisson binomial distribution, and using simulated data, we characterize its performance in approximating the generic null distribution. Moreover, we demonstrate the result of applying our methods to four examples of interval overlap problem from previously published studies, and compare our results with the p-values reported in these studies.

1.2 Genome skimming

The ability to quickly and inexpensively study the taxonomic diversity in an environment is critical in this era of rapid climate and biodiversity changes. In North America alone, the bird population has declined by over a quarter since 1970[10]. Simply understanding the scope and extent of bio-diversity changes remains a challenging problem. Genomic sequence based biodiversity sampling provides an attractive alternative to physical sampling and cataloging. *However, the analysis typically requires assembling and finishing a reference genome, which can still be prohibitively costly.* Despite the many projects aimed at high quality genome sequencing of eukaryotic species [11], it could be many decades before we have acquired high-quality data

so that biodiversity measurements for each population can be acquired on an ongoing, routine basis.

While low costs have kept PCR-based pipelines attractive, decreasing costs of shotgun sequencing have now made it possible to shotgun sequence 1-2Gb of total DNA per reference specimen sample for as low as $80 [12], even after including sample preparation and labor costs. This has lead researchers to propose an alternate method that uses low-pass sequencing to generate *genome-skims* [13, 12], and subsequently identifies chloroplast or mitochondrial marker genes or assembles the organelle genome. Reconstructing plastid and mtDNA genomes from low-pass shotgun data is possible because organelle DNA tends to be heavily overrepresented in shotgun sequencing data; for example, 10.4% of all reads from the Apocynaceae family of flowering plants were from the chloroplast in one genome-skimming study [13]. Large reference databases based on genome-skimming techniques are under construction by projects such as PhyloAlps [14], NorBol [15], and DNAmark [16].

Most current applications of genome-skimming to species identification require organelle genome assembly, a task that requires relatively time-consuming manual curation steps to ensure that assembly errors are avoided [17]. This approach discards a vast proportion of the non-target data, reducing the discriminatory power. For these reasons, the DNAmark project [16] is considering alternative methods, where, instead of only relying on organelle markers, one could use the entire set of reads generated in a genome-skim as the identifier of a species. This approach poses an interesting methodological question: can the unassembled data be used to taxonomically profile reference and query samples in a similar manner to conventional barcoding, but using all available genomic information and saving us from the labor-intensive task of mitochondria/plastid genome assembly?

In Chapter 3, we introduce a new assembly-free method to directly use low coverage genome-skims of both reference and query samples. By avoiding the assembly step, our approach also reduces the amount of data processing needed for expanding the reference database. We treat genome-skims simply as low-coverage "bags of reads", both for a collection of reference

species and for query samples. The problem is to find the reference genome-skim that matches the query; if an exact match is not found, we seek the closest available match. We achieve this goal by estimating a *distance* between two genome-skims for low and varied coverage using assembly-free and alignment-free approaches.

In Chapter 4, we revisit the problem of estimating genomic parameters from genome-skim data: specifically, genome length, sequencing depth, and repeat content. We use a mix of theoretical and empirical analysis to understand the fundamental limitations to estimating the genomic parameters. We get around these intrinsic limitations using a novel constrained optimization approach, where the constraints are learned empirically from available assembled genomes.

Chapter 2

ISTAT: Computing the Statistical Significance of Overlap between Genome Annotations

In this chapter, we consider the following problem: Let I and I_f each describe a collection of n and m non-overlapping intervals on a line segment of finite length. Suppose that k of the m intervals of I_f are intersected by some interval(s) in I. Under the null hypothesis that intervals in I are randomly arranged w.r.t I_f, what is the significance of this overlap? This is a natural abstraction of statistical questions that are ubiquitous in the post-genomic era. The interval collections represent annotations that reveal structural or functional regions of the genome, and overlap statistics can provide insight into the correlation between different structural and functional regions. However, the statistics of interval overlaps have not been systematically explored. In this chapter, we formulate a statistical significance problem which considers the length and structure of intervals. We describe a combinatorial algorithm for a constrained interval overlap problem that can accurately compute very small p-values. We also propose a fast approximate method to facilitate problems consisted of very large number of intervals. These methods are all implemented in a tool, ISTAT. We applied ISTAT to simulated interval data to obtain precise estimates of low p-values, and characterize the performance of our methods. We also test ISTAT on real datasets from previous studies, and compare ISTAT results with the reported p-values using basic permutation or parametric tests.

2.1 Introduction

Annotating the genome is a central problem in biology. Subsequent to the sequencing and assembly of the human genome, and the development of deep sequencing technologies, researchers have employed creative ideas to develop better insight into the regions that support genome structure and function.

Examples of annotation include repeat elements [1], protein coding genes [2], non-coding RNA [3], regulatory regions [4], sites with specific epigenetic modifications [5], transcription start sites [4], ribosome initiation sites [6, 7]. Annotation may also involve structural features, such as the regions with a change in copy number and other structural variation [8, 9]. These regions can be dynamic and change depending upon tissue type and experimental conditions (e.g., histone methylation marks, regions with high gene expression, etc.), or relatively static (e.g., location of protein coding genes).

In all of these annotations, we can work with an abstract representation by considering the genome as a line segment, and any annotation as a collection of non-overlapping intervals on that line. Having a pair of annotations modeled by two sets of intervals, enables us to evaluate their overlap which is very useful in uncovering biological principles, and widely used by scientists.

Epigenetics is among the areas that extensively apply such models, in order to study the potential association between epigenetic modifications and functional elements in the genome. For instance, Guenther et al. 2007 [18] observed that about $3/4$ of all known promoters were overlapped by the intervals highly enriched for the methylation of lysine 4 on histone H3, showing that a large fraction of genes are enriched for H3K4me3 modification, including genes without any detected transcript. Assuming that the presence of histone H3K4me3 is correlated with transcription initiation, they hypothesized that transcription initiation occurs in all genes, but only in active genes it is accompanied by transcriptional elongation.

This model can also be useful to study functional impact of segmental duplications and copy number variations (CNVs). Zarrei et al. 2015 [19] performed a meta-analysis and provided

an updated map of CNVs in healthy individual. They have considered several sets of genes and genomic sequences such as protein-coding and non-coding genes, cancer genes, lincRNAs, Promoters, etc., and computed the enrichment of copy number variant regions in each of these annotations to assess the variability of different functional regions of the genome.

In experiments related to genome annotations, such questions are ubiquitous, and they all distill down to the underlying statistical question of significantly overlapping intervals. Hence, it has been a standard practice to formulate the problems as a hypothesis test and compute the p-value to substantiate the statistical significance of the observed overlap. Randomization tests are known to be exact significance tests when the space of all possible random samples can be enumerated. Random annotations can be generated by randomizing the position of intervals while preserving the coherence of each region. However, in many real-life examples including the above studies, the sample space is enormous, and naive sampling-based methods cannot achieve adequate resolution to distinguish between rare events in feasible running times. On the other hand, while parametric tests used in the literature are computationally efficient, they oversimplify the problem by casting intervals as points and ignoring the dimension of annotated regions on the genome, which often lead to more significant p-values. In this chapter, we propose an algorithm which efficiently enumerates over all possible randomized samples to find the exact null distribution, under the assumption that the order of intervals is preserved when randomizing their position. Using simulated data, we show that the impact of our assumption on p-value calculation is limited. We also provide a fast approximate solution based on Poisson binomial distribution, and using simulated data, we characterize its performance in approximating the generic null distribution. Moreover, we demonstrate the result of applying our methods to four examples of interval overlap problem from previously published studies, and compare our results with the p-values reported in these studies.

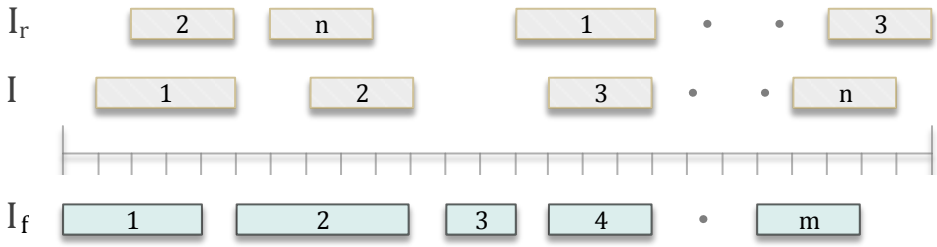


Figure 2.1. A Schematic of the interval overlap problem. I_f denotes the reference collection of intervals, and I represents the query collection. The randomized set I_r is generated by relocating the intervals in I such that all possible non-overlapping random sets are equiprobable.

2.2 Methods

Let us first introduce the notation frequently used throughout this chapter. Let I_f denote a 'reference' collection of intervals, and I denote a 'query' collection of intervals (Figure 2.1). We use the space-counted, zero-start convention for the genomic coordinates. Namely, we count the space between bases starting from 0 (the one before the first base) up to g (the one after the last base), where g denotes the length of the genomic region of interest. Thus, each interval is denoted by a pair of indices (u_1, u_2) with $0 \leq u_1 < u_2 \leq g$, and is composed of the nucleotides between u_1 and u_2. We use 'i' to index the intervals in query set I, which has total number of n intervals, and designate 'j' to index the intervals in reference set I_f, which consists of m intervals in total. The length of i-th query interval and j-th reference interval are represented by l_i and x_j, respectively. Two intervals (u_1, u_2) and (v_1, v_2) overlap iff they share common nucleotide(s). A collection of intervals is *non-overlapping* if no pair of intervals in the collection overlap.

Problem formulation

Let $I_f \sqsubseteq I$ denote the subset of intervals in I_f that are *hit* (overlap with intervals in I). Suppose $|I_f \sqsubseteq I| = k$. We measure the significance (p-value) of this observation by sampling a random set of intervals I_r with the following properties (See Figure 2.1)

- $|I_r| = |I|$. I_r has exactly n elements.
- Intervals in I_r have the same lengths as the intervals in I.

- The location of intervals in I_r are drawn from a distribution (implicitly) such that all possible random sets are equally likely.

Let I_r be drawn according to the process above, then p-value is defined as

$$P\text{-value}(k) = \Pr(|I_f \sqsubseteq I_r| \geq k).$$

While the computational complexity of the problem is not known, we can argue that it is hard. Clearly, the number of possible random sets is very large; ranging from $\binom{g+n-\sum_i l_i}{n}$ when all l_i's are identical, to $\binom{g+n-\sum_i l_i}{n} n!$ when all l_i are distinct. For typical values of $g = 2 \cdot 10^8$ (length of a chromosome), $n = 100$ (number of annotated regions), and $\sum l_i = 10^6$ (total length of regions covered by an annotation), counting all possibilities naively to compute $\Pr(|I_f \sqsubseteq I| \geq k)$ is computationally intractable. Thus, we impose the restriction that the intervals in I_r must retain the same order as the intervals in I, and present a dynamic programming (DP) algorithm to compute the number of distinct random sets with $|I_f \sqsubseteq I_r| = k$, for all k. In practice, to apply the algorithm to large genomes with abundant annotation we use a practical interval 'scaling' scheme by considering the natural partitioning of the genome into intervals and the gaps amidst them, and scale each interval and gap in I and I_f by a fraction ν. Ideally, we want to have $\nu = 1$, but large problems require smaller fractions to make the computation feasible from both running time and memory usage aspects. Nevertheless, we show that the algorithm still yields a close approximation of p-value.

2.2.1 Dynamic programming algorithm

For interval i in I_r, genomic location h, $(1 \leq h \leq g)$, $0 \leq k \leq m$, $a \in 0,1$, let $N(i,h,k,a)$ denote the number of arrangements of the first i intervals in I_r such that (see Figure 2.2):

- The i-th interval ends exactly at location h.
- k intervals in I_f are hit by the first i intervals in I_r.

- $a = 0$ if the interval from I_f that spans h (if any) has not been counted earlier; $a = 1$ otherwise.

We also define $N_1(i,h,k,a)$ identically to $N(i,h,k,a)$ with the exception that the i-th interval ends at or before location h. Note that if the j-th interval in I_f spans h, it is counted as a hit, but

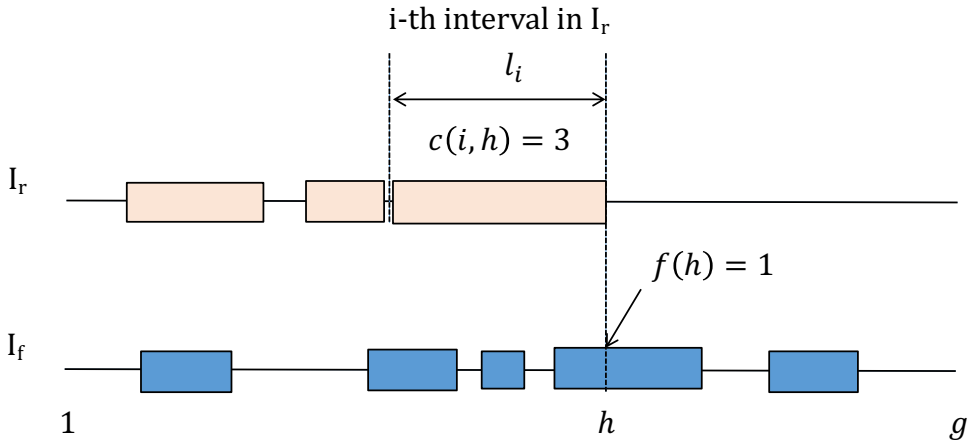


Figure 2.2. Cartoon for dynamic programming.

may have already been counted by some other interval in I_r. Although a separate function can be defined to store that information, we use a as an indicator in dynamic programming for the sake of brevity. In order to compute $N_1(i,h,k,a)$, we must define some auxiliary functions. Let $c(i,h)$ denote the number of intervals in I_f which intersect with $(h-l_i,h)$ in I_r. While evaluating $c(i,h)$, (j_1,j_2) in I_f is counted as an intersecting interval with $(h-l_i,h)$ if $j_1 < h$ and $j_2 > h-l_i$. We also define binary function $f : (0,g] \to \{0,1\}$, where $f(h) = 1$ if some interval in I_f spans h, and $f(h) = 0$ otherwise. See Figure 2.2. For the simplicity of exposition, it is assumed that a single nucleotide overlap between two intervals from I_r and I_f is sufficient to count the reference interval as intersected. The generalization of algorithm to accommodate stricter conditions is straightforward and can be done by modifying the definition of $c(i,h)$ and $f(h)$ (Appendix A.1).

To explain the recurrences, note that $N_1(i,h,k,a)$ can be computed by adding cases where the i-th interval ends exactly at h, and cases where the i-th interval ends strictly before h. To compute $N(i,h,k,a)$ we need to consider all arrangements where the first $i-1$ intervals in I_r

ends before the start of the i-th interval at $h - l_i$.

$$N_1(i,h,k,a) = \begin{cases} N(i,h,k,a) & h = 1 \\ N(i,h,k,a) + N_1(i,h-1,k,\min\{a, f(h-1)\}) & \text{Otherwise} \end{cases}$$

$$N(i,h,k,a) = \begin{cases} 0 & h < \sum_{x=1}^{i} l_x \text{ or } k < c(i,h) - a \\ 1 & i = 1 \text{ and } k = c(i,h) - a \\ N_1(i-1,h-l_i,k-c(i,h)+a, f(h-l_i)) & \text{Otherwise} \end{cases}$$

$$1 \le i \le n, \quad 1 \le h \le g, \quad 0 \le k \le m, \quad a \in \{0,1\} \tag{2.1}$$

We note a technical difference between non-overlapping and 'disjoint'. Intervals (i_1, i_2) and (i_2, i_3) are non-overlapping as they do not share any nucleotide, but are not disjoint because we cannot distinguish them from interval (i_1, i_3). The case where I_r is restricted to be disjoint is described in Appendix A.4. The *DP* p-value ($\Pr(|I_f \sqsubseteq I_r| \ge k)$) can be computed using the ratio

$$P\text{-value}(k) = \frac{\sum_{\kappa=k}^{m} N_1(n,g,\kappa,0)}{\sum_{\kappa=0}^{m} N_1(n,g,\kappa,0)}\,.$$

Recall that the total number of configurations is

$$\sum_{\kappa=0}^{m} N_1(n,g,\kappa,0) = \binom{g - \sum_{i=1}^{n} l_i + n}{n}.$$

which can be very large and surpass the upper limit of ordinary data types. Therefore, we perform all calculations using a logarithmic scale (Appendix A.5).

Time complexity

The number of iterations to complete the table of values for $N_1(i,h,k,a)$ is $\mathcal{O}(ngm)$. The functions $c(i,h)$ and $f(h)$ can be pre-computed (using a modified version of binary search algorithm), so each iteration is computed in a constant time. Therefore, the total time complexity

is $\mathcal{O}(ngm)$ which is pseudo-polynomial because the input size is $\mathcal{O}((n+m)\log g)$. The running time can be reduced to $\mathcal{O}(ng\nu m)$ by scaling the genome using scaling factor $\nu < 1$. We also use a number of tricks to improve the speed of computations, including lowering memory usage from $\mathcal{O}(ngm)$ to $\mathcal{O}(gm)$. We should note that this time complexity is achieved under the assumption that the order of intervals in I_r is same as I. In Results, we show that choosing different orders does not significantly change the p-value.

Multiple chromosomes

In many cases of interest, the intervals reported are on multiple chromosomes, with a non-uniform distribution across chromosomes. Therefore, the appropriate random interval set I'_r may only allow permutation of interval positions within the chromosome it is originally assigned to. For this alternative null model, the DP algorithm is applied to each chromosome to enumerate rearrangements of intervals within each chromosome, and then the results are combined to compute the overall p-value. See Appendix A.6.

The general case

We finally tackle the case of computing p-values with no assumptions using sampling techniques. The simplest method is a permutation like test where many (e.g., 10^7) random trials are used and we count the fraction of times when the statistic (k) is exceeded. As the number of trials required is $\Omega\left(\frac{1}{p\text{-value}(k)}\right)$, the direct approach does not work when the event is very rare. Conditional sampling offers a way out of this conundrum. Define

$$f_s = \Pr\left(|I_f \sqsubseteq I_r| \geq s \;\middle|\; |I_f \sqsubseteq I_r| \geq s-1\right) = \frac{\Pr(|I_f \sqsubseteq I_r| \geq s)}{\Pr(|I_f \sqsubseteq I_r| \geq s-1)} \,.$$

Using telescoping products and the fact that $\Pr(|I_f \sqsubseteq I_r| \geq 0) = 1$, we get

$$p\text{-value}(k) = \prod_{s=1}^{k} f_s \,. \tag{2.2}$$

If we had a procedure to estimate f_s efficiently (time $\mathcal{O}(1/f_s)$), we could use Eqn. 2.2 to estimate the p-value in time $\mathcal{O}\left(\sum_{s=1}^{k}\frac{1}{f_s}\right) = \mathcal{O}\left(\frac{m}{\min_s f_s}\right)$ instead of $\mathcal{O}\left(\frac{1}{p\text{-value}(k)}\right)$. If $\min_s f_s \gg m \cdot p\text{-value}(k)$, we get a significant time reduction.

Let $Z_{\geq s}$ denote the set of all configurations I_r s.t. $|I_f \sqsubseteq I_r| \geq s$. If for all $s > 1$, we could sample uniformly at random from configurations in $Z_{\geq s-1}$, then f_s could be estimated simply by keeping track of the fraction of configurations in which $\geq s$ intervals were hit. We use a novel Markov Chain Monte Carlo based method to sample uniformly from $Z_{\geq s}$. At a high level, we choose a permutation π of intervals in I, and a subset S of I_f with $|S| \geq s$ so that all and only the intervals in S are hit using the last $|S|$ intervals in π. The other intervals in π can be assigned arbitrary locations constrained only in that they cannot hit any interval in I_f, and maintain the order dictated by π. By definition, any configuration I_r s.t. $|I_f \sqsubseteq I_r| \geq s$ can be assigned to a unique tuple $\langle \pi, S \rangle$. Let $\mathscr{C}(\pi, S)$ denote the set of configurations in $Z_{\geq s}$ assigned to $\langle \pi, S \rangle$. We consider a markov chain with each state designated by $\langle \pi, S \rangle$, and the target probability given by

$$\rho(\langle \pi, S \rangle) = \frac{|\mathscr{C}(\pi, S)|}{|Z_{\geq s}|}$$

In the MCMC procedure, we sample states from the markov chain according to their target probabilities. Next, for each state $\langle \pi, S \rangle$ that was sampled, we output a configuration uniformly at random from the set $\mathscr{C}(\pi, S)$. By construction, each configuration is output with probability $\frac{1}{|Z_{\geq s}|}$. While this procedure described the main idea, it does not quite work because $|\mathscr{C}(\pi, S)|$ is difficult to estimate efficiently. In this section, we describe the details of the modified MCMC procedure with a proposal distribution and a transition probability that satisfy detailed balance, ensuring convergence to the target distribution.

2.2.2 Poisson binomial approximation

For the case that annotations contain too many intervals such that the processing resources to run DP algorithm cannot be afforded, we provide an approximation which is reasonably close

under certain condition. For simplicity, we remove the non-overlapping assumption on I_r. Thus, I_r is a randomly located collection of n intervals of lengths $l_1, l_2, l_3, \ldots, l_n$ with arbitrary order. Let E_{ij} denote the event that the j-th interval in I_f is intersected by the i-th interval in I_r. Then,

$$p_{ij} \coloneqq \Pr(E_{ij}) = \frac{l_i + x_j - 1}{g}$$

As before, we assumed that a single nucleotide overlap is sufficient, but it can be easily

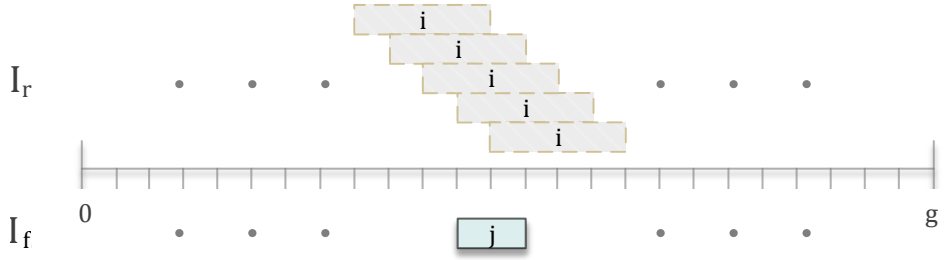


Figure 2.3. Illustration of the cases that i-th interval from I_r intersects j-th interval in I_f.

generalized to a more strict overlap condition (Appendix A.1). Let $\overline{E}_{ij}$ be the event that the i-th interval in I_r does not intersect the j-th interval in I_f. In the absence of the non-overlapping assumption on I_r, the events $\overline{E}_{ij}$, $i = 1, 2, \ldots, n$, are independent, and the probability of their intersection is given by the product of individual probabilities. Therefore, the probability of $E_j = \cup_{i=1}^{n} E_{ij}$, which is the event where interval $j \in I_f$ is hit by I_r, can be calculated as

$$P_j \coloneqq \Pr(E_j) = \Pr(\cup_{i=1}^{n} E_{ij}) = 1 - \Pr(\cap_{i=1}^{n} \overline{E}_{ij}) = 1 - \prod_{i=1}^{n} \Pr(\overline{E}_{ij}) = 1 - \prod_{i=1}^{n} (1 - \Pr(E_{ij})). \quad (2.3)$$

Now consider the binary indicator variable X_j, where $X_j = 1$ iff event E_j occurs. We have m Bernoulli experiments with success probabilities $P_1, P_2, \ldots, P_m$, and we are interested in computing $\Pr(\sum_j X_j = k)$. In general, there are dependencies between E_j's for different values of j. However, under certain condition where intervals are not too close or far away, we can approximately assume independence between different intervals. The sum of m independent

Bernoulli trials with different success probabilities is a Poisson binomial (PB) distribution [20].

$$\Pr(\sum_{j=1}^{m} X_j = k) = \sum_{A \in F_k} \prod_{u \in A} P_u \prod_{v \in A^c} (1 - P_v) \tag{2.4}$$

where F_k is the set of all subsets of $\{1, 2, \ldots, m\}$ with k elements. Eqn. 2.4) allows us to compute the p-value as

$$P\text{-value}(k) = \Pr(\sum_{j=1}^{m} X_j \geq k).$$

We cannot directly use Eqn. 2.4 by enumerating over all elements in F_k, but use a recursive approach to compute it, following Hong [21]. It is reproduced here for completeness. Let $\pi_{k,j} = \Pr(\sum_{u=1}^{j} X_u = k)$ denote the probability of getting k hits in the first j intervals in I_f. Our goal is to compute $\Pr(\sum_{u=1}^{m} X_u = k) = \pi_{k,m}$. All values $\pi_{k,j}$ can be computed in $\mathcal{O}(m^2)$ time using

$$\pi_{k,j} = P_j \pi_{k-1,j-1} + (1 - P_j)\pi_{k,j-1}, \quad 0 \leq k \leq m, 0 \leq j \leq m \tag{2.5}$$

with the boundary conditions $\pi_{-1,j} = \pi_{j+1,j} = 0, j = 0, 1, \ldots, m$ and $\pi_{0,0} = 1$. Other FFT based methods are also applicable [21].

With the above PB approximation, we assume that the event of an interval in I_f being hit is independent of other intervals being hit, greatly reducing the computational complexity of the problem. To understand the impact of this assumption, we introduce a new parameter. Recall that $P_j = \Pr(E_j) = \Pr(X_j = 1)$ is the probability that interval j (length x_j) in I_f is hit by some interval in I_r. Let d_j denote the distance of interval j from interval $j-1$. Define $\Delta \coloneqq (m-1) \cdot \textbf{median}\{d_j | \, j = 2, 3, \ldots, m\}$, and $\eta \coloneqq \frac{\Delta}{g}$. Parameter η is a measure of the 'spread' of intervals in I_f. For $\eta \ll 1$, and j' sufficiently close to j, we expect to have

$$\Pr(X_j = 1 | X_{j'} = 1) > \Pr(X_j = 1) \, .$$

In other words, if intervals in I_f are clumped, then E_j, E'_j are not statistically independent but

positively correlated, and we will underestimate the true p-value. For larger values of η, and j, j' sufficiently distant,

$$Pr(X_j = 1 | X_{j'} = 1) < Pr(X_j = 1) \ ,$$

The negative correlation leads to an over-estimation of the p-value. To better recognize this effect, imagine an extreme case where $n < m$ and due to the size and spread of intervals in I_f, at most n intervals in I_f can be hit. Therefore, p-value$(n+1) = \Pr(\sum_j X_j > n) = 0$. The independence assumption in PB computation, though, will lead to a non-zero value (over-estimate) for p-value$(n+1)$.

2.2.3 MCMC sampling

Recall that we generate a configuration I_r by randomly reassigning locations of intervals in I. For parameter s, let $Z_{\geq s}$ (respectively Z_s) denote the set of distinct configurations that hit at least s (respectively, exactly s) intervals from I_f. If we can get $z_{\geq s}$ samples (uniformly sampled) from $Z_{\geq s}$ and count the number of sample z_s in which exactly s intervals were hit (which means that they are in $Z_s \cap Z_{\geq s}$), then we can estimate f_s as (SSdifferent from the definition of f_s in the main text):

$$f_s = 1 - \frac{z_s}{z_{\geq s}} \ .$$

Consider a permutation π of the intervals in I. Define a configuration satisfying π as an assignment of starting coordinates p_i to interval π_i for all i. Similarly, we denote the starting coordinate of interval j in I_f as f_j and its ending coordinate as e_j. For simplicity of exposition, we will consider the case where any gap in I_f is larger than the largest interval in I. Consider an ordered subset $S \subseteq I_f$, with $|S| \geq s$, and a randomly chosen positioning $A_{\pi,S}$ of the last $|S|$ intervals $(a_{n-|S|+1}, \ldots, a_n)$ in π so that for all $1 \leq j \leq |S|$, interval $\pi_{n-|S|+j}$ of I overlaps with the j-th interval in S. Let $\mathscr{C}(\pi, S, A_{\pi,S})$ denote the set of configurations of π so that none of the first

$n-|S|$ intervals in π hit any interval in I_f, while $p_j = a_j$ for all $n-|S| < j \leq n$. By definition,

$$\sum_{\pi,S,A_{\pi,S}} |\mathscr{C}(\pi,S,A_{\pi,S})| = |Z_{\geq s}|$$

We will design a Markov Chain as follows: each state ζ is characterized by the triple $\zeta = \langle \pi, S, A_{\pi,S} \rangle$, where $|S| \geq s$. Define a *target distribution* on the states of the markov chain by

$$\rho(\zeta) = \frac{|\mathscr{C}(\zeta)|}{|Z_{\geq s}|} \,. \tag{2.6}$$

To sample uniformly from $Z_{\geq s}$, we do the following:

1. Use a Monte Carlo procedure to sample each state $\zeta = \langle \pi, S, A_{S,\pi} \rangle$ according to its target probability $\rho(\zeta)$.
2. Output a configuration uniformly at random from $\mathscr{C}(\zeta)$.

By construction, each configuration is output with probability $\frac{1}{|Z_{\geq s}|}$.

Monte Carlo Sampling from the Markov Chain

Consider an arbitrary $\zeta = \langle \pi, S, A_{\pi,S} \rangle$ with $|S| \geq s$. A configuration $\zeta' = \langle \pi', S', A_{\pi',S'} \rangle$ is a neighbor of ζ if either

$\pi' = \pi$. In this case, either $S' = S - \{x\} + \{y\}$ for $x \in S, y \notin S$, or $S' = S - \{x\}$ for $x \in S$, or $S' = S + \{y\}$ for $y \notin S$, or $S' = S$. Each of the $|S|(m-|S|) + (|S|) + (m-|S|) + 1$ choices is picked with equal probability, except when $s' = s$, or $s' = m$, OR

$\pi' \neq \pi$. In this case, a new permutation π' is chosen by randomly exchanging two elements, and set $\boldsymbol{S' = S}$.

We choose each of the two possibilities above with equal probability to get π', and choose S' uniformly from available choices. Given a choice of π', S', we choose $A_{\pi',S'}$ uniformly from all available choices. With this procedure, we can compute the *proposal distribution* $\Pr(\zeta'|\zeta)$.

For example, suppose we have the case where $\pi' = \pi$, $S' = S - \{x\} + \{y\}$ for some $x \in S$, $y \notin S$. Then,

$$\Pr(\zeta'|\zeta) = \frac{1}{2} \cdot \frac{1}{|S|(m-|S|)+m+1} \cdot \frac{1}{|A_{\pi',S'}|} .$$

Note that we are free to choose any proposal distribution as long as it can be computed efficiently. However, the choice might impact convergence time of the Markov Chain. The *Acceptance probability* $A(\zeta \to \zeta')$ is given by the Metropolis-Hastings rule.

$$A(\zeta \to \zeta') = \min\left\{1, \frac{\Pr(\zeta|\zeta')\rho(\zeta')}{\Pr(\zeta'|\zeta)\rho(\zeta)}\right\} = \min\left\{1, \frac{\Pr(\zeta|\zeta') \cdot |\mathscr{C}(\zeta')|}{\Pr(\zeta'|\zeta) \cdot |\mathscr{C}(\zeta)|}\right\} \tag{2.7}$$

The transition probability $T(\zeta \to \zeta')$ is given by

$$T(\zeta \to \zeta') = \Pr(\zeta'|\zeta)A(\zeta \to \zeta') ,$$

and satisfies the detailed balance condition,

$$\rho(\zeta)T(\zeta \to \zeta') = \rho(\zeta')T(\zeta' \to \zeta) ,$$

ensuring convergence to the Target distribution.

Computing $|\mathscr{C}(\zeta)|$

The algorithm above (Eqn. 2.7) assumes that we can compute $|\mathscr{C}(\pi, S, A_{\pi,S})|$ for all ζ, and we describe a dynamic programming approach to achieve that goal. In the following let $s = |S|$. Once we have $A_{\pi,S}$, we can create a new set I'_f, where *the starting positions, and/or the ending positions* for the s intervals in S have changed (Figure 2.4). Denote the new fixed set as I'_f with starting positions f'_j, and ending positions e'_j. Similarly, the new query set π' is the same as π, but restricted to the first $n-s$ intervals. Let $\mathscr{N}(\pi', I'_f)$ denote the number of assignments of $p_1, p_2, \ldots, p_{n-s}$ so that none of these $n-s$ intervals in π' overlaps with any of the intervals in I'_f. Computing $\mathscr{N}(\pi', I'_f)$ and multiplying it by $|A_{\pi,S}|$, we get $|\mathscr{C}(\pi, S, A_{\pi,S})|$. Hence, we propose

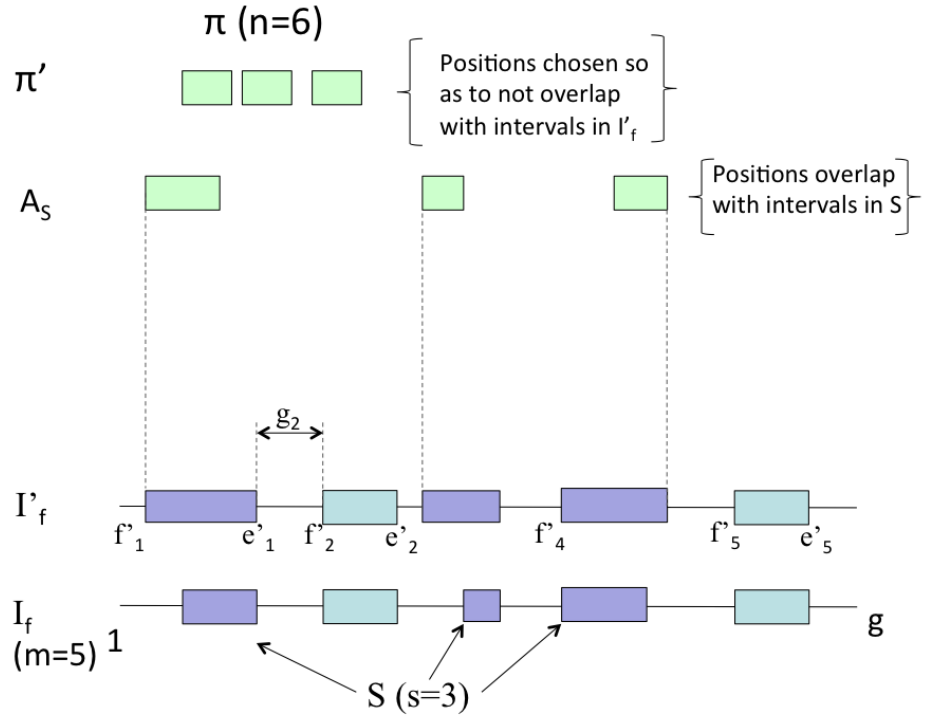


Figure 2.4. Counting $|\mathscr{C}(\zeta)|$ for any state $\zeta = \langle \pi, S, A_{\pi,S} \rangle$.

an algorithm to compute $\mathscr{N}(\pi', I'_f)$.

Think of the intervals in I'_f as barriers. For all $1 \leq i_1 \leq n-s$, $1 \leq i_2 \leq i_1+1$, $1 \leq j \leq m+1$, let $N(i_2, i_1, j)$ denote the number of ways of configuring the first i_1 intervals so that exactly the intervals between i_2 and i_1 (inclusive) lie between the barriers $j-1$ and j. In other words, intervals from i_2 to i_1 must lie in the interval (e'_{j-1}, f'_j). In this notation, $N(i_1+1, i_1, j)$ denotes the number of configurations of the first i_1 intervals so that no interval falls in (e'_{j-1}, f'_j). Let $g_j = f'_j - e'_{j-1}$, and $l(i_2, i_1) = \sum_{i_2 \leq i \leq i_1} l_i$. The number of possible ways of configuring intervals from i_2 to i_1 such that they fall in (e'_{j-1}, f'_j), denoted by $C(i_2, i_1, j)$, is given as

$$C(i_2, i_1, j) = \begin{cases} 1 & \text{if } i_2 = i_1 + 1 \\ 0 & \text{if } l(i_2, i_1) \geq g_j \text{ and } i_2 \leq i_1 \\ \binom{g_j - l(i_2,i_1) + (i_2 - i_1 + 1)}{(i_2 - i_1 + 1)} & \text{otherwise.} \end{cases}$$

Then,

$$N(i_2, i_1, j) = \left(\sum_{i_3 \leq i_2} N(i_3, i_2 - 1, j - 1) \right) \cdot C(i_2, i_1, j)$$

Note that, $\sum_{1 \leq i_2 \leq n-s+1} N(i_2, n-s, m+1)$ gives us $\mathscr{N}(\pi', I'_f)$. Finally, $|\mathscr{C}(\pi, S, A_{\pi,S})| = \mathscr{N}(\pi', I'_f) \cdot |A_{\pi,S}|$.

2.3 Results

2.3.1 Performance on simulated data

We simulated intervals in a randomly generated chromosome to test the performance of ISTAT. To study the impact of scaling and fixed-order assumption on the DP algorithm, we chose $g = 200$Mbp, and the two sets of intervals I and I_f with $n = m = 100$ intervals. The intervals in I and I_f are generated with random lengths l_i and x_j distributed uniformly over $[1\text{Kbp}, 10\text{Kbp}]$. The intervals in I_f were placed uniformly at random along the chromosome, while ensuring no overlap between them. To benchmark the speed of DP algorithm, we changed n, m, and g over a range of values and measured the running time of ISTAT. We also simulated intervals in I_f distibuted non-uniformly over the chromosome to study how their positional distribution impacts the quality of PB approximation.

The impact of scaling on DP p-value

The algorithm has substantial demands on memory and time. To allow it to work on the human genome, we scale down the intervals and the gaps between them by a fraction ν. To test the impact of scaling, we considered the example of a chromosome described above, with $g = 200$Mbp, and $n = m = 100$. The impact on DP p-values due to scaling with $\nu \in \{1, 10^{-1}, 10^{-2}, 10^{-3}\}$ is shown in Figure 2.5a. As can be observed, scaling preserves the p-values tightly. To further investigate robustness of DP p-value computation to the scaling, we also considered an adversarial example where I and I_f contain intervals smaller than ν^{-1}. For that purpose, the length of intervals generated from a uniform distribution over $[100\text{bp}, 4\text{Kbp}]$,

thus when we apply scaling factor $\nu = 10^{-3}$ about one fourth of intervals are smaller than the resolution ν^{-1} and become unit intervals. Nevertheless, p-values obtained with $\nu = 10^{-3}$ tightly followed finer-scale p-values (Figure A.1a), verifying that we can apply reasonable scaling factor (with respect to the distribution of the length of intervals and gaps) to facilitate p-value computation using DP algorithm.

Effect of order on p-value

To test the effect of fixed order on p-value, we used a scaling factor $\nu = 10^{-3}$, and applied the DP method to 100 random instances of simulated intervals described before, each with a random permutation of I. In Figure 2.5b, we plotted the mean p-value for all k, as well as the standard error of the mean. We observe that the standard error is distributed tightly around the mean, while its ratio to the mean increases slightly for smaller p-values. The mean p-value range from $0.4320 \pm 2.279 \cdot 10^{-4}$ for $k = 1$ to $1.017 \cdot 10^{-269} \pm 6.246 \cdot 10^{-271}$ for $k = 100$. The results suggest that fixing the order in DP algorithm to compute the p-value is an acceptable compromise for many real data-sets.

Running time

Using a desktop PC with Intel Core i7-6700K CPU and 32GB DDR4 RAM, the running time of our DP algorithm (in a logarithmic scale) versus the number of query intervals is plotted in Figure 2.5c for a number of scaling factors. The running time scales almost linearly with the number of query intervals n. It also scales linearly with the number of reference intervals m (Figure A.1b) and the size of chromosome g (Figure A.1c), and when larger scaling factors is used.

PB versus DP

To test the role of η in p-value estimation, we compared the p-values of the Poisson binomial method against the DP method for different values of η. See Figure 2.5d–f. Relative to the DP, the PB approximation underestimates p-values when $\eta = 0.005445$ (Figure 2.5d), and over-estimates for $\eta = 0.6197$ (Figure 2.5f). However, this over-estimation is not as pronounced

as the under-estimation in the case of clumping, and reduces with large n (Figure A.1ef). In our simulations, we changed the number of query and reference intervals as well as the spread of reference intervals over the genome, and compared the p-values computed using each method. Although the closeness of PB approximation is a complicated function of the distribution of intervals and its exact characterization is hard, as a rule of thumb we suggest to consider using DP (with the largest computationally-feasible scaling ν) when $\eta < 0.06$, to avoid liberal p-values (more significant). In the case that we have multiple chromosomes, the minimum η among all chromosomes can be considered to be as conservative as possible.

2.3.2 Enrichment analysis on real data

We also took four examples from the literature and applied ISTAT to to test its performance on interval data from previously published studies and compare the p-values estimated by ISTAT with the reported p-values. The first example comes from [22], relating to matching of focal copy number changes in tumor genomes. The second dataset is from [19] where a map of copy number variation (CNV) in the human genome is provided, and different genomic elements are investigated for the presence/absence of CNVs. We also ran ISTAT on an example from epigenetics context [18], where the promoters are found to be enriched for H3K4 methylation. The last example was extracted from an effort to systematically annotate genome by the means of characterizing chromatin states [23].

TCGA-CNV enrichment in HIRT (extra-chromosomal data)

For I_f, we chose a collection of intervals with recurrent copy number amplifications in the TCGA array CGH data-set (named TCGA-CNV) [24, 25]. For I, we used amplified genomic regions from a whole genome sequencing experiment with an experimental protocol, HIRT, that preferentially selected extra-chromosomal fragments. A strong enrichment of TCGA-CNV intervals in the HIRT intervals would suggest that *many copy number amplifications can be attributed to the formation and independent replication of episomes (extra-chromosomal*

elements). The number of intervals in query and reference sets were not large, $n = 116$ and $m = 101$, so we did not scale the intervals and the resulting p-value is $8.679 \cdot 10^{-6}$. For comparison, we applied scaling factor $\nu = 10^{-1}$ and the change in p-value was very small. As expected from $\eta = 0.001$, PB approximation gives more significant p-value $= 2.642 \cdot 10^{-10}$ (Figure 2.6a).

Non-coding genes enrichment in CNVs

We chose the set of all CNV gains from the inclusive map as I, and the set of all non-coding genes as I_f, containing $n = 3132$ and $m = 9058$ intervals, respectively. Using the scaling factor $\nu = 10^{-2}$, we obtained p-value $= 5.216 \cdot 10^{-18}$, confirming high enrichment of non-coding genes in CNV gains. After applying an order of magnitude smaller scaling factor $\nu = 10^{-3}$, we get very close p-value $= 2.532 \cdot 10^{-18}$ which shows that scaling with $\nu = 10^{-2}$ is fine (Figure 2.6b). PB approximation p-value is $1.370 \cdot 10^{-52}$, much smaller p-value that is consistent with $\eta = 0.024$. In the paper, they consider the exons of non-coding genes as I_f, and report p-value $= 0.0001$ from a 10000 randomized dataset, which shows the limited resolution of basic permutation tests. The result of our algorithm indicates that computing the exact p-value in this case requires at least about 10^{18} randomized samples, which is impossible. In the supplementary they have also reported a binomial p-value $= 2.32 \cdot 10^{-54}$.

Enrichment of H3K4me3 in promoters

In [18] authors found that 74% of all annotated promoters were enriched for H3K4 methylation, concluding that a large fraction of genes with no detected transcript have promoter-proximal nucleosomes enriched for H3K4me3 modification. To evaluate the statistical significance of this observation, we took the set of regions highly enriched for H3K4me3 in ES cells (provided as supplementary information in [18]) as the query set. However, they have not provided the coordinates of promotors, and so for the reference intervals, we used the collection of all promoters (-5.5Kbp to 2.5Kbp relative to TSS of all RefSeq genes) as the reference set of intervals. Although with I_f that we used we did not get the same ratio of overlap as reported

in the paper, but still the p-value is quite significant. At the observed overlap, PB p-value is $1.775 \cdot 10^{-76}$, while DP p-value with $\nu = 10^{-2}$ is $2.734 \cdot 10^{-82}$. For this example, $\eta = 0.1$ so PB approximation gives conservative p-values as expected (Figure 2.6c).

Enrichment of promoters in promoter-associated chromatin states

Among 51 identified chromatin states, states 1 to 11 were referred to as promoter-associated states because of high enrichment for promoter regions. We tried to compute the p-value of enrichment by considering the set of all promoter regions (within 2Kbp of RefSeq TSS) as the query set I, and 200-bp intervals identified with state 9 as the reference set I_f. The p-value $= 1.588 \cdot 10^{-8}$ (under the scaling factor $\nu = 10^{-2}$) shows that it would be very unlikely to observe such overlap only by chance, yet it is much less significant than the p-value reported by the authors ($\leq 10^{-200}$), computed using hypergeometric distribution. As $\eta = 0.01$, PB approximation expectedly gives liberal p-value (Figure 2.6d).

2.4 Discussion

Our results explore the statistics of interval overlaps. The question is quite natural in the post genomic era where annotating the genome for function, structure, and variation and identifying correlated annotations is a key problem. While scientists have used many different ways to compute the significance of overlap between two sets of intervals, their computations often do not explicitly state the assumptions on the null model, or accurately compute the p-values given specific assumptions.

To the best of our knowledge, the p-value computation for sets of overlapping intervals has been limited to either permutation tests which do not scale to computation of small p-values, or simple parametric tests such as hypergeometric or binomial tests which are based on simplifying assumptions about the length and structure of intervals. Our method, however, formulates a null model where the size of intervals and their relative arrangement are considered when the significance of overlap is evaluated. We explicitly state the assumptions that we have

made in our proposed model, and assess the impact of our assumptions thorough the experiments on simulated and real datasets. Computation of exact p-values may be necessary in some cases. For example, p-values can be used to compare the significance of two 'competing' annotations with different numbers of intervals (n) and intersections (k). We develop a novel frame-work that makes exact computation of p-value possible, even for very small p-values.

The proposed DP method is able to compute very small p-values by efficiently counting the number of possible random rearrangements of intervals resulting in specific amount of overlap. Although we assume that the order of intervals is not changed, and it may be possible to construct adversarial examples where changing the order has a material impact on p-values, but our simulation of typical examples of interval data show that the resulting change in p-values is not significant. Our experiments on simulated and real datasets also suggest that to improve the speed and memory usage, we can employ reasonable scaling factors and sill obtain accurate p-values.

The Poisson binomial approximation is very efficient to compute. However, our results suggest that for typical values found in real-life examples, the independence assumption is too strong, and might result in under-estimated p-values, or the false reporting of some overlap as being significant. Nevertheless, we have introduced parameter η which can be readily computed from the data before running the DP method, to estimate the accuracy of PB method compared to DP algorithm results. Future work should look into more systematic characterization of PB approximation.

Throughout our experiments, we let the intervals to be uniformly distributed over the whole extent of chromosomes. However, one might be interested in a non-uniform distribution of intervals under the null model, to account for confounding variables such G/C content, sequence context, or intergenic/genic region. Our methods can be used in such cases by confining the problem to the specific regions of interest. Hence, only intervals falling into such regions are considered, and g would be the total length of the segments that intervals are allowed to be distributed there. Moreover, we considered the overlap of two intervals as a binary event, and

defined the statistic based on the number of overlapping intervals. However, DP method can be modified to compute the *p*-value when the overlap statistic is defined based on the total amount of shared base pairs instead. Thus, we provide this as an option in ISTAT software and give the user the flexibility of choosing the appropriate measure of overlap for their specific application.

Chapter 2, in full, is a reprint of the material as it appears in Cell systems 8, no. 6 (2019): 523-529. "Computing the statistical significance of overlap between genome annotations with iStat". Shahab Sarmashghi and Vineet Bafna. The dissertation author was the primary investigator and author of this paper.

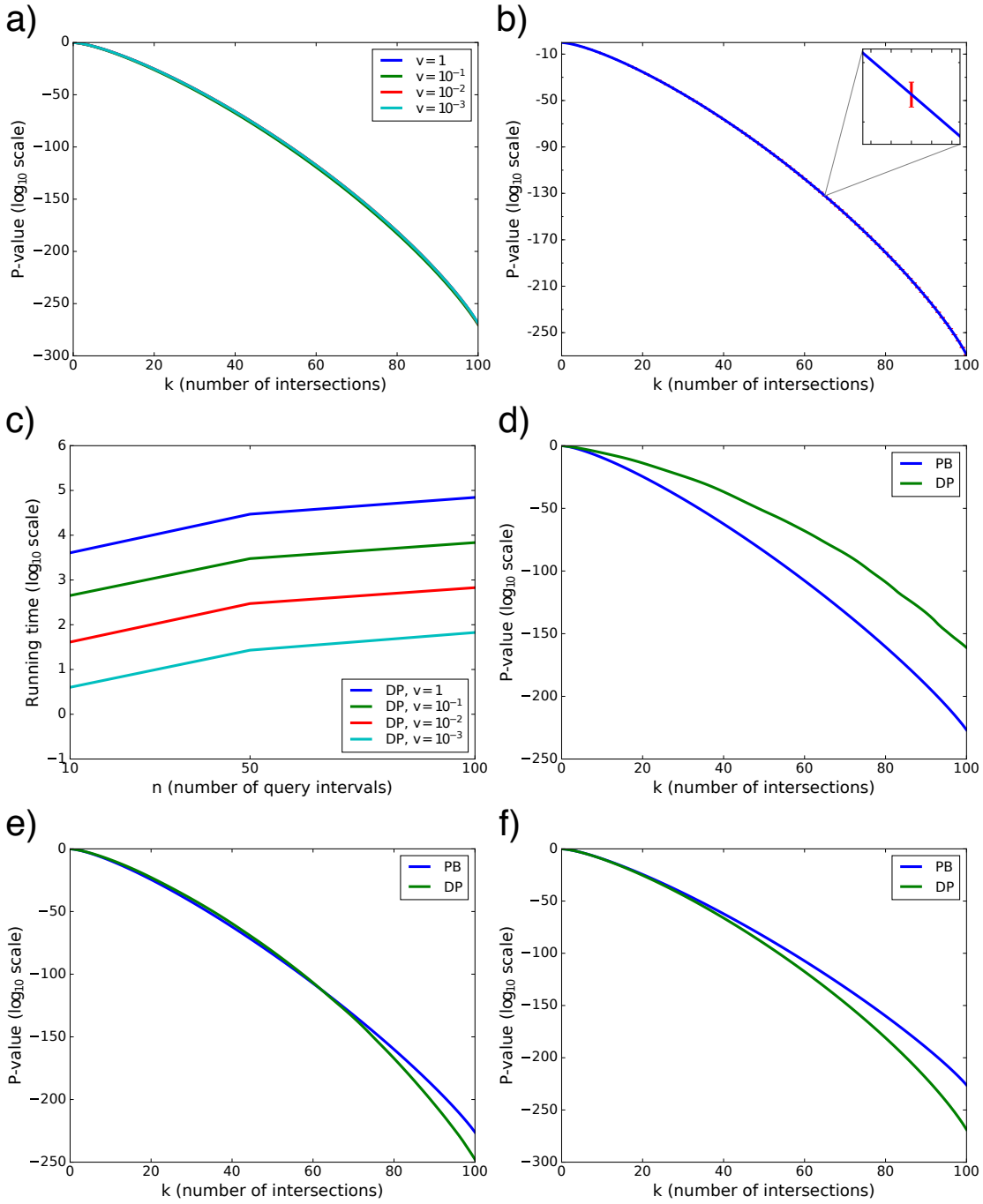


Figure 2.5. Testing methods on simulated data. (a) Impact of scaling parameter ν on DP p-value when $l_i, x_j \sim \mathscr{U}[1\text{Kbp}, 10\text{Kbp}]$. (b) Impact of ordering on DP p-value, with $\nu = 10^{-3}$. The mean of 100 p-value computations for random orderings is plotted, and the error bars represent the standard error of the mean. (c) Running time (in secs.) of DP algorithm as a function of n, with $m = 100$ and $g = 200\text{Mbp}$. (d–f) Impact of approximation on p-value computation. Simulations are run with $g = 200\text{Mbp}$, $m = 100$, $n = 100$, $l_i, x_j \sim \mathscr{U}[1\text{Kbp}, 10\text{Kbp}]$; (d) $\eta = 0.0054$, (e) $\eta = 0.053$, (f) $\eta = 0.62$.

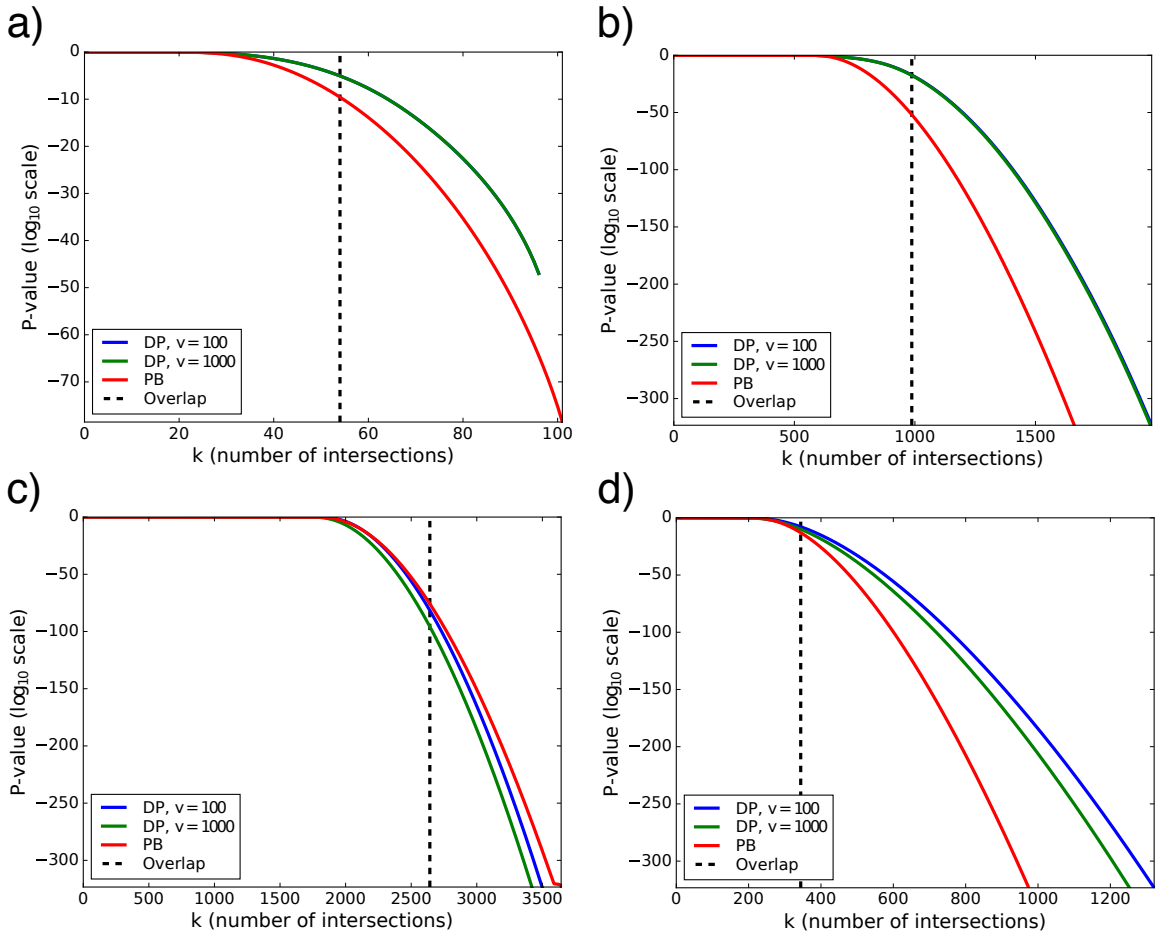


Figure 2.6. Enrichment analysis on real datasets. (a) TCGA-CNV enrichment in HIRT (extra-chromosomalcdata). (b) Non-coding genes enrichment in CNVs. (c) Enrichment of H3K4me3 in promoters. (d) Enrichment of promoters in promoter-associated chromatin states.

Chapter 3

Skmer: Assembly-free and Alignment-free Sample Identification using Genome Skims

The ability to inexpensively describe taxonomic diversity is critical in this era of rapid climate and biodiversity changes. The recent genome-skimming approach extends current barcoding practices beyond short markers by applying low-pass sequencing and recovering whole organelle genomes computationally. This approach discards the nuclear DNA, which constitutes the vast majority of the data. In contrast, we suggest using all unassembled reads. We introduce an assembly-free and alignment-free tool, Skmer, to compute genomic distances between the query and reference genome-skims. Skmer shows excellent accuracy in estimating distances and identifying the closest match in reference datasets. Skmer software is publicly available on https://github.com/shahab-sarmashghi/Skmer

3.1 Introduction

The ability to quickly and inexpensively study the taxonomic diversity in an environment is critical in this era of rapid climate and biodiversity changes. The current molecular technique of choice is (meta)barcoding [26, 27, 28]. Traditional (meta)barcoding is based on DNA sequencing of taxonomically informative and group-specific marker genes (e.g., mitochondrial COI [29, 26] and 12S/16S [30, 31] for animals, chloroplast genes like matK for plants [32], and ITS [33] for

fungi) that are variable enough for taxonomic identification, but have flanking regions that are sufficiently conserved to allow for PCR amplification using universal primers. Barcoding is used for taxonomic identification of single-species samples. In the case of metabarcoding, the goal is to deconstruct the taxonomic composition of a mixed sample consisting of multiple species [28]. Beyond the barcoding application, the barcoding marker genes have also been used to delimitate species [34] and to infer phylogenies [35, 36].

The accuracy of (meta)barcoding depends on the coverage of the reference database and the method used to search queries against it [28]. To increase coverage, reference databases with millions of barcodes have been generated (e.g., Barcode of Life Data System, BOLD, for COI [37]). Computational methods for finding the closest match in a reference dataset (e.g., TaxI [38]), and for placement of a query into existing marker trees [39, 40, 41] have been developed. However, the traditional approach to (meta)barcoding, despite its success, has some drawbacks. PCR for marker gene amplification requires relatively high quality DNA and thus cannot be applied to samples in which the DNA is heavily fragmented. Moreover, since barcode markers are relatively short regions, their phylogenetic signal and identification resolution can be limited [42]. For example, in a recent study, 896 out of 4,174 wasp species could not be distinguished from each other using COI barcodes [43].

While low costs have kept PCR-based pipelines attractive, decreasing costs of shotgun sequencing have now made it possible to shotgun sequence 1-2Gb of total DNA per reference specimen sample for as low as $80 [12], even after including sample preparation and labor costs. This has lead researchers to propose an alternate method that uses low-pass sequencing to generate *genome-skims* [13, 12], and subsequently identifies chloroplast or mitochondrial marker genes or assembles the organelle genome. Reconstructing plastid and mtDNA genomes from low-pass shotgun data is possible because organelle DNA tends to be heavily overrepresented in shotgun sequencing data; for example, 10.4% of all reads from the Apocynaceae family of flowering plants were from the chloroplast in one genome-skimming study [13]. Large reference databases based on genome-skimming techniques are under construction by projects such as

PhyloAlps [14], NorBol [15], and DNAmark [16].

Most current applications of genome-skimming to species identification require organelle genome assembly, a task that requires relatively time-consuming manual curation steps to ensure that assembly errors are avoided [17]. This approach discards a vast proportion of the non-target data, reducing the discriminatory power. For these reasons, the DNAmark project [16] is considering alternative methods, where, instead of only relying on organelle markers, one could use the entire set of reads generated in a genome-skim as the identifier of a species. This approach poses an interesting methodological question: can the unassembled data be used to taxonomically profile reference and query samples in a similar manner to conventional barcoding, but using all available genomic information and saving us from the labor-intensive task of mitochondria/plastid genome assembly? In this chapter, we introduce a new assembly-free method to directly use low coverage genome-skims of both reference and query samples. By avoiding the assembly step, our approach also reduces the amount of data processing needed for expanding the reference database.

We treat genome-skims simply as low-coverage "bags of reads", both for a collection of reference species and for query samples. The problem is to find the reference genome-skim that matches the query; if an exact match is not found, we seek the closest available match. A more advanced problem, not directly addressed here, is placing the query in a phylogeny of reference species. An even more difficult challenge, also not addressed here, is decomposing a query genome-skim that contains DNA from several different taxa into its constituent species.

Central to solving these problems is the ability to estimate a *distance* between two genome-skims for low and varied coverage using assembly-free and alignment-free approaches. Alignment-free sequence comparison has been widely studied [44, 45, 46, 47, 48, 49], including for phylogenetic reconstruction [50, 51, 52, 53, 44, 54, 55, 56, 57, 58, 59, 60, 61, 62, 63]. Most existing methods, such as Kr [47], spaced words [63], and kmacs [64] compute evolutionary distances using the length distribution of matched substrings or the count of certain words and thus require assembled genomes to produce accurate results. These methods will not work

with high accuracy when both the query and the reference are a set of reads and not assembled contigs. Other methods, such as andi [60] and FSWM [62], use micro-alignments to compute distances. Even though it may be possible to extend the idea of using micro-alignments to the assembly-free case, both andi and FSWM software currently require assemblies as input. However, several assembly-free methods also exist. Co-phylog [58] makes micro-alignments and calculates distances to reconstruct phylogenetic trees; Mash [65] computes the Jaccard index and an evolutionary distance using the k-mers; Simka [66] computes several distance measures based on the whole k-mer content of reads. However, these methods all assume high enough coverage, ensuring that most of the genome is covered. These levels of coverage are currently not economically feasible for building up large reference databases or for obtaining many query samples. Among existing methods, AAF [52] is the only one that aims to work even at lower coverage. AAF first infers a phylogeny and then corrects its branch lengths to reflect a given estimate of the coverage.

Here, we show that high levels of coverage are not necessary. We focus on a distance measure defined as the proportion of mismatches between the global alignment of two genomes. The mismatch rate, called genomic distance hereafter, is useful for species identification because it reflects the evolutionary divergence between two species. We introduce a new method, Skmer, for accurately computing the genomic distance even from low coverage genome-skims. In extensive test, we show that Skmer dramatically improves estimates of genomic distance based on genome-skims and accurately places genome-skim queries on to a reference collection. This assembly-free approach can therefore be considered a viable complement to currently available DNA barcoding and genome-skimming tools.

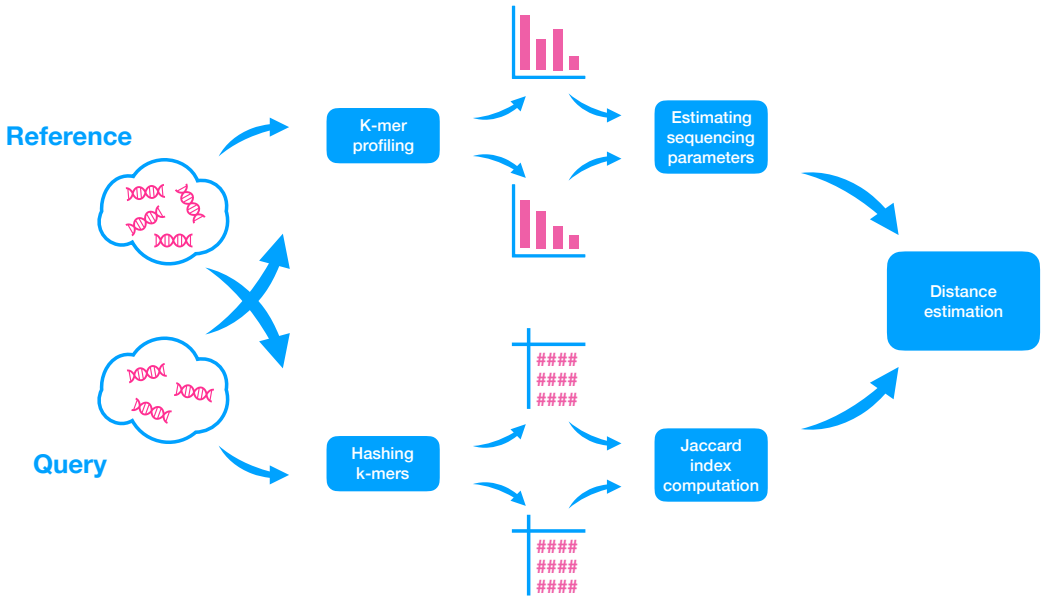


Figure 3.1. Overview of Skmer pipeline. For both query and reference genome-skims, first, the k-mer frequency profiles are used to estimate the sequencing error and coverage (top). Then, the k-mers are hashed, and a subset is retained and used to estimate the Jaccard index between the two genomes (bottom). Finally, the estimated Jaccard index and estimated sequencing coverage and error are used to compute the corrected genomic distance between the query and the reference.

3.2 Results

3.2.1 Skmer

We decomposed reads into fixed length oligomers (denoted *k-mers* with length k), a technique used by many existing alignment-free methods [67, 60]. Recall that the *Jaccard index* J is a similarity measure between any two sets (e.g. k-mer collections) defined as the size of their intersection divided by the size of their union. Ondov *et al.* describe a tool, Mash [65], in which (a) J is estimated efficiently using a hashing procedure; and, (b) J is used to estimate the genomic distance between two genomes. Mash, however, assumes sufficiently high coverage. Unfortunately, J, in addition to the true distance, is impacted by coverage, sequencing error, and genome length. Skmer accounts for the impact of these factors on J.

Skmer has two stages (Fig. 3.1): first we use k-mer frequency profiles (computed using

JellyFish [68]) to estimate the amount of sequencing error and the coverage (neither of which is known) using a novel method. Let M_i be the number of k-mers observed i times in the genome-skim. Let $h = \text{argmax}_{i \geq 2} M_i$. Then, defining $\xi = \frac{M_{h+1}}{M_h}(h+1)$, we derive (see Methods):

$$\lambda = \frac{M_1}{M_h}\frac{\xi^h}{h!}e^{-\xi} + \xi(1-e^{-\xi}) \tag{3.1}$$

$$\varepsilon = 1-(\xi/\lambda)^{1/k} \tag{3.2}$$

where λ and ε are our estimates of the k-mer coverage and the sequencing error rate, respectively.

In stage two, we use the hashing technique of Mash to compute J. Finally, given these estimates, we compute the genomic distance using

$$D = 1 - \left(\frac{2(\zeta_1 L_1 + \zeta_2 L_2)J}{\eta_1 \eta_2 (L_1+L_2)(1+J)}\right)^{1/k} \tag{3.3}$$

where for $i \in \{1,2\}$, $\eta_i = 1-e^{-\lambda_i(1-\varepsilon_i)^k}$ and $\zeta_i = \eta_i + \lambda_i(1-(1-\varepsilon_i)^k)$ (for high coverage, we define ζ_i and η_i differently; see Methods for details), and L_i is the estimated genome length.

We used a series of experiments to study the accuracy of Skmer compared to existing methods with respect to (i) the error in computed distances, and (ii) the ability to find the closest match to a query sequence in a reference dataset of genome-skims, and (iii) phylogenetic inference. We compared the performance against *Mash* and *AAF* [52]. AAF is a method that uses k-mers to estimate phylogenetic distances among a set of at least four sequences. We conclude by comparing Skmer against the results of using COI barcodes from available barcode databases.

3.2.2 Distance accuracy for pairs of genome-skims

We first compare the accuracy of Mash and Skmer in estimating distances between two genome skims. Since AAF outputs a phylogenetic tree and so requires at least four species, we cannot include it in our first set of analyses on pairs of genomes.

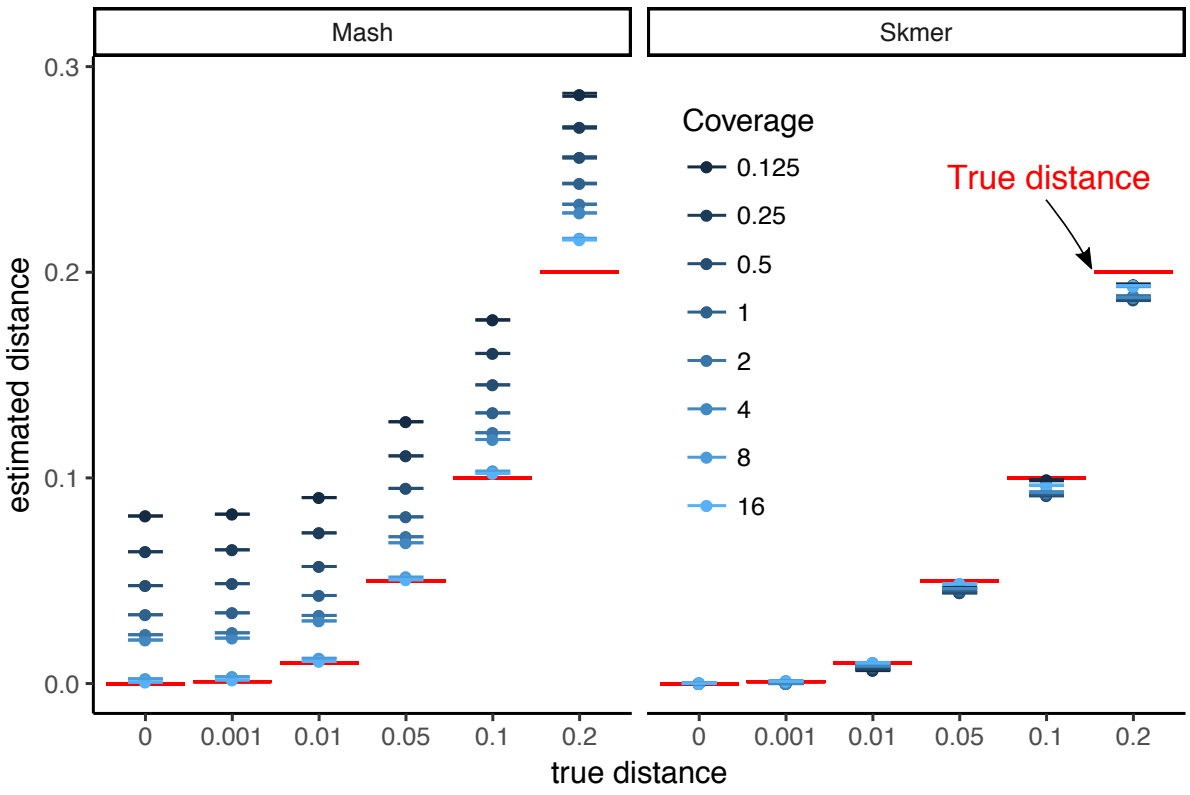


Figure 3.2. Comparing the accuracy of Mash and Skmer on simulated genomes. Genome-skims are simulated using ART with read length $\ell = 100$. Substitutions applied to the assembly of *C. vestalis* at six different rates (x-axis), and genome-skims simulated at varying coverage range from $\frac{1}{8}$X to 16X. The estimated distance (y-axis) by Mash (left) and Skmer (right) is plotted versus the real distances for each coverage level (color). The mean (dots) and standard error (lines) of distances are shown (10 repeats). True distance is shown in red. See Supplementary Fig. B.1 for a scaled representation.

Simulated genomes with controlled distance

Starting from the highly repetitive genome assembly of the wasp species *Cotesia vestalis*, we simulated new genomes with controlled true distance d by randomly adding SNPs, and then we simulated genome-skims by randomly sub-sampling reads and adding error (see Methods). On these simulated genomes, distances are computed with high accuracy by Mash when coverage is high (Fig. 3.2), except where the true distance is also high (i.e., 0.2). However, the accuracy of Mash quickly degrades when the coverage is reduced to 4X or less. In contrast, even when the coverage is reduced to $\frac{1}{8}$X, Skmer has high accuracy. For example, with the true distance set to 0.05, Mash estimates the distance as 0.081 with 1X coverage (an overestimation by 62%)

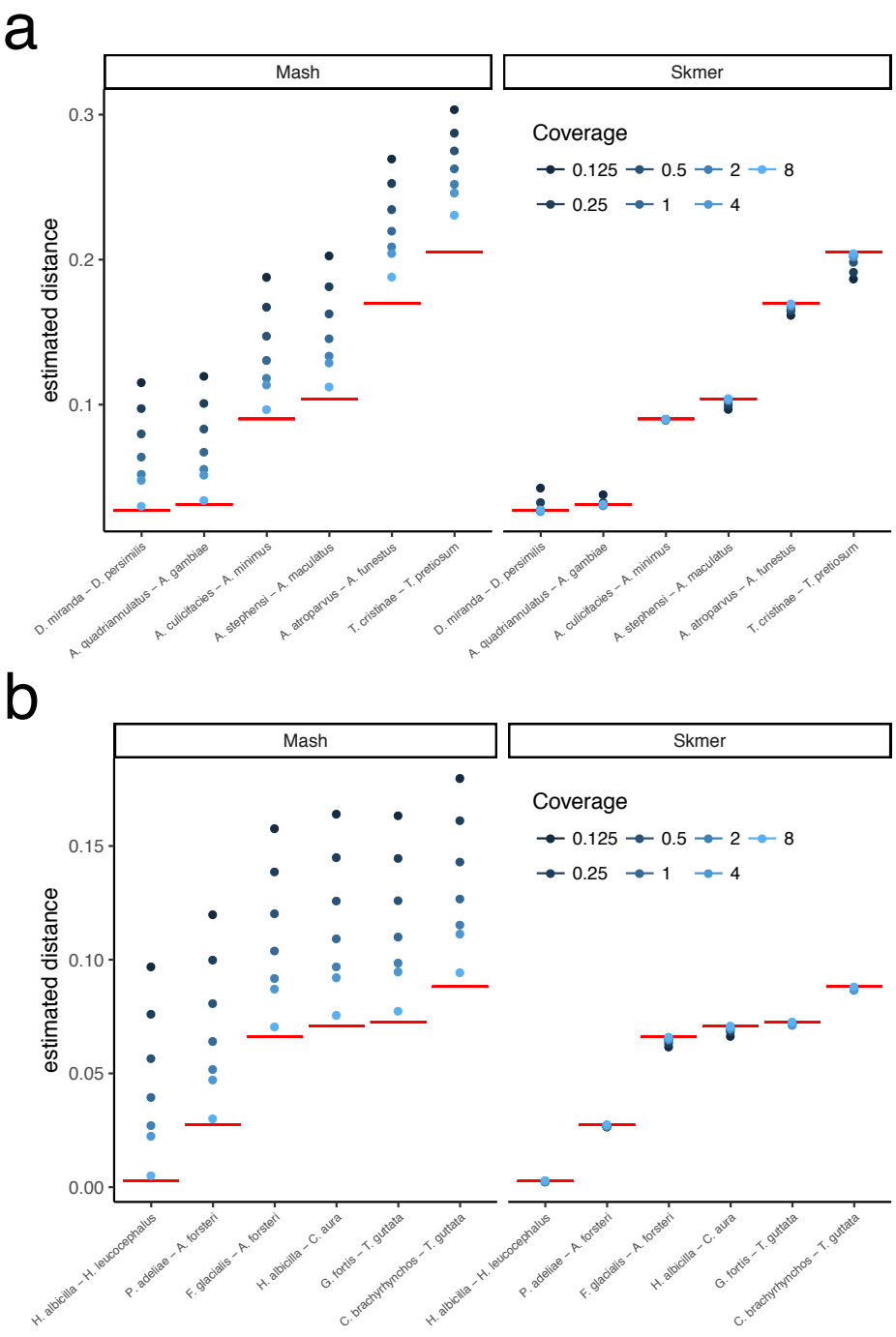


Figure 3.3. Comparing the accuracy of Mash and Skmer on pairs of insects (a) and birds (b) genomes. Genome-skims are simulated at coverage $\frac{1}{8}$X to 8X (shades of blue). The estimated distance (y-axis) is plotted for Mash (left) and Skmer (right) for each pair of species (x-axis). The results of Mash* run on assemblies, which is taken as the ground truth, is shown in red. Mash overestimates at lower coverages. Skmer estimates are closer to the ground truth and are less sensitive to the coverage. See also Supplementary Fig. B.5.

while Skmer corrects the distance to 0.045 (an underestimation by 10%). Note that applying Mash* (Mash without the unnecessary approximation $(1-D)^k \approx e^{-kD}$ used by default in Mash) to the complete assemblies generally generates very accurate results, as expected, but even given the full assembly, Mash* still has a small but noticeable error when $d = 0.2$. Note that results are extremely consistent across our ten different runs of subsampling (Fig. 3.2). We repeated the simulation with a lower range of coverage ($\frac{1}{64}$X to 1X). Interestingly, even with very low coverage, the absolute distance error is small in many cases (Supplementary Fig. B.2); however, for $d \geq 0.1$, Skmer estimates start to degrade below $\frac{1}{8}$X coverage.

Repeating the process with the *Drosophila melanogaster* genome as the base genome also produces similar results (Supplementary Fig. B.3). The only condition where Skmer has an absolute error larger than 0.01 is with coverage below 1X and $d = 0.2$ (Fig. 3.2). However, we note that for $d = 0.001$, the relative error is not small with low coverage (Supplementary Fig. B.4b) indicating that distinguishing very small distances (perhaps below species-level) requires high coverage. Estimating the right order of magnitude when the true distance is 0.001 seems to require 2X coverage (preferably 8x) while 1X coverage is sufficient to distinguish distances at or above 0.01 (Supplementary Fig. B.4).

Pairs of insect and bird genomes

We now test methods on several pairs of insect and avian genomes, subsampled to create genome-skims. Note that unlike the simulated datasets, here, genomes can undergo all types of genetic variations and complex rearrangements, and thus, do not have the same length. We carefully selected several pairs of genomes to cover a wide range of mutation distance and genome length. Here, the true genomic distance is not known, but we use the distance estimated by Mash* on the full assemblies as the true distance d. For all pairs of insect and avian genomes (Fig. 3.3), Mash has high error for coverage below 8X while Skmer successfully corrects the estimated distance and obtains values extremely close to the results of running Mash* on the full assembly. For example, the distance between *A. stephensi* with length ∼196Mbp and *A.*

maculatus with length ∼132Mbp is estimated to be 0.104 based on the full assembly and 0.102 (2% underestimation) with only $\frac{1}{2}$X coverage using Skmer, while Mash would estimate the distance to be 0.163 (∼57% overestimation).

3.2.3 Distance accuracy for all pairs genome-skims

We now turn to datasets with sets of genome-skims, evaluating the accuracy of all pairs of distances. Here, since we have at least four sequences in each test, in addition to Mash, we also compare our results with AAF.

Fixed sequencing effort

So far, our experiments have controlled for the coverage by subsampling varying amount of sequence data, proportional to the genome length. In our genome-skimming application, coverage will not be fixed. Often, the amount of sequence data obtained for each species will be relatively similar. As a result, genomes of different length end up being sequenced with different coverage depth proportional to the inverse of their length. We therefore performed a study where all species are subsampled to produce 100Mb of sequence data in total resulting in varying levels of coverage (based on the genome length, Supplementary Table B.5). The error in the distance estimated by Mash relative to the ground truth can be quite large (higher than 300% in the worst case) while Skmer consistently makes accurate estimates close to the true distance even at the lowest amount of coverage (Fig. 3.4, Figs. 3.5, and Supplementary Table B.6). Repeating the analysis with 0.5Gb or 1Gb total sequence data produced similar patterns, but as expected, increasing the sequencing effort reduces the error for all methods (Supplementary Figs. B.6–B.8).

Before error correction, AAF has error levels that are comparable to Mash (Figs. 3.4b, Fig. 3.5b). The correction applied by AAF, similar to Skmer, reduces the negative impact of low coverage but not to the same extent. Thus, Skmer has less error compared to corrected AAF (with 100Mb sequence and across all datasets, the mean error of Skmer is 3.13% and AAF-corrected is

Table 3.1. Tree error. For each method, we show normalized weighted RF distance (%) of trees inferred from genome-skim distances to trees inferred from full assembly distances. Boldface: the lowest error.

Dataset	Sequencing effort	Mash	Skmer	AAF (uncorrected)	AAF (corrected)
Anopheles	0.1G	23.19%	**1.07**%	19.92%	6.36%
	0.5G	12.84%	**0.45**%	9.74%	4.9%
	1G	8.92%	**0.37**%	9.59%	3.3%
	Mixed	14.75%	**0.58**%	8.46%	8.45%
Drosophila	0.1G	23.87%	**2.05**%	20.29%	5.85%
	0.5G	13.33%	**0.72**%	10.37%	5.25%
	1G	7.11%	**0.58**%	10.84%	2.2%
	Mixed	16.58%	**1.11**%	11.36%	10.87%
Birds	0.1G	37.03%	**5.64**%	31.81%	21.13%
	0.5G	25.16%	**1.91**%	20.8%	6.86%
	1G	19.42%	1.19%	15.54%	**1.05**%
	Mixed	28.14%	**3.08**%	18.15%	7.57%

22.7%). For example, in the *Drosophila* dataset, the worst-case error of AAF between any two pairs of genome-skims is 31%, whereas the error never exceeds 8% for Skmer. Note that when computing the error of AAF, we use the result of running AAF on full assemblies as the ground truth.

To quantify the impact of distance estimates on downstream analyses, we used FastME [69] to infer phylogenetic trees using distances computed by Mash and Skmer on genome skims and with correction using the JC69 model [70]. AAF by default generates trees as part of its output. We compare these trees to those computed by Mash/AAF run on the full assemblies (taken as the ground truth) using the weighted Roubinson-Foulds (WRF) distance [71] (Table 3.1). WRF is the sum of branch length differences between the two trees (using zero length for missing branches), and we normalized WRF by the sum of branch lengths of both trees. In all three datasets, Skmer distances lead to trees with lower WRF distance to the ground truth compared to Mash and AAF/uncorrected. AAF correction reduces WRF compared to uncorrected AAF; however, Skmer trees have two to 14 times less error compared to the corrected AAF, except in

one case where AAF/corrected has 1.05% error and Skmer has 1.19% (Table 3.1). Increasing the size of skims to 0.5Gb and 1Gb helps all methods to produce more accurate trees.

Heterogeneous sequencing effort

In addition to changes in the genomic length, the sequencing effort per species may also vary across sequencing protocols, experiments and research labs, and so a database of reference genome-skims may consist of samples with heterogeneous sequencing efforts. To capture this, for each species, we choose its total sequencing effort from three possible values 0.1Gb, 0.5Gb, and 1Gb, uniformly at random, and estimate all pairs of distances within each dataset as before (Fig. 3.6 and Supplementary Fig. B.9). Similar to the case of fixed sequencing effort, Skmer mitigates large relative error in the distances estimated by Mash and produces more accurate results than both Mash and AAF, (Table 3.2, Fig. 3.6, and Supplementary Fig. B.9). For example, comparing to the case of fixed 100Mb genome-skims of the *Drosophila* dataset, the worst-case error of AAF is increased to 70%, while using Skmer it remains almost the same (8%). Comparing trees inferred from distances estimated by various methods also confirms the higher accuracy of Skmer (Table 3.1). For instance, on the Anopheles dataset, Skmer has only 0.58% WRF distance to the reference tree whereas Mash and AAF-corrected trees have 14.75% and 8.45% WRF distance.

Table 3.2. Comparing the average error of Mash, Skmer, and AAF in estimating distances over three datasets with heterogeneous sequencing effort.

Dataset	Mash	Skmer	AAF (uncorrected)	AAF (corrected)
Anopheles	28.72% (1.10%)	**0.84**% (0.03%)	13.48% (0.56%)	11.36% (0.44%)
Drosophila	29.05% (0.59%)	**0.84**% (0.04%)	15.25% (0.38%)	10.94% (0.33%)
Birds	64.29% (0.54%)	**2.21**% (0.04%)	36.02% (0.29%)	5.28% (0.16%)

* The standard error of the mean is provided in parentheses.

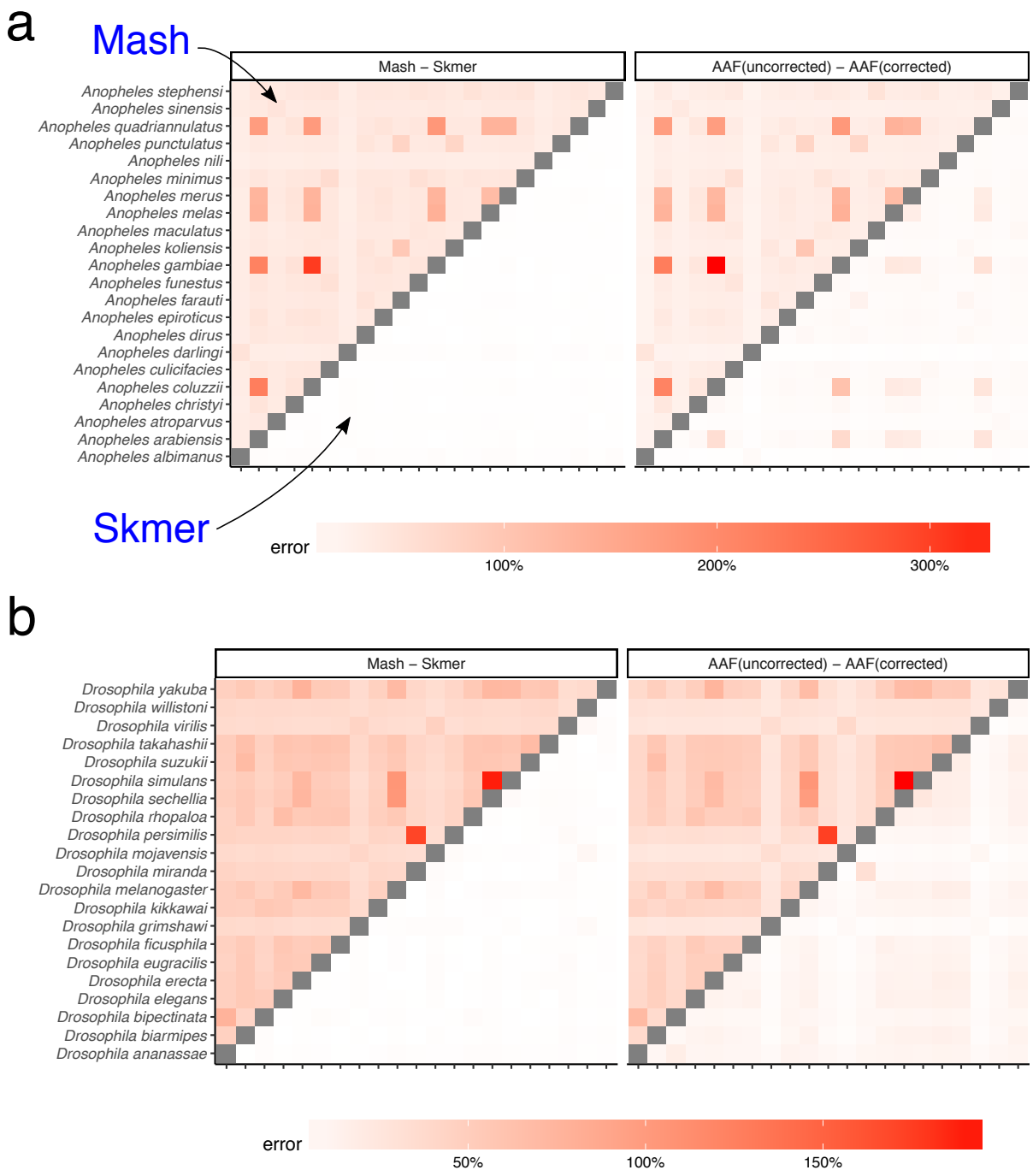


Figure 3.4. Distance error with fixed 100Mb sequence per genome for (a) 22 Anopheles, (b) 21 Drosophila. Each genome is skimmed with 100Mb sequence and distances are computed using Mash, Skmer, and AAF. True distance used in calculating the error is computed by applying each method (AAF and Mash) to the full genome assemblies. The heatmaps on the left show the error of Mash (upper triangle) and Skmer (lower triangle), and the heatmaps on the right are for AAF before correction (upper) and after correction (lower).

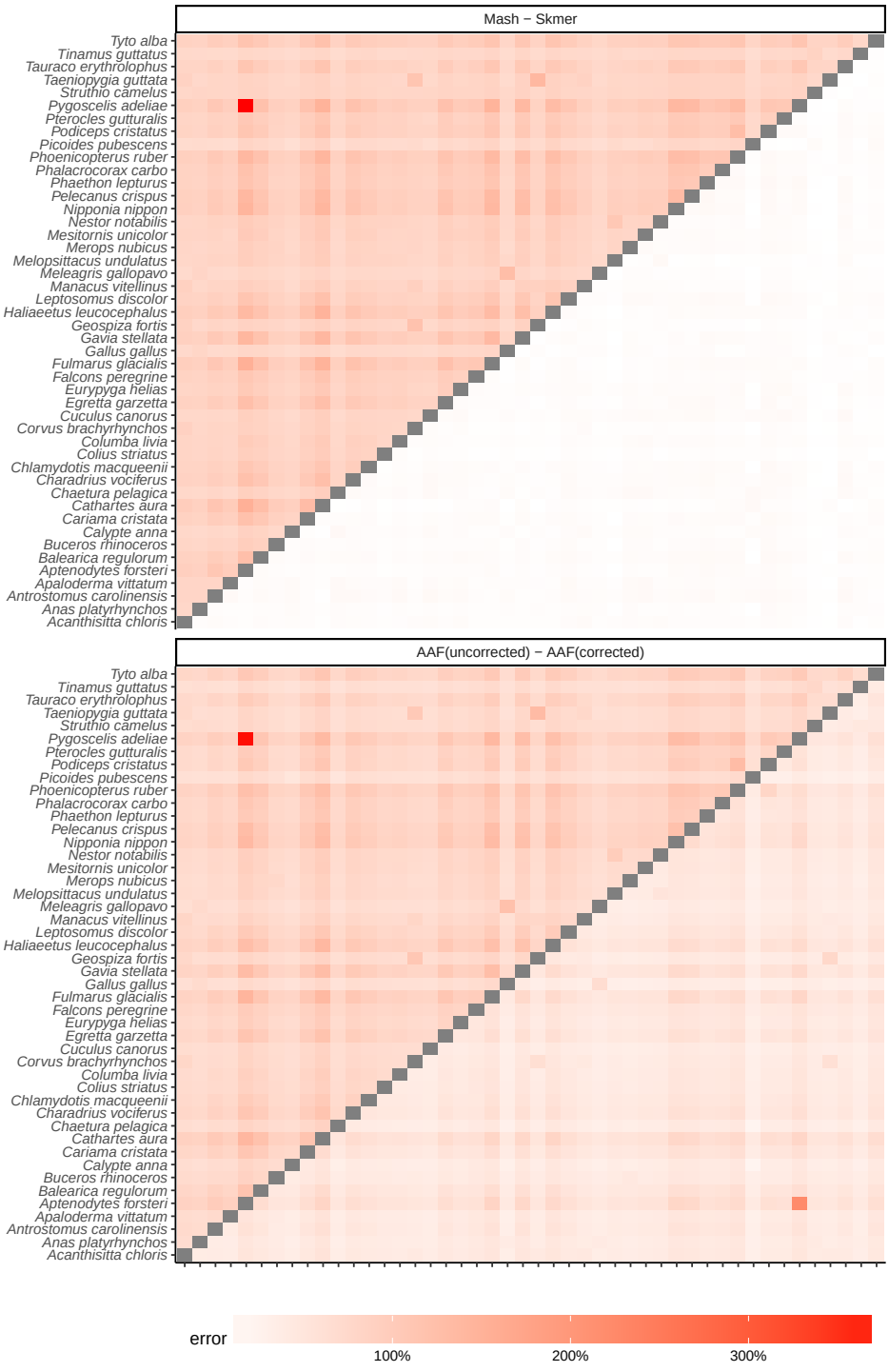


Figure 3.5. Distance error with fixed 100Mb sequence per genome for the avian dataset. The errors of Mash and AAF for the two eagle species (*H. albicilla* and *H. leucocephalus*) were extremely large (Mash: $\approx 4000\%$, AAF $> 3000\%$ error), dominating the color spectrum; we excluded *H. albicilla* to help readability; for the eagles, Skmer's estimate is 0.00244 ($\sim 9\%$ error).

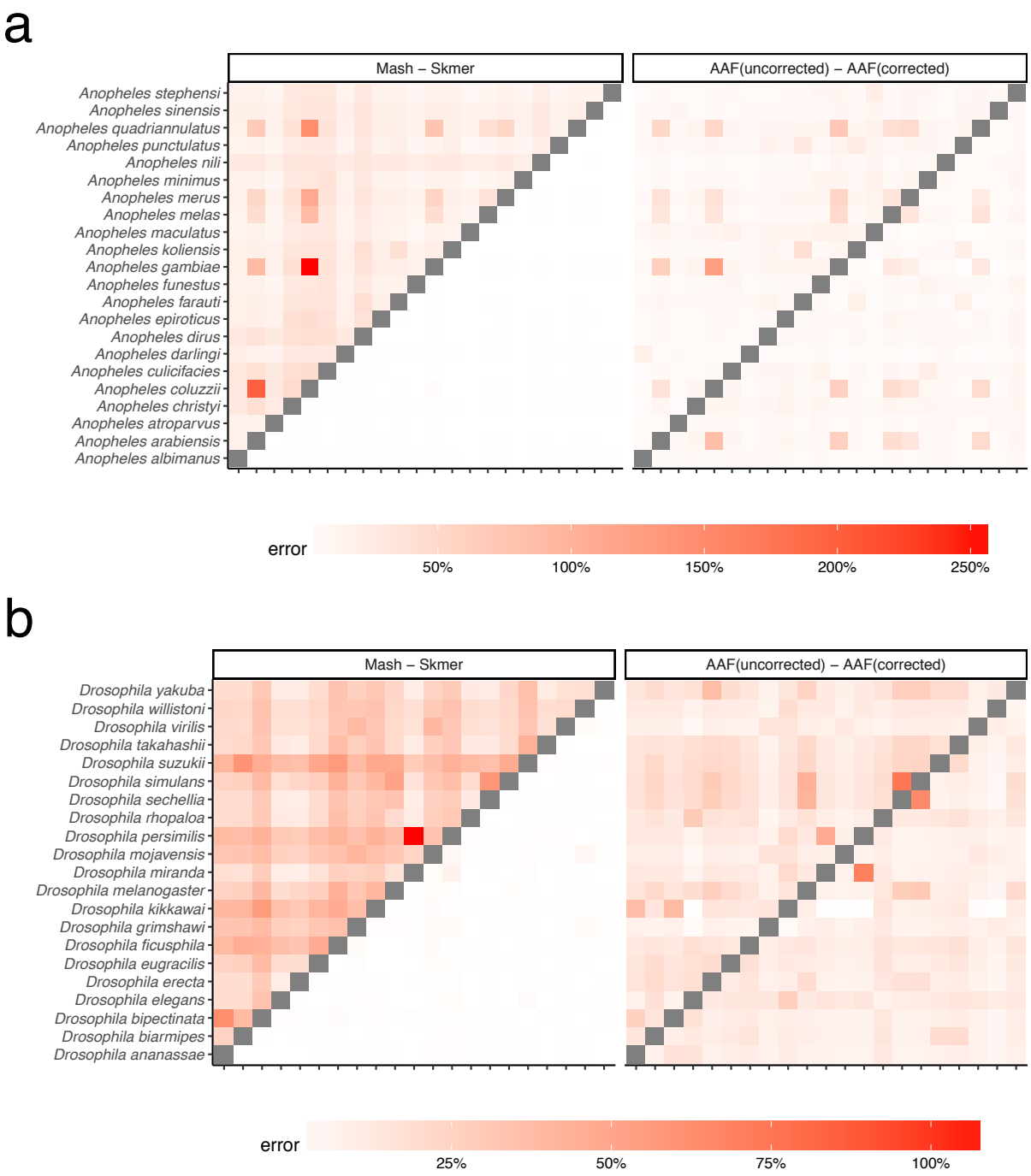


Figure 3.6. Distance error with heterogeneous sequencing effort for (a) Anopheles and (b) Drosophila. Species have random amount of sequence chosen uniformly among 0.1Gb, 0.5Gb, and 1Gb. See Supplementary Fig. B.9 for birds.

Figure 3.7. Comparing the error of Mash and Skmer on a dataset of 14 Drosophila genome-skims. Each SRA is subsampled to 100Mb and then filtered to remove contamination. True distances are computed from the assemblies.

3.2.4 Genome skims from real reads

So far, all of our tests used simulated reads. When analyzing real genome skims, there are additional complications such as extraneous DNA (real or artifactual) and the over representation of organelle genome. We next tested Skmer using real reads. We created 100Mb skims of 14 Drosophila genomes by subsampling short-read data produced in a recent Drosophila genome assembly study [72]. Before running Skmer or Mash, we filtered reads that (even partially) aligned to 12 Drosophila-associated microbial genomes as reported in previous studies [73, 74, 75] (see Supplementary Table B.1), to the human genome, or to the mitochondrial

genome of respective Drosophila species. We then estimated all pairs of distances as before and computed the error relative to the distances computed from the assemblies (Fig 3.7). Consistent with the results we obtained on the simulated skims, Skmer has less error compared to Mash. The average error of Mash on this dataset is 43.48% ($\pm$ 2.29%) with maximum error of 217%. Skmer, on the other hand, has an average error of 4.21% ($\pm$ 0.35%) and its maximum error is 22.2%.

Running time

Skmer and Mash have comparable running time, while AAF is much slower. In the experiment with heterogeneous sequencing effort, the total running time (using 24 CPU cores) to compute distances based on genome-skims for all $\binom{47}{2}$ pairs of birds using Mash, Skmer, and AAF was roughly 8, 33, and 460 minutes, respectively.

3.2.5 Leave-out search against a reference database of genome-skims

We now study the effectiveness of using genomic distance to search a database of genome-skims to find the closest match to a query genome-skim. Given a query genome-skim and a reference dataset of genomes, we can order the reference genomes based on their distance to the query. The results can be provided to the user as a ranking. When the query genome is available in the reference dataset, finding the match is relatively easy. To study the effectiveness of the search as the distance of the closest available match increases, we use a leave-out experiment, as described in Methods. Figure 3.8 shows the mean rank error as well as the mean distance error of the best remaining match in a leave-out experiment when removing genomes closer than d for $0.01 \leq d \leq 0.1$. A rank error (or distance error) equal to zero corresponds to a perfect match to the best available genome.

On all three datasets, Skmer consistently and often substantially outperforms Mash and AAF in terms of finding the best remaining match, except the *Drosophila* dataset where Mash and Skmer have comparable rank error, while both are better than AAF (Fig 3.8). Even in that

case, on average, the distance of the best match found by Skmer is closer to the distance of the true best match compared to the best hit found by Mash. Moreover, the mean rank error of Skmer is smaller than Mash (Supplementary Fig. B.10) if we exclude only one species *Drosophila willistoni* (which is at distance $0.1565 \leq d \leq 0.1622$ from other species). It is also notable that over the avian dataset, Skmer has mean rank error less than 0.5 for all range of distances, while Mash and AAF can be off by more than 2.5 on average. These results demonstrate that correcting the distance not only impacts our understanding of the absolute distance, but also, impacts results of searching a reference library.

3.2.6 Phylogeny reconstruction and comparison to organelle markers

As the last experiment, we estimated phylogenetic trees for *Anopheles* and *Drosophila* datasets after transforming the genomic distances estimated by Skmer to Jukes-Cantor (JC) distances [70]. For each dataset, we also built another tree based on available COI barcodes, using an identical method. We compare the results against a reference tree obtained from Open Tree of Life [76]. We restricted the results to species for which COI barcodes were available (Fig. 3.9ab).

For the *Anopheles* species, Skmer distances produce a tree that is almost identical to the reference tree (with only one branch difference out of nine), while COI tree differs from the reference in seven branches. Similarly, for the *Drosophila* species, Skmer differs from the reference in three branches (with small local changes) out of 13 total branches in the reference tree, whereas COI tree is very inconsistent with the reference tree (seven branches are different). We also built maximum-likelihood trees from COI barcodes (Supplementary Fig. B.11), but the number of incorrect branches did not reduce. Comparing the distribution of all pairwise genomic distances obtained from genome-skims and barcodes (Fig. 3.9c), Skmer has larger distances and fewer pairs with zero or close to zero distance, indicating that Skmer has a higher resolution in differentiating between samples. For example, four species of the *Anopheles* genus *A. coluzzii*, *A. gambiae*, *A. arabiensis*, and *A. melas* have very small pairwise distances based on COI barcodes,

while using Skmer, the estimated distances are in the range 0.02–0.04 for these species.

3.3 Methods

Consider an idealized model where two genomes are the outcome of a random process that copies a genome and introduces mutations at each position with fixed probability d. Moreover, substitutions are the only allowed mutation. In this case, the per-nucleotide hamming distance D between the two genomes is a random variable (r.v.) with expected value d. We would like to estimate d. While this is a simplified model, we will test the method on real pairs of genomes that differ due to complex mutational processes (also, see Appendix B.2 for extensions). We start with known results connecting the Jaccard index and the hamming distance and then show how these results can be generalized to low coverage genome-skims. Throughout, we present our results succinctly and present derivations and more careful justifications in Appendix B.1 of the supplementary material.

3.3.1 Jaccard index versus genomic distance

The Jaccard index of subsets A_1 and A_2 is defined as

$$J = \frac{|A_1 \cap A_2|}{|A_1 \cup A_2|} = \frac{|A_1 \cap A_2|}{|A_1| + |A_2| - |A_1 \cap A_2|} \,. \tag{3.4}$$

Let W be the number of shared k-mers between the two genomes. Note that: $J = \frac{W}{2L-W} \Rightarrow \frac{2J}{1+J} = \frac{W}{L}$, where L is the genome length. Assuming random genomes and no repeats, perhaps justifiably [77], the probability that a changed k-mer exists elsewhere in the genome is vanishingly small for sufficiently large k. Thus, we assume a k-mer is in the shared k-mers set only if no mutation falls on it, an event that has probability $(1-d)^k$. Thus, we can model W as a binomial with probability $(1-d)^k$ and L trials. As Ondov *et al.* [65] pointed out, we can estimate

$$D = 1 - \left(\frac{2J}{J+1}\right)^{\frac{1}{k}} \tag{3.5}$$

and they further approximate D as $\frac{1}{k}\ln\left(\frac{J+1}{2J}\right)$. To be able to estimate large distances, we avoid the unnecessary approximation and use Equation 3.5 directly. We skim each genome to obtain k-mer sets A_1, A_2 and estimate J using Equation 3.4, which can be computed efficiently using a hashing technique used by Mash [65]. Note that, however, Equation 3.5 assumes a high coverage of the genome so that each k-mer is sampled at least once with very high probability. This assumption is violated for genome-skims in consequential ways. As a simple example, suppose the coverage is low enough that a k-mer is sampled with probability 0.5. Then, even for identical genomes, we estimate J as $\frac{1}{3}$, resulting in a distance estimate of $D \approx 0.032$ for $k = 21$.

3.3.2 Extending to genome-skims with known low coverage and error

We now show how Equation 3.5 can be refined to handle genome-skims despite low and uneven coverage, sequencing error, and varying genome-lengths. We first assume that coverage and error are known and later show how to compute these.

Low coverage

When the genome is not fully covered, three sources of randomness are at work: mutations and sampling of k-mers from each of the two genomes. Each genome of length L is sequenced independently using randomly distributed short reads of length ℓ at coverages c_1 and c_2 to produce two genome-skims. Under the simplifying assumption that genomes are not repetitive, we choose k to be large enough so that each k-mer is unique with high probability. Therefore, the number of distinct k-mers in each genome is $L - k \simeq L$. The probability of covering each k-mer can be approximated as $\eta_i = 1 - e^{-\lambda_i}$ where $\lambda_i = c_i(1 - k/\ell)$. Modeling the sampling of k-mers as independent Bernoulli trials, $|A_i|$ becomes binomially distributed with parameters η_i and L. By independence, $W = |A_1 \cap A_2|$ also becomes binomially distributed with parameters $\eta_1\eta_2(1-d)^k$ and L. Moreover, $U = |A_1 \cup A_2|$ can also be modeled approximately as a Gaussian with mean $(\eta_1 + \eta_2 - \eta_1\eta_2(1-d)^k)L$. Treating η_1 and η_2 as known and dividing $\frac{W}{L}$ by $\frac{U}{L}$ gives us:

$$J = \frac{W}{U} = \frac{\eta_1 \eta_2 (1-D)^k}{\eta_1 + \eta_2 - \eta_1 \eta_2 (1-D)^k} ;$$

thus,

$$D = 1 - \Big(\frac{(\eta_1 + \eta_2)}{\eta_1 \eta_2} \frac{J}{(1+J)} \Big)^{\frac{1}{k}} .$$

Sequencing error

Each error reduces the number of shared k-mers and increases the total number of observed k-mers, and thus can also change the Jaccard index. Let ε_i denote the base-miscall rate for genome skim i. For large k and small ε_i, the probability that an erroneous k-mer produces a non-novel k-mer is negligible. The probability that a k-mers is covered by at least one read, without any error, is approximately

$$\eta_i = 1 - e^{-\lambda_i (1-\varepsilon_i)^k} . \tag{3.6}$$

Adding up the number of error-free and erroneous k-mers, the total number of k-mers observed from both genomes can again be approximately modeled as a Gaussian with mean $\zeta_i L$ for

$$\zeta_i = \eta_i + \lambda_i (1 - (1-\varepsilon_i)^k) . \tag{3.7}$$

Just as before, we can simply estimate D by solving for it in

$$J = \frac{\eta_1 \eta_2 (1-D)^k}{\zeta_1 + \zeta_2 - \eta_1 \eta_2 (1-D)^k} . \tag{3.8}$$

When the coverage is sufficiently high, each k-mer will be covered by multiple reads with high probability, and low-abundance k-mers can be safely considered as erroneous. Mash has an option to filter out k-mers with abundances less than some threshold m to remove k-mers

that are likely to be erroneous. In this case,

$$\zeta_i = \eta_i = 1 - \sum_{t=0}^{m_i-1} \frac{(\lambda_i(1-\varepsilon_i)^k)^t}{t!} e^{-\lambda_i(1-\varepsilon_i)^k} \tag{3.9}$$

assuming all erroneous k-mers are removed. For instance, filtering single-copy k-mers (i.e., $m = 2$) gives us:

$$\zeta_i = \eta_i = 1 - e^{-\lambda_i(1-\varepsilon_i)^k} - \lambda_i(1-\varepsilon_i)^k e^{-\lambda_i(1-\varepsilon_i)^k}$$

and the Jaccard index follows the same equation as (3.8). Since this filtering approach only works for high coverage, we filter low coverage k-mers only when our estimated coverage is higher than a threshold (described below). Note that the genome-skims compared may use different filtering schemes yet Eqn. 3.8 holds regardless.

Differing genome lengths

Based on a model where the genomic distance between genomes of different lengths is defined to be confined to the mutations that are falling on homologous sequences, we can drive

$$J = \frac{\eta_1\eta_2 \min(L_1,L_2)(1-D)^k}{\zeta_1 L_1 + \zeta_2 L_2 - \eta_1\eta_2 \min(L_1,L_2)(1-D)^k} .$$

This computation does not penalize for genome length difference. While a rigorous modeling of evolutionary distance for genomes of different length require sophisticated models of gene gain, duplication, and loss, we take the heuristic approach used by Ondov *et al.* [65] and simply replace $\min(L_1,L_2)$ with $(L_1+L_2)/2$. This ensures that the estimated distance increases as genome lengths becomes successively more different. This leads us to our final estimate of distance given by:

$$D = 1 - \left(\frac{2(\zeta_1 L_1 + \zeta_2 L_2)J}{\eta_1\eta_2(L_1+L_2)(1+J)} \right)^{1/k} \tag{3.10}$$

3.3.3 Estimating sequencing coverage and error rate

So far we have assumed a perfect knowledge of sequencing depth and error. However, for genome-skims, the genome length is not known; thus, we need to estimate the coverage in order to apply our distance correction. We also assume a constant base error rate, and co-estimate it with the coverage.

The sequencing depth, which is the average number of reads covering a position in the genome, can be estimated from the k-mer coverage profiles. The probability distribution of the number of reads covering a k-mer is a Poisson r.v. with mean λ, where λ is defined as k-mer coverage. As we look into the histogram data, it is easier to work with counts instead of probabilities. Let M denote the total number of k-mers of length k in the genome, and M_i count the number of k-mers covered by i reads. Thus, for $i \geq 0$, $\mathbb{E}[M_i] = M\frac{\lambda^i}{i!}e^{-\lambda}$. For a given set of reads, we can count the number of times that each k-mer is seen, and assuming zero sequencing error, it equals the number of reads covering that k-mer. Then, we can aggregate the number of k-mers covered by i reads and find M_i for $i \geq 1$. However, since in a genome-skim, large parts of the genome may not be covered, both M and M_0 are unknown. To deal with this issue, we could take the ratio of consecutive counts to get a series of estimates of λ as $\tilde{\lambda}_i = \frac{M_{i+1}}{M_i}(i+1)$ for $i = 1, 2, \ldots$. In practice, sequencing errors change the frequency of k-mers and has to be considered when estimating the coverage. Assuming that the error is introduced at a constant rate along the reads, we can use the information in the k-mer counts to co-estimate ε and λ. Like before, we assume that the k-mer length k is large enough that any error will introduce a novel k-mer, so the count of all erroneous k-mers is added to the count of single-copy k-mers. Moreover, for k-mers with more than one copy, the number of times that each kmer is seen equals the number of reads covering that k-mer without any error. Formally, let $\hat{M}_i$ denote the count of k-mers seen i times in the presence of error, and $\rho = (1-\varepsilon)^k$ denote the probability of error-free

k-mer.

$$\mathbb{E}[\hat{M}_i] = \begin{cases} \sum_{j \geq i} M \frac{\lambda^j}{j!} e^{-\lambda} \binom{j}{i} \rho^i (1-\rho)^{j-i} & i \geq 2 \\ \sum_{j \geq 1} M \frac{\lambda^j}{j!} e^{-\lambda} \left(j\rho(1-\rho)^{j-1} + j(1-\rho) \right) & i = 1 \end{cases} = \begin{cases} M \frac{\xi^i}{i!} e^{-\xi} & i \geq 2 \\ M \left(\xi e^{-\xi} + \lambda - \xi \right) & i = 1 \end{cases} \tag{3.11}$$

where $\xi = \lambda\rho$ is the average number of error-free reads covering a k-mer. A family of estimates for ξ is obtained by taking the ratio of consecutive counts of error-free k-mers as $\tilde{\xi}_i = \frac{\hat{M}_{i+1}}{\hat{M}_i}(i+1)$ for $i \geq 2$. Then, using an estimate of ξ and the count of single-copy k-mers, we get a series of estimates of λ for $i \geq 2$ as

$$\tilde{\lambda}_i = \frac{\hat{M}_1}{\hat{M}_i} \frac{\tilde{\xi}^i}{i!} e^{-\tilde{\xi}} + \tilde{\xi}(1 - e^{-\tilde{\xi}}) \,. \tag{3.12}$$

Moreover, we can estimate the error rate from the estimates of λ and ξ as

$$\tilde{\varepsilon} = 1 - (\tilde{\xi}/\tilde{\lambda})^{1/k} \,. \tag{3.13}$$

While any of these $\tilde{\xi}_i$ and $\tilde{\lambda}_i$ can be used in principle, the empirical performance can be affected by the choice; in our tool, we use heuristic rules (described below) that seek to use large M_i values.

3.3.4 Skmer: implementation

Skmer takes as input two or more genome-skims. It uses JellyFish [68] to compute M_i values, which are then used in estimating λ and ε based on Equations 3.12 and 3.13, by setting $\tilde{\xi} = \tilde{\xi}_h$ and $\tilde{\lambda} = \tilde{\lambda}_h$, where $h = \text{argmax}_{i \geq 2} M_i$. Then, Mash is used to estimate the Jaccard index, with $k = 31$ (selected empirically; Supplementary Fig. B.14) and sketch size 10^7. Finally, we use Equation 3.10 to compute the hamming distance with η and ζ values computed using Equations 3.6, 3.7 if $c < 5$ or else using Equation 3.9. The genome length L is estimated as the

total sequence length divided by the coverage c.

3.3.5 Experimental setup

Method settings

For Skmer, we use default parameters described above. For Mash, similar to Skmer, we used $k = 31$ (selected empirically; Supplementary Fig. B.14) and sketch size 10^7. As Mash handles errors by removing low copy k-mers, we set the minimum cardinality for k-mers to be included as $\lfloor \frac{c}{5} \rfloor + 1$ with our estimate of c.

AFF has an algorithm to correct hamming distances for low coverage, but the correction relies on adjusting the length of tip branches in a distance-based inferred phylogeny. As such, it cannot run on a pair of genomes and requires at least four genomes. Also, AAF leaves coverage estimation to the user with some guidelines, which we fully follow (Appendix B.3).

For building phylogenetic trees, we transformed Skmer distances using the JC69 [70] model and used FastME [69] to construct the distance-based trees via BIONJ [78] method.

Genomic Datasets

We used an assembly of *Cotesia vestalis* [79] (GenBank accession: GCA_000956155.1) as well as three sets of publicly available assembled genomes (Supplementary Tables B.2–B.4) and used ART [80] to simulate genome-skims of read length $\ell = 100$ with default sequencing error profile, controlling for the sequencing depth (coverage) (Appendix B.3). Specifically, the data included 21 *Drosophila* genomes (flies) and 22 genomes from the *Anopheles* genus (mosquitoes) obtained from InsectBase[81], and 47 avian species from the Avian Phylogenomic Project [82, 83].

For the experiment on real genome skims, high-coverage SRA's of 14 *Drosophila* species were obtained from NCBI database under project number PRJNA427774 [84] and then subsampled to 100Mb. Assemblies used to compute true distances for these 14 *Drosophila* species were obtained from the Drosophila project [85]. We used the tool fastp [86] for filtering low-quality

reads and adapter removal. We also used Megablast [87] to search against a database of bacterial and mitochondrial genomes and remove contaminant reads. We used Bowtie2 [88] with the highest sensitivity to remove the reads aligning (even partially) to the human reference genome.

To simulate genomes with controlled genomic distance, we introduced random mutations. As a challenging case, we took the highly repetitive assembly of the wasp species *Cotesia vestalis*, and mutated it artificially; we only applied single nucleotide mutations distributed uniformly at random across the genome. We repeated the study on the simpler case of the fly species *D. melanogaster*. We generate genome-skims using ART with $\ell = 100$, default error profile of Illumina sequencer, and varying coverage between $\frac{1}{64}$X and 16X. For simulated genomes, we repeated the subsampling 10 times and reported the mean and standard error.

In order to compare with DNA barcoding method, we downloaded available COI barcodes for the *Drosophila* and *Anopheles* species in BOLD database [37]. Out of 21 *Drosophila* and 22 *Anopheles* species in our dataset, 16 *Drosophila* and 19 *Anopheles* species had one or more barcodes in BOLD. For each species, we selected a barcode, and using MUSCLE [89], aligned all barcodes within each dataset and constructed the phylogenetic tree assuming the Jukes-Cantor model. Under the same model of substitution, we transformed Skmer distances and built the Skmer tree. We used FastME [69] to construct the distance-based trees via BIONJ [78] method. The maximum-likelihood COI trees were built using PhyML [90].

Evaluation Metrics

For simulated data, the true distance is controlled and is thus known. For biological datasets, the ground truth is unknown. Instead, we use the distance measured on the full assembly by each method as its ground truth; thus, the ground truth for AAF is computed using AAF. We show both absolute error and the relative error, measured as $|\frac{\hat{d}-d}{d}|$ where d and $\hat{d}$ are the true and the estimated distances.

Leave-out

We used a leave-out strategy to study the accuracy of searching for a query genome in a reference set. For a query genome G_q in a set of n genomes $\{G_1 \dots G_n\}$, we ordered all genomes based on their distances to G_q calculated using the full assemblies, which represents the ground truth; let $G_q^1 \dots G_q^n$ denote the order, and $d_q^1 \dots d_q^n$ be the respective distances from the query (note $G_q^1 = G_q$ and $d_q^1 = 0$). For $0.01 \leq d \leq 0.10$, we removed genomes $1 \dots i$ from the datasets where i is the largest value such that $d_q^i \leq d$, leaving us with $G_q^{i+1} \dots G_q^n$. We then ordered the remaining genomes by each method; let $x_1 \dots x_{n-i}$ be the order obtained by a method and let r be the the rank of the best remaining genome according to the ground truth in the estimated order (i.e., $x_1 = G_q^{i+r}$). Since $r = 1$ implies perfect performance, and $r > 1$ indicates error, we measured rank error as the mean of $r-1$ across all query genomes ($1 \leq q \leq n$). Moreover, the mean (relative) distance error is defined as the mean of $\frac{d_q^{i+r} - d_q^{i+1}}{d_q^{i+1}}$ over all queries.

3.4 Discussion

We showed that Skmer can compute the genomic distance between a pair of species from genome-skims with very low coverage (at or even below 1X), with much better accuracy than the main two alternatives, Mash and AAF. We also showed that the distances computed by Skmer can accurately place a voucher genome-skim within a reference database of genome-skims, and can be used to infer the phylogenetic tree with reasonable accuracy. While Skmer is not the first k-mer based approach for distance estimation or phylogenetic reconstruction, as we showed, the alternatives have low accuracy given low coverage data. We compare with Mash because it is used within Skmer and is one of the most widely-used alignment and assembly-free methods. However, we note that authors of Mash do no claim it can handle low coverage, and so our results are not a criticism of their approach. Besides the methods we discussed, many other alignment-free sequence comparison and phylogeny reconstruction algorithms exist [50, 51, 53, 44, 54, 55, 56, 57, 58, 59, 47, 60, 48, 61, 62]. However, these methods take

as input assembled (but unaligned) sequences, and thus, are not applicable in an assembly-free pipeline. In other words, their goal, is to avoid the alignment step and not the assembly step.

Compared to using COI markers, currently used in practice, we showed that using *all* k-mers, including those from the nuclear genome, improves the phylogenetic accuracy. These improvements are resulting from distances that have a larger range and more resolution compared to COI. Also, the increased resolution should not be surprising given that the entire genome is much larger than any single locus, reducing the variance in estimates of the distance. Beyond the question of resolution, gene trees and species trees need not match [91], a fact that can further reduce the accuracy of marker genes for both species identification and phylogeny reconstruction. By using the entire genome, Skmer ensures that an average distance across the genome is computed, reducing the sensitivity to gene tree/species tree discordances. Moreover, a recent result shows that the JC-transformed genomic distance is a statistically consistent estimator of the species distances despite gene tree discordance due to incomplete lineage sorting [92], further encouraging our use of the genomic distance as a measure of the evolutionary divergence.

We showed that genomic distances as small as 0.01 can be estimated accurately from genome-skims with 1X or lower coverage. What does a distance of 0.01 mean? The answer will depend on the organisms of interest. For example, two eagle species of the same genus (*H. albicilla* and *H. leucocephalus*) have $D \approx 0.003$ but two *Anopheles* species of the same species complex (*A. gambiae* and *A. coluzzii*) have $D \approx 0.018$. Broadly speaking, for eukaryotes, detecting distances in the 10^{-2} order is often enough to distinguish between species (Supplementary Fig. B.12). On the other hand, to differentiate individuals in a population, or very similar species, we may need to reliably estimate distances of the order 10^{-3}. Detection at these lower levels seems to require $>$ 1X coverage using Skmer (Supplementary Fig. B.4b) but future work should study the exact level of sequencing required for accurate ordering of species at distances in the order of 10^{-3} or less. Moreover, the question of the minimum coverage required may avail itself to information-theoretical bounds and near-optimal solutions, similar to those established for the assembly problem [93, 94].

Although most of our tests simulated genome skims simulated from assemblies, we also tested Skmer on genome skims simulated by subsampling previous whole genome sequencing experiments. Several complications have to be addressed in real applications. The actual coverage of real genome skims may not be uniform and randomly distributed and they can have an overrepresentation of mitochondrial or plastid sequence. More importantly, other sources of DNA originating from for example, parasites, diet, fungi, commensals, bacteria, and human contamination may all be present in the sample and may cause a bias in the estimation of distances. In our test, we simply searched all reads in a genome-skim against a few bacterial genomes and the human reference genome; this simple scheme filtered out up to $\sim$10% of reads (for *D. virilis*). These filtering strategies were sufficient to produce reliable distance estimates in the case of Drosophila genomes. We recommend that before using Skmer, such database searches should be used to find and eliminate bacterial or fungal contamination (using BLAST [95] or perhaps metagenomic tools such as Kraken [96]), as well as removing contaminant reads with human origin (using for example Bowtie2 [88]). However, in future, it will be beneficial to develop better methods for finding extraneous reads without reliance on known sources.

A related direction of future work is to explore whether Skmer can be extended to environmental DNA analyses, i.e., queries consisting of genome-skims of multi-taxa samples. While Skmer is presented here in a general setting, its best use is for eukaryotic organisms, where the notion of species is better established and species can be separated with reasonable effort. We tested Skmer on birds and insects, but we predict it will work equally well for plants, a prediction that we plan to test in future work.

Throughout our experiments, we used Mash* run on the assemblies to compute the ground truth. Given the true alignment of the two genomes, we can compute the true genomic distance as the proportion of mismatches among *aligned* orthologous positions (i.e., ignoring gaps). To ensure that Mash* closely approximates true distances, we used simulated genomes of Rat and Mouse from the Mammalian dataset of the Alignathon competition [97]. This simulation uses Evolver [98] and includes many forms of mutation, including indels, rearrangement, duplications,

and losses. On this dataset, the true distance based on the known true alignment is 0.145 and Mash* estimated the distance as 0.143, which is a very good approximation. In contrast, FastANI [99], an alignment-free sequence mapping tool for estimating average nucleotide identity, computes the distance as 0.189. If we count gaps as non-matching positions in the definition of distance, then the true distance would be 0.287, which also does not match FastANI. Presumably, FastANI, which relies on alignment of short blocks, counts short gaps (with *some* definition of short) as mismatch but excludes larger ones. Thus, on real data, Mash* is the best available option to approximate the true distance. Finally, note that, for real genomes, we chose not to use estimated whole genome alignments (WGA) to compute the ground truth because WGA is a difficult problem, and WGAs that are available are not necessarily accurate. We get inconsistent estimates of distance when we use pairwise or multiple WGAs. For example, between *D. melanogaster* and *D. yakuba*, the distance changes from 0.10 when using the multiple WGA [100], to 0.21 if we use the pairwise WGAs [101] from the UCSC genome browser [102], which is the state-of-the-art.

The connection between genomic distance and phylogenetic distance depends on mutation processes considered. If only substitutions are allowed and assuming the Jukes-Cantor model, the phylogenetic distance is $-\frac{3}{4}\ln(1-\frac{4}{3}d)$; note this transformation is monotonic and does not change rankings of matches to a query search. Assuming a more complex model such as GTR [103], genomic distance is not enough to estimate the phylogenetic distance. However, we have devised a simple procedure to estimate GTR distances using the log-det approach [104] by repeated applications of Skmer to perturbed reads (Appendix B.2). The GTR distances can rank matches to a query differently from the genomic distance; the accuracy of the two distances should be compared in future work.

Insertions, deletions, duplications, losses, and repeats can all lead to differences between genomes, thereby reducing the Jaccard index and increasing the genomic distance. They also impact genomic length. Interestingly, in our experiments, Skmer run with the true coverage is *less* accurate than with estimated coverage (Supplementary Fig. B.13). We speculate that on

genomes with repeats, by overestimating coverage, our method gives an estimate of the "effective" coverage, reducing the impact of repeats on the Jaccard index. Nevertheless, with these complex mutations, the correct definitions of the evolutionary distance and genomic distance are not straightforward; nor is it clear how the Jaccard index should be translated to the genomic distance. Here, we used a heuristic approach that simply averaged the length of the two genome, leaving these broader questions about the best definition of genomic distance in the presence of large structural variations to future work.

Chapter 3, in full, is a reprint of the material as it appears in Genome biology 20, no. 1 (2019): 1-20. "Skmer: assembly-free and alignment-free sample identification using genome skims". Shahab Sarmashghi, Kristine Bohmann, M. Thomas P. Gilbert, Vineet Bafna, and Siavash Mirarab. The dissertation author was the primary investigator and author of this paper.

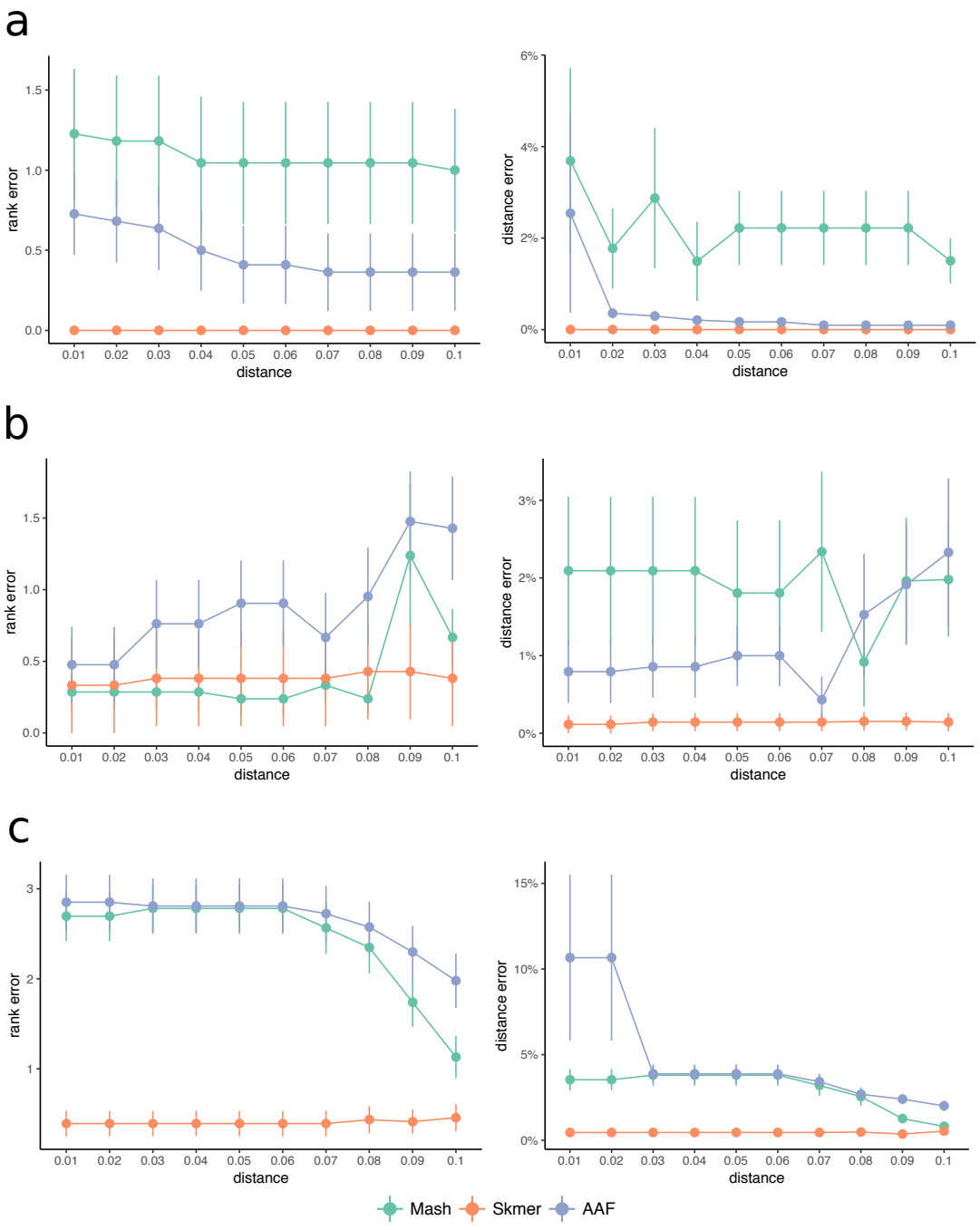


Figure 3.8. The mean rank and distance error of the best remaining match in leave-out experiments. The distance of closest genome in the reference to a query is varied from 0.01 to 0.1 (x-axis). The rank and distance errors (y-axis) of the best match to a query, are computed by comparing the order given by each method with the order obtained by applying Mash* to the full assemblies (ground truth). For each dataset, the experiment is repeated by taking each species as the query, and then the errors are averaged. Three methods, Mash, Skmer, and AAF, are compared on: (a) the *Anopheles* dataset, (b) the *Drosophila* dataset, and (c) the avian dataset.

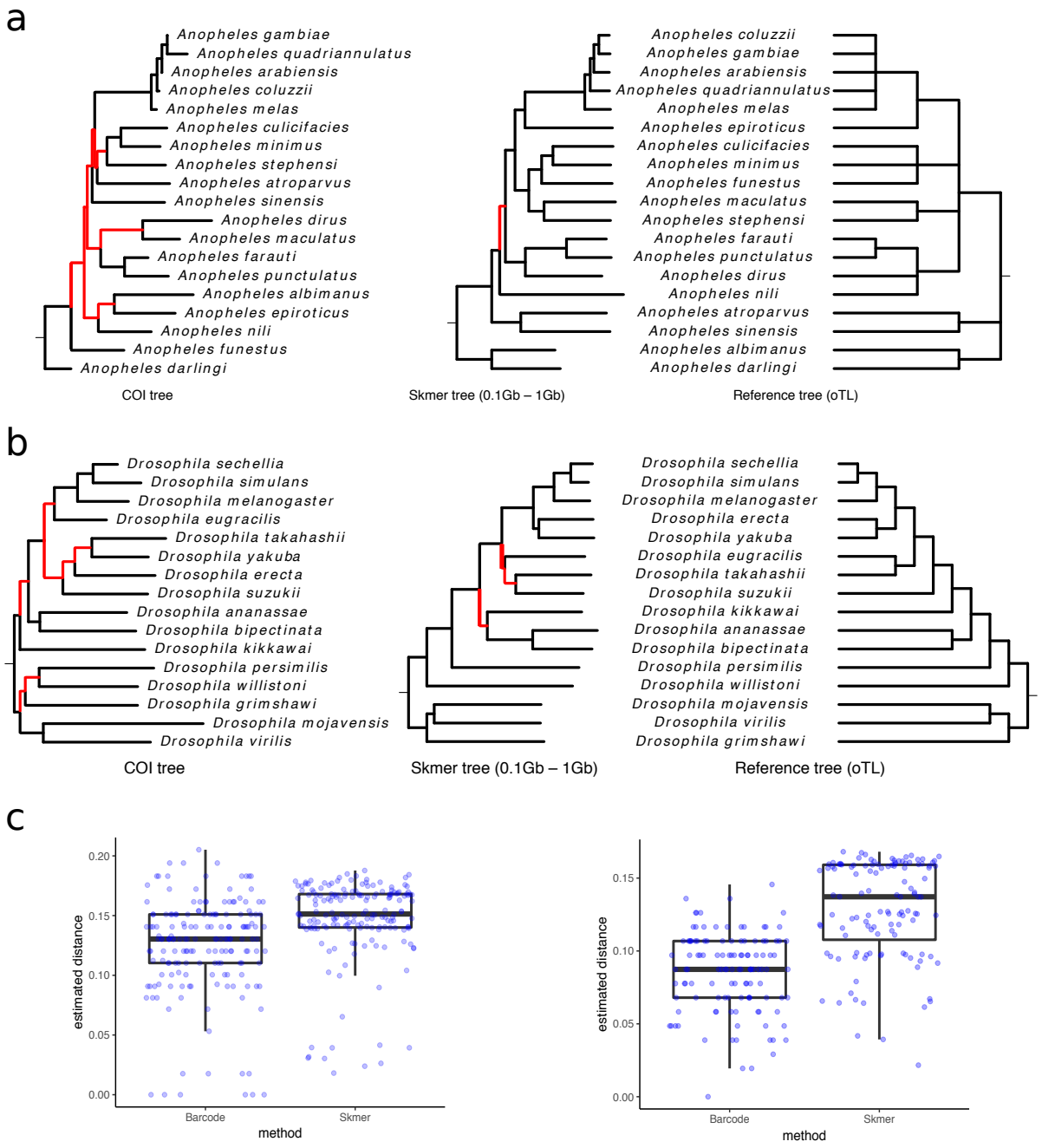


Figure 3.9. Comparing distances and phylogenetic trees from COI barcodes and simulated genome-skims. Shown in red are wrong internal branches corresponding to the bipartitions that are not found in the reference tree. Genome-skim size is randomly chosen among 0.1Gb, 0.5Gb, and 1Gb. (a) *Anopheles* trees. (b) *Drosophila* trees. (c) Distribution of distances for *Anopheles* (left) and *Drosophila* (right) genomes

Chapter 4

RESPECT: Estimating Repeat Spectra and Genome Length from Low-coverage Genome Skims

The cost of sequencing the genome is dropping at a much faster rate compared to assembling and finishing the genome. The use of lightly sampled genomes (genome-skims) could be transformative for genomic ecology, and results using k-mers have shown the advantage of this approach in identification and phylogenetic placement of eukaryotic species. Here, we revisit the basic question of estimating genomic parameters such as genome length, coverage, and repeat structure, focusing specifically on estimating the k-mer repeat spectrum. We show using a mix of theoretical and empirical analysis that there are fundamental limitations to estimating the k-mer spectra due to ill-conditioned systems, and that has implications for other genomic parameters. We get around this problem using a novel constrained optimization approach (Spline Linear Programming), where the constraints are learned empirically. On reads simulated at 1X coverage from 66 genomes, our method, REPeat SPECTra Estimation (RESPECT), had $< 1.5\%$ error in length estimation compared to 34% error previously achieved. In shotgun sequenced read samples with contaminants, RESPECT length estimates had median error 4%, in contrast to other methods that had median error 80%. Together, the results suggest that low-pass genomic sequencing can yield reliable estimates of the length and repeat content of the genome. The RESPECT software will be publicly available at https://github.com/shahab-sarmashghi/RESPECT.git

4.1 Introduction

Anthropogenic pressure and other natural causes have resulted in severe disruption of global ecosystems in recent years, including loss of biodiversity[105]. In North America alone, the bird population has declined by over a quarter since 1970[10]. Simply understanding the scope and extent of bio-diversity changes remains a challenging problem. Genomic sequence based biodiversity sampling provides an attractive alternative to physical sampling and cataloging, as falling costs have made it possible to shotgun sequence a reference specimen sample for at most $10 per Gb (with another $60 for sample prep). *However, the analysis typically requires assembling and finishing a reference genome, which can still be prohibitively costly.* Despite the many projects aimed at high quality genome sequencing of eukaryotic species [11], it could be many decades before we have acquired high-quality data so that biodiversity measurements for each population can be acquired on an ongoing, routine basis.

While (meta)barcoding [26, 27, 28] methods can be used for species identification and biodiversity measurements, they have many drawbacks including limited phylogenetic resolution [42, 43]. Organelle assembly based methods [106, 16, 14] similarly cannot be used for populations and often require whole genome sequences but discard the nuclear reads (the vast majority of data). Therefore, there is renewed interest in the development of methods that use all nuclear DNA from *genome-skims*–low-coverage (0.5-2Gb) sequencing, providing 0.2-4$\times$ coverage[107]. The low coverage of skims makes them cost-effective, but insufficient for assembling, and calls for assembly-free methods. Such methods, based on analysis of k-mers are being actively developed[108], and have been used for species identification (Skmer[109]); for phylogenetic placement of a new species not in the library (APPLES[110]), and contaminant filtering (CONSULT[111]). While k-mer analysis works well for species identification, it cannot be applied easily for the analysis of populations (individuals from the same species) using genome-skims, a key component of genomic ecology. Specifically, it ignores the effect of repeats, and uses heuristics to estimate sequencing error and coverage, neither of which is known.

In this chapter, we revisit the problem of estimating genomic parameters from genome-skim data: specifically, genome length L, sequence-coverage c, and repeat content. From genome-skim data, we have as input, abundance values of k-mers denoted by $\mathbf{o}$, where o_h denotes the number of distinct k-mers of multiplicity h. A key latent variable is the *k-mer-repeat-spectrum* (denoted hereafter as the k-mer spectrum) of the genome described by $\mathbf{r}$, where r_h denotes the number of distinct k-mers that appear exactly h times in the genome. As the value of o_h depends upon $\mathbf{r}, c, L$, and also on sequencing error, we consider the inverse problem of estimating genomic parameters given $\mathbf{o}$ as input. The problem was studied in a seminal paper by Li and Waterman[112] who mostly considered the case of high coverage and no sequencing errors. Williams et al.[113] improved upon this model by ignoring o_1 assuming that a large proportion of unique k-mers can be attributed to sequencing errors. This assumption works better for high coverage because at low coverage, many informative k-mers are also seen only once. Hozza *et al.*[114] point this out, and focus attention on k-mer spectra. Their method, CovEst, models spectra using a geometric distribution of unknown parameters, uses that parameterized model to estimate both parameters and r_1, r_2, r_3, and improves estimates even for low coverage and high error.

A distinct but related line of research relates to estimating $\mathbf{o}$ itself by sub-sampling or streaming reads. Melsted and colleagues[115, 116] describe streaming algorithms to estimate o_1 as well as moments $F_k = \sum_i i^k o_i$. Interestingly, these moments can also be used to estimate genome parameters. For example, $\mathbb{E}[F_1] = \lambda L$, where $\lambda = (1-(k-1)/\ell)c$ denotes the *k-mer coverage*, or the average number of k-mers covering a position derived from reads of length ℓ. We note that streaming is akin to low-coverage sampling and consider the case of estimating parameters over a range of λ.

4.1.1 Estimating genome repetitiveness and other parameters using k-mers

While previous research has emphasized the estimation of genome length and coverage, we focus specifically on estimating the k-mer spectrum $\mathbf{r}$, defined below. Consider a genome of length L. Decompose the genome into a collection of all fixed-length (overlapping) sequences of length k, called k-mers. Let variables r_j $(j \geq 1)$ denote the number of k-mers that occur exactly j times in the genome. When k is large enough ($k \geq \log_4 L$), high values of r_j, for $j \geq 2$, can be attributed to the repetitive structures in the genome rather than chance similarities. Therefore, we define $\mathbf{r} = [r_1, r_2, \cdots]$ as the *(k-mer)-repeat-spectrum* of the genome.

While the repetitive sequences occur in a variety of arrangements in terms of their multiplicity, complexity and the size of repeating unit, the repeat spectrum provides a valuable summary of the extent of repetition in the genome as well as other parameters. For example, the genome length can be estimated as $L = k - 1 + \sum_j j r_j \simeq \sum_j j r_j$. Define the *uniqueness ratio* of a genome as r_1/L, or the ratio of the number of k-mers seen only once to the genome length (which is the total number of k-mers in the genome). We computed the uniqueness ratio for 622 eukaryotic genomes in RefSeq using $k = 31$ (Supplementary Fig C.1). The ratio revealed a broad spectrum of values, ranging from 0.287 for *A. tauschii* (Tausch's goatgrass) to 0.995 for a mite species, *V. jacobsoni* (Fig 4.1A). Expectedly, there is some phylogenetic correlation and the variation of uniqueness ratio within a genus (intra-generic) is significantly lower than inter-generic variation of uniqueness ratios (Supplementary Fig C.2). At higher taxonomic ranks, we observed that plants had a significantly lower uniqueness ratio compared to other groups (Fig 4.1B), consistent with a prevalence of whole genome duplication (WGD) events (see Methods). Nevertheless, the correlation is not strong enough to predict uniqueness ratios solely from taxonomy. For example, rice species *O. sativa* and *O. brachyantha* have different ratios 0.91 and 0.75, respectively.

The repeat spectrum provides other insights. In genomes composed largely of unique

sequences, $r_1/L \simeq 1$ and r_j values decrease rapidly for $j \geq 2$ with $\log r_1/r_5 \geq 4.5$ (Fig 4.1C). On the other hand, genomes with higher repetitive content have a smoother decrease of r_j values (Fig 4.1D) with $\log \frac{r_1}{r_5} \leq 2.5$ (Supplementary Fig C.3). Additionally, a genome that has duplicated very recently will have $r_1 \simeq 0$ and a very high value of r_2. Over time, however, r_1 increases due to the accumulation of mutations. Similarly, $r_j > 0$ for large values of j suggest the presence of interspersed repeats.

Our method RESPECT (Repeat Spectrum identification) derives genomic length and coverage from low-coverage genome skims, while also providing insight into the repeat structure. We showed, through a mix of theoretical reasoning and empirical evidence, that the k-mer repeat spectra estimation problem is fundamentally difficult because of severe ill-conditioning of the system. In fact, the spectra are hard to estimate even when the coverage and sequencing error rate are known. We resolve this problem for the case of known coverage and sequencing error by imposing constraints on r_h and solving a constrained optimization problem. This approach provides greatly improved estimates of $\mathbf{r}$, which in turn lead to even better estimation of coverage, genome length and sequencing error through a stochastic iteration method. Results on genomes sampled from different parts of the tree of life and with differing repeat structures illustrate the validity of our approach. RESPECT is available at https://github.com/shahab-sarmashghi/RESPECT.git

4.2 Results

4.2.1 A simple model for estimating repeat spectra from unassembled data performs poorly

Assume that reads in the genome-skim are sequenced with a fixed mean error rate of ε per bp, and that the read start positions follow a Poisson distribution with a mean coverage of λ per bp. Denote the *observed* k-mer data as the vector $\mathbf{o} = [o_1, o_2, \cdots]$, where o_h denotes the number of k-mers observed exactly h times in the genome-skim input. The value o_h is the outcome of a

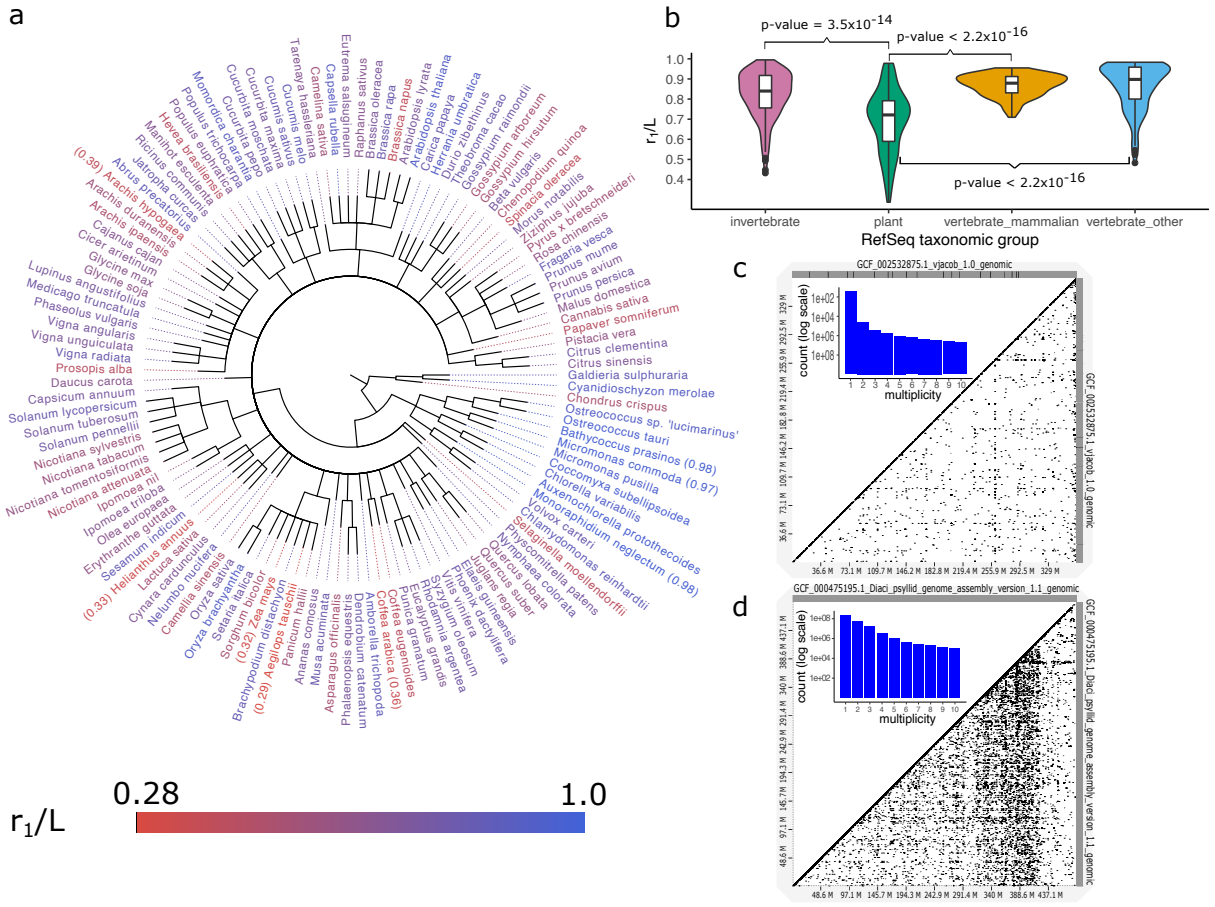


Figure 4.1. Characterizing repeats at k-mer level. (a) RefSeq plant taxonomy. The species are color-coded based on the uniqueness ratio, from red (highly repetitive) to blue (non-repetitive). (b) Uniqueness ratio distribution among four major taxonomic groups of eukaryotes in RefSeq. Plants (green) have significantly lower $r1/L$ compared to invertebrates (pink), mammals (yellow), and other vertebrates (blue). P-values shown on the figure, are the result of statistical tests that the uniqueness ratio is lower among plants compared to other groups. Also, to understand the extent of difference, we tested if the ratios are lower among plants by $X\%$ margin. The results are 5% p-value $= 1.1 \times 10^{-6}$, 10% p-value $= 4.3 \times 10^{-6}$, and 10% p-value $= 4.2 \times 10^{-6}$ when comparing plants against invertebrates, mammals, and other vertebrates, respectively. (c) Dot-plot of *V. jocobsoni* genome's (self)alignment with very few off-diagonal points, and a rapidly decaying repeat spectrum ($r_1/L = 0.99$). (d) Dot-plot of *D. citri*'s highly-repetitive genome marked by many off-diagonal elements and a smoothly decreasing repeat spectrum ($r_1/L = 0.51$).

random variable O_h that depends upon the parameter set $\Phi = \{\lambda, \varepsilon, \mathbf{r}\}$ (See Methods: 'Modeling genomic parameters'). Specifically, we assume that each k-mer with copy number j in the genome is sampled h times according to a Poisson distribution with rate dependent upon k, Φ.

Let P_{hj} represent the probability of h observances of a k-mer with copy number j. Then, in expectation,

$$\mathbb{E}[\mathbf{O}] = \mathbf{r}\mathbf{P}^{\mathbf{T}} + \mathbf{1}_{h=1}E \tag{4.1}$$

where E is the expected number of erroneous k-mers that in turn depends upon Φ. Φ could be estimated using:

$$\Phi = \arg\min_{\Phi} \|\mathbf{o} - \mathbb{E}[\mathbf{O}]\| = \arg\min_{\Phi} \|\mathbf{o} - (\mathbf{r}\mathbf{P}^{\mathbf{T}} + \mathbf{1}_{h=1}E)\| \tag{4.2}$$

In principle, an iterative procedure could be used to solve the optimization; we start with initial estimates of λ and ε, and use them to compute $\mathbf{P}$ and E. Then, we can use the least-square (LS) method to find $\mathbf{r}$ which minimizes $\|\mathbf{o} - (\mathbf{r}\mathbf{P}^{\mathbf{T}} + \mathbf{1}_{h=1}E)\|$ (Eqn. 4.2) (See Methods: 'Least-squares estimate of repeat spectrum').

To study the accuracy of this model for repeat spectra estimation, we simulated genome skims at 1X coverage with no sequencing errors ($E = 0$) for all 622 genomes in RefSeq in four major taxonomic groups of eukaryotes. A subset of 66 species was selected as the test set. The test genomes were sampled such that their uniqueness-ratio (r_1/L) values matched the distribution of uniqueness-ratios of all 622 RefSeq genomes (Supplementary Fig C.4, see Methods: 'Comparing $r1/L$ distribution over different sets'). In the following text, all parameters were trained on the 556 training genomes, and all test results shown on the 66 test genomes.

For a baseline test, we assumed that the coverage λ was known, so that $\mathbf{r}$ could be estimated using $\|\mathbf{o} - \mathbf{r}\mathbf{P}^{\mathbf{T}}\|_2$ (Eqn. 4.2). Using an LS solver (see Methods: 'Least-squares estimate of repeat spectrum'), we obtained highly accurate estimates of r_1 on the test data (Fig 4.2A; LS method). However, even in this simple case with perfect knowledge of coverage and no sequencing error, the error in estimating r_j increased rapidly with increasing j, as the LS solution was often sparse and the estimation set $r_j = 0$ for many j's, contrary to its true value in the genome.

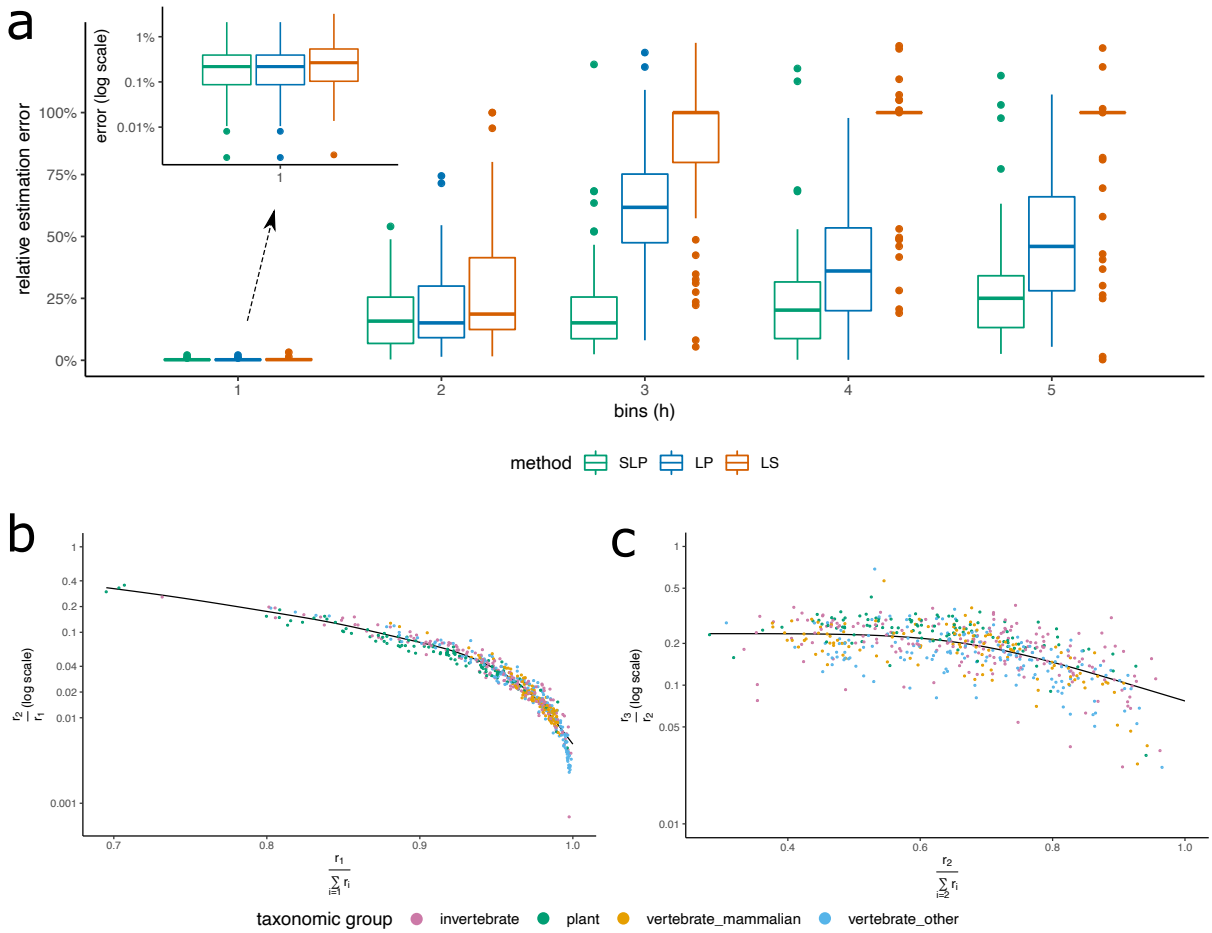


Figure 4.2. Repeat spectra estimation. (a) The relative error in estimating repeat spectra using Least-Squares (LS), constrained Linear Programming (LP), and Spline Linear Programming (SLP). The genome-skims are simulated at 1X with no sequencing error. (b) Correlation between true r_2/r_1 ratios, and our estimates of $r_1/\sum_{i=1} r_i$ for each genome. (c) Similar correlation plot between true r_3/r_2 and estimated $r_2/\sum_{i=2} r_i$. In both (b) and (c), true spectral ratios on Y axis are computed from the assemblies, and the estimated indices on X axis are obtained by applying the LP method to the simulated skims described in (a).

Empirical and theoretical results showed that the poor performance could be attributed to severe ill-conditioning. We proved that the condition number of $\mathbf{P}$ grows exponentially with the number of spectra (see Supplementary Methods). Therefore, small changes in $\mathbf{o}$ relative to $\mathbb{E}[\mathbf{O}]$ (Eqn. 4.1), for example due to the sampling variability or the simplifying assumptions of model, led to very large errors in estimates of $\mathbf{r}$.

4.2.2 Overview of RESPECT algorithm.

The negative result suggested a fundamental limitation to the use of k-mer based methods for estimating repeat spectra. Regularization is a proposed remedy for ill-conditioned matrices. However, most regularization methods enforce sparsity and $\mathbf{r}$ is known to be not sparse. A second challenge is that both observed counts and k-mer spectra are very skewed towards lower indices. Thus, a small (even 1%) relative error in r_1 could lead to a larger error in r_j for $j > 1$. To get around the ill-conditioning problem, we focused on *constraining* possible values of $\mathbf{r}$. We observed empirically that ratio of consecutive spectral values r_{j+1}/r_j was tightly constrained. Fig 4.2B traces r_2/r_1 as a function of $\frac{r_1}{\sum_{i\geq 1} r_i}$ on the training data and notes the tight correlation across all taxonomic groups. A similar, albeit less tight, constraint was observed for r_3/r_2 (Fig 4.2C) and other values as well (Supplementary Figs C.5-C.7).

These ideas provided the basis of a constrained linear-program for estimating $\mathbf{r}$. As a first step, we added the constraint that $\mathscr{L}_j \leq \frac{r_j}{r_{j+1}} \leq \mathscr{U}_j$ for each j, where $\mathscr{L}_j$ and $\mathscr{U}_j$ are the smallest and the largest $\frac{r_j}{r_{j+1}}$ ratios over the training genomes, and solved the following LP to find $\mathbf{r}$ (see Methods: 'Linear programming for constrained optimization based estimates')

$$\mathbf{r} = \arg\min_{\mathbf{r}} \mathscr{E} = \arg\min_{\mathbf{r}} \sum_{h=2}^{n} \left| o_h - \sum_{j=1}^{n} P_{hj} r_j \right| \tag{4.3}$$

This approach significantly improved the average error in estimating the spectra at multiplicity $j = 3$ and higher (Fig 4.2A; LP method), and resulted in small improvement at $j = 1, 2$ as well.

Using the repeat spectra from 556 training genomes, we observed a strong correlation between r_2/r_1 and $r_1/\sum_{i\geq 1} r_i$ (Fig 4.2B). Therefore, we estimated r_2/r_1 by using the LP estimate of $r_1/\sum_{i\geq 1} r_i$ and a spline fitted on the training data based on a generalized additive model [117, 118] (see Methods: 'Spline Linear programming'). The estimated r_2/r_1 value and the LP estimated r_1 value provided a new estimate (named SLP) of r_2. In a similar fashion, we computed SLP estimates of r_{j+1} from LP estimate of r_j and $r_j/\sum_{i\geq j} r_i$ for $j = 2, 3, 4, 5$ (Fig 4.2C,

Supplementary Figs C.5-C.7, and Methods: 'Spline Linear programming'). Using the additional information learned from the training genomes captured by the fitted splines, we obtained significant reduction in the average error of repeat spectra estimation (SLP vs. LP in Fig 4.2A). To solve the full optimization problem in Eqn. 4.2, we used a simulated annealing procedure. Specifically, starting with initial estimates of parameters obtained under no-repeat assumption, at each iteration a new values for λ is suggested, and SLP method is used to estimate $\mathbf{r}$. If a candidate λ results in a reduction in error, the algorithm accepts the move. Moreover, to avoid getting stuck at local minima, occasionally moves to states with higher error are also accepted. Lastly, the initial estimate of ε is corrected for the repetitiveness of genome using a regression learned over a subset of training genomes (Supplementary Fig C.8). The algorithm is outlined below (also see Methods: 'RESPECT algorithm' for a detailed description).

1. Generate initial estimates of λ, ε, and $\mathbf{r}$.
2. Compute the initial values of $\mathbf{P}$ and error function $\mathscr{E}$.
3. For $t = 1, \cdots, N$ repeat:
   3.1. Choose λ_{next} randomly within a neighborhood of current λ, and compute $\mathbf{P}_{\text{next}}$.
   3.2. Solve for $\mathbf{r}_{\text{next}}$ using SLP method.
   3.3. Use $\mathbf{P}_{\text{next}}$ and $\mathbf{r}_{\text{next}}$ to compute $\mathscr{E}_{\text{next}}$.
   3.4. Set $\lambda \leftarrow \lambda_{\text{next}}$, $\mathscr{E} \leftarrow \mathscr{E}_{\text{next}}$, and $\mathbf{r} \leftarrow \mathbf{r}_{\text{next}}$ with probability $\min\{1, \exp(-(\mathscr{E}_{\text{next}} - \mathscr{E})t/N)\}$.
4. Correct the initial estimate of ε, and update λ
5. Output $c = \lambda\ell/(\ell - k + 1)$, $\mathbf{r}$, $L = B/c$, and ε at the end of iterations (B is the total amount of nucleotides sequenced).

4.2.3 Estimating genome lengths

We applied RESPECT and CovEst to simulated genome-skims–Illumina reads sampled from the 66 test genomes skimmed at 1X coverage with 1% sequencing-error rate–and compared

their relative error in the estimation of r_1 through r_5 and genome length (Fig 4.3), after their convergence (see Supplementary Figs C.9-C.14 for the convergence of RESPECT's estimates). The median RESPECT error in estimating r_1 was less than 1.5% (average: 2.9%), while the median error of CovEst was 15% (average: 34%). The error profile extended to higher multiplicities, where, as noted earlier, CovEst used a parametric model. The tight relation between r_1 and r_2 and the large absolute differences between the two values implied that a small error in r_1 would translate into a large relative error for r_2, and we observed that for r_2. Similarly, the RESPECT estimates of genome length were highly accurate with median error 2.2% (average: 4.1%), in contrast to 27% (average: 40%) for CovEst (Fig 4.3B). RESPECT estimates were better than CovEst in 62 out of 66 species, often by considerable margins (Fig 4.3C). For example, in 54/66 species, RESPECT error was less than 5%, while CovEst error exceeded 50% in one third of test genomes. In fact, CovEst severely underestimates the length for these genomes (Supplementary Fig C.15). For 18/66 test genomes, the CovEST estimate was less than the true length by a factor of 4 or higher (Supplementary Fig C.16). SSRESPECT relies on models trained using available assemblies. We tested if the performance depended on the amount of training data and the taxonomic composition of the training data. RESPECT performance remained robust in these scenarios (Supplementary Fig C.17a). Moreover, its performance improved slightly (had fewer outliers) with additional training data (Supplementary Fig C.17b).

We repeated the same experiment at sequence level coverage of 0.5X, 2X, and 4X (Supplementary Fig C.18). At 0.5X coverage, the median error of RESPECT was 16% (average: 18%), while CovEst had 88% median error (average: 75%) and underestimated the length by a factor of 8 or more in half of the species (Supplementary Fig C.19). CovEst performance improved at higher coverage but RESPECT continued to have lower error (Supplementary Fig C.20). At 4X, CovEst had median error 3.3% (average: 7.6%), while RESPECT median error was $< 1\%$ (average: 1.9%). Moreover, CovEst error exceeded 10% error in a third of species, while RESPECT had $< 10\%$ error in 64/66 species (Supplementary Fig C.21).

SSWe also compared the performance of RESPECT among different taxonomic groups.

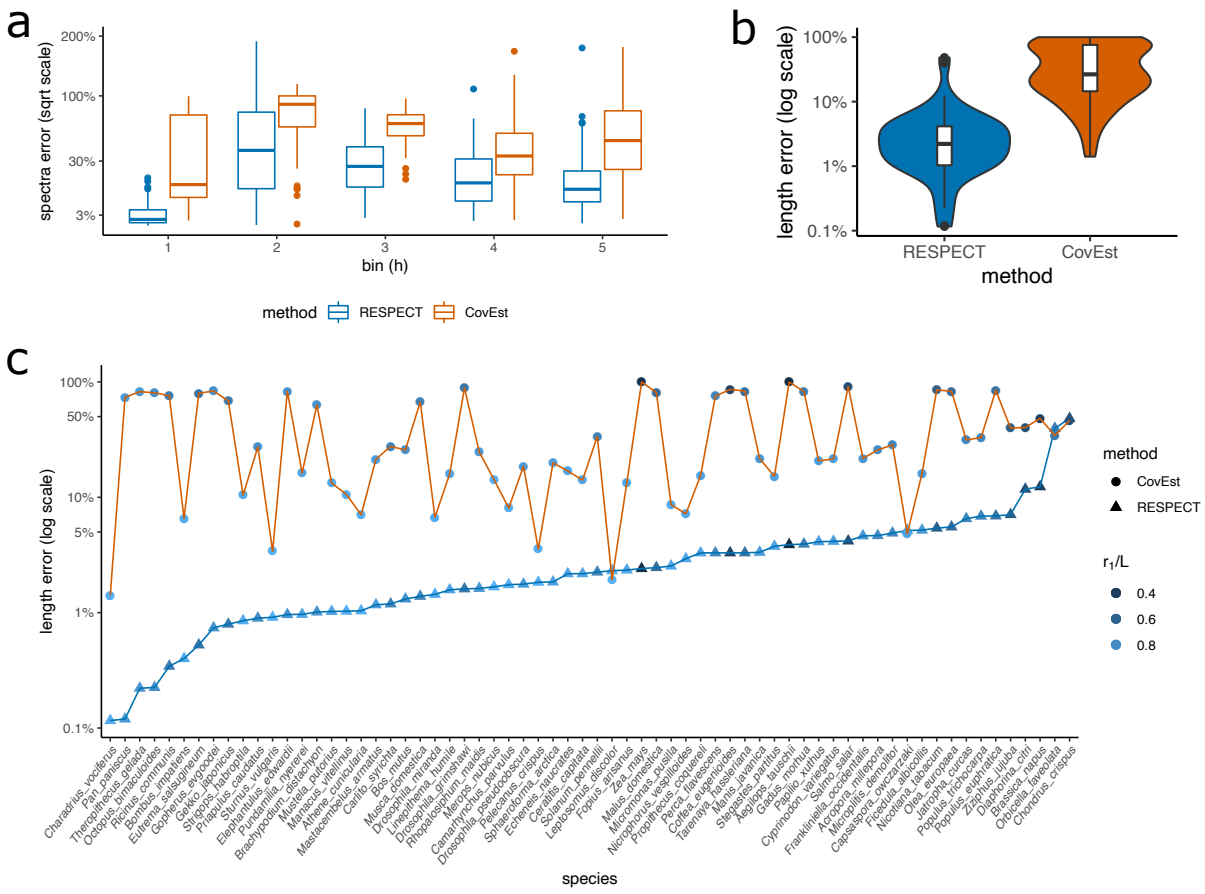


Figure 4.3. Iterative estimation of genome length. (a) Comparing the error of RESPECT and CovEst in estimating the repeat spectrum. The first 5 spectra are shown. (b) The distribution of error in CovEst and RESPECT. The absolute value of relative error in genome length estimation is used (in logarithmic scale). (c) Per-genome error of RESPECT and CovEst in estimating the genome length of 66 species with genomes skimmed at 1X coverage.

In general, plants and invertebrates had higher error rates compared to both vertebrate groups (Supplementary Fig C.22), consistent with their lower uniqueness ratios (Fig 4.1B). In fact, we observed a statistically significant negative correlation between the estimation error and the uniqueness ratio (Supplementary Fig C.23). We additionally tested RESPECT on simulated genome-skims at 1X coverage from 10 bacterial genomes, and the results did not suggest any bias against prokaryotic genomes (Supplementary Fig C.24), despite the fact that we trained our model on eukaryotic genomes.

4.2.4 Estimating genome length using sequenced short reads

A key difference between sequenced reads versus simulated reads is the presence of 'contaminants' or reads from non-target species. Differences may also include presence of adapter sequences, duplications of reads from the sequencing platform, lower or higher sequencing error rates due to DNA quality, and length variation of reads. Therefore, we tested RESPECT in genome-skims obtained from NCBI's Sequence Read Archive (SRA) database [119]. We downloaded high-coverage raw reads from 29 test species (from all four major taxonomic groups of eukaryotes in RefSeq) including highly repetitive plant genomes, and compared the results with the corresponding genome assemblies of the same data. After preprocessing the raw reads using BBTools [120] to remove adapter sequences and duplicate reads, we used Kraken[121] to remove contaminant reads with microbial or human origin (see Methods: 'SRA preprocessing and contamination filtering'). We note that this is an imperfect process as these tools work only when the contaminating organisms have a highly related member in the reference databases[122]. We discarded 10 samples because $> 40\%$ of reads (after removing adapters) were either duplicates of other reads, or came from external DNA sources (Supplementary Table C.1). For the remaining 19 samples, duplicates and reads classified as contaminant were removed, and unclassified reads were sub-sampled to 1X coverage. In 16 out of 19 samples, RESPECT error was less than 11% (median: 4%), including highly repetitive genomes such as *A. tauschii* ($r_1/L = 0.29$), *Z. mays (maize)* ($r_1/L = 0.32$), *S. salar (salmon)* ($r_1/L = 0.48$), and *N. tabacum* ($r_1/L = 0.57$), where the abundance of repeats made the length estimation challenging (Fig 4.4, Table 4.1). In contrast, CovEst had less than 30% error in only 4 samples (median error 80%) (Fig 4.4). For the highly repetitive genomes, CovEst length estimates ranged from 1/11 to 1/7 of the assembled sequence lengths or 10 to 30 times larger error compared to RESPECT (see Table 4.1). In 3 samples, RESPECT had relatively high errors. For SRR085103 (domestic ferret), 99.9% of the reads did not in fact map to the available reference assembly of the domestic ferret *M. putorious*. Together with the relatively low percentage of duplication (9%) the data

suggest a mislabeling of the sample species. For Coquerel's sifaka (*P. coquereli*), we observed a large gap between the total sequence length (2.8 Gbp) and the total ungapped length (2.1 Gbp) of the assembly, suggesting some challenges with the assembly. Cape elephant shrew (*E. edwardii*) was the last sample where RESPECT length estimate of 4.5Gbp exceeded the RefSeq (GCF_000299155.1) assembly length (3.8Gbp) by over 10%. Interestingly, the uniqueness ratio of the assembly was $r_1/L = 0.72$, which contrasted with the RESPECT estimated uniqueness ratio of $r_1/L = 0.65$ from the short-read data. Upon investigation, we found that a more recent assembly for *E. edwardii* (GCA_004027355.1), not yet in RefSeq, had an assembled length equal to 4.3 Gbp, with $r_1/L = 0.66$, matching the RESPECT estimates (4.5Gb, 0.65, respectively). The difference between total sequence length and ungapped length in GCA_004027355.1 was only 1 Mbp, in contrast to $>$ 500 Mbp for GCF_000299155.1. Together, these data suggest that GCA_004027355.1 better assembles repetitive regions, and the RESPECT length estimation error was $< 5\%$, despite using only 1X coverage.

Table 4.1. Comparing RESPECT and CovEst accuracy on SRA's of highly repetitive genomes. The numbers in parentheses are the percentage errors.

Species	*A. tauschii* (goat grass)	*Z. mays* (maize)	*S. salar* (salmon)	*N. tabacum* (tobacco)
r_1/L	0.29	0.32	0.48	0.57
Assembly length (Gbp)	4.3	2.1	3.0	3.6
RESPECT	3.9 (-10.7%)	2.0 (-8.2%)	2.8 (-4.9%)	3.7 (2.6%)
CovEst	0.4 (-90%)	0.2 (-90%)	0.3 (-90%)	0.5 (-86%)

4.2.5 The role of WGD versus high copy repeat elements in shaping genome repeat structure

Predicting polyploidy and recent WGD is challenging because mutation and gene loss after a WGD event can reduce the polyploidy signal. Specifically, a WGD event results in the uniqueness ratio (r_1/L) becoming 0. Subsequently, as mutations accumulate, r_1/L ratio moves gradually towards 1 in a process that may be specific to the species, and hard to predict. Nevertheless, it should be skewed toward smaller values for recent WGD events. Independently,

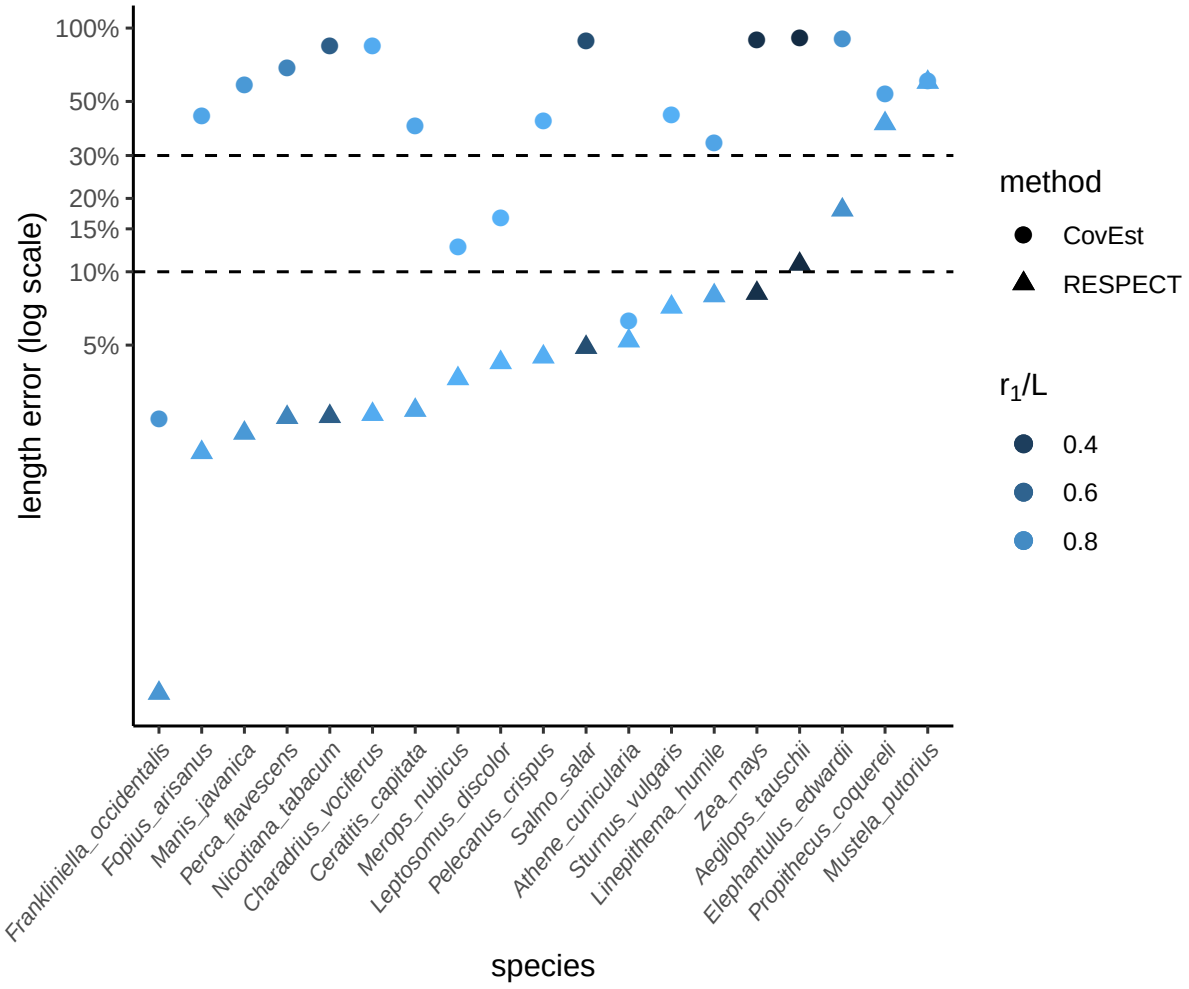


Figure 4.4. Estimating genome length using SRA data. Comparing the error of CovEst and RESPECT. High coverage SRA were preprocessed and later downsampled to 1X coverage. Both methods are applied to genome skims (after preprocessing) and the absolute values of the relative error in estimating the genome lengths are compared.

the presence of high copy repeats due to DNA transposons and retrotransposons can lead to very high copy numbers of a small set of oligomers. To capture the contribution of high copy repeat elements, we defined the 'High Copy Repeats per Million (HCRM)' value as the average count (per million base-pairs) of the 10 most highly repetitive k-mers. HCRM values varied across the species, ranging from 2 to 3738 among our set of 622 RefSeq genomes (Supplementary Fig C.25). We observed some correlation between HCRM values of species of the same genus, especially among vertebrates (Supplementary Fig C.26). However, similar to the case of uniqueness ratios,

the phylogenetic signal was not pronounced enough to predict HCRM based on the taxonomy.

Analytical calculations showed that the probability of high HCRM values ≥ 200 in a genome with random set of k-mers was negligibly small ($P \leq 10^{-100}$) (See Methods: 'Statistical analysis of the repeat structure'), suggesting that high HCRM values could not be explained solely by WGD events, and were likely due to high copy (transposon) repeats. Fig 4.5 shows the (r_1/L, HCRM) value of 622 genome-skims, which tightly matched the true values computed from assembled genomes (Supplementary Fig C.27). To analyze the r_1/L and HCRM values of genomes with recent WGD, we compiled a partial list of species with known WGD events within the last 150M years based on the available literature[123, 124, 125] (See Methods: 'Selecting species with known recent WGD events' and Supplementary Table C.2).

Species with known recent WGD events had expectedly low r_1/L. For example, only 14% of species with recent WGD had r_1/L values ≥ 0.8, in contrast with 64% of all species that had r_1/L values higher than 0.8. Surprisingly, 93% of species with recent WGD also had low HCRM values (≤ 200) (Fig 4.5), and there was a strong association between the occurrence of recent WGD events and the (r_1/L, HCRM) values (p-value: 1.8×10^{-23}; See Methods: 'Statistical analysis of the repeat structure'). Our results suggest that genomes with low HCRM and r_1/L are strong candidates for WGD events.

4.3 Methods

Comparing $r1/L$ distribution over different sets

To compare two sets of values and see if the values in one set are greater than the other set, we used the Mann–Whitney U test. Formally, if X and Y are random samples from populations $\mathscr{X}$ and $\mathscr{Y}$, the test statistic, U, is given by the number of times x is greater than y for all $(x,y) \in \mathscr{X} \times \mathscr{Y}$. The Mann–Whitney U test is non-parametric and does not restrict the samples to be from a certain family of distributions. The test also allows the user to specify a location shift μ and examine the alternative hypothesis that $X - Y > \mu$. By gradually increasing μ and computing the p-value, we can understand the extent of difference between X and Y.

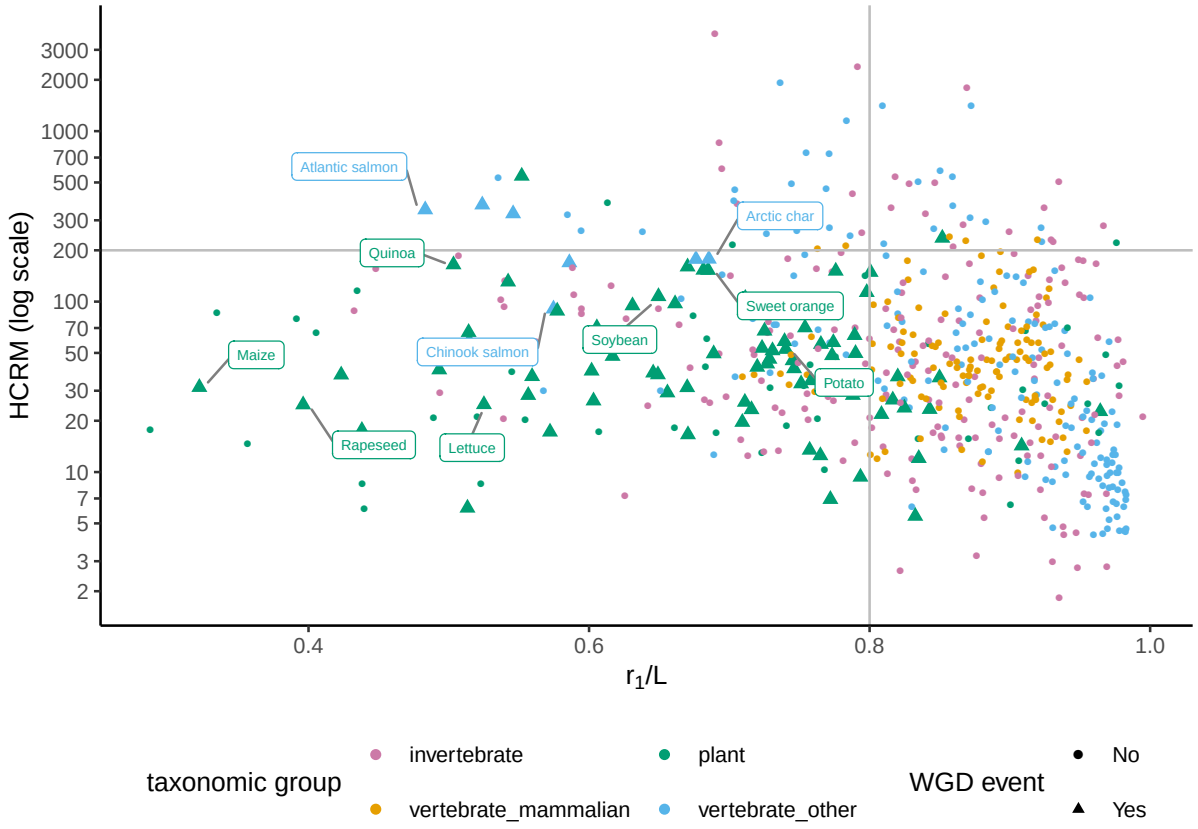


Figure 4.5. High copy repeats per million versus uniqueness ratio among genomes with and without known recent WGD events. Most of genomes with known recent WGD events had $r_1/L < 0.8$ and HCRM < 200. The y-axis is in a logarithmic scale. HCRM values are computed from genome-skims simulated at 1X coverage with no sequencing error. Some of the species with a recent WGD are labeled by their common names.

To test if two sets of numbers are drawn from the same distribution, we used the two-sample Kolmogorov–Smirnov (KS) test. The test statistic is a distance between the empirical distributions functions of the samples from the two sets. We used R 'stats' package [126] to compute the p-values for both tests.

Modeling genomic parameters

We consider k-mers in a genome of length L and assume that $k \gg \log_4 L$ so that any k-mer is unlikely to appear more than once, unless it is part of a repeated sequence. Denote the (unknown) k-mer spectrum of a genome that contains repeats using $\mathbf{r}$, where r_j describes the

number of distinct k-mers that appear exactly j times in the genome.

The genome is shotgun sequenced using reads of length ℓ with average sequencing depth c. The total number of nucleotides sequenced is given by $B = cL$. As there are $l-k+1$ k-mers in each read, the k-mer coverage is given by

$$\lambda = (1-(k-1)/\ell)c = \frac{(1-(k-1)/\ell)B}{L} . \tag{4.4}$$

Let $\mathbf{o}$ denote the histogram of observed k-mer counts. The observed number of k-mers of abundance h, o_h, can be thought of as a sample allocation to random variable O_h, whose expected value, $m_h = \mathbb{E}[O_h]$, depends upon $\mathbf{r}$, λ, L, and also on sequencing error. We assume that any base-pair is sequenced erroneously with probability ε, and sequencing errors only result in novel k-mers. We further assume that the number of times a unique k-mer repeated j times is sampled follows a Poisson distribution with rate $\lambda j(1-\varepsilon)^k$. Therefore

$$\mathbf{m} = \mathbf{r}\mathbf{P}^{\mathbf{T}} + \mathbf{1}_{h=1}E , \tag{4.5}$$

where $P_{hj} = e^{-j\lambda(1-\varepsilon)^k}\frac{(j\lambda(1-\varepsilon)^k)^h}{h!}$ denotes the probability that a k-mer repeated j times in the genome is observed with count h in the genome skim, $\mathbf{1}_{h=1} = [1,0,0,\ldots]$, and $E = L\lambda(1-(1-\varepsilon)^k)$ is the expected number of erroneous k-mers. As λ and L are connected through Eqn. 4.4, we choose λ as the independent variable and consider L as a function of λ. Under this model, we would like to estimate $(\mathbf{r},\lambda,\varepsilon) = \arg\min_{\mathbf{r},\varepsilon}\mathscr{E}(\mathbf{P},\mathbf{r},\varepsilon,\mathbf{o})$, where $\mathscr{E}$ is a weighted p-norm of the difference between expected and observed counts

$$\mathscr{E}_{\mathbf{w},p}(\mathbf{P},\mathbf{r},\varepsilon,\mathbf{o}) = \left(\sum_h w_h\left|m_h - o_h\right|^p\right)^{1/p} = \left(\sum_h w_h\left|(\mathbf{r}\mathbf{P}^{\mathbf{T}} + \mathbf{1}_{h=1}E)_h - o_h\right|^p\right)^{1/p} . \tag{4.6}$$

Note that the optimization is non-trivial because $\mathbf{P}$ and E are functions of $(\mathbf{r},\lambda,\varepsilon)$, and must be simultaneously estimated.

A generic iterative optimization for parameter estimation

The dimensions of $\mathbf{o}$ and $\mathbf{r}$ in Eqn. 4.5 are determined entirely by data and are not necessarily identical. However, we truncated both to a common dimension $n = 50$ for computational expediency. A generic optimization method could be described as below.

1. Generate initial estimates of λ, ε, L.
2. Solve for $\mathbf{r}$ using Eqn. 4.6.
3. Use estimated $\mathbf{r}$ and grid-search to re-estimate λ, ε.
4. Repeat step 2 onwards until the error has converged.

Step 2 is the key step in this procedure, and we devised a number of approaches to solve it.

Least-squares estimate of repeat spectrum

Choosing $p = 2$ (Euclidean norm) and $w_h = 1, \forall h$ in Eqn. 4.6, the problem is turned into a Least-Squares (LS) optimization. To test an LS method for estimating $\mathbf{r}$, we considered the simplest sequencing-error-free case ($\varepsilon = 0$), where coverage λ was known. Therefore, $\mathbb{E}[\mathbf{O}] = \mathbf{m} = \mathbf{r}\mathbf{P}^{\mathbf{T}}$, where $\mathbf{P}$ is an $n \times n$ matrix with

$$P_{hj} = e^{-j\lambda} \frac{(j\lambda)^h}{h!} .$$

We showed (Supplementary Methods) that $\mathbf{P}$ is non-singular and in the error-free case, it should be possible to use the estimate $\mathbf{r}^{(\text{est})} = \mathbf{o}\mathbf{P}^{-\mathbf{T}}$. However, we observed that its effective rank was very small as Λ, E each have rapidly diminishing eigenvalues. Therefore, instead of decomposing $\mathbf{P}$ and explicitly computing $\mathbf{P}^{-\mathbf{1}}$, we used the non-negative least squares (NNLS) method[127] to solve

$$\mathbf{r}^{(\text{est})} = \arg\min_{\mathbf{r}} \|\mathbf{o} - \mathbf{r}\mathbf{P}^{\mathbf{T}}\|_2 .$$

We used nnls method from SciPy's [128] Optimize library. Unfortunately, the LS estimates were very unreliable and showed high error. In fact, we proved, for $\lambda = 1$ (see Supplementary

Methods), that

$$cond(\mathbf{P}) \geq \frac{2^n}{n} \ .$$

The condition number grows exponentially with n suggesting a highly ill-conditioned matrix $\mathbf{P}$ where small changes in $\mathbf{o}$ from the expected values $\mathbf{m}$ would lead to large errors in estimate of $\mathbf{r}$. For these reasons, we adopted constrained optimization methods to solve for $\mathbf{r}$.

Linear programming for constrained optimization based estimates

We used Eqn. 4.6 with $\mathbf{w} = [0, 1, 1, \ldots, 1]$ and $p = 1$ to design a Linear programming estimate of $\mathbf{r}$ as:

$$\min_{\mathbf{r}} \sum_{h=2}^{n} \left| o_h - \sum_{j=1}^{n} P_{hj} r_j \right| , \tag{4.7}$$

such that

$$\mathscr{L}_h \leq \frac{r_h}{r_{h+1}} \leq \mathscr{U}_h, \ \ h = 1, 2, \cdots, n-1$$

The rationale behind setting $w_1 = 0$ was that o_1 contains a large number of erroneous k-mers, so we exclude it from the objective function and use the rest of the bins to estimate $\mathbf{r}$. As ε is not known in general, o_1 was used to estimate the (average) sequencing error rate, and subsequently the k-mer coverage λ.

The lower and upper bounds on $\frac{r_j}{r_{j+1}}$ were determined based on the distribution R_j of spectral ratios in 556 training genomes, and therefore we only search for candidate solutions $\mathbf{r}$ that satisfy the constraints. Specifically, we profiled the repeat spectra of the training genomes and set $[\mathscr{L}_j, \mathscr{U}_j]$ equal to the empirical support of R_j distribution, i.e., $\mathscr{L}_j$ and $\mathscr{U}_j$ are the smallest and the largest samples observed from R_j over the training genomes. We use Gurobi Optimizer [129] to solve the constrained optimization problem formulated in Eqn. 4.7.

Spline Linear programming

The final method of estimating $\mathbf{r}$ is based on the LP estimate of $\mathbf{r}$ and the splines fitted on spectral ratios r_j/r_{j+1} as functions of $\frac{r_j}{\sum_{i \geq j} r_i}$. Formally, let r_j^{LP} denote the LP estimate of r_j by constraining the spectral ratios to be within the support of R_j among the training genomes,

as discussed above. For each $j \in \{1,2,3,4,5\}$, we used a generalized additive model (GAM), learned from 556 training genomes, to predict r_j/r_{j+1} based on $\frac{r_j^{\mathrm{LP}}}{\sum_{i\geq j} r_i^{\mathrm{LP}}}$. Specifically, we model $y_j = r_j/r_{j+1}$ for different genomes as samples drawn from dependent random variable Y_j, which follows gamma distribution and its mean is determined by

$$g_j(\mathbb{E}\left[Y_j\right]) = s_j(\frac{r_j}{\sum_{i\geq j} r_i}) \,, \tag{4.8}$$

where g_j is called the link function, and s_j is the smoothing spline. These functions allow us to capture nonlinear dependencies between the variables in our model. For $j = 1,2$, we use a logarithmic link function to account for the large dynamic range of r_j/r_{j+1} over the training set, and use identity link for $j = 3,4,5$. For each fitted GAM, we empirically set the smoothing parameter to balance the over-fitting against the goodness of fit. We used R 'mgcv' package [130] for GAM fitting.

Using the LP estimates of r_j's and plugging them into Eqn. 4.8, we predict the spectral ratios. Let y_j^{SLP} denote the estimate of y_j using Eqn. 4.8 on previous estimates of $\mathbf{r}$. We *recursively* re-estimate r_j for $j \in \{2,3,4,5,6\}$ and call them r_j^{SLP}:

$$r_j^{\mathrm{SLP}} = \begin{cases} r_j^{\mathrm{LP}} & j = 1 \text{ and } j > 6 \\ r_{j-1}^{\mathrm{SLP}}/y_{j-1}^{\mathrm{SLP}} & 2 \leq j \leq 6 \end{cases} \tag{4.9}$$

RESPECT algorithm

For the RESPECT algorithm, we replaced the basic iterative method described above with a simulated annealing procedure outlined in Algorithm 1 to speed up the computations. To initialize the algorithm, we started with the assumptions that genome has no repeats $\mathbf{r} = [L,0,0,\ldots]$, and the error-free k-mer counts follow a Poisson distribution (Eqn. 4.5). Defining $\lambda_{\mathrm{ef}} = \lambda(1-\varepsilon)^k$ as the *error-free k-mer coverage*, we estimate its initial value from the ratio of

observed counts

$$\lambda_{\text{ef}} = \frac{(h^*+1)o_{h^*+1}}{o_{h^*}}, \text{ where } h^* = \underset{h>1}{\operatorname{argmax}}\, o_h \,,$$

and set

$$\lambda = e^{-\lambda_{\text{ef}}} \frac{\lambda_{\text{ef}}^{h^*}}{h^*!} \frac{o_1}{o_{h^*}} + \lambda_{\text{ef}}(1 - e^{-\lambda_{\text{ef}}}), \;\; \varepsilon = 1 - (\lambda_{\text{ef}}/\lambda)^{1/k}$$

(see Supplementary Methods). The above estimate of ε is used throughout the algorithm, but is corrected at the end based on the estimated uniqueness ratio (described below). Using the estimate of λ_{ef}, we compute $\mathbf{P}$, and thus the error function $\mathscr{E}$ at the start of the algorithm. For $\mathscr{E}$, we chose $\mathbf{w} = [0, 1, 1, \ldots, 1]$ and $p = 1$ in Eqn. 4.6, so

$$\mathscr{E} = \sum_{h=2}^{n} \left| o_h - \sum_{j=1}^{n} P_{hj} r_j \right|$$

With the initial values of the parameters known, RESPECT runs a simulated annealing optimization until the error converges. At each iteration, a candidate λ_{next} in $[\frac{1}{2}\lambda, 3\lambda]$ is selected uniformly at random, and $\mathbf{P}_{\text{next}}$ is computed from $\lambda_{\text{next}}(1-\varepsilon)^k$. Next, we run SLP method on ($\mathbf{o}$, $\mathbf{P}_{\text{next}}$) to get $\mathbf{r}_{\text{next}}$. Throughout the algorithm, we used truncated $\mathbf{o}_{1\times m}$, $\mathbf{r}_{1\times n}$, and $\mathbf{P}_{m\times n}$ where the number of spectra is fixed at $n = 50$ (a reasonable compromise between accuracy and speed), and the number of observed counts $m = n \cdot \max(1, \lambda_{\text{ef}})$ scales proportionally with the initial estimate of error-free k-mer coverage. Using ($\mathbf{o}$, $\mathbf{P}_{\text{next}}$, $\mathbf{r}_{\text{next}}$), error function for the candidate state $\mathscr{E}_{\text{next}}$ is calculated. If moving to the candidate state results in a reduction in the error ($\mathscr{E}_{\text{next}} < \mathscr{E}$), the algorithm accepts the move and updates the current estimate of parameters. In addition, to help the algorithm deal with local minima and find better solutions, a simulated annealing scheme is implemented such that the algorithm probabilistically decides to move to states with higher error. Specifically, at iteration t, even if $\mathscr{E}_{\text{next}} > \mathscr{E}$, the algorithm accepts the move with probability $\exp(-(\mathscr{E}_{\text{next}} - \mathscr{E})t/N)$.

At the end of iterations, the initial estimate of ε (obtained under no-repeats assumption) is corrected based on the estimated value of r_1/L. The correction was learned over 120 genomes

Algorithm 1: The RESPECT method.

Start with $\lambda_{\text{ef}} = \lambda^{(0)}(1-\varepsilon)^k = \frac{(h^*+1)o_{h^*+1}}{o_{h^*}}$, where $h^* = \text{argmax}_{h>1}\, o_h$;

Compute $\mathbf{P}^{(0)}$, $\mathscr{E}^{(0)} = \min_{\mathbf{r}} \mathscr{E}(\mathbf{P}^{(0)}, \mathbf{r}^{(0)}, \mathbf{o})$, and $\mathbf{r}^{(0)} = \text{argmin}_{\mathbf{r}} \mathscr{E}(\mathbf{P}^{(0)}, \mathbf{r}^{(0)}, \mathbf{o})$;

Find $E = o_1 - \sum_j P_{1j}^{(0)} r_j^{(0)}$;

Set $\lambda^{(0)} = e^{-\lambda_{\text{ef}}} \frac{\lambda_{\text{ef}}^{h^*}}{h^*!} \frac{o_1}{o_{h^*}} + \lambda_{\text{ef}}(1 - e^{-\lambda_{\text{ef}}})$, and compute ε from λ_{ef} and $\lambda^{(0)}$;

for $1 \le t \le N$ **do**

  $\lambda^{(t)} \leftarrow \mathscr{U}[\frac{1}{2} \cdot \lambda^{(t-1)}, 3 \cdot \lambda^{(t-1)}]$;

  Use $\lambda^{(t)}$ and ε to compute $\mathbf{P}^{(t)}$, $\mathbf{r}^{(t)} = \text{argmin}_{\mathbf{r}} \mathscr{E}(\mathbf{P}^{(t)}, \mathbf{r}^{(t)}, \mathbf{o})$, and $\mathscr{E}^{(t)} = \min_{\mathbf{r}} \mathscr{E}(\mathbf{P}^{(t)}, \mathbf{r}^{(0)}, \mathbf{o})$;

  Move to $\lambda^{(t)}$ with probability $\min\left\{1, \exp\left(\frac{\mathscr{E}^{(t-1)} - \mathscr{E}^{(t)}}{N-t+1}\right)\right\}$;

end

Correct ε and set $\lambda = \lambda^{(N)}(1-\varepsilon)^k / (1-\varepsilon_{\text{corrected}})^k$;

Output $c = \frac{\ell}{\ell-k+1}\lambda$, $L = B/c$, $\varepsilon_{\text{corrected}}$, and $\mathbf{r}^{(N)}$

randomly selected from the training set, and applied if the estimated coverage is smaller than 1.5X. Then, λ is re-computed based on the corrected ε, and is used to compute the final estimates of coverage and genome length. The estimated sequencing error rate and repeat spectrum are also provided by the algorithm.

SRA preprocessing and contamination filtering

After downloading SRA accessions and converting them to FASTQ using SRA Toolkit [131], we used BBDuk and Dedupe from BBTools package to trim adapter sequences and remove duplicate reads. We then ran Kraken2 to remove contamination with prokaryotic or human origin. For plant and invertebrate samples, we filtered out any read that was classified to the Kraken database at 0 confidence level (very sensitive, a single matched k-mer is enough for the classification). For vertebrates, due to their smaller evolutionary distance to homo sapiens, we required 0.5 confidence level (more specific, half of the read's k-mers should match) for human classification, and 0 confidence level for everything else in the database.

Implementation details and running time

We use 'count' and 'histo' commands from Jellyfish [68] command line tool to compute the k-mer histogram of input genome-skims. In each iteration of RESPECT algorithm, we

solve a constrained optimization problem using the tools provided by Gurobi Python interface in 'gurobipy' package. The running time of RESPECT slowly increases with the coverage as the size of **P** (and hence the size of optimization problem at each iteration) scales with the (initial) estimate of coverage. On average, for a typical 0.5X-4X coverage of genome-skims, it takes about 2 hours for RESPECT algorithm to converge and produce the final estimate of the parameters.

Selecting species with known recent WGD events

From the total of 83 RefSeq genomes in our database, we obtained the WGD annotation (with estimated age) for 44 plant species [124]. WGD annotations for the remaining 32 plant species in our database were based on the data provided by the 1000 plants project [125], where either the exact same species or a species from the same genus is identified to have undergone a WGD event using transcriptomic data. We also have 7 Salmonid genomes where their common ancestor is thought to have had a WGD event about 80My ago [123].

Statistical analysis of the repeat structure

In a random genome with length L, there are $L-k+1 \simeq L$ k-mers, and assuming the random selection of k-mers is uniform over the space of all 4^k possible k-mers, the probability distribution for the copy number (CN) of each k-mer is

$$\text{Prob}[\text{CN} = x] = \binom{L}{x} (\frac{1}{4^k})^x (1 - \frac{1}{4^k})^{L-x} .$$

For typical values of $L \sim 100 - 1000$ Mbp and $k = 31$, the conditions to use a Poisson distribution to approximate a Binomial (see e.g., Section 5.4 of [132]) are met, i.e., $L \gg 1$ and $4^{-k} \ll 1$, hence we have

$$\text{Prob}[\text{CN} = x] = e^{-L/4^k} \frac{(L/4^k)^x}{x!} .$$

If the genome subsequently undergoes n_w whole genome duplication events, the genome length is multiplied by 2^{n_w}. However, the multiplicity of each k-mer increases by at most 2^{n_w}, as mutations reduce the copy number of k-mers. Therefore, to have an HCRM value of H, there should exist at least a k-mer with copy number $x \geq HL$ in the original random genome. Now, considering that under random-genome model the selection of *any* k-mer is equally likely, we can use the union bound (see e.g., Section 1.5 of [132]) and have

$$\begin{aligned} \text{Prob}[\text{HCRM} \geq H] &< \sum_{\substack{\text{all} \\ k\text{-mers}}} \sum_{x=HL} e^{-L/4^k} \frac{(L/4^k)^x}{x!} \\ &< 4^k \sum_{x=HL} e^{-L/4^k} \frac{(L/4^k)^x}{x!} \,. \end{aligned} \tag{4.10}$$

We used WolframAlpha [133] to compute the bound in (4.10) for several values of H. For $H = 200$ and $L \in [100 - 1000]$ Mbp, the resulting p-values were less than 10^{-100}.

To test the association between WGD events and the values of r_1/L and HCRM, we used the assembled genomes of 622 RefSeq species and constructed a two by two contingency table where columns represent the species with or without an identified recent WGD, and the rows specify whether or not the genome has r_1/L and HCRM values less than 0.8 and 200, respectively. We filled the table by the count of genomes that satisfied each of these four conditions, and performed a Fisher's exact test (using R 'stats' package [126]) and got the p-value $= 1.8 \times 10^{-23}$ for the correlation between the rows and columns of the table.

4.4 Discussion

In this chapter, we revisited the problem of estimating genomic parameters (length, sequence coverage, k-mer spectra) based on low coverage shotgun sequencing data. The problem has been studied previously and was considered challenging due to the need for simultaneous inference of coverage and sequencing errors along with the k-mer spectra. However, our results

suggest that the problem remains challenging even when there is no error and the coverage is known. This is due to two factors. (a) The linear system is ill-conditioned, so that a small change in the k-mer counts due to random sampling can lead to large changes in the estimated k-mer spectra (b) Values in the k-mer spectra show a skewed and non-sparse distribution, where r_1 dominates; r_1 is important for length estimation, but controlling for small errors in r_1 leads to larger errors in the other r_h values. We provide evidence of both, but future work will clarify the importance of each facet of the identification.

Proposed solutions for ill-conditioning use regularization but those methods generally enforce sparse solutions. However, the true k-mer distribution is not sparse. Our work resolved this issue through an empirical estimation of k-mer ratios based on finished genomes. This approach is viable given the many finished genomes with different repeat characteristics. Our study, with 662 genomes of which around 10% were isolated for testing, is the largest empirical study of its kind.

As expected, accurately estimated k-mer spectra led to better estimation of genomic parameters such as length, with Skmer-genome performing significantly better than the previous best method, sometimes by orders of magnitude. Our results also have lower variance than those of other methods.

As coverage increases, all methods perform well. However, at coverage 8X and higher, partial assemblies are possible and small contigs can start to be assembled. In those cases, alternative methods to estimate genome lengths may be possible, but our methods work well even for 0.5X coverage.

SSWe had used every genome for which the assembled sequence and the raw-reads were available at the time of submission. Recently, new data has been been released, and we tested our method on 10 additional samples with very similar performance (Supplementary Fig C.28).

The presence of contaminants is a significant barrier to accurate estimations, and in fact is challenging even for assembling the data. As data sampling and DNA extraction methods improve, this problem will likely be less problematic. In parallel, we are also working to improve

computational approaches to removing contamination.

While most k-mer based statistics were developed as an initial first step prior to deep sequencing and assembly, they may have an important role to play in independent analysis of genomes. Many genomes are $\leq$ 1Gb or lower. Therefore acquiring genome-skims for a majority of organisms and even multiple individuals in a population is a feasible goal. Methods that work on these reduced representations can be transformative for studying dramatic and short-term changes in bio-ecology. We can envision technologies where a sampled individual's genome-skim can be used to quickly estimate its genome-length, repeat structure, remove contaminating reads, identify the organism or place it confidently in the tree of life, and finally, identify the robustness of population through analysis of heterozygosity. Our work contributes to the first step of this vision.

Chapter 4, in full, is a reprint of the material as it appears in PLOS Computational Biology 17(11): e1009449. "Estimating repeat spectra and genome length from low-coverage genome skims with RESPECT". Shahab Sarmashghi, Metin Balaban , Eleonora Rachtman, Behrouz Touri, Siavash Mirarab, and Vineet Bafna. The dissertation author was the primary investigator and author of this paper.

Appendix A

Supplementary material: ISTAT

A.1 Generalized overlap

We can be more strict about declaring an intersection by accepting only those overlaps which include z or more base pairs (units). The dynammic programming algorithm and Poisson binomial approximation can both be easily generalized for that:

A.2 DP algorithm

For $c(i,h)$, the intersection conditions should change to $j_1 \leq h-z$ and $j_2 \geq h-l_i+z$, which can be compressed into the single condition $\min\{j_2,h\}-\max\{j_1,h-l_i\} \geq z$. For $f(h)$ we need to modify the defintion of "span", so interval (j_1,j_2) spans h if $j_1 \leq h-z$ and $j_2 \geq h+z$, which allows it to have the opportunity of overlap (under this new criteria) with both intervals starting and ending at h.

A.3 Poisson binomial approximation

In this case, p_{ij} is given by

$$p_{ij} = \begin{cases} 0 & \text{if } z > \min\{x_j, l_i\} \\ \frac{l_i+x_j-2z+1}{g} & \text{Otherwise} \end{cases}$$

A.4 Dynamic programing with disjointed Intervals

As described earlier, for i-th interval in I_r, genomic location h, $(1 \leq h \leq g)$, $0 \leq k \leq m$, $a \in 0,1$, let $N(i,h,k,a)$ denote the number of arrangements of the first i intervals in I_r such that (See Figure 2.2):

- The i-th interval ends exactly at location h.
- k intervals in I_f are hit by the first i intervals in I_r.
- $a = 0$ if the interval from I_f that spans j (if any) has not been counted earlier; $a = 1$ otherwise.

We also define $N_1(i,h,k,a)$ identically to $N(i,h,k,a)$ with the exception that the i-th interval ends at or before location h. If we consider the restriction that the intervals in I_r must be disjoint, which means that for any ordered pair of intervals (i_1,i_2) and (i_3,i_4), i_2 has to be strictly less than i_3, then the recurrence relation for $N(i,h,k,a)$ has to be modified as:

$$N(i,h,k,a) = \begin{cases} 0 & h < \sum_{x=1}^{i} l_x + i - 1 \text{ or } k < c(i,h) - a \\ 1 & i = 1 \text{ and } k = c(i,h) - a \\ N_1(i-1, h-l_i-1, k-c(i,h)+a, \min\{f(h-l_i-1), f(h-l_i)\}) & \text{Otherwise} \end{cases}$$

A.5 Log-scale computations

Let $a = \log A$ and $b = \log B$, then the following simple math trick enables us to calculate $c = \log(A \pm B)$ without explicitly converting a and b to their intractably large counterparts $A = \exp(a)$ and $B = \exp(b)$

$$c = \begin{cases} a + \log(1 \pm \exp(b-a)) & \text{if } b > a \\ b + \log(1 \pm \exp(a-b)) & \text{if } a > b \end{cases}$$

As a matter of fact, this trick is useful when A and B are both large, but the ratio $\frac{A}{B} = \exp(a-b)$ is computable, which is the case in the recurrence relation given by Eqn. 2.2.1. In fact, as we proceed along the four dimensions of $N_1(.,.,.,.)$, configurations accumulate and their count increases gradually. Therefore, whenever two numbers are added, their ratio is within the admissible range, even if their absolute values are not. The multiplication and division can be also done trivially.

A.6 The null model with multiple chromosomes

Consider Q chromosomes. For arbitrary chromosome q, let $I_q \subseteq I$ and $I_{f,q} \subseteq I_f$ denote the subsets of intervals paced on q, containing n_q and m_q intervals, respectively. Similarly, we can define $I_{r,q}$ to be a random reordering of I_q on chromosome q. Let $N_q(k_q)$ denote the number of configurations of intervals in $I_{r,q}$ s.t. $|I_{f,q} \sqsubseteq I_{r,q}| = k_q$. Using dynamic programming on each of Q chromosomes, we can obtain $N_q(k_q)$ $1 \leq q \leq Q, 0 \leq k_q \leq m_q$. For $k \in [0,m]$ we define the p-value to be

$$P\text{-value}(k) = \Pr(\sum_{q=1}^{Q} k_q \geq k).$$

With the equiprobability assumption and using simple arguments based on multiplication principle to count the number of desired configurations, we can compute the p-value as

$$P\text{-value}(k) = \frac{\sum_{(k_1,k_2,\ldots,k_Q)\in T_k} \prod_{q=1}^{Q} N_q(k_q)}{\sum_{(k_1,k_2,\ldots,k_Q)\in T_0} \prod_{q=1}^{Q} N_q(k_q)},$$

where T_k is the set of all Q-tuples $(k_1,k_2,\ldots,k_Q)$ such that $\sum_{q=1}^{Q} k_q \geq k$. While the denominator can be easily computed via the following identity

$$\sum_{(k_1,k_2,\ldots,k_Q)\in T_0} \prod_{q=1}^{Q} N_q(k_q) = \prod_{q=1}^{Q} \sum_{k_q=0}^{m_q} N_q(k_q),$$

it is not efficient to iterate over T_k to compute the numerator for each k. Instead, we use a simple recursive procedure to compute it. Let $M(q,k)$ be the number of configurations that the first q chromosomes have k intersections. The p-value can be expressed in terms of $M(q,k)$ as

$$P\text{-value}(k) = \frac{\sum_{\kappa=k}^{m} M(Q,\kappa)}{\sum_{\kappa=0}^{m} M(Q,\kappa)}.$$

The following recurrence relation lets us to efficiently compute the p-value for all $k \in [0,m]$

$$M(q,k) = \sum_{l=0}^{\min\{k,m_q\}} M(q-1,k-l)N_q(l)$$
$$M(q,0) = \prod_{u=1}^{q} N_u(0), \quad M(1,k) = N_1(k)$$

where the time complexity is $\mathcal{O}(Qm^2)$. Nevertheless, the total time complexity of calculating the p-value is definitely dominated by the compexity of applying DP algorithm to each chromosome to compute all $N_q(k_q)$. As DP algorithm on each chromosome is done independently, we can take advantage of parallel computing and the total running time would be $\mathcal{O}(\max_q\{n_q g_q m_q\})$.

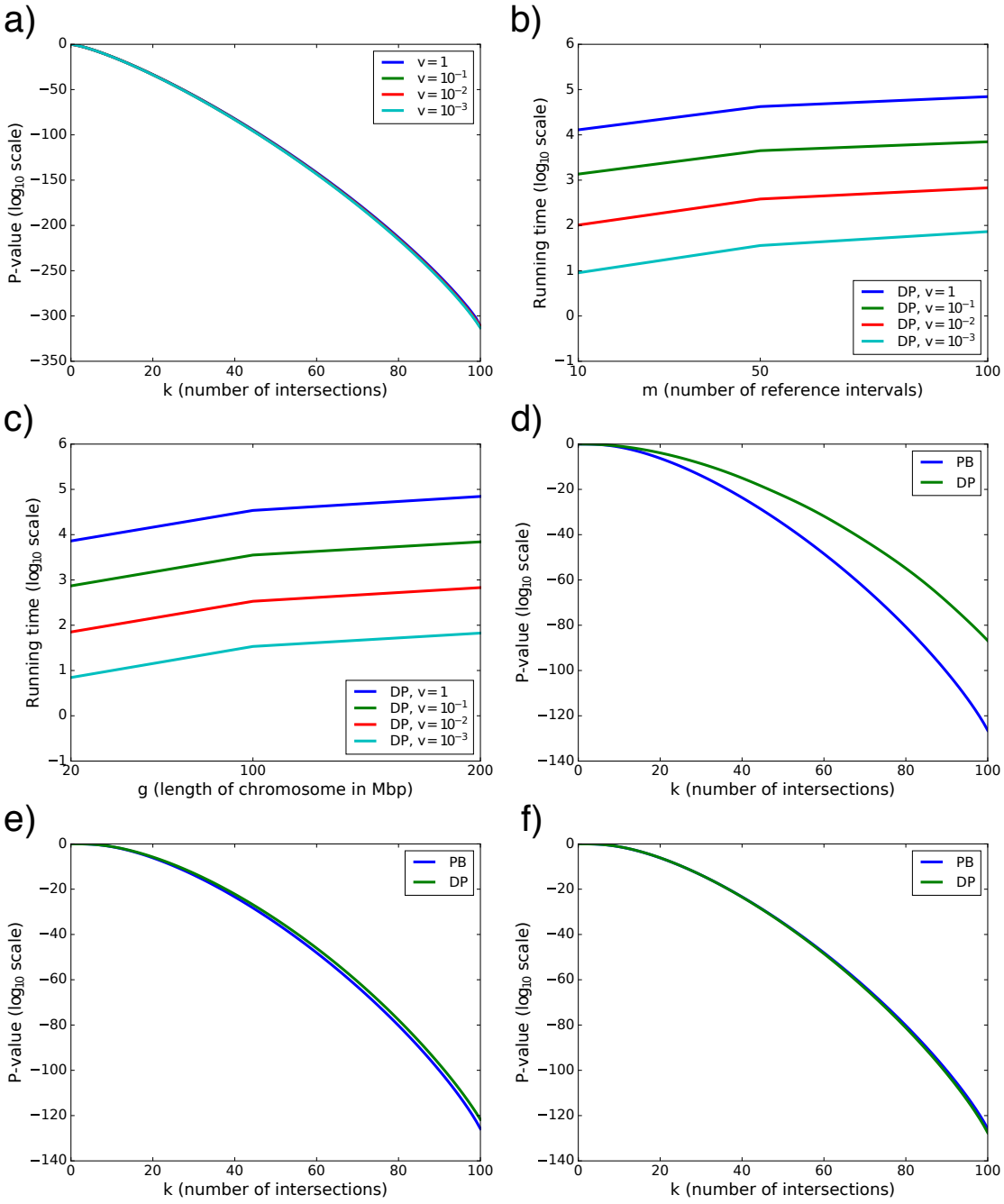


Figure A.1. Further evaluation of methods using simulated datasets. (a) Impact of scaling parameter ν on DP p-value when $l_i, x_j \sim \mathscr{U}[100\text{bp}, 4\text{Kbp}]$. (b) Running time (secs.) of DP algorithm as a function of m, with $n = 100$, $g = 200\text{Mbp}$. (c) Running time (secs.) of DP algorithm as a function of g, with $n = m = 100$. (d–f) Impact of approximation on p-value computation. Simulations are run with $g = 200\text{Mbp}$, $m = 100$, $n = 1000$, $l_i, x_j \sim \mathscr{U}[1\text{Kbp}, 2\text{Kbp}]$; (d) $\eta = 0.0079$. (e) $\eta = 0.062$. (f) $\eta = 0.68$.

Appendix B

Supplementary material: Skmer

B.1 Theoretical results

Consider two genomes of identical length L and separated by hamming distance D where the hamming distance is defined as the fraction of variant sites between the perfect alignment of the two genomes. We would like to estimate D from two genome-skims.

Mutations

We model the two genomes as the outcome of a random process that copies a genome and introduces mutations at each position i.i.d with a fixed probability d. Indexing from left to right, we can define $n = L - k + 1$ k-mers (note that $n \approx L$ for any reasonable choice of k and genome length). Let X_i be a binary random variable (r.v.) that indicates whether k-mer i is identical between the two genomes. Clearly, in our model, $X_i \sim \text{Bern}(p)$ where $p = (1-d)^k$. Then, $W = \sum_1^n X_i$ gives the number of shared k-mers. If J is defined as the Jaccard index over the set of all k-mers from both genomes, it's easy to see that $J = \frac{W}{2n-W}$ and thus, $\frac{W}{n} = \frac{2J}{1+J}$. We further make a simplifying assumption. We assume all X_i r.v.s are independent, an assumption that is true for most pairs of k-mers but ignores the fact that each k-mer overlaps with k-1 other k-mers. With this assumption, the maximum likelihood estimate of p is simply

$$\hat{p} = \frac{W}{n} = \frac{2J}{1+J}\,.$$

By the functional invariance of maximum likelihood, the ML estimate of d is given by:

$$\hat{d} = 1 - \left(\frac{2J}{1+J}\right)^{\frac{1}{k}} .$$

k-mer sampling

We now assume that each genome is covered uniformly at random. Thus, k-mers are also sub-sampled and we assume each k-mer is sampled at least once with probability η_1 in the first genome and η_2 in the second genome; we derive the relationship between these probabilities and genome coverage below. We estimate η values separately (also described below) and here consider them as given. For each $1 \leq i \leq n$ and $j \in \{1,2\}$, let $Y_{j,i} \sim \text{Bern}(\eta_j)$ be the indicator of whether the k-mer i is sampled at least once in the genome j. Under this scenario, the number of k-mers shared between the two genomes is given by the r.v. $W = \sum_1^n X_i Y_{1,i} Y_{2,i}$. Defining $Z = X_i Y_{1,i} Y_{2,i}$, we get $W = \sum_1^n Z_i$ and $Z_i \sim \text{Bern}(r)$ where $r = p\eta_1\eta_2$ by the independence of the mutation process and each of the two k-mer sampling processes. Assuming independence between Z_i r.v.s (again ignoring the overlap between consecutive k-mers) we get the ML estimate $\hat{r} = \frac{W}{n}$, and thus (for a given η_1 and η_2) we have

$$\hat{r} = \hat{p}\eta_1\eta_2 = \frac{W}{n} \tag{B.1}$$

Let $U = \sum_1^n S_i$ where $S_i = Y_{1,i} + Y_{2,i} - Y_{1,i}Y_{2,i}X_i$. It is easy to see that U gives the total number of sampled k-mers in both genomes. However, S_i is not a Bernoulli and thus, U is not Binomial. Nevertheless, the same assumptions that we used to treat X_i and Z_i r.v.s as independent also give us independence between S_i values; therefore, by the central limit theorem, $\frac{U}{n}$ can be approximated by a Gaussian with mean $q = \mathbb{E}[S_i]$. Moreover, $\mathbb{E}[S_i] = \mathbb{E}[Y_{1,i}] + \mathbb{E}[Y_{2,i}] - \mathbb{E}[Y_{1,i}Y_{2,i}X_i] = \eta_1 + \eta_2 - \eta_1\eta_2 p$ (note that X_i, $Y_{1,i}$ and $Y_{2,i}$ are independent).

By this Gaussian approximation, the ML estimate of q given η_1, η_2 is given by:

$$\hat{q} = \eta_1 + \eta_2 - \eta_1\eta_2\hat{p} = \frac{U}{n} \,. \tag{B.2}$$

Note that $J = \frac{W}{U}$. Equations B.1 and B.2 give two different ML estimators of the same parameter p given two different types of data (W and U). While the two estimators are not the same, because n is extremely large, both estimators have a very low variance. Exploiting the low variance, we treat the two estimates of p as equal and divide both sides of Equation B.1 by Equation B.2 to get:

$$\frac{\hat{r}}{\hat{q}} = \frac{W}{U} = J = \frac{\hat{p}\eta_1\eta_2}{\eta_1 + \eta_2 - \eta_1\eta_2\hat{p}} \,.$$

Solving for $\hat{p}$ and replacing $\hat{d} = 1 - \hat{p}^{\frac{1}{k}}$ gives

$$\hat{d} = 1 - \big(\frac{(\eta_1 + \eta_2)J}{\eta_1\eta_2(1+J)}\big)^{\frac{1}{k}} \,.$$

Note that we have assumed a known coverage and thus we are not co-estimating η_j's and d. In practice, we need to first estimate η_1 and η_2, and we do it as we will describe.

Connection of η to read coverage

A k-mer stretching from position y to $y+k$ on the genome is covered by the reads that start in the interval $[y+k-\ell, y]$. Assuming that there is no sequencing error, and a uniform spread of of the N reads across the genome of length L. We show that the probability η that a k-mer is sampled by at least one read is given by

$$\eta = 1 - e^{-c(1-\frac{k}{\ell})}$$

Let X be a r.v. denoting the number of reads that cover a specific k-mer. Assuming a uniform spread of N reads across the genome of length L, the probability of x reads covering a

k-mer (starting in an interval of length $\ell - k$) is given by

$$Prob(X = x) = \binom{N}{x} (\frac{l-k}{L})^x (1 - \frac{l-k}{L})^{N-x}$$

As N is large and $\frac{N(l-k)}{L}$ is constant, it can be closely approximated by

$$Prob(X = x) = \frac{\lambda^x}{x!} e^{-\lambda}$$

where $\lambda = \frac{N(l-k)}{L}$ is the *k-mer* coverage, and is related to the coverage c by

$$\lambda = \frac{l-k}{l} c$$

As the number of reads covering a k-mer follows Poisson distribution, the fraction of k-mers covered by 1 or more reads is

$$\eta = 1 - e^{-\lambda} \tag{B.3}$$

Sequencing error

We model the sequencing error as an i.i.d process that corrupts each position of each read with a fixed probability ε. To extend our previous results to cover this scenario, we need to see how the intersection r.v. (W) and the union r.v. (U) get affected.

We start with the intersection (W). We change the meaning of η to denote the probability that a k-mer is covered by at least one error-free read. The probability of a k-mer within a read being error-free is clearly

$$\rho = (1 - \varepsilon)^k \simeq e^{-k\varepsilon} \tag{B.4}$$

By conditioning on the number of reads covering a k-mer, the probability of not covering

a k-mer with an error-free read is given by

$$\begin{aligned} Prob(\text{no error-free read}) &= \sum_{i=0}^{\infty} Prob(\text{all reads have error}|i \text{ reads})\, Prob(i \text{ reads}) \\ &= \sum_{i=0}^{\infty} (1-\rho)^i\, Prob(i \text{ reads}) \\ &= \sum_{i=0}^{\infty} (1-\rho)^i \frac{\lambda^i}{i!} e^{-\lambda} \\ &= e^{-\lambda\rho} \end{aligned} \tag{B.5}$$

Hence, the probability that a k-mer is covered by at least one error-free read is given by

$$\eta = 1 - e^{-\lambda\rho} \tag{B.6}$$

Note that Eqn. B.6 reduces to Eqn. B.3 when there is no sequencing error, i.e., $\rho = 1$. Similar to the case of no error, given η_1 and η_2, the r.v. $\frac{W}{n}$ (where W is the number of shared k-mers) can be used with Equation B.1 to estimate r.

We now turn to the union (r.v. U). For large enough k, and for genomes that are random and repeat-free, with high probability ($> 1 - \frac{2L}{4^k}$) an error produces a new k-mer that is not observed in either of the input genomes. Ignoring the exceedingly unlikely event that two errors produce the same k-mer or that they produce a k-mer present in one of the two genomes, we can assume that the sequencing error generates as many new k-mers as the number of reads being affected by errors.

In the regime that includes errors, $U = \sum_1^n (T_{1,i} + T_{2,i}) - W$ where the r.v.s $T_{1,i}$ and $T_{2,i}$ give the total number of k-mers generated from the position i from the first and second genomes, respectively. W.l.o.g, consider $T_{1,i}$. By conditioning on the number of reads covering a k-mer we have

$$\mathbb{E}[T_{1,i}] = \mathbb{E}[\mathbb{E}[T_{1,i}|x \text{ reads}]] = \sum_{x=0}^{\infty} \mathbb{E}[T_{1,i}|x \text{ reads}]\, Prob(x \text{ reads}) \tag{B.7}$$

Given that x reads are covering a k-mer, $T_{1,i}$ equals the number of erroneous k-mers E, plus 1 if there is any error-free k-mer. As $E \sim Binom(x, 1-\rho)$

$$\begin{aligned}\mathbb{E}[T_{1,i}|x \text{ reads}] &= \sum_{j=0}^{x} (j + \mathbf{1}_{j\neq x}) \binom{x}{j} (1-\rho)^j \rho^{x-j} \\ &= x(1-\rho) + (1-(1-\rho)^x)\end{aligned} \tag{B.8}$$

and substituting into (B.7)

$$\begin{aligned}\mathbb{E}[T_{1,i}] &= \sum_{x=0}^{\infty} ((1-(1-\rho)^x) + x(1-\rho))\, Prob(x \text{ reads}) \\ &= \sum_{x=0}^{\infty} ((1-(1-\rho)^x) + x(1-\rho)) \frac{\lambda_1^x}{x!} e^{-\lambda_1} \\ &= 1 - e^{-\lambda_1 \rho} + \lambda_1(1-\rho) \\ &= \eta_1 + \lambda_1(1-\rho) \\ &= \eta_1 + \lambda_1(1-(1-\varepsilon)^k)\end{aligned} \tag{B.9}$$

Letting $\zeta_1 = \mathbb{E}[T_{1,i}]$ and using the same central limit argument we used before, $\frac{U}{n}$ becomes approximately a Gaussian with expectation $\zeta_1 + \zeta_2 - \eta_1\eta_2 p$. Similar to Equation B.2, given ζ_1, ζ_2, η_1, and η_2, the Gaussian approximation gives us:

$$\zeta_1 + \zeta_2 - \eta_1\eta_2\hat{p} = \frac{U}{n}\,. \tag{B.10}$$

Again, assuming that estimates of p in Equation B.1 (with the new definition of η) and Equation B.10 are the same (due to low variance), we divide the two equations and solve for d to get the estimator:

$$D = 1 - \left(\frac{(\zeta_1+\zeta_2)J}{\eta_1\eta_2(1+J)}\right)^{1/k}\,.$$

Excluding low-copy k-mers from the Jaccard index calculation

If we discard k-mers observed less than m times, then a k-mer will survive if it is covered by m or more error-free reads. Hence, η becomes the probability of m or more error-free reads covering a k-mer

$$\begin{aligned}
\eta &= 1 - \sum_{t=0}^{m-1} Prob(t \text{ error-free read}) \\
&= 1 - \sum_{t=0}^{m-1} \sum_{i=t}^{\infty} Prob(t \text{ error-free read} | i \text{ reads}) \, Prob(i \text{ reads}) \\
&= 1 - \sum_{t=0}^{m-1} \sum_{i=t}^{\infty} \binom{i}{t} p^t (1-p)^{i-t} \frac{\lambda^i}{i!} e^{-\lambda} \\
&= 1 - \sum_{t=0}^{m-1} \frac{(\lambda p)^t}{t!} e^{-\lambda p}
\end{aligned} \tag{B.11}$$

In general, we have shown that the probability distribution of the number of error-free k-mers is a Poisson with parameter λp.

B.2 Computing GTR distances

To compute the GTR matrix using the log-det approach, we need a 4×4 matrix F where each element is the fraction of sites where one genome has one letter while the other genome has the other letter. Given this matrix, $d = -\log(\det(F))$.

As elsewhere, we assume a no-indel scenario so that each k-mer mismatch can be attributed to a single nucleotide substitution. For $i, j \in \{\text{A,C,G,T}\}$, let $x_{ij} = x_{ji}$ denote the number of mutations of the form $i \leftrightarrow j$. Our goal is to estimate x_{ij} for all i, j. However, the paradigm of computing distance by hashing/sketching k-mers treats all mutations alike. Formally, the estimated distance d equals

$$d = x_{\text{AC}} + x_{\text{AG}} + x_{\text{AT}} + x_{\text{CG}} + x_{\text{CT}} + x_{\text{GT}}$$

We do the following:

1. Replace G and T with C, and compute distance $d_{\text{A}} = x_{\text{AC}} + x_{\text{AG}} + x_{\text{AT}}$.
2. Replace G and T with A, and compute distance $d_{\text{C}} = x_{\text{AC}} + x_{\text{CG}} + x_{\text{CT}}$.
3. Replace G with T, and compute distance $d_{\text{AC}} = x_{\text{AC}} + x_{\text{AG}} + x_{\text{AT}} + x_{\text{CG}} + x_{\text{CT}}$.

Combining, we get

$$x_{\text{AC}} = d_{\text{A}} + d_{\text{C}} - d_{\text{AC}}$$

A similar procedure can be used to compute all x_{ij} and normalization gives us F.

Note that this procedure reduces the space of possible k-mers of length k to 2^k possibilities instead of 4^k. Therefore, it will likely be required that k is increased for high accuracy when this approach is used.

B.3 Supplementary method details and commands

Here we provide the exact procedures and commands that we used to run external softwares throughout our experiments.

Simulating genome-skims using ART

To simulate short reads with length $\ell = 100$ and (default) error profiles of Illumina HiSeq2000, we ran

```
art_illumina -i FASTA_FILE -l 100 -f c -o FASTQ_FILE
```

To simulate reads with constant error rate $\varepsilon = 0.01$ (Phred score $= 20$) at coverage c, we used

```
art_illumina -i FASTA_FILE -l 100 -qL 20 -qU 20 -f c -o FASTQ_FILE
```

Computing k-mer frequencies using JellyFish

To count all k-mers of length $k = 31$ in a genome-skim, we used

```
jellyfish count -m 31 -s 100M -C -o COUNT_FILE FASTQ_FILE
```

and to get the histogram of k-mer counts

```
jellyfish histo COUNT_FILE
```

Computing Jaccard index and estimating distance using Mash

We first *sketch* input genome-skims or assemblies with k-mer length $k = 31$ and sketch size $s = 10^7$. For genome-skims (in FASTQ format) when no k-mer filtering is applied, we run

```
mash sketch -r -k 31 -s 10000000 -o SKETCH_FILE FASTQ_FILE
```

To sketch genome-skims while filtering k-mers with less than C copies, we use

```
mash sketch -m C -k 31 -s 10000000 -o SKETCH_FILE FASTQ_FILE
```

For genome assemblies (in FASTA format), we used

```
mash sketch -k 31 -s 10000000 -o SKETCH_FILE FASTA_FILE
```

Then, the Jaccard index and Mash distance between sketches is computed by running

```
mash dist SKETCH_FILE_1 SKETCH_FILE_2
```

Estimating distances using AAF

To count the k-mers ($k = 31$) in a dataset of genome-skims using 24 cores and 120GB memory, we first ran

```
python PATH_to_FILE/aaf_phylokmer.py -k 31 -t 24 -o KMER_COUNT_FILE \
       -d INPUT_DIR -G 120
```

Next, to get the (uncorrected) distances and phylogeny, we used

```
python PATH_to_FILE/aaf_distance.py -i KMER_COUNT_FILE -t 24 -G 120 \
       -o OUTPUT_FILE_PREFIX -f KMER_DIVERSITY_FILE
```

where `KMER_DIVERSITY_FILE` is an output of previous command. Finally, to correct tip branches of phylogeny tree for low coverage and sequencing error, we used

```
python PATH_to_FILE/aaf_tip.py -i TREE_FILE -k 31 \
       --tip TIP_INFO_FILE -f KMER_DIVERSITY_FILE
```

where we had to provide `TIP_INFO_FILE` containing estimates of coverage and sequencing error. To estimate coverage, we followed the procedure suggested in AAF user manual. We first used JellyFish to find the k-mer counts M_i's as described before. They suggest when there is a clear peak in the k-mer frequency distribution, estimate k-mer coverage λ to be the maximum bin. As they do not suggest a specific rule for that, we first find $j = \text{argmax}_{i>1} M_i$, excluding the count of the first bin M_1, which is always large because of erroneous k-mers due to sequencing error. If $j > 2$, it means that we can see a peak in k-mers distribution at j, so we use $\lambda = j$.

Otherwise, if $j = 2$, we follow their suggested formula $\lambda = \frac{\sum iM_i}{\sum M_i}$ for the case of low coverage or high sequencing error that there is no clear peak in the k-mer frequency distribution. We should also mention that no k-mer filtering used for AAF, as the coverage was heterogeneous over genome-skims. In fact, in AAF the filtering is applied to all genome-skims if used, and so they suggest to not apply filtering when there is any taxon with low coverage ($c < 5$) within the dataset.

Preprocessing raw reads using fastp

We used the following command to filter low-quality reads and trim the adapter sequences

```
fastp -t 1 -i INPUT_READS_R1 -I INPUT_READS_R2 \
      -o OUTPUT_READS_R1 -O OUTPUT_READS_R2
```

Contamination removal

To remove bacterial and mitochondrial sequences, we first created a BLAST database from the assemblies of contaminant genomes by running

```
makeblastdb -in CONTAMINANTS_FASTA_FILE -dbtype nucl -out BLAST_DB
```

and then searched the reads against these genomes using Megablast

```
blastn -db BLAST_DB -query READS_FILE -outfmt 6 -out MEGABLAST_OUTPUT
```

We also used Bowtie2 to find the reads aligned to the human reference genome

```
bowtie2 -x HUMAN_REFERENCE -U READS_FILE \
        -S BOWTIE_OUTPUT --very-sensitive-local
```

We then removed any read found in `MEGABLAST_OUTPUT` or `BOWTIE_OUTPUT`.

B.4 Supplementary figures and tables

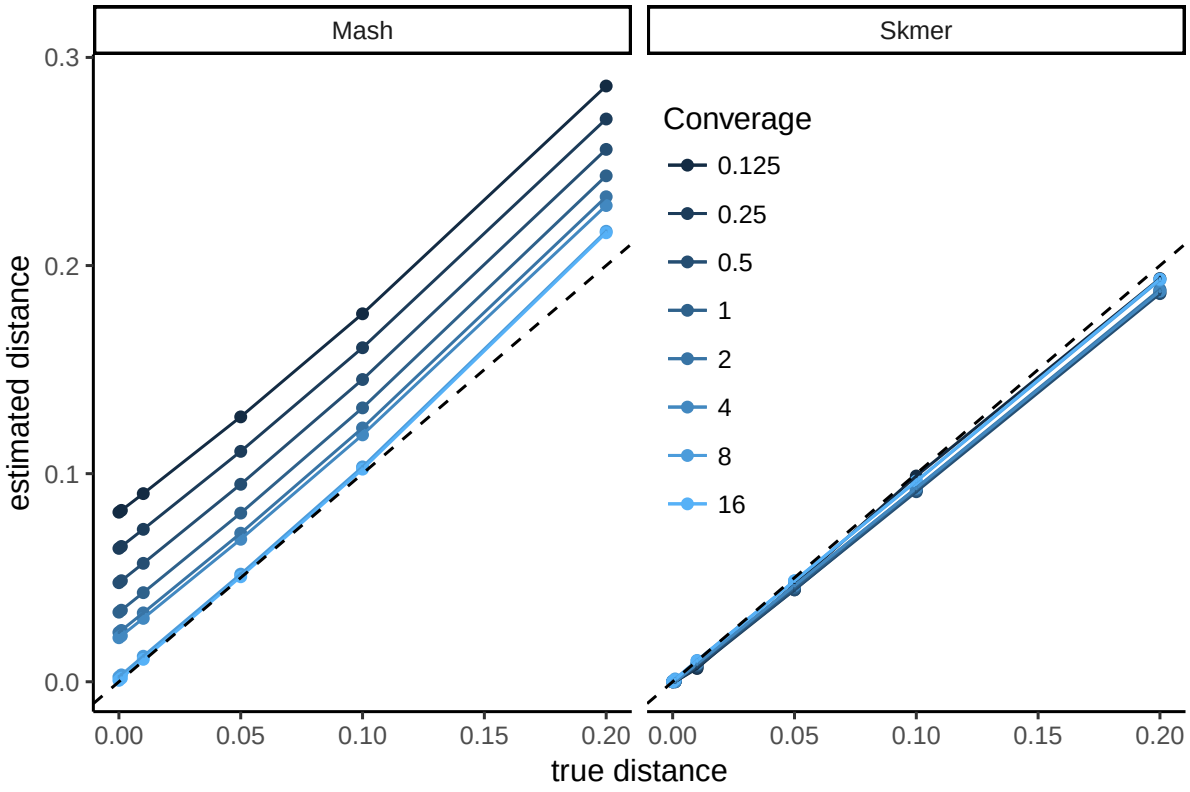


Figure B.1. Comparing the accuracy of Mash and Skmer on simulated genomes. Genome-skims are simulated using ART with read length $\ell = 100$. Substitutions applied to the assembly of *C. vestalis* at six different rates (x-axis), and genome-skims simulated at varying coverage range from $\frac{1}{8}$X to 16X (colors). The estimated distance (y-axis) by Mash (left) and Skmer (right) is plotted versus the real distances (x-axis). The mean (dots) distances are shown as dots (10 repeats) but standard errors are too small to see. The unit line is shown as a dashed line.

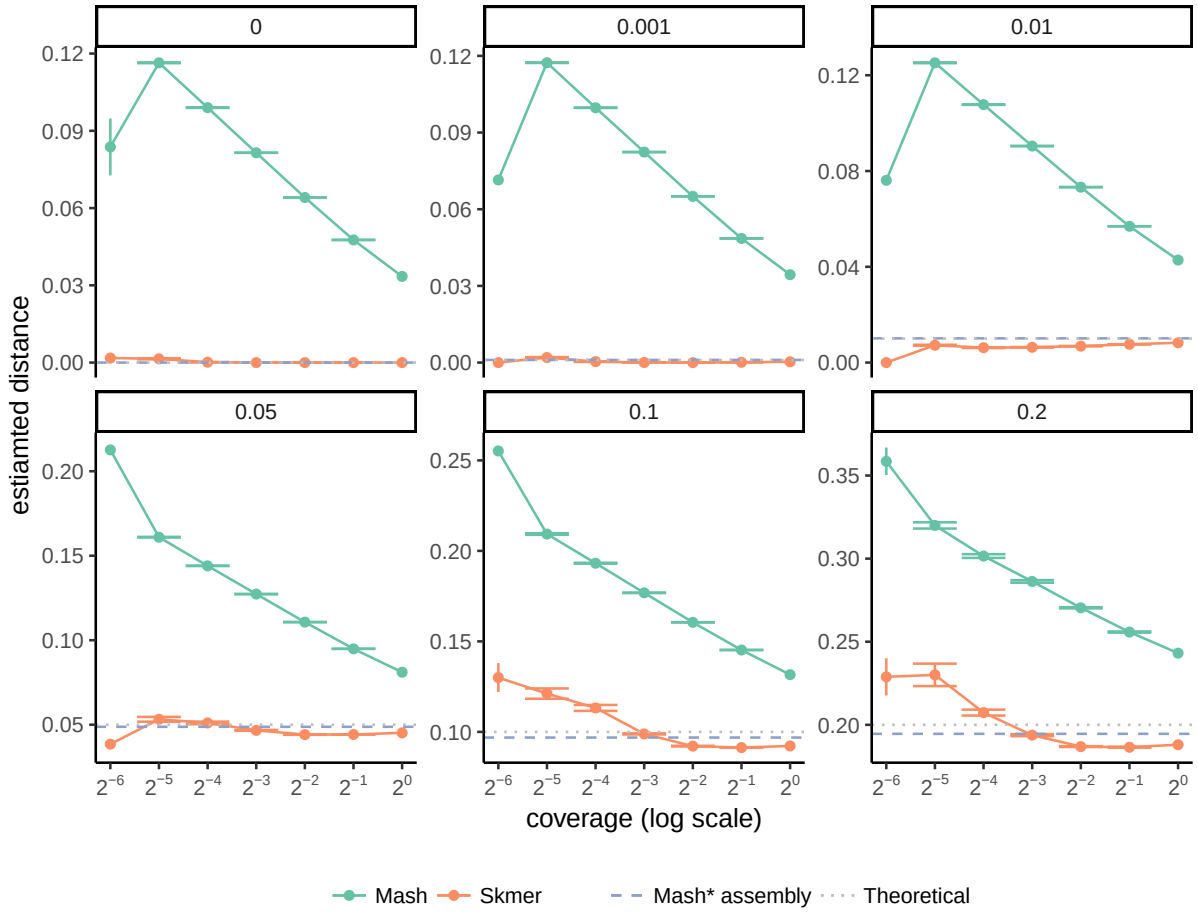


Figure B.2. Comparing distances estimated by Mash and Skmer for simulated data at very low coverages. Skims of *C. vestalis* v.s. genomes simulated to be at different distances from *C. vestalis*, with varying coverage. The mean and standard error of distances are shown over 10 repeats of the experiment. The coverage ranges from $\frac{1}{64}$X to 1X.

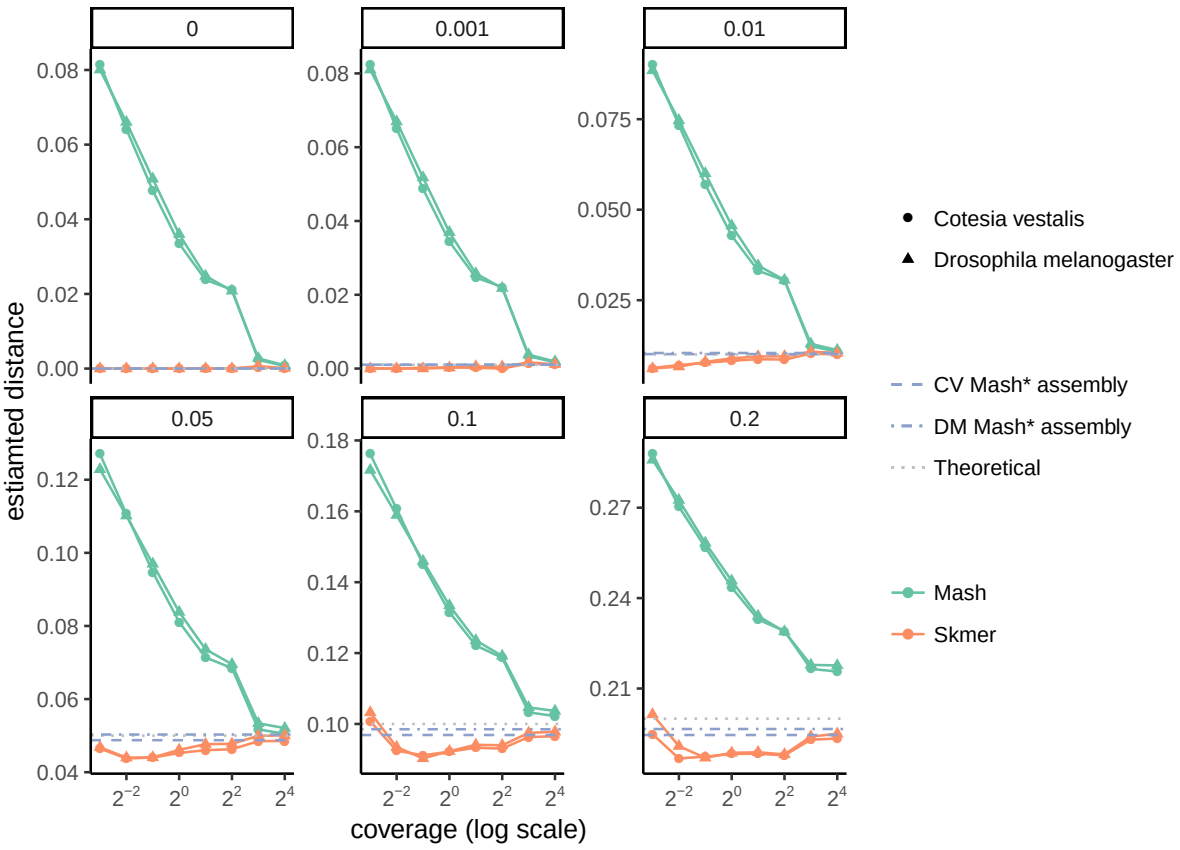
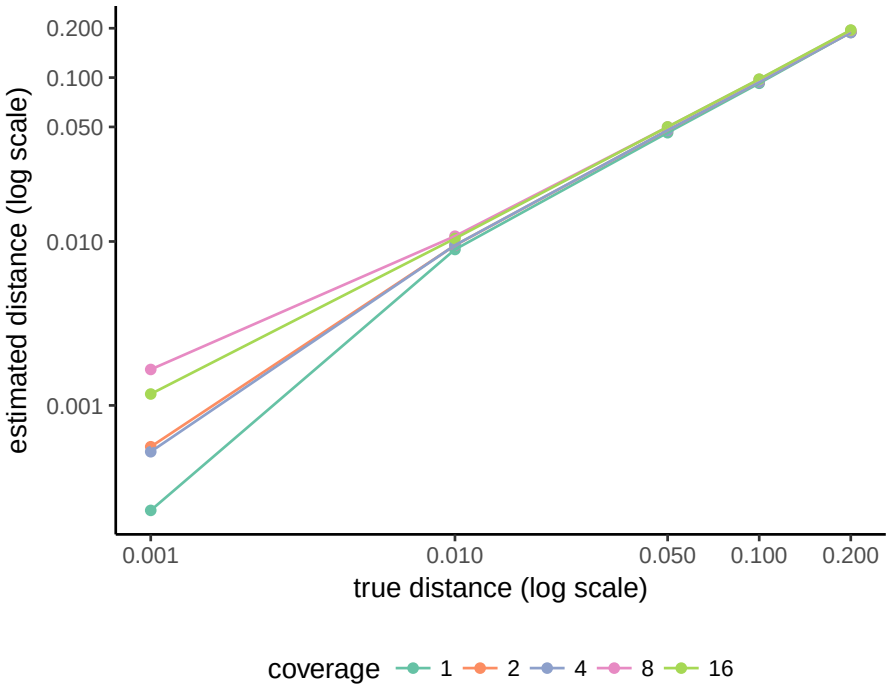
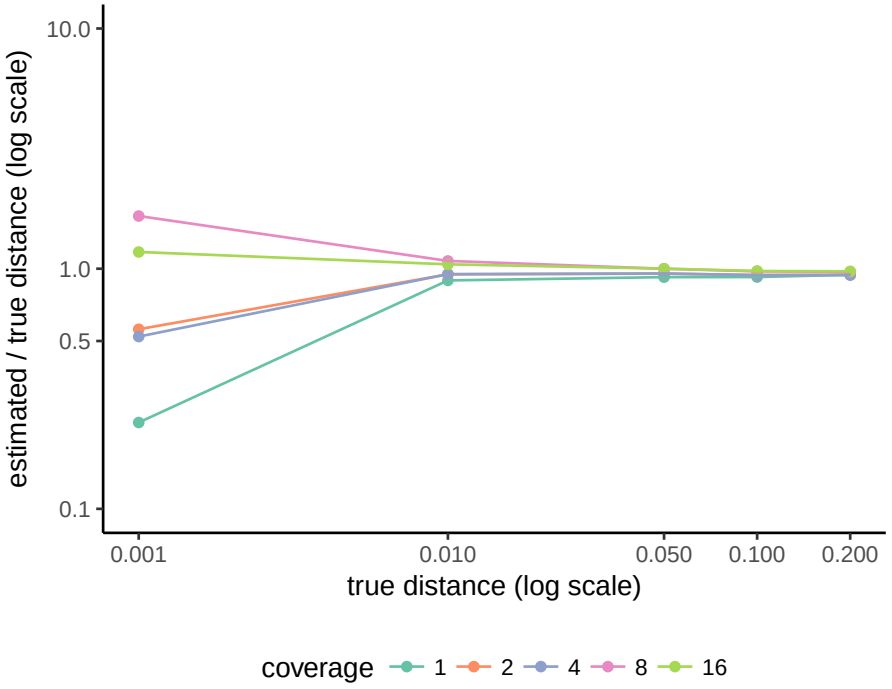


Figure B.3. Comparing distances estimated for genome-skims of two different species. Genomes simulated at different distances from the genomes of C. vestalis and D. melanogaster and subsampled at a range of coverage from $\frac{1}{8}$X to 16X.



(a)



(b)

Figure B.4. The resolution of Skmer at different genomic distances. Skims of *D. melanogaster* v.s. genomes simulated to be at different distances from *D. melanogaster*, with varying coverage. (a) Estimated distance versus the true distance. (b) The ratio of estimated distance to the true distance.

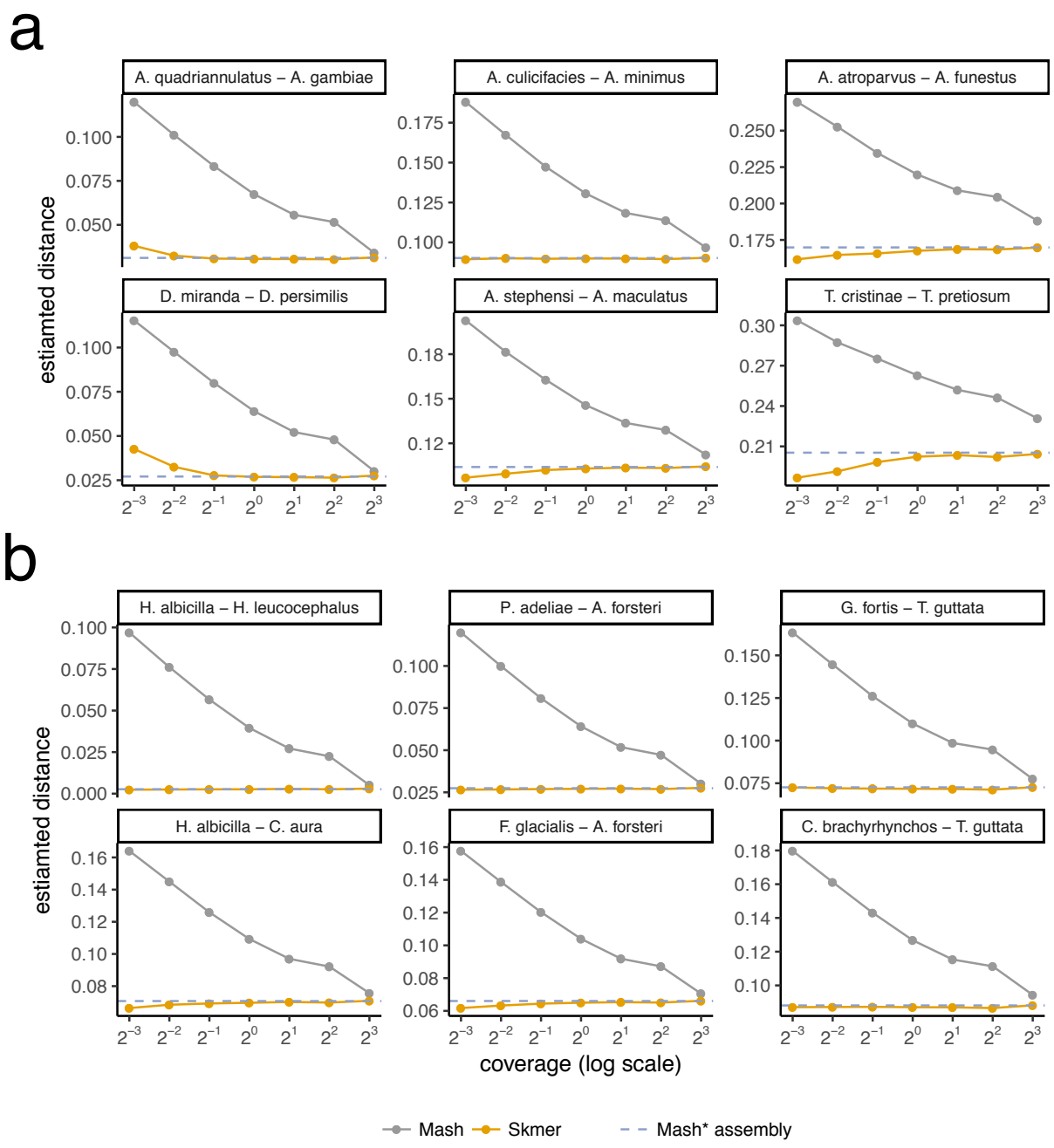


Figure B.5. Comparing the accuracy of Mash and Skmer on pairs of insects and birds genomes. Genome-skims simulated at coverage $\frac{1}{8}$X to 8X. On each subplot, the estimated distance (y-axis) is plotted versus the coverage (x-axis) for a pair of species. Dashed line shows Mash* run on assemblies, which is taken as the true distance. Skmer estimates (light-colored curves) are very close to the true distance while Mash (gray curves) largely overestimates at lower coverages. (a) Six pairs of insects. (b) Six pairs of birds.

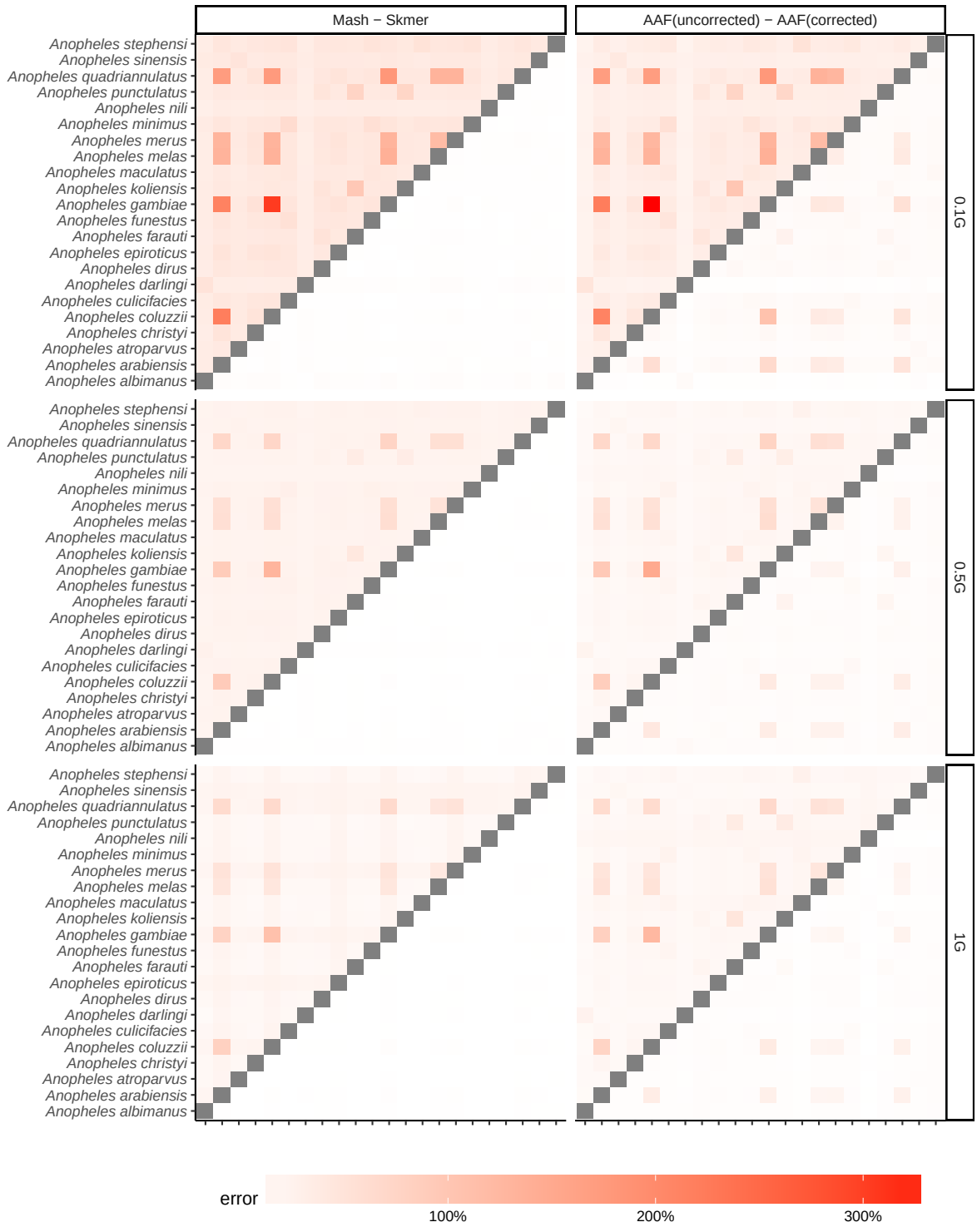


Figure B.6. Comparing the error of Mash, Skmer, and AAF in distance estimation with fixed amount of sequence from each species. The dataset of 22 Anopheles genomes, subsampled with 0.1Gb, 0.5Gb, and 1Gb sequence.

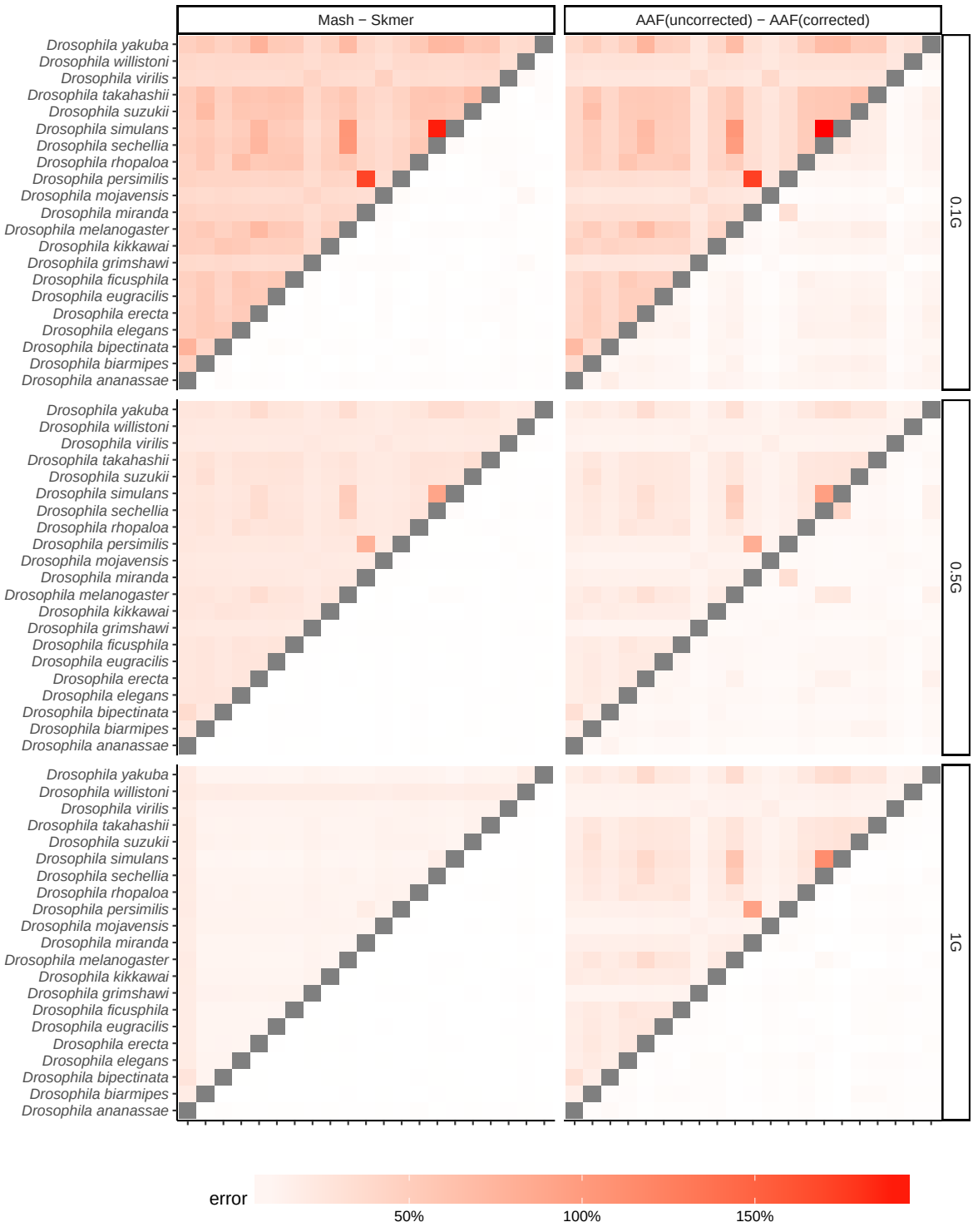


Figure B.7. Comparing the error of Mash, Skmer, and AAF in distance estimation with fixed amount of sequence from each species. The dataset of 21 Drosophila genomes, subsampled with 0.1Gb, 0.5Gb, and 1Gb sequence.

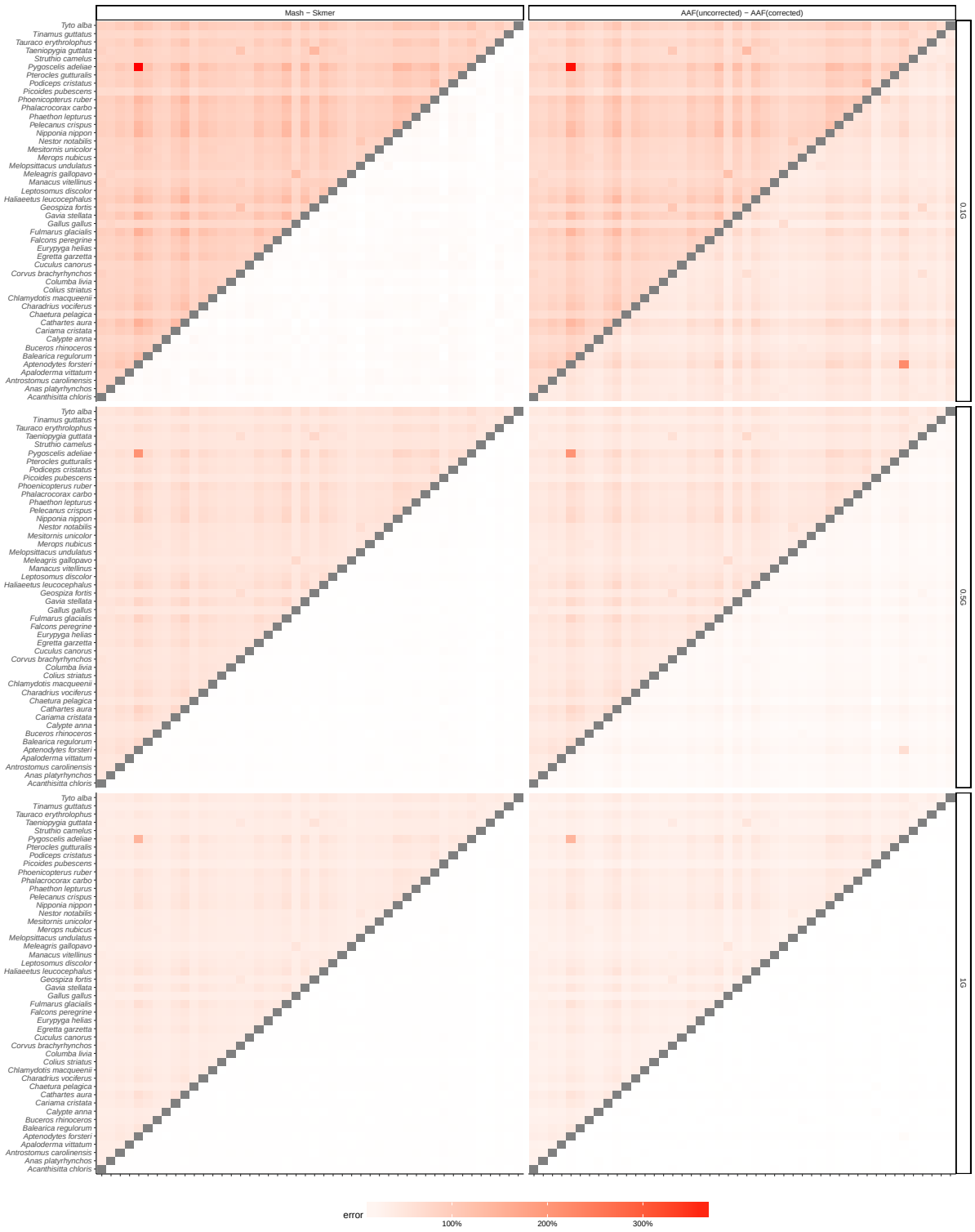


Figure B.8. Comparing the error of Mash, Skmer, and AAF in distance estimation with fixed amount of sequence from each species. The dataset of 47 avian genomes, subsampled with 0.1Gb, 0.5Gb, and 1Gb sequence.

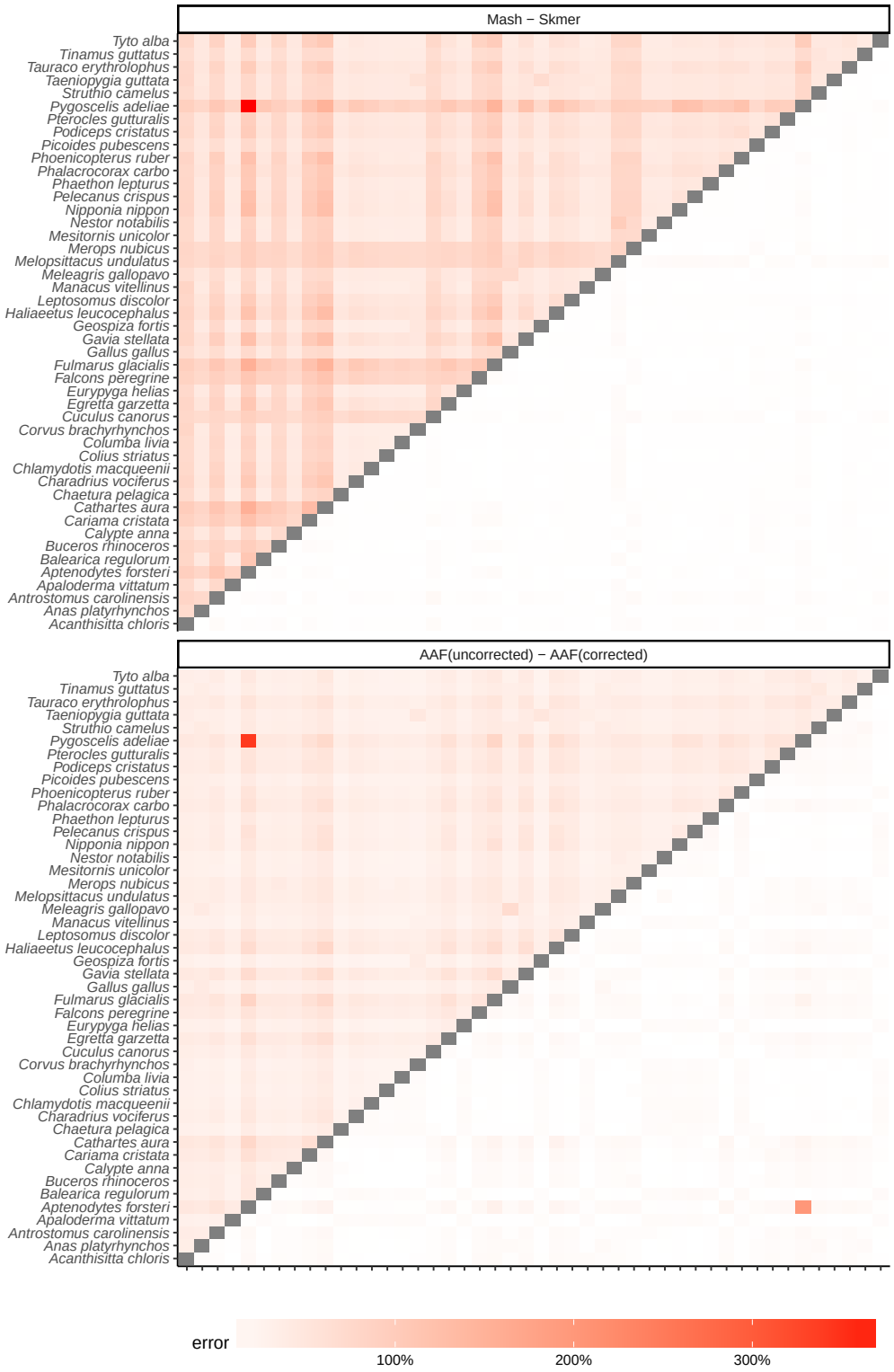


Figure B.9. Comparing the error of Mash, Skmer, and AAF on the Avian dataset with mixed coverage. Species have random amount of sequence chosen uniformly among 0.1Gb, 0.5Gb, and 1Gb. Similar to (Fig. 3.5), we have excluded one of the eagles (*H. albicilla*). The error of Mash, AAF, and Skmer in estimating the distance between the two eagles are 2193%, 884%, and 4.2%, respectively (both of the eagles are subsampled at 0.5Gb here).

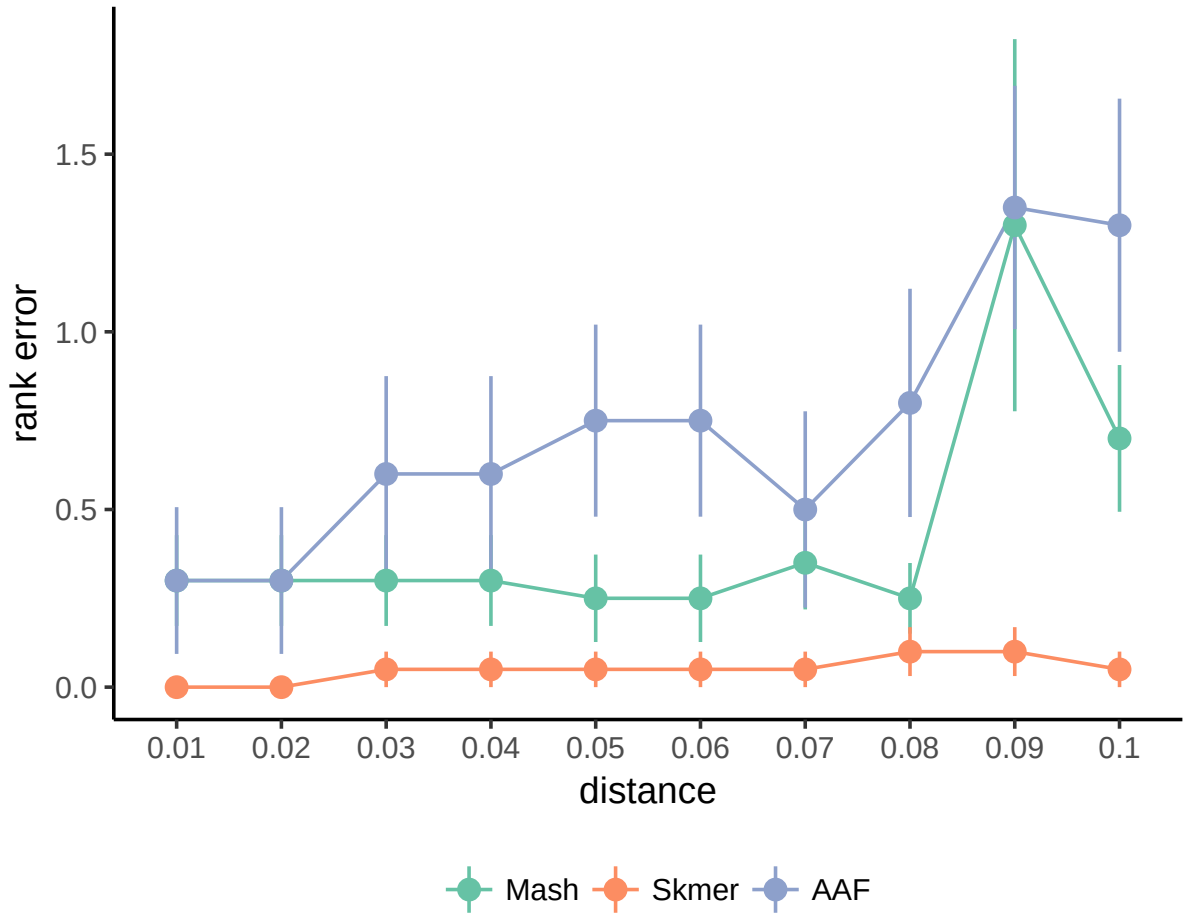


Figure B.10. The mean rank error of the best remaining match in leave-out experiments on the *Drosophila* dataset. *Drosophila willistoni* has been excluded.

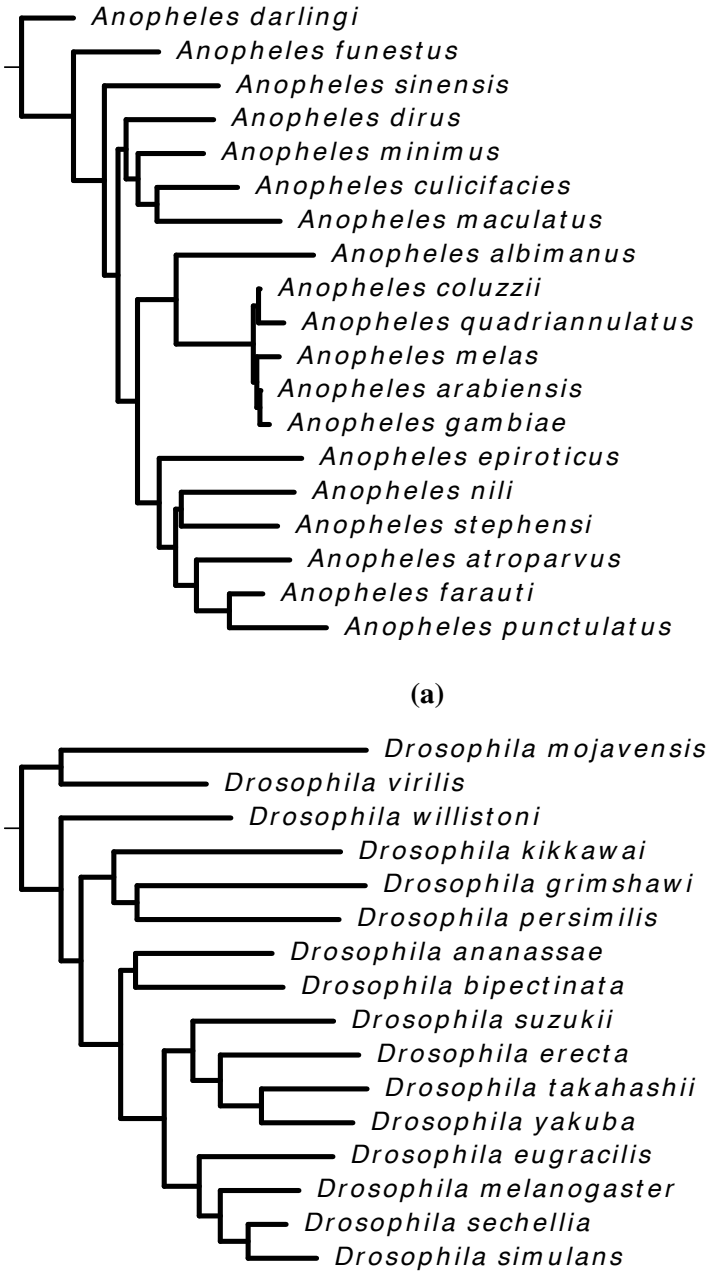


Figure B.11. Maximum-likelihood trees inferred from COI barcodes. (a) *Anopheles* tree. (b) *Drosophila* tree.

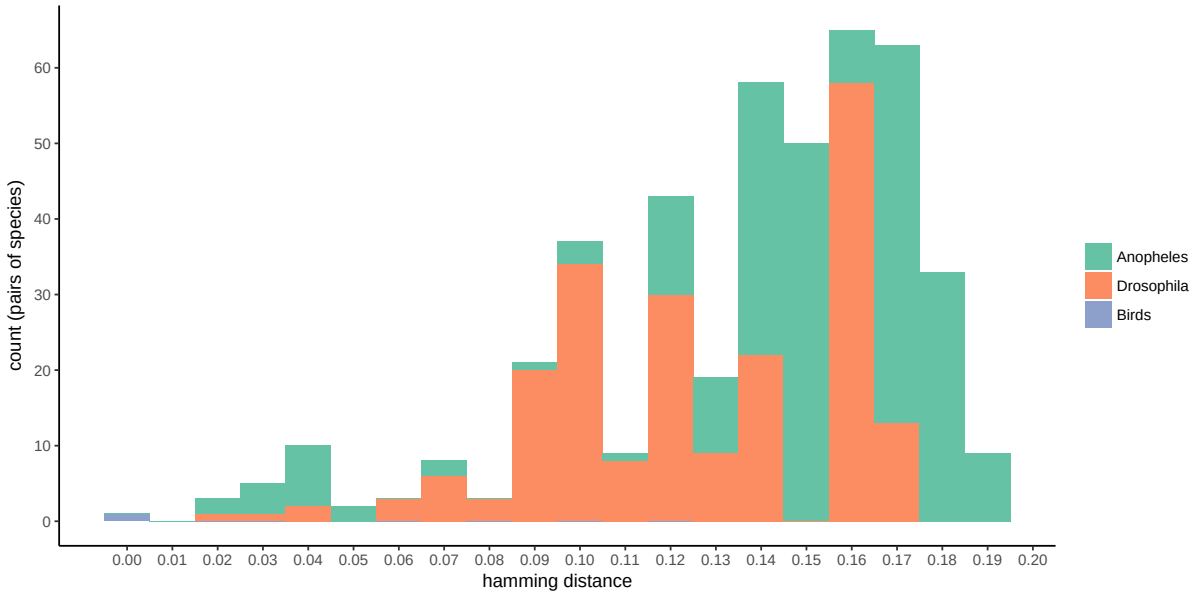


Figure B.12. The histogram of genomic distances between species from the same genus among the Anopheles, Drosophila, and birds datasets. Distances computed based on full assemblies. The only species from the same genus with hamming distance less than 0.01 were the two eagle species (*H. albicilla* and *H. leucocephalus*).

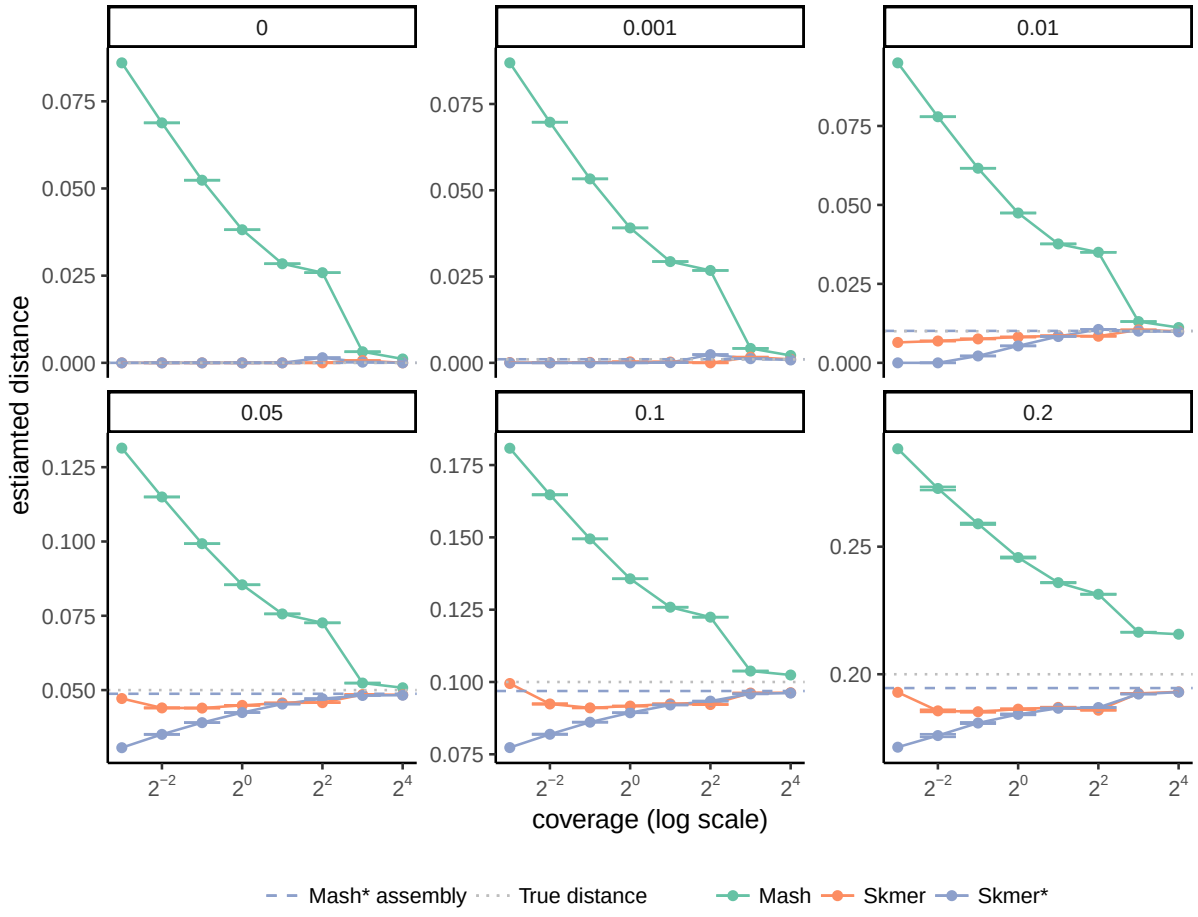


Figure B.13. The performance of Skmer coverage estimation. Comparing distances estimated by Mash, Skmer with estimated coverages, and Skmer with true coverages (Skmer*), on genome-skims of *C. vestalis* and genomes simulated at different distances from it.

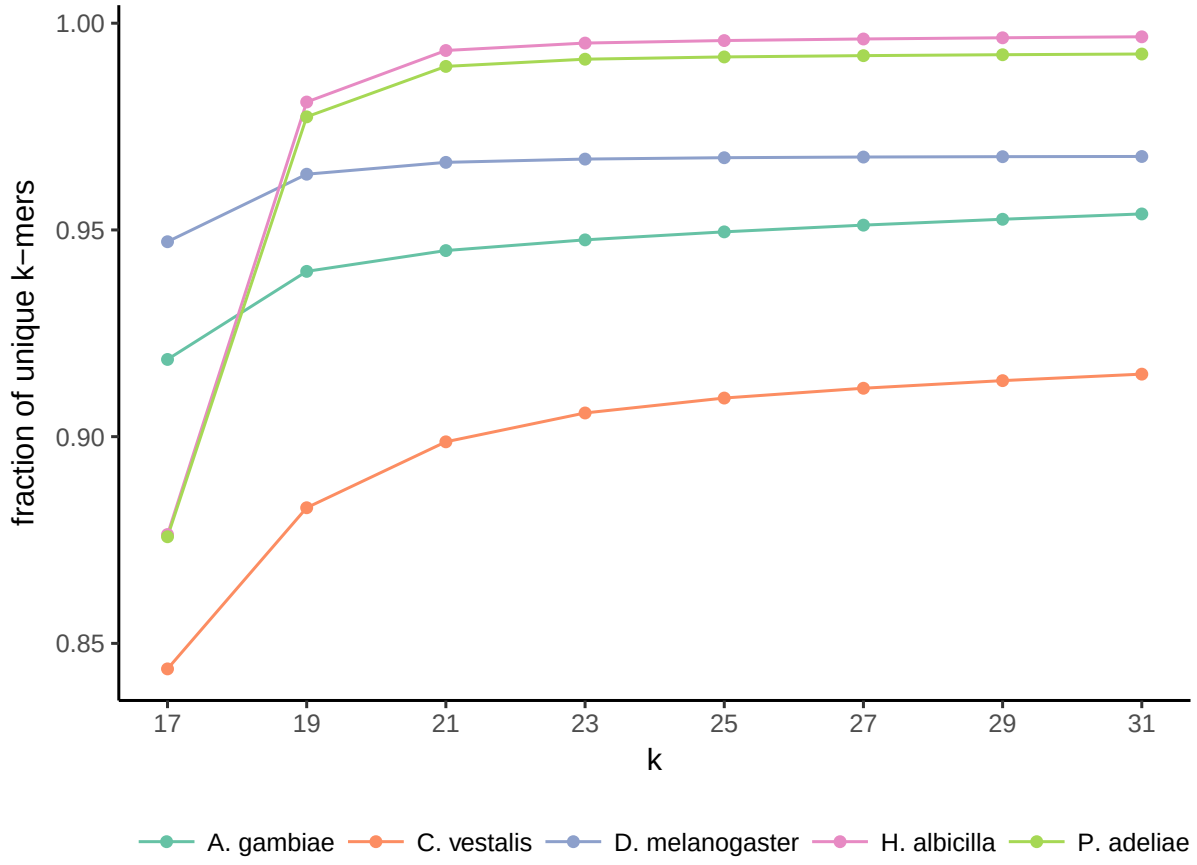


Figure B.14. The fraction of unique *k*-mers in selected species of insects and birds.

Table B.1. GenBank accession numbers of microbial species used in contamination removal.

Species	GenBank assembly accession
Pasteurella langaaensis	GCA_003096995.1
Providencia stuartii	GCA_001558855.2
Serratia marcescens	GCA_000783915.2
Shigella flexneri	GCA_000006925.2
Commensalibacter intestini	GCA_002153535.1
Acetobacter malorum	GCA_002153605.1
Acetobacter pomorum	GCA_002456135.1
Lactobacillus plantarum	GCA_000203855.3
Lactobacillus brevis	GCA_003184305.1
Enterococcus faecalis	GCA_002208945.2
Vagococcus teuberi	GCA_001870205.1
Wolbachia	GCA_000022285.1

Table B.2. GenBank accession numbers and URLs for Anopheles genomes.

Species	GenBank assembly accession	URL
Anopheles albimanus	GCA_000349125.1	http://www.insect-genome.com/data/genome_download/Anopheles_albimanus/Anopheles_albimanus_genomic.fasta.gz
Anopheles arabiensis	GCA_000349185.1	http://www.insect-genome.com/data/genome_download/Anopheles_arabiensis/Anopheles_arabiensis_genomic.fasta.gz
Anopheles atroparvus	GCA_000473505.1	http://www.insect-genome.com/data/genome_download/Anopheles_atroparvus/Anopheles_atroparvus_genomic.fasta.gz
Anopheles christyi	GCA_000349165.1	http://www.insect-genome.com/data/genome_download/Anopheles_christyi/Anopheles_christyi_genomic.fasta.gz
Anopheles coluzzii	-	http://www.insect-genome.com/data/genome_download/Anopheles_coluzzii/Anopheles_coluzzii_genomic.fasta.gz
Anopheles culicifacies	GCA_000473375.1	http://www.insect-genome.com/data/genome_download/Anopheles_culicifacies/Anopheles_culicifacies_genomic.fasta.gz
Anopheles darlingi	GCA_000211455.3	http://www.insect-genome.com/data/genome_download/Anopheles_darlingi/Anopheles_darlingi_genomic.fasta.gz
Anopheles dirus	GCA_000349145.1	http://www.insect-genome.com/data/genome_download/Anopheles_dirus/Anopheles_dirus_genomic.fasta.gz
Anopheles epiroticus	GCA_000349105.1	http://www.insect-genome.com/data/genome_download/Anopheles_epiroticus/Anopheles_epiroticus_genomic.fasta.gz
Anopheles farauti	GCA_000956265.1	http://www.insect-genome.com/data/genome_download/Anopheles_farauti/Anopheles_farauti_genomic.fasta.gz
Anopheles funestus	GCA_000349085.1	http://www.insect-genome.com/data/genome_download/Anopheles_funestus/Anopheles_funestus_genomic.fasta.gz
Anopheles gambiae	GCA_000150785.1	http://www.insect-genome.com/data/genome_download/Anopheles_gambiae/Anopheles_gambiae_genomic.fasta.gz
Anopheles koliensis	GCA_000956275.1	http://www.insect-genome.com/data/genome_download/Anopheles_koliensis/Anopheles_koliensis_genomic.fasta.gz
Anopheles maculatus	GCA_000473185.1	http://www.insect-genome.com/data/genome_download/Anopheles_maculatus/Anopheles_maculatus_genomic.fasta.gz
Anopheles melas	GCA_000473525.2	http://www.insect-genome.com/data/genome_download/Anopheles_melas/Anopheles_melas_genomic.fasta.gz
Anopheles merus	GCA_000473845.2	http://www.insect-genome.com/data/genome_download/Anopheles_merus/Anopheles_merus_genomic.fasta.gz
Anopheles minimus	GCA_000349025.1	http://www.insect-genome.com/data/genome_download/Anopheles_minimus/Anopheles_minimus_genomic.fasta.gz
Anopheles nili	GCA_000439205.1	http://www.insect-genome.com/data/genome_download/Anopheles_nili/Anopheles_nili_genomic.fasta.gz
Anopheles punctulatus	GCA_000956255.1	http://www.insect-genome.com/data/genome_download/Anopheles_punctulatus/Anopheles_punctulatus_genomic.fasta.gz
Anopheles quadriannulatus	GCA_000349065.1	http://www.insect-genome.com/data/genome_download/Anopheles_quadriannulatus/Anopheles_quadriannulatus_genomic.fasta.gz
Anopheles sinensis	GCA_000441895.2	http://www.insect-genome.com/data/genome_download/Anopheles_sinensis/Anopheles_sinensis_genomic.fasta.gz
Anopheles stephensi	GCA_000300775.2	http://www.insect-genome.com/data/genome_download/Anopheles_stephensi/Anopheles_stephensi_genomic.fasta.gz

Table B.3. GenBank accession numbers and URLs for Drosophila genomes.

Species	GenBank assembly accession	URL
Drosophila ananassae	GCA_000005115.1	http://www.insect-genome.com/data/genome_download/Drosophila_ananassae/Drosophila_ananassae_genomic.fasta.gz
Drosophila biarmipes	GCA_000233415.2	http://www.insect-genome.com/data/genome_download/Drosophila_biarmipes/Drosophila_biarmipes_genomic.fasta.gz
Drosophila bipectinata	GCA_000236285.2	http://www.insect-genome.com/data/genome_download/Drosophila_bipectinata/Drosophila_bipectinata_genomic.fasta.gz
Drosophila elegans	GCA_000224195.2	http://www.insect-genome.com/data/genome_download/Drosophila_elegans/Drosophila_elegans_genomic.fasta.gz
Drosophila erecta	GCA_000005135.1	http://www.insect-genome.com/data/genome_download/Drosophila_erecta/Drosophila_erecta_genomic.fasta.gz
Drosophila eugracilis	GCA_000236325.2	http://www.insect-genome.com/data/genome_download/Drosophila_eugracilis/Drosophila_eugracilis_genomic.fasta.gz
Drosophila ficusphila	GCA_000220665.2	http://www.insect-genome.com/data/genome_download/Drosophila_ficusphila/Drosophila_ficusphila_genomic.fasta.gz
Drosophila grimshawi	GCA_000005155.1	http://www.insect-genome.com/data/genome_download/Drosophila_grimshawi/Drosophila_grimshawi_genomic.fasta.gz
Drosophila kikkawai	GCA_000224215.2	http://www.insect-genome.com/data/genome_download/Drosophila_kikkawai/Drosophila_kikkawai_genomic.fasta.gz
Drosophila melanogaster	GCA_000778455.1	http://www.insect-genome.com/data/genome_download/Drosophila_melanogaster/Drosophila_melanogaster_genomic.fasta.gz
Drosophila miranda	GCA_000269505.2	http://www.insect-genome.com/data/genome_download/Drosophila_miranda/Drosophila_miranda_genomic.fasta.gz
Drosophila mojavensis	GCA_000005175.1	http://www.insect-genome.com/data/genome_download/Drosophila_mojavensis/Drosophila_mojavensis_genomic.fasta.gz
Drosophila persimilis	GCA_000005195.1	http://www.insect-genome.com/data/genome_download/Drosophila_persimilis/Drosophila_persimilis_genomic.fasta.gz
Drosophila rhopaloa	GCA_000236305.2	http://www.insect-genome.com/data/genome_download/Drosophila_rhopaloa/Drosophila_rhopaloa_genomic.fasta.gz
Drosophila sechellia	GCA_000005215.1	http://www.insect-genome.com/data/genome_download/Drosophila_sechellia/Drosophila_sechellia_genomic.fasta.gz
Drosophila simulans	GCA_000259055.1	http://www.insect-genome.com/data/genome_download/Drosophila_simulans/Drosophila_simulans_genomic.fasta.gz
Drosophila suzukii	GCA_000472105.1	http://www.insect-genome.com/data/genome_download/Drosophila_suzukii/Drosophila_suzukii_genomic.fasta.gz
Drosophila takahashii	GCA_000224235.2	http://www.insect-genome.com/data/genome_download/Drosophila_takahashii/Drosophila_takahashii_genomic.fasta.gz
Drosophila virilis	GCA_000005245.1	http://www.insect-genome.com/data/genome_download/Drosophila_virilis/Drosophila_virilis_genomic.fasta.gz
Drosophila willistoni	GCA_000005925.1	http://www.insect-genome.com/data/genome_download/Drosophila_willistoni/Drosophila_willistoni_genomic.fasta.gz
Drosophila yakuba	GCA_000005975.1	http://www.insect-genome.com/data/genome_download/Drosophila_yakuba/Drosophila_yakuba_genomic.fasta.gz

Table B.4. GenBank accession numbers and URLs for avian genomes.

Species	GenBank assembly accession	URL
Acanthisitta chloris	GCA_000695815.1	http://dx.doi.org/10.5524/101015
Anas platyrhynchos	GCA_000355885.1	http://dx.doi.org/10.5524/101001
Antrostomus carolinensis	GCA_000700745.1	http://dx.doi.org/10.5524/101019
Apaloderma vittatum	GCA_000703405.1	http://dx.doi.org/10.5524/101016
Aptenodytes forsteri	GCA_000699145.1	http://dx.doi.org/10.5524/100005
Balearica regulorum	GCA_000709895.1	http://dx.doi.org/10.5524/101017
Buceros rhinoceros	GCA_000710305.1	http://dx.doi.org/10.5524/101018
Calypte anna	GCA_000699085.1	http://dx.doi.org/10.5524/101004
Cariama cristata	GCA_000690535.1	http://dx.doi.org/10.5524/101020
Cathartes aura	GCA_000699945.1	http://dx.doi.org/10.5524/101021
Chaetura pelagica	GCA_000747805.1	http://dx.doi.org/10.5524/101005
Charadrius vociferus	GCA_000708025.2	http://dx.doi.org/10.5524/101007
Chlamydotis macqueenii	GCA_000695195.1	http://dx.doi.org/10.5524/101022
Colius striatus	GCA_000690715.1	http://dx.doi.org/10.5524/101023
Columba livia	GCA_000337935.1	http://dx.doi.org/10.5524/100007
Corvus brachyrhynchos	GCA_000691975.1	http://dx.doi.org/10.5524/101008
Cuculus canorus	GCA_000709325.1	http://dx.doi.org/10.5524/101009
Egretta garzetta	GCA_000687185.1	http://dx.doi.org/10.5524/101002
Eurypyga helias	GCA_000690775.1	http://dx.doi.org/10.5524/101024
Falcons peregrine	GCA_000337955.1	http://dx.doi.org/10.5524/101006
Fulmarus glacialis	GCA_000690835.1	http://dx.doi.org/10.5524/101025
Gallus gallus	GCA_000002315.3	ftp://climb.genomics.cn/pub/10.5524/100001_101000/101000/chicken/
Gavia stellata	GCA_000690875.1	http://dx.doi.org/10.5524/101026
Geospiza fortis	GCA_000277835.1	http://dx.doi.org/10.5524/100040
Haliaeetus albicilla	GCA_000691405.1	http://dx.doi.org/10.5524/101027
Haliaeetus leucocephalus	GCA_000737465.1	http://dx.doi.org/10.5524/101040
Leptosomus discolor	GCA_000691785.1	http://dx.doi.org/10.5524/101028
Manacus vitellinus	GCA_000692015.2	http://dx.doi.org/10.5524/101010
Meleagris gallopavo	GCA_000146605.3	ftp://climb.genomics.cn/pub/10.5524/100001_101000/101000/turkey/
Melopsittacus undulatus	GCA_000238935.1	http://dx.doi.org/10.5524/100059
Merops nubicus	GCA_000691845.1	http://dx.doi.org/10.5524/101029
Mesitornis unicolor	GCA_000695765.1	http://dx.doi.org/10.5524/101030
Nestor notabilis	GCA_000696875.1	http://dx.doi.org/10.5524/101031
Nipponia nippon	GCA_000708225.1	http://dx.doi.org/10.5524/101003
Pelecanus crispus	GCA_000687375.1	http://dx.doi.org/10.5524/101032
Phaethon lepturus	GCA_000687285.1	http://dx.doi.org/10.5524/101033
Phalacrocorax carbo	GCA_000708925.1	http://dx.doi.org/10.5524/101034
Phoenicopterus ruber	GCA_000687265.1	http://dx.doi.org/10.5524/101035
Picoides pubescens	GCA_000699005.1	http://dx.doi.org/10.5524/101012
Podiceps cristatus	GCA_000699545.1	http://dx.doi.org/10.5524/101036
Pterocles gutturalis	GCA_000699245.1	http://dx.doi.org/10.5524/101037
Pygoscelis adeliae	GCA_000699105.1	http://dx.doi.org/10.5524/100006
Struthio camelus	GCA_000698965.1	http://dx.doi.org/10.5524/101013
Taeniopygia guttata	GCA_000151805.2	ftp://climb.genomics.cn/pub/10.5524/100001_101000/101000/zebrafinch/
Tauraco erythrolophus	GCA_000709365.1	http://dx.doi.org/10.5524/101038
Tinamus guttatus	GCA_000705375.2	http://dx.doi.org/10.5524/101014
Tyto alba	GCA_000687205.1	http://dx.doi.org/10.5524/101039

Table B.5. The coverage of genomes over three datasets. Each genome is skimmed with 100Mb sequence.

Dataset	Min	Mean	Max
Drosophila	0.45X	0.60X	0.79X
Anopheles	0.37X	0.57X	1.02X
Birds	0.082X	0.090X	0.107X

Table B.6. Comparing the average error of Mash, Skmer, and AAF over three datasets. Fixed sequencing effort from each species.

Dataset	Sequencing effort	Mash	Skmer	AAF (uncorrected)	AAF (corrected)
	0.1Gb	48.02% (1.54%)	2.02% (0.05%)	40.22% (1.67%)	9.62% (0.52%)
Anopheles	0.5Gb	24.89% (0.59%)	0.75% (0.02%)	17.60% (0.70%)	7.35% (0.26%)
	1Gb	18.43% (0.54%)	0.55% (0.02%)	16.94% (0.61%)	4.74% (0.22%)
	0.1Gb	47.98% (0.82%)	1.65% (0.06%)	40.67% (0.94%)	9.00% (0.20%)
Drosophila	0.5Gb	25.25% (0.34%)	0.72% (0.03%)	18.63% (0.45%)	7.00% (0.19%)
	1Gb	13.00% (0.16%)	0.50% (0.02%)	19.69% (0.52%)	2.18% (0.06%)
	0.1Gb	95.57% (2.54%)	5.72% (0.06%)	86.45% (3.18%)	49.48% (1.94%)
Birds	0.5Gb	56.61% (1.40%)	2.14% (0.02%)	49.13% (1.75%)	13.73% (0.56%)
	1Gb	41.25% (0.97%)	1.32% (0.01%)	34.33% (1.22%)	1.05% (0.09%)

* The standard error of the mean is provided in parentheses.

Appendix C

Supplementary material: RESPECT

C.1 Supplementary methods

Initial estimate of parameters

With the assumption that a genome has no repeating k-mers $\mathbf{r} = [L, 0, 0, \ldots]$, Eq. (4.5) reduces to

$$\mathbb{E}[o_h] = m_h = \begin{cases} L\lambda(1-\varepsilon)^k e^{-\lambda(1-\varepsilon)^k} + L\lambda(1-(1-\varepsilon)^k) & h = 1 \\ L\frac{(\lambda(1-\varepsilon)^k)^h}{h!} e^{-\lambda(1-\varepsilon)^k} & h > 1 \end{cases} . \tag{C.1}$$

We use the method of moments (see e.g., Section 7.6 of [132]) and set $m_h = o_h$ to estimate the underlying parameters λ and ε. Specifically, let $h^* = \text{argmax}_{h>1}\, o_h$ be the multiplicity with the largest number of observed k-mers (excluding the unique ones $h = 1$). We use o_{h^*+1}/o_{h^*} to estimate $\lambda_{\text{ef}} = \lambda(1-\varepsilon)^k$

$$\lambda_{\text{ef}} = \frac{(h^*+1)o_{h^*+1}}{o_{h^*}} . \tag{C.2}$$

Then, using o_{h^*}/o_1, we estimate λ as

$$\lambda = \lambda_{\text{ef}}^h e^{-\lambda_{\text{ef}}} \frac{o_1}{h^*!\, o_{h^*}} - \lambda_{\text{ef}} e^{-\lambda_{\text{ef}}} + \lambda_{\text{ef}} , \tag{C.3}$$

and estimate ε from the ratio of λ_{ef} and λ

$$\varepsilon = 1 - (\frac{\lambda_{\text{ef}}}{\lambda})^{1/k} . \tag{C.4}$$

Least-squares estimate of repeat spectrum

Consider the cost function defined in Eq. (4.6) with $p = 2$ and $w_h = 1$ for all h

$$\mathscr{E}_{\mathbf{w},p}(\mathbf{P},\mathbf{r},\varepsilon,\mathbf{o}) = \left(\sum_h \left| m_h - o_h \right|^2\right)^{1/2} = \left(\sum_h \left| (\mathbf{r}\mathbf{P}^{\mathbf{T}} + \mathbf{1}_{h=1}E)_h - o_h \right|^2\right)^{1/2} . \tag{C.5}$$

We considered the simplest sequencing-error-free case ($\varepsilon = 0$), where coverage λ was known. Therefore, $\mathbb{E}[\mathbf{O}] = \mathbf{m} = \mathbf{r}\mathbf{P}^{\mathbf{T}}$, where $\mathbf{P}$ is an $n \times n$ matrix with

$$P_{hj} = e^{-j\lambda} \frac{(j\lambda)^h}{h!}. \tag{C.6}$$

$\mathbf{P}$ can be decomposed as $\mathbf{P} = \mathbf{\Lambda}\mathbf{V}\mathbf{E}$ where $\mathbf{\Lambda}$ is a diagonal matrix with $\mathbf{\Lambda}_{hh} = \frac{\lambda^h}{h!}$, $\mathbf{E}$ is a diagonal matrix with $\mathbf{E}_{jj} = je^{-j\lambda}$, and $\mathbf{V}$ is the transpose of a Vandermonde matrix with the second column given by the vector $(1,2,3,\ldots,n)^T$; thus $V_{hj} = j^{h-1}$, and

$$\mathbf{V} = \begin{bmatrix} 1 & 1 & 1 & \cdots & 1 \\ 1 & 2 & 3 & \cdots & n \\ 1 & 2^2 & 3^2 & \cdots & n^2 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & 2^{n-1} & 3^{n-1} & \cdots & n^{n-1} \end{bmatrix} . \tag{C.7}$$

Note that $\mathbf{\Lambda}$ and $\mathbf{E}$ are diagonal matrices with non-zero diagonal elements and hence, they are non-singular. Also, since V is a Vandermonde matrix, we have $\det(\mathbf{V}) = \prod_{1 \le i < j \le n}(j-i) > 0$ which renders $\mathbf{V}$ a non-singular matrix. Thus, it enables us to use the estimate $\mathbf{r}^{(\text{est})} = \mathbf{o}\mathbf{P}^{-\mathbf{T}}$.

However, for the case $\lambda = 1$ we prove that

$$\text{cond}(\mathbf{P}) \geq c\frac{2^n}{n},$$

suggesting a highly ill-conditioned matrix, and making the LS estimates unreliable.

Bound on $\text{cond}(\mathbf{P})$

To establish the aforementioned bound for $\text{cond}(\mathbf{P})$, as $\mathbf{P}$ is non-singular, we have (see e.g., Section 3.8 of [134])

$$\text{cond}(\mathbf{P}) = \|\mathbf{P}\|\|\mathbf{P}^{-1}\| = \|\mathbf{\Lambda V E}\| \cdot \|\mathbf{E}^{-1}\mathbf{V}^{-1}\mathbf{\Lambda}^{-1}\|, \tag{C.8}$$

where $\|\mathbf{A}\|$ is the induced 2-norm of the matrix $\mathbf{A}$. Since both $\mathbf{E}$ and $\mathbf{\Lambda}$ are diagonal, we have

$$\text{cond}(\mathbf{\Lambda}) = \text{cond}(\mathbf{\Lambda}^{-1}) = \frac{\max_i \{\lambda^i/i!\}}{\min_i \{\lambda^i/i!\}} = \frac{\lambda^{\lfloor\lambda\rfloor}/\lfloor\lambda\rfloor!}{\min(1, \lambda^n/n!)}$$
$$\text{cond}(\mathbf{E}) = \text{cond}(\mathbf{E}^{-1}) = \frac{\max_i \left\{ie^{-i\lambda}\right\}}{\min_i \left\{ie^{-i\lambda}\right\}} = \frac{f(\lambda)}{\min(e^{-\lambda}, ne^{-n\lambda})},$$

where

$$f(\lambda) = \begin{cases} \max(\lfloor\lambda\rfloor e^{-\lambda\lfloor\lambda\rfloor}, (\lfloor\lambda\rfloor + 1)e^{-\lambda(\lfloor\lambda\rfloor+1)}) & \lambda < 1 \\ e^{-\lambda} & \lambda \geq 1 \end{cases}.$$

For the simplicity of exposition, we use $\lambda = 1$. Hence,

$$\begin{aligned}\text{cond}(\mathbf{\Lambda}) &= \text{cond}(\mathbf{\Lambda}^{-1}) = n!,\\ \text{cond}(\mathbf{E}) &= \text{cond}(\mathbf{E}^{-1}) = \frac{1}{n}e^{n-1}.\end{aligned} \tag{C.9}$$

Note that for the 2-norm $\|\cdot\|$, we have

$$\|\mathbf{AB}\| \leq \|\mathbf{A}\|\|\mathbf{B}\| \tag{C.10}$$

hence, for any matrix $\mathbf{B}$ and any invertible matrix $\mathbf{A}$, we have $\|\mathbf{B}\| = \|\mathbf{A}^{-1}\mathbf{AB}\| \leq \|\mathbf{A}^{-1}\|\|\mathbf{AB}\|$, implying

$$\|\mathbf{AB}\| \geq \frac{\|\mathbf{B}\|}{\|\mathbf{A}^{-1}\|}. \tag{C.11}$$

By repeated application of this inequality to Eq. (C.8) and using (C.9), we have

$$\text{cond}(\mathbf{P}) \geq \frac{\text{cond}(\mathbf{V})}{\text{cond}(\mathbf{E})\cdot\text{cond}(\mathbf{\Lambda})} = \frac{n}{n!e^{n-1}}\,\text{cond}(\mathbf{V}). \tag{C.12}$$

To show that $\text{cond}(\mathbf{P})$ grows exponentially, we use the following lemma.

Lemma 1. *For the matrix* $\mathbf{V}$ *given in* (C.7)*, we have*

$$c\frac{1}{n^{3/2}}(2n)^n \leq \text{cond}(\mathbf{V}) \leq C(2n)^n. \tag{C.13}$$

for some $c, C > 0$*.*

Proof. Since $\mathbf{V}$ is non-singular, $\text{cond}(\mathbf{V}) = \|\mathbf{V}\|\|\mathbf{V}^{-1}\|$ and hence, it remains to bound $\|\mathbf{V}\|$ and $\|\mathbf{V}^{-1}\|$. To do this, we use the following inequalities relating several norms of a matrix $\mathbf{A}$ (see e.g., Section 10.4.4 of [135])

$$\frac{1}{\sqrt{n}}\|\mathbf{A}\|_F \leq \|\mathbf{A}\| \leq \|\mathbf{A}\|_F, \tag{C.14}$$

$$\frac{1}{\sqrt{n}}\|\mathbf{A}\|_\infty \leq \|\mathbf{A}\| \leq \sqrt{n}\|\mathbf{A}\|_\infty, \tag{C.15}$$

where $\|\mathbf{A}\|_F$ and $\|\mathbf{A}\|_\infty$ are the Frobenius and the induced ∞-norm of $\mathbf{A}$, respectively.

1. **Bounding** $\|\mathbf{V}\|$: Using (C.14) and $\|\mathbf{V}\|_F^2 = \sum_{i=1}^n \sum_{j=1}^n j^{2i-2} \geq n^{2(n-1)}$, we get

$$\|\mathbf{V}\|_F \geq n^{n-1}.$$

For an upper bound on $\|\mathbf{V}\|_F$, we have

$$\begin{aligned}
\|\mathbf{V}\|_F^2 &= \sum_{i=1}^n \sum_{j=1}^n j^{2i-2} \\
&\leq \sum_{i=1}^n \left(n^{2i-2} + \int_1^n x^{2i-2} dx \right) \\
&= \sum_{i=1}^n \left(n^{2i-2} + \frac{n^{2i-1}-1}{2i-1} \right) \\
&= \sum_{i=1}^n n^{2i-2} + \sum_{i=1}^n \frac{n^{2i-1}}{2i-1},
\end{aligned}$$

where the inequality follows from x^k being a monotonically increasing function for positive x and all $k \geq 1$. But

$$\sum_{i=1}^n n^{2i-2} = \sum_{i=1}^n (n^2)^{i-1} = \frac{n^{2n}-1}{n^2-1} \leq \frac{n^{2n}}{n^2-1}. \tag{C.16}$$

Therefore, for $n \geq 3$, we have

$$
\begin{aligned}
\|\mathbf{V}\|_F^2 &\leq \frac{n^{2n}}{n^2-1} + \sum_{i=1}^{n} \frac{n^{2i-1}}{2i-1} \\
&= \frac{n^{2n}}{n^2-1} + \frac{n^{2n-1}}{2n-1} + \sum_{i=1}^{n-1} \frac{n^{2i-1}}{2i-1} \\
&\leq \frac{n^{2n}}{n^2-1} + \frac{n^{2n-1}}{2n-1} + (n-1)\frac{n^{2n-3}}{2(n-1)-1} \\
&\leq (\frac{1}{1-n^{-2}} + \frac{1}{2-n^{-2}} + \frac{1}{2n-3})n^{2n-2} \\
&\leq 2n^{2n-2},
\end{aligned}
$$

where the second inequality follows from $\frac{n^{2i-1}}{2i-1}$ being an increasing function of i for $n \geq 3$. It can be verified that the above bound holds for $n = 1, 2$ too. Using this fact, and the lower bound on $\|\mathbf{V}\|_F$, we have

$$
\frac{1}{\sqrt{n}} n^{n-1} \leq \|\mathbf{V}\| \leq \sqrt{2} n^{n-1}. \tag{C.17}
$$

2. **Bounding** $\|\mathbf{V}^{-1}\|$: Using Theorem 1 by Gautschi (1962) [136], we have:

$$
\|\mathbf{V}^{-1}\|_\infty = \max_i \prod_{j \neq i} \frac{1+j}{|j-i|}. \tag{C.18}
$$

Note that for a fixed i,

$$
\prod_{j \neq i} (1+j) = \frac{(n+1)!}{i+1},
$$

and

$$
\prod_{j \neq i} \frac{1}{|j-i|} = \frac{1}{(i-1)!(n-i)!}.
$$

Therefore,

$$
\prod_{j \neq i} \frac{1+j}{|j-i|} = \frac{(n+1)!}{(i+1)(i-1)!(n-i)!} = \binom{n}{i} \frac{i}{i+1}(n+1).
$$

Replacing this in (C.18), we get

$$\frac{1}{2}(n+1)\max_i \binom{n}{i} \leq \|\mathbf{V}^{-1}\|_\infty \leq (n+1)\max_i \binom{n}{i}. \tag{C.19}$$

But $\max_i \binom{n}{i} = \binom{n}{\lfloor n/2 \rfloor}$ and hence, by the Sterling's approximation [137], we have

$$a\frac{1}{\sqrt{n}}2^n \leq \max_i \binom{n}{i} \leq A\frac{1}{\sqrt{n}}2^n$$

for some $a < A$. Using this in (C.19), we arrive at

$$b\sqrt{n}2^n \leq \|\mathbf{V}^{-1}\|_\infty \leq B\sqrt{n}2^n,$$

for some $b, B > 0$. Therefore, using (C.15), we get

$$b2^n \leq \|\mathbf{V}^{-1}\| \leq Bn2^n. \tag{C.20}$$

Finally, combining the bounds (C.17) and (C.20) on $\|\mathbf{V}\|$ and $\|\mathbf{V}^{-1}\|$, respectively, and using $\text{cond}(\mathbf{V}) = \|\mathbf{V}\|\|\mathbf{V}^{-1}\|$, we get the desired result

$$c\frac{1}{n^{3/2}}(2n)^n \leq \text{cond}(\mathbf{V}) \leq C(2n)^n,$$

for some constants $c, C > 0$.

□

Now, we are ready to show that $\mathbf{P}$ is a highly ill-conditioned matrix.

Lemma 2. *For the matrix* $\mathbf{P}$ *(given by* (C.6)*), we have*

$$\text{cond}(\mathbf{P}) \geq c\frac{2^n}{n}, \tag{C.21}$$

for some $c > 0$.

Proof. By the application of the lower-bound of Lemma 1 to (C.12), we have

$$\mathrm{cond}(\mathbf{P}) \geq \frac{n}{n!e^{n-1}}\,\mathrm{cond}(\mathbf{V}) \geq \frac{n}{n!e^{n-1}} \cdot c\frac{1}{n^{3/2}}(2n)^n = \frac{c(2n)^n}{\sqrt{n}n!e^{n-1}}.$$

Therefore, using the Sterling's approximation $n! \leq n^{n+1/2}e^{-(n-1)}$, we get

$$\mathrm{cond}(\mathbf{P}) \geq c\,\frac{2^n}{n}$$

for some constant $c > 0$. □

Since the bound (C.21) can be written as

$$\mathrm{cond}(\mathbf{P}) \geq c\,2^{n-\log_2 n},$$

and for any $\varepsilon > 0$, we have $n - \log_2 n \geq (1-\varepsilon)n$ for sufficiently large n, we have $\mathrm{cond}(\mathbf{P}) \geq c2^{(1-\varepsilon)n}$. Therefore, $\mathrm{cond}(\mathbf{P})$ grows exponentially and in fact, $\mathrm{cond}(\mathbf{P}) = \Omega(\alpha^n)$ for any $\alpha < 2$.

C.2 Supplementary figures and tables

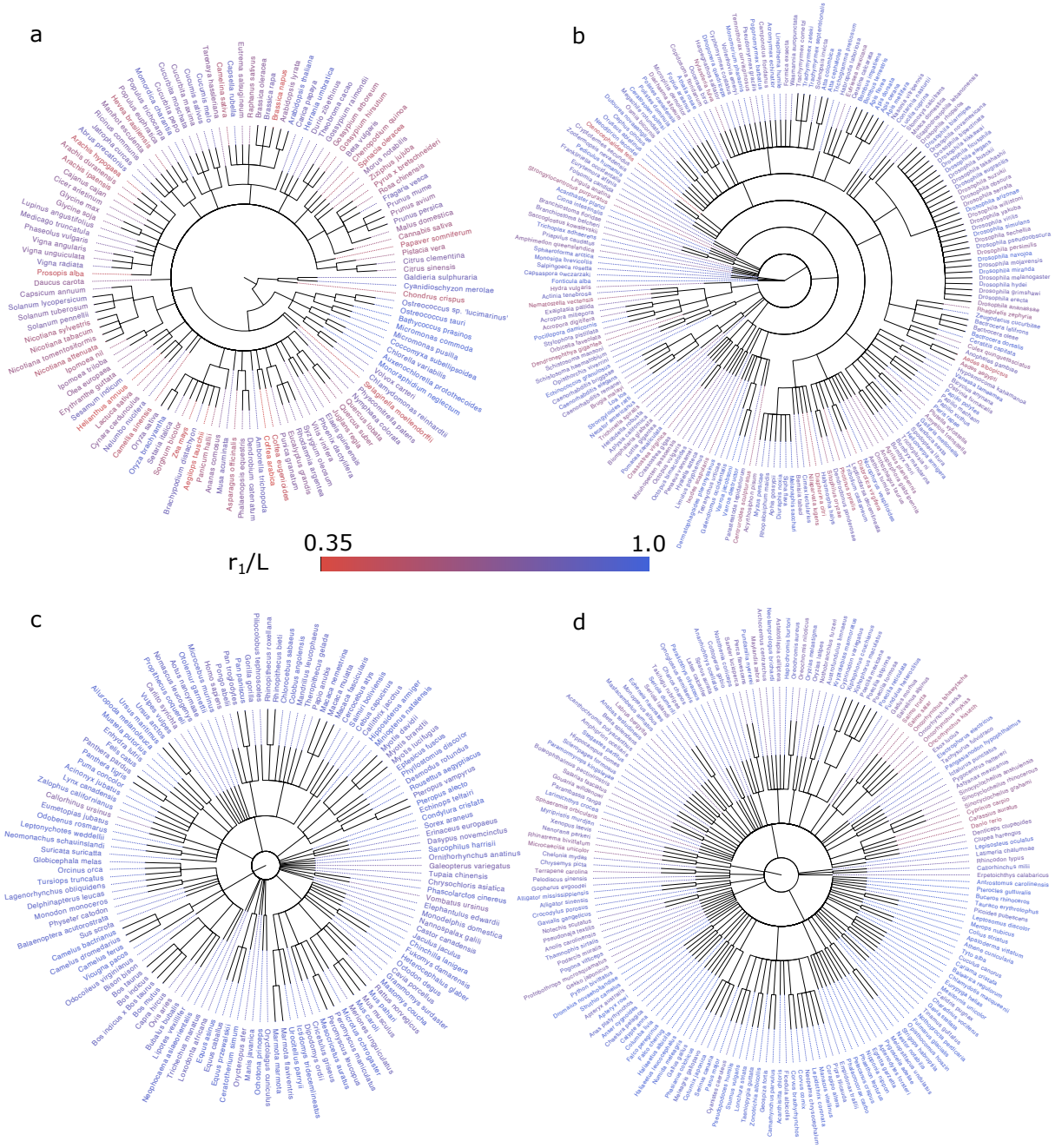


Figure C.1. Whole RefSeq taxonomy with r_1/L annotation. (a) Plants. (b) Invertebrates. (c) Mammals. (d) Other vertebrates.

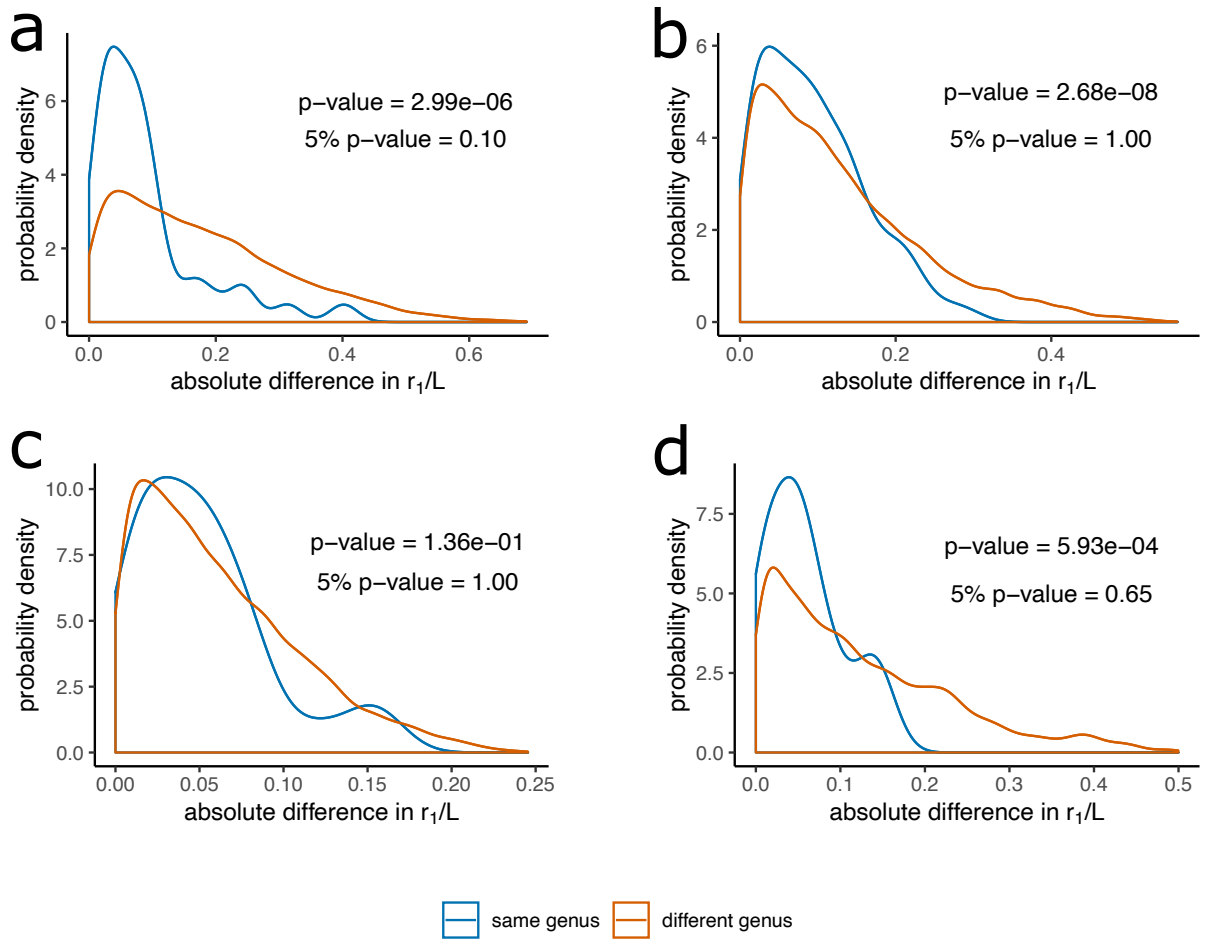


Figure C.2. Distributions of intra-generic versus inter-generic differences in $r1/L$ for pairs of RefSeq species. (a) Plants. (b) Invertebrates. (c) Mammals. (d) Other vertebrates.

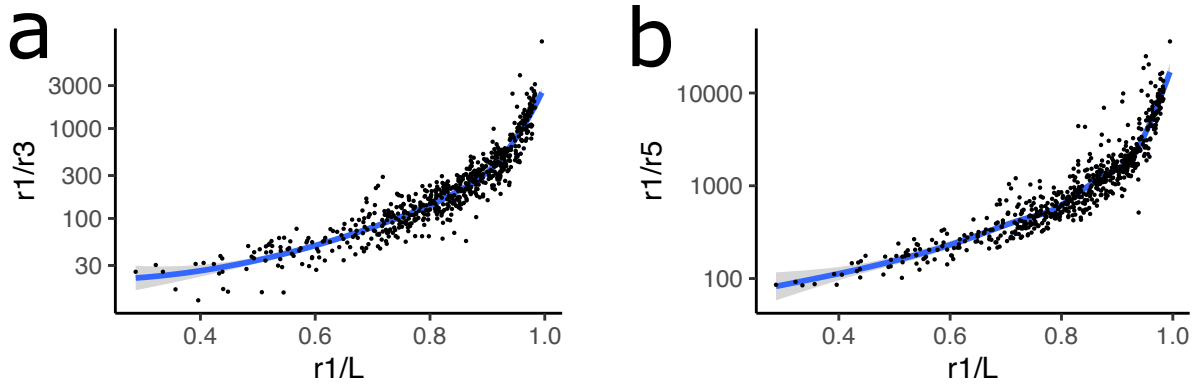


Figure C.3. Correlation of r_1/L with spectral ratios. (a) r_1/r_3 versus r_1/L, (b) r_1/r_5 versus r_1/L.

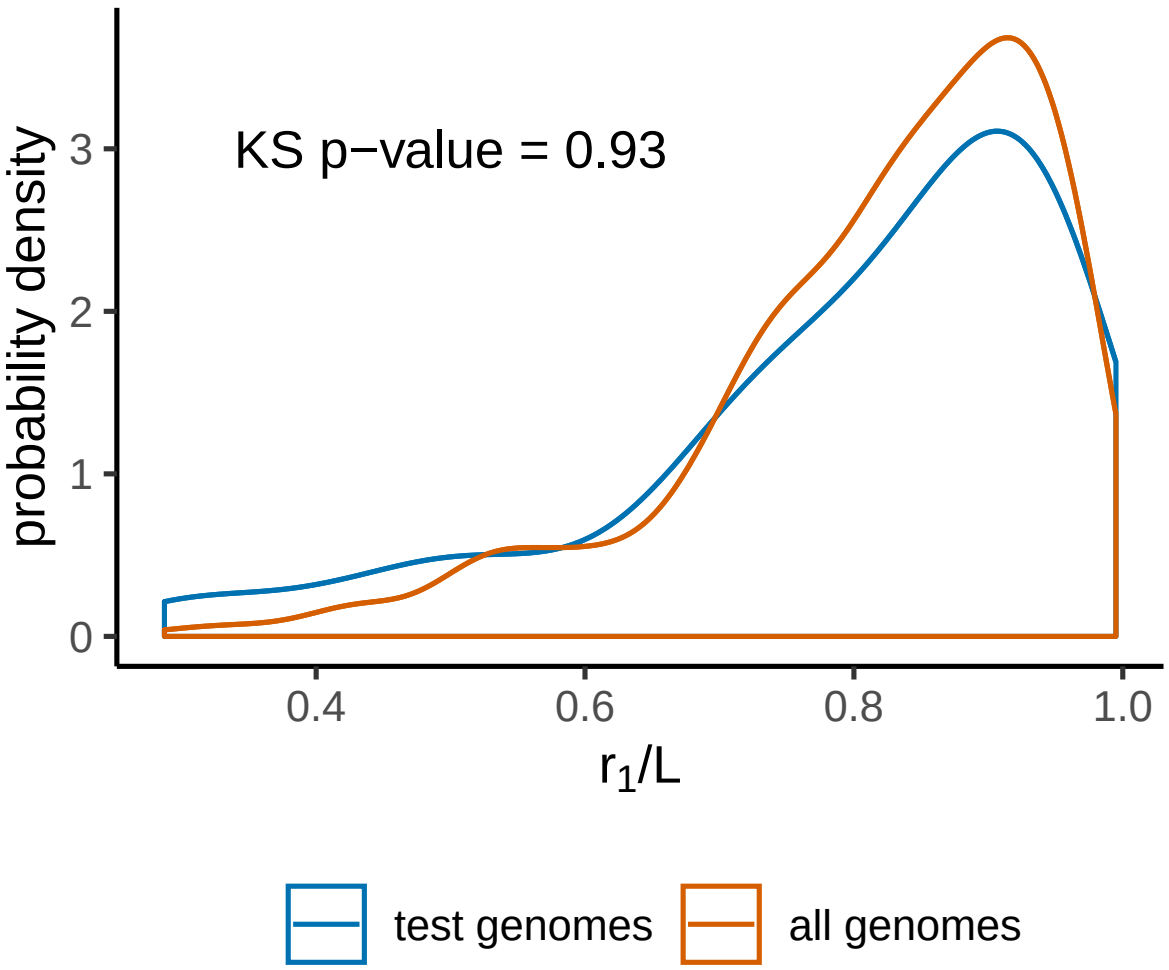


Figure C.4. Comparing the distributions of $r1/L$ among test and all RefSeq genomes. The p-value for the hypothesis that the distributions are different using two-sided Kolmogorov–Smirnov test is 0.93. Highly-repetitive genomes are slightly over-represented in the test set.

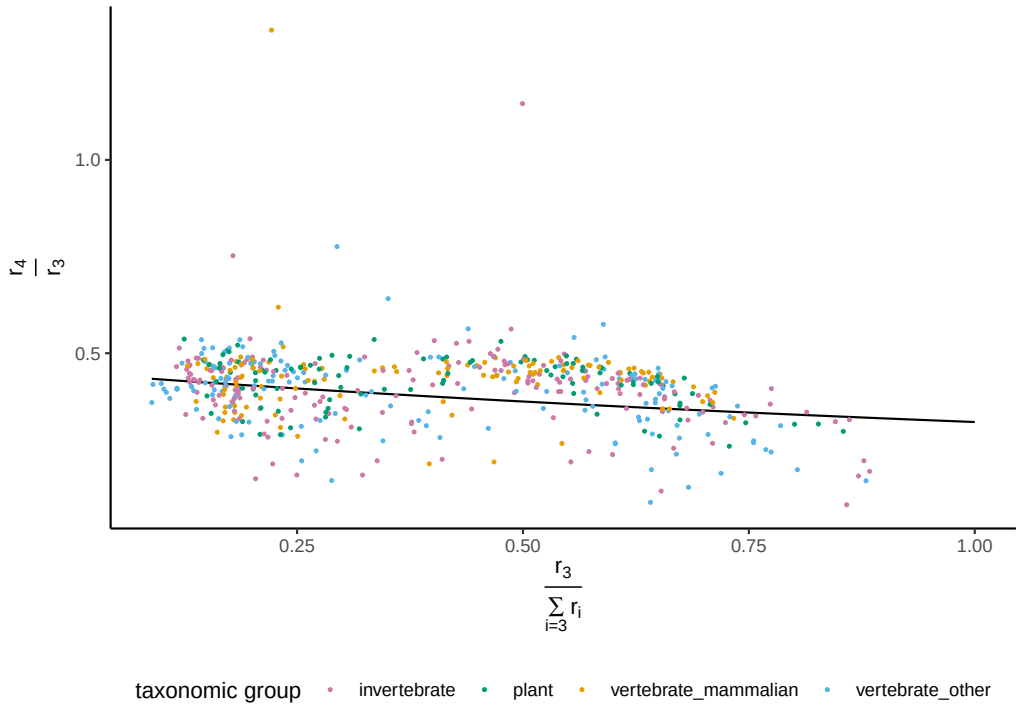


Figure C.5. Correlation between true r_4/r_3 and estimated $r_3/\sum_{i=3} r_i$.

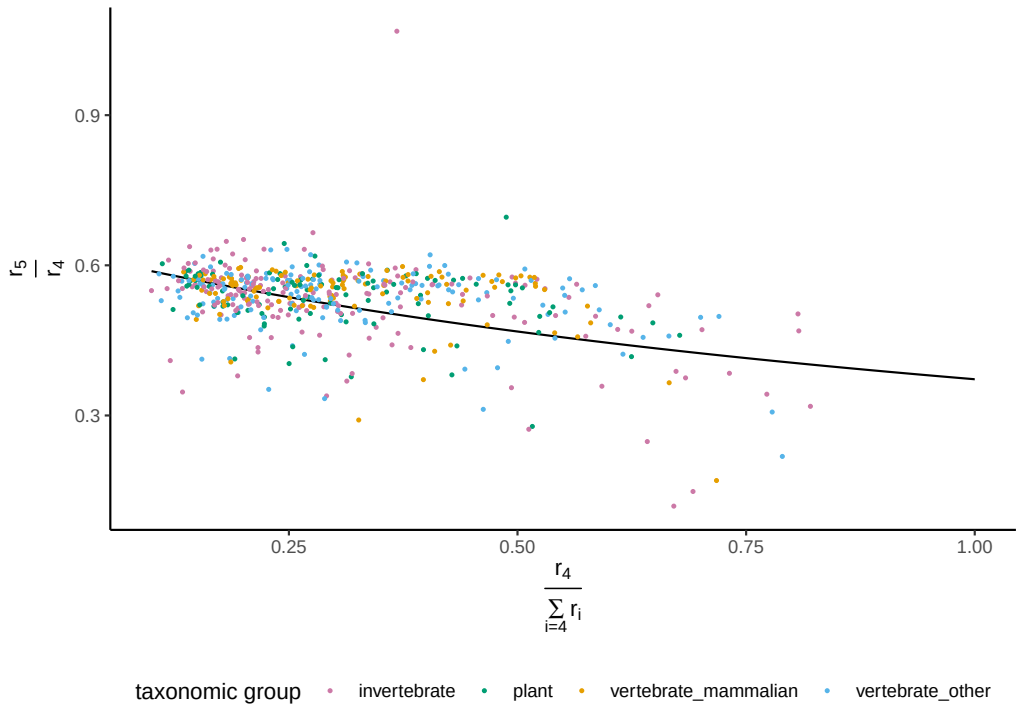


Figure C.6. Correlation between true r_5/r_4 and estimated $r_4/\sum_{i=4} r_i$.

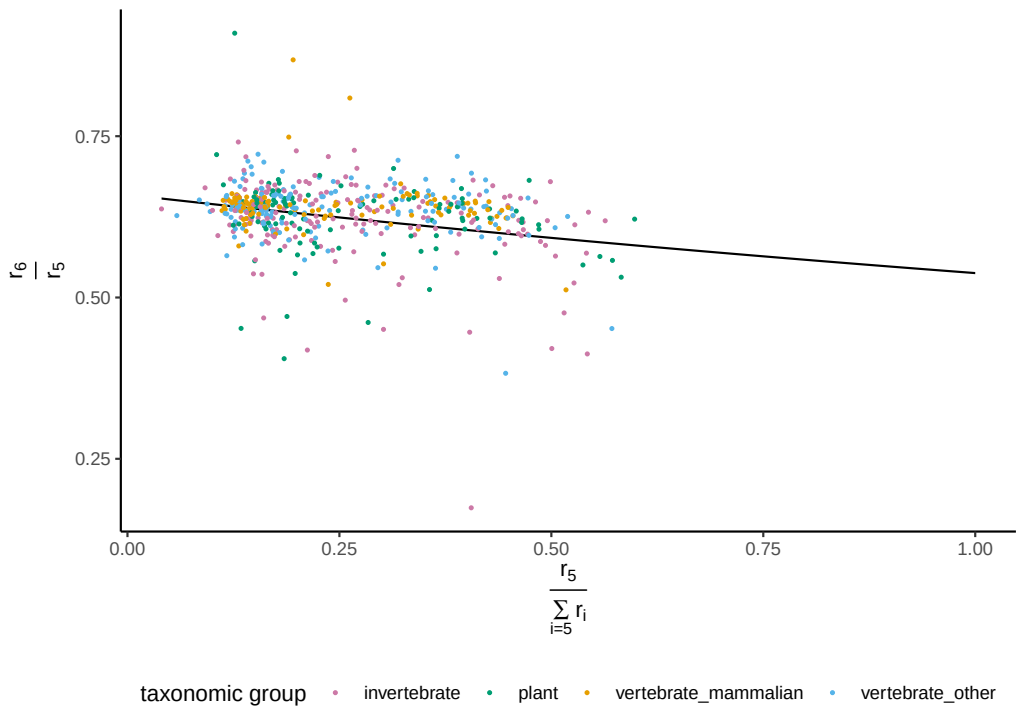


Figure C.7. Correlation between true r_6/r_5 and estimated $r_5/\sum_{i=5} r_i$.

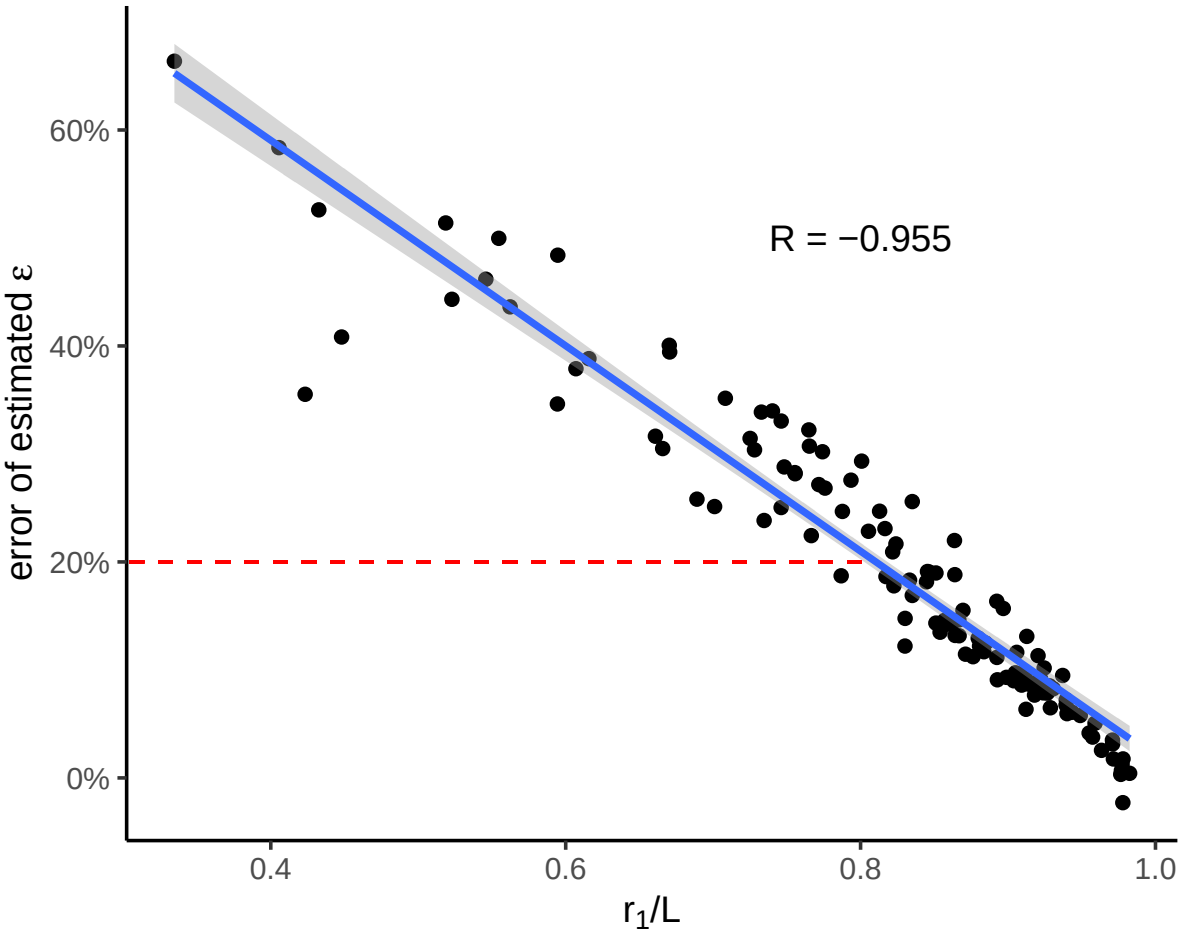


Figure C.8. Correlation between the relative error in the estimated sequencing error and the uniqueness ratio. A subset of 120 training genomes were selected as the cross-validation set, and genome-skims were simulated at 1X coverage with 1% sequencing error rate. There is a strong correlation ($R = -0.995$) between the error in estimating ε and r_1/L ratio. We capped the correction at 20% (red dashed line).

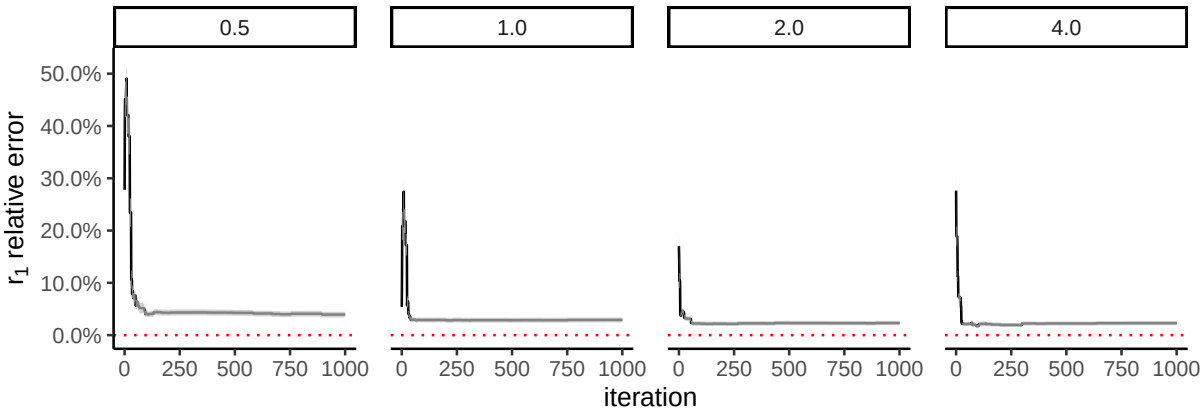


Figure C.9. r_1 **estiamtion convergence with time.**

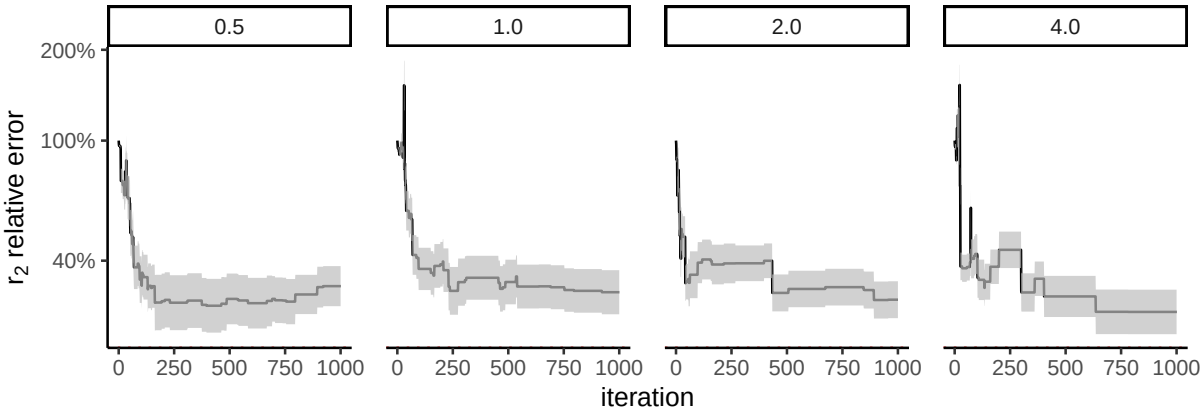


Figure C.10. r_2 **estiamtion convergence with time.**

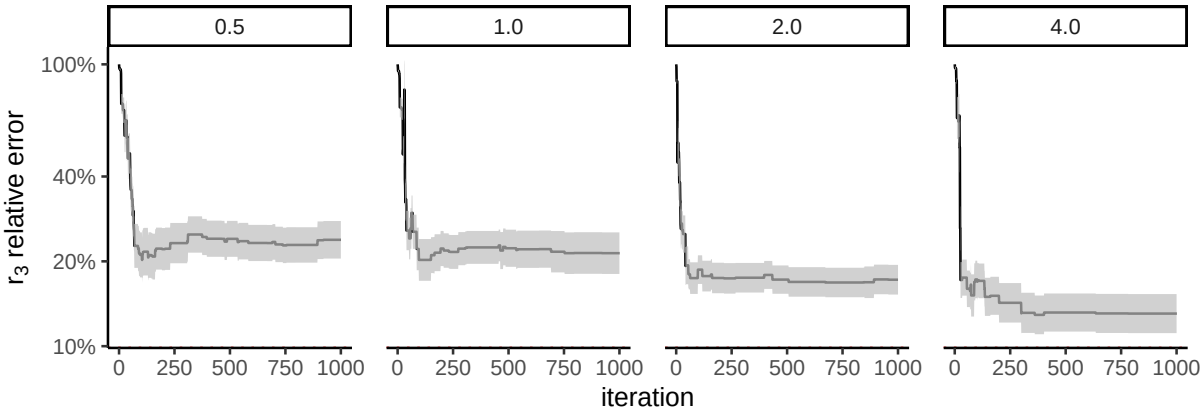


Figure C.11. r_3 **estiamtion convergence with time.**

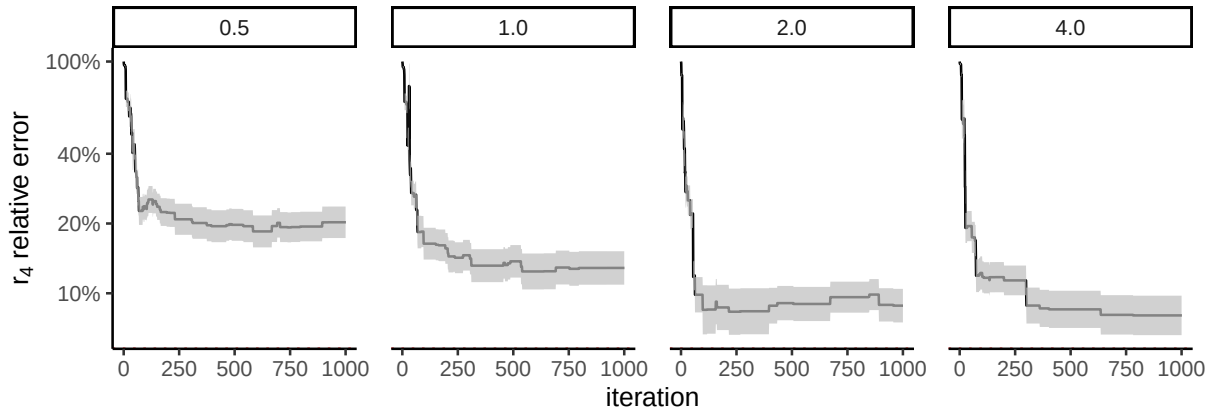


Figure C.12. r_4 **estiamtion convergence with time.**

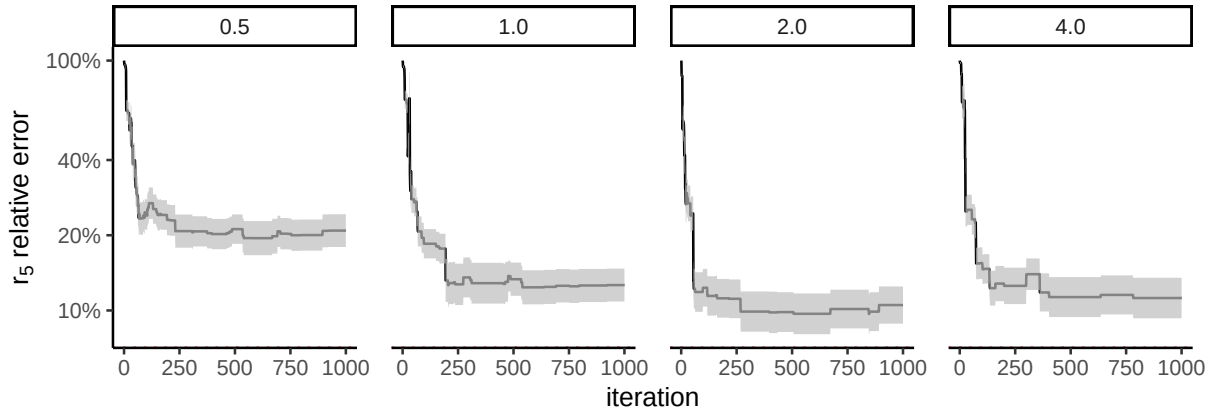


Figure C.13. r_5 **estiamtion convergence with time.**

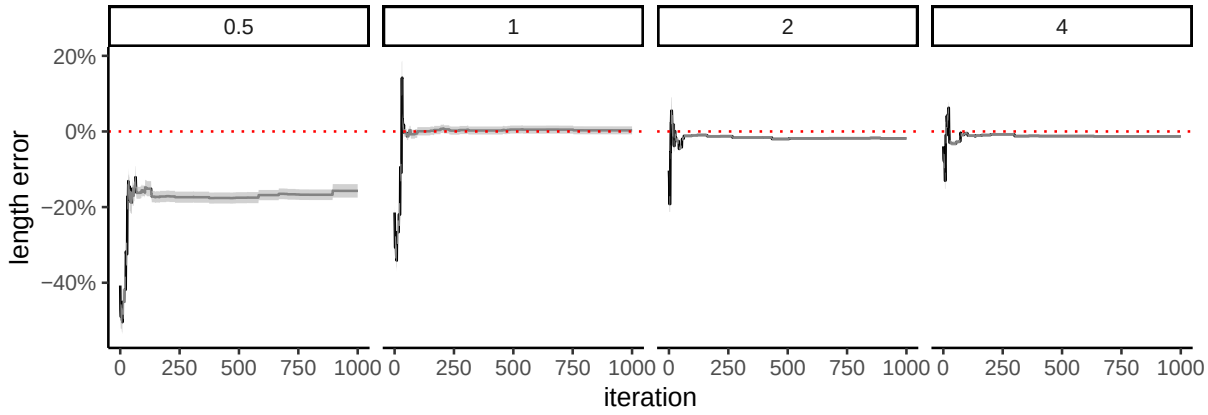


Figure C.14. Genome length convergence with time.

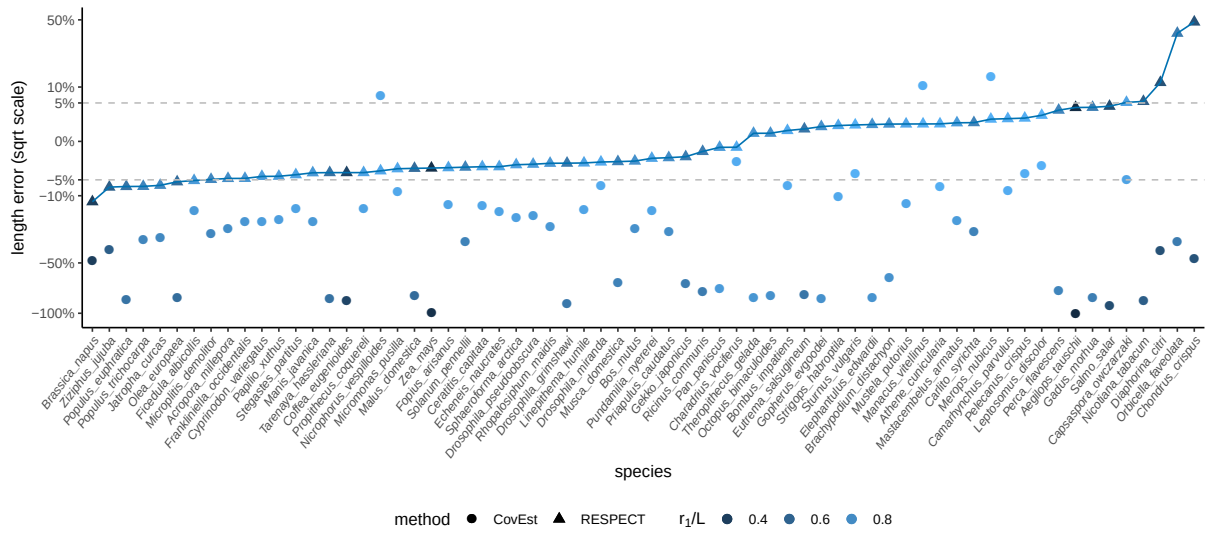


Figure C.15. Genome length estimation error of RESPECT and CovEst. The coverage is 1X, and the y-axis is in square-root scale. The sign of error indicates overestimation or underestimation. The dashed lines mark the region that the absolute value of error is less than 5%.

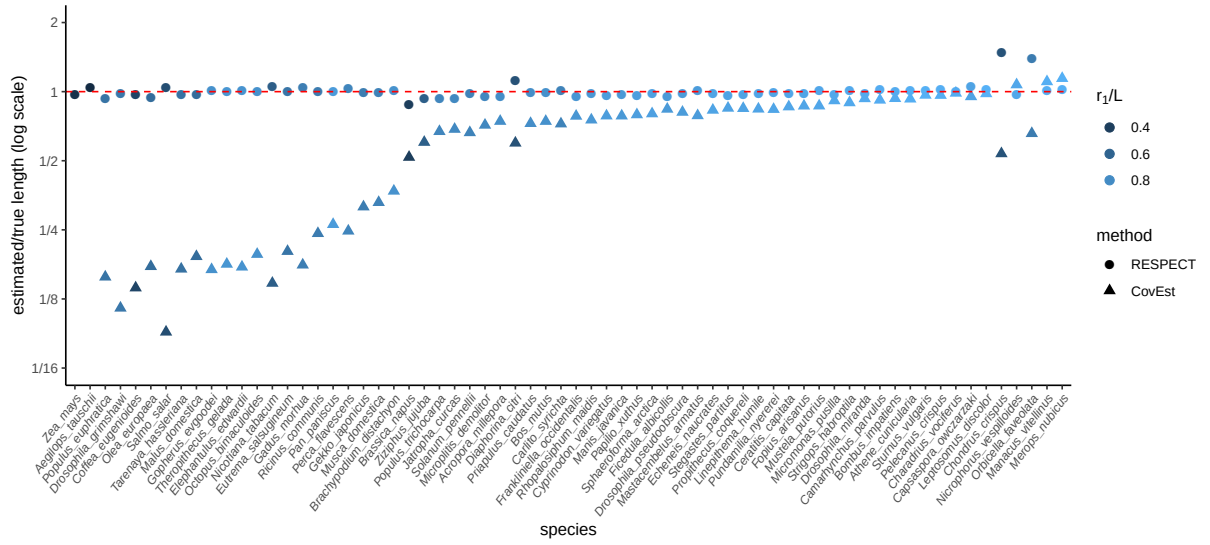


Figure C.16. Estimated to true genome length ratio. Comparing RESPECT and CovEst over 66 test species with genomes skimmed at 1X coverage. The y-axis is plotted in log scale, and the red dashed line at $y = 1$ is the grand truth (no error). Two genomes (*A. tauschii* (0.002) and *Z. mays* (0.003)) that CovEst had extremely low estimated to true ratios were removed to improve readability

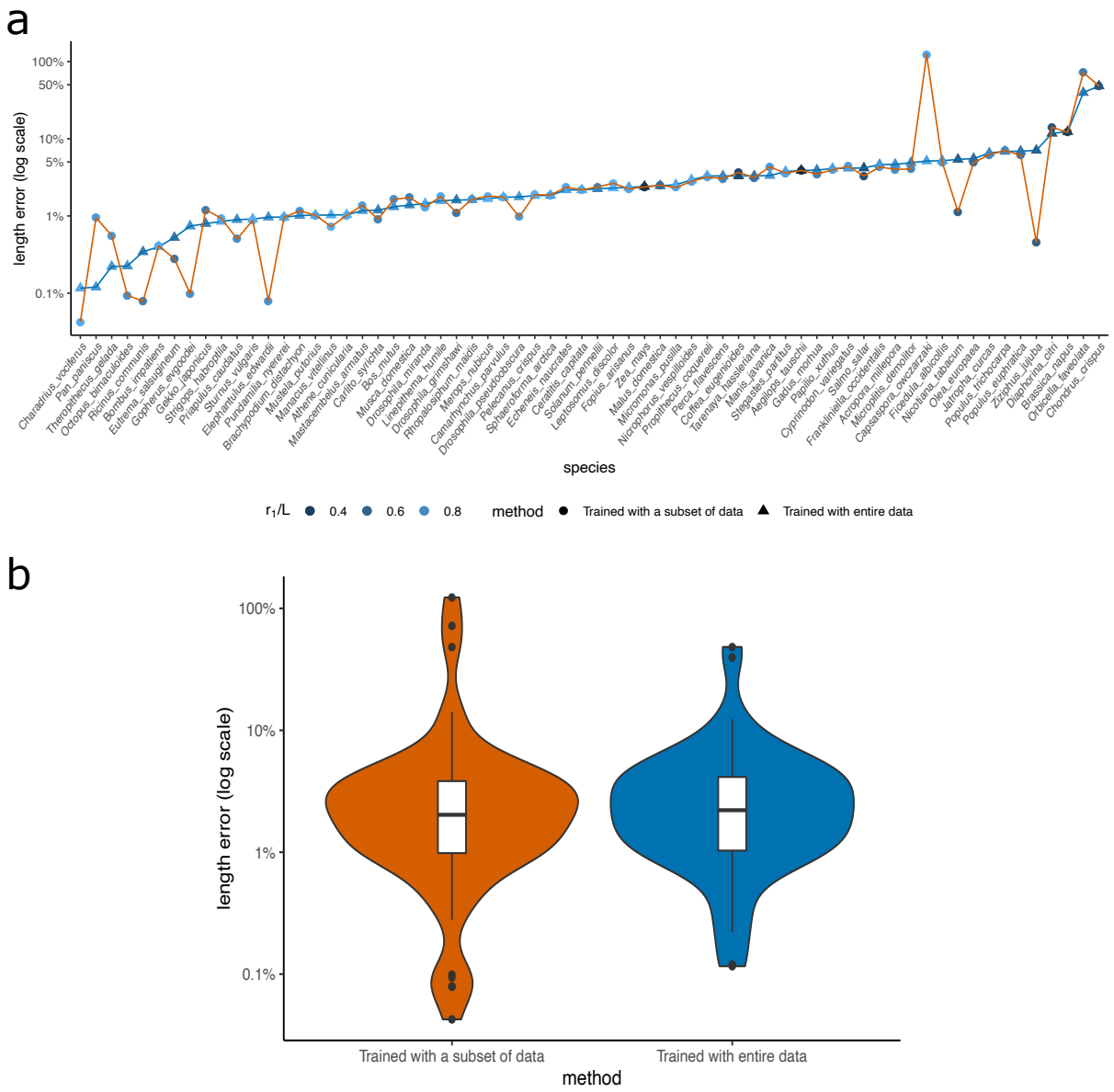


Figure C.17. Impact of training data on length estimation accuracy. RESPECT was trained on a subset of genomes (50 of 129 mammalian genomes and 50 of 195 invertebrate genomes were removed), and the error plotted (circles) along with the error on the original training set (triangles). (a) The error per genome is plotted in log scale on the y-axis. (b) The distribution of error values with RESPECT trained on the subset (blue) and the entire data set (red).

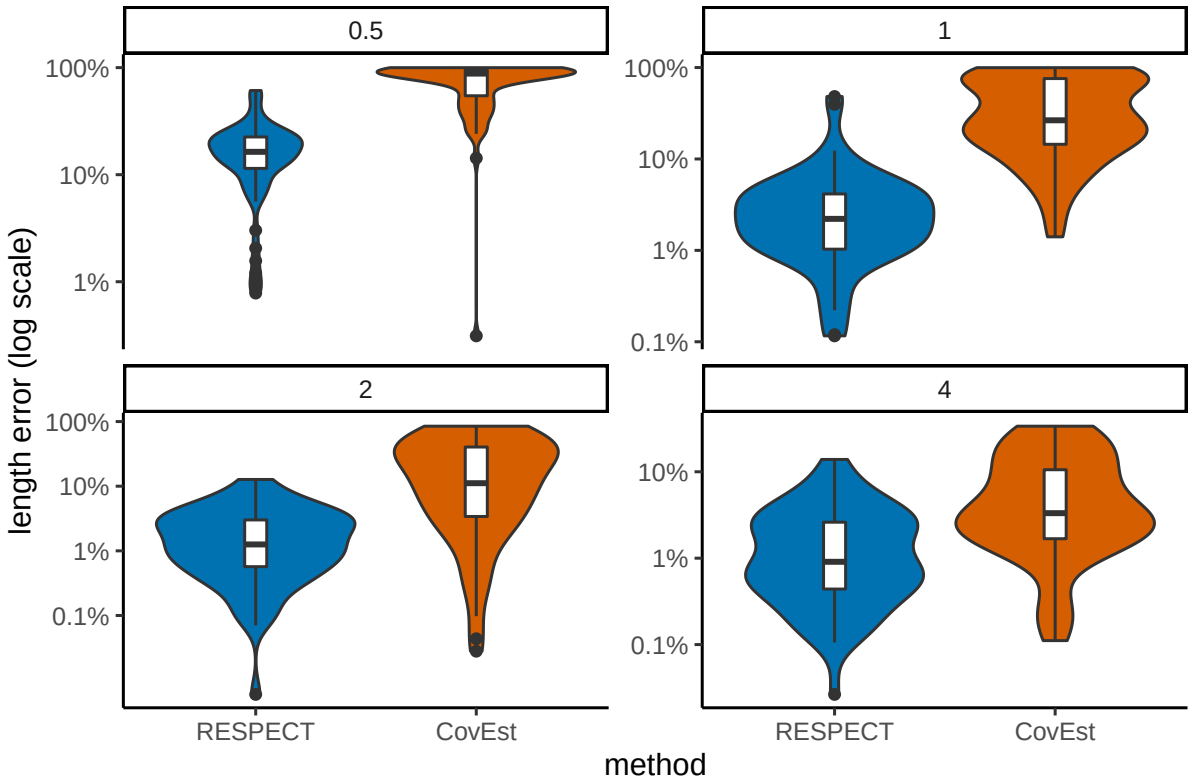


Figure C.18. Length estimation error on simulated data at different coverages. The distribution of error made by RESPECT and CovEst in estimating the length of 66 test genomes skimmed at 0.5X, 1X, 2X, and 4X coverage. The y-axis is plotted in log scale.

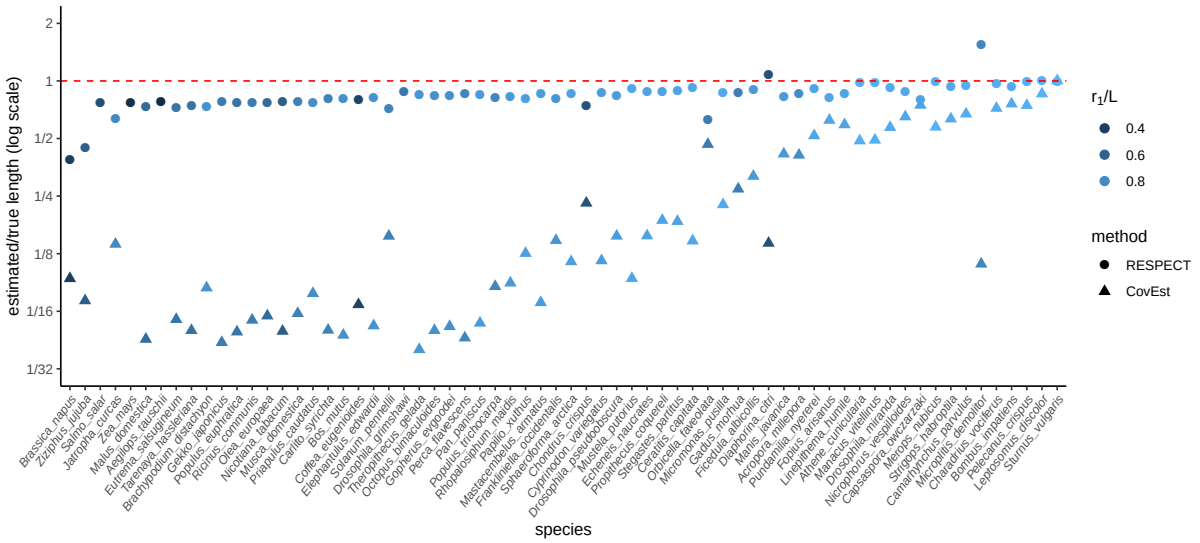


Figure C.19. Estimated to true genome length ratio. Comparing RESPECT and CovEst over 66 test species with genomes skimmed at 0.5X coverage. The y-axis is plotted in log scale, and the red dashed line at $y = 1$ is the grand truth (no error). Four genomes (*D. grimshawi* (0.0004), *S. salar* (0.0006), *A. tauschii* (0.0012), and *Z. mays* (0.0016)) that CovEst had extremely low estimated to true ratios were removed to improve readability.

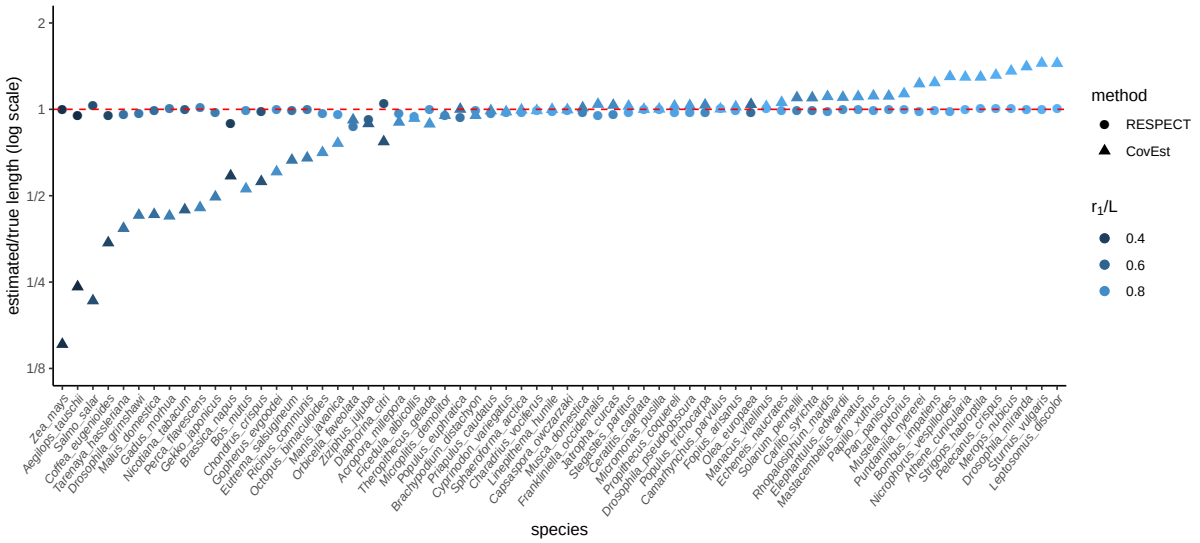


Figure C.20. Estimated to true genome length ratio. Comparing RESPECT and CovEst over 66 test species with genomes skimmed at 2X coverage. The y-axis is plotted in log scale, and the red dashed line at $y = 1$ is the grand truth (no error).

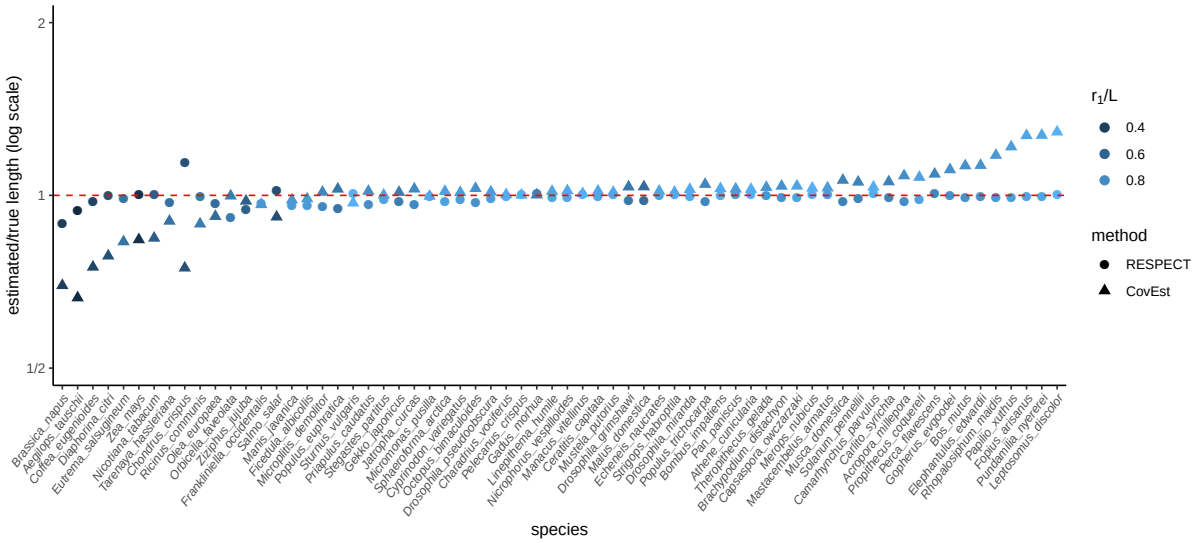


Figure C.21. Estimated to true genome length ratio. Comparing RESPECT and CovEst over 66 test species with genomes skimmed at 4X coverage. The y-axis is plotted in log scale, and the red dashed line at $y = 1$ is the grand truth (no error). Four genomes (*D. grimshawi* (0.0004), *S. salar* (0.0006), *A. tauschii* (0.0012), and *Z. mays* (0.0016)) that CovEst had extremely low estimated to true ratios were removed to improve readability.

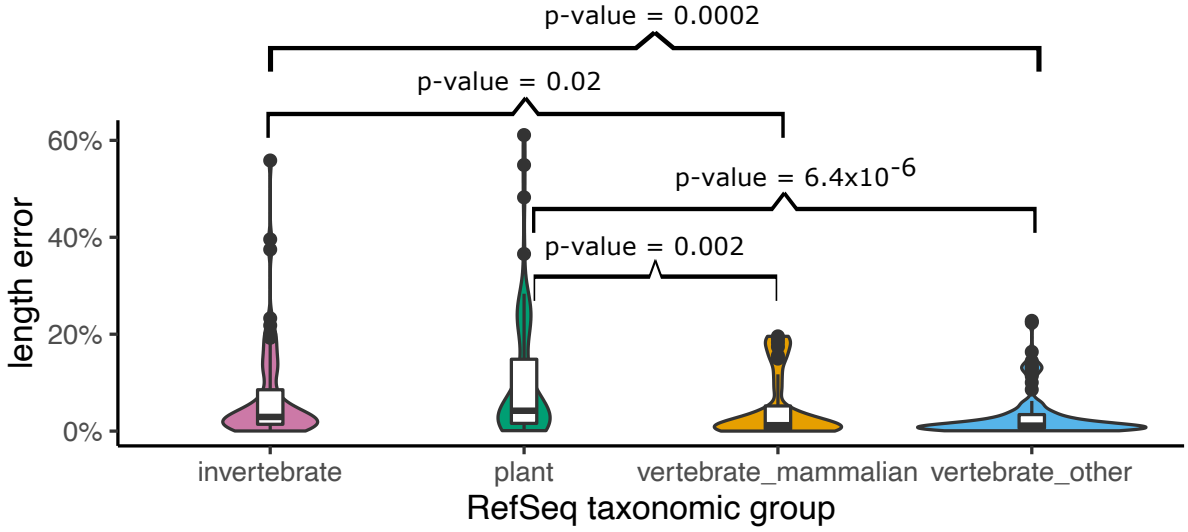


Figure C.22. Distribution of length estimation error over four major taxonomic groups. Significant p-values (0.05 threshold) computed using Mann-Whitney U test are added to the plot. Plants and invertebrates have higher error rates compared to vertebrates species in our test dataset.

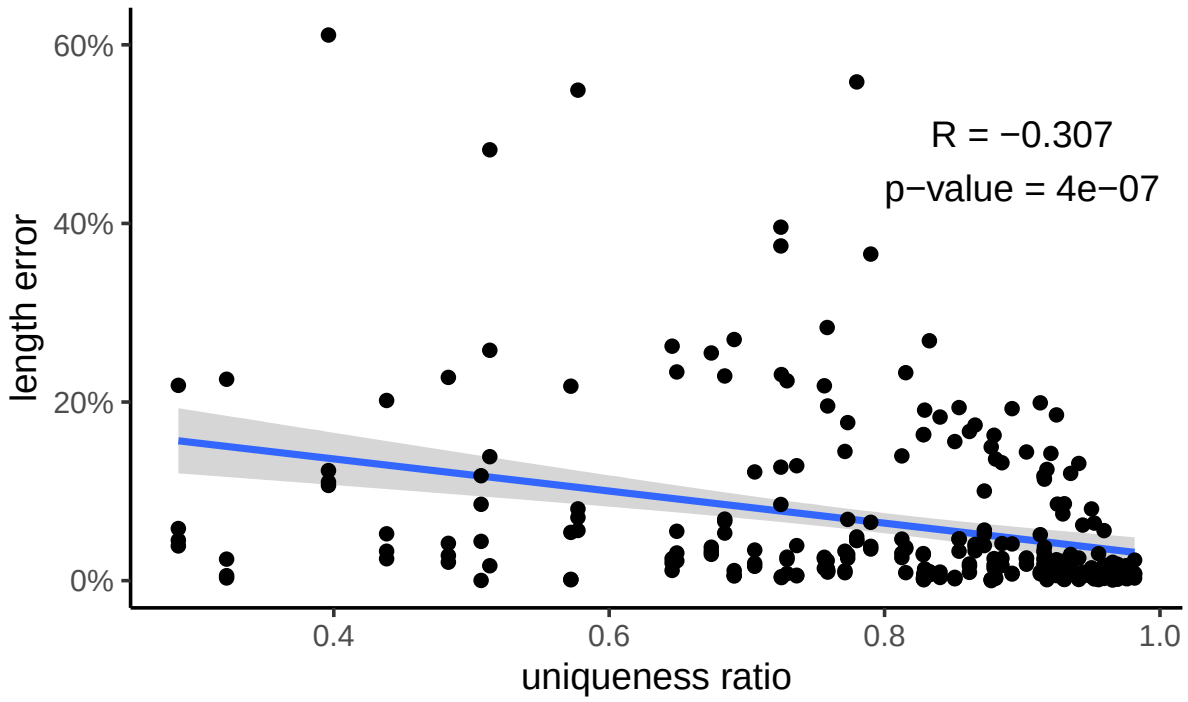


Figure C.23. Length estimation error vs. uniqueness ratio. Negative correlation between RESPECT's error and uniqueness ratio of the genome.

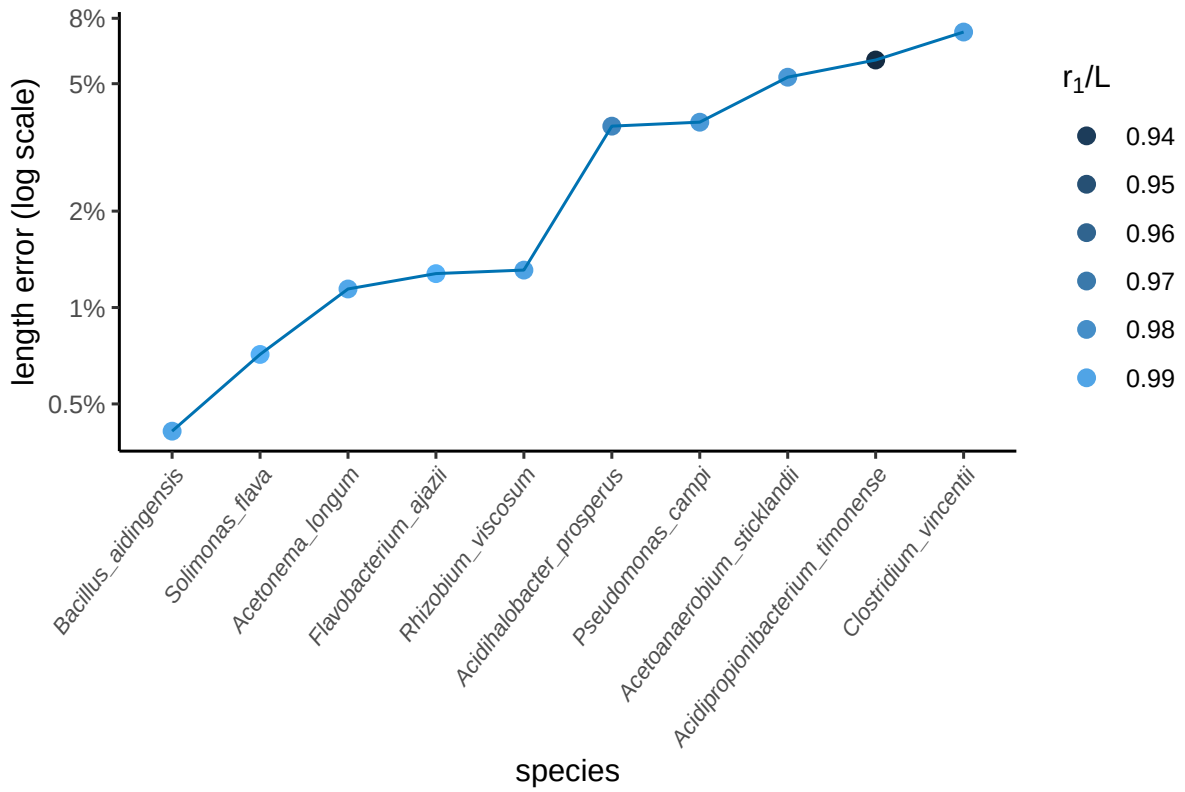


Figure C.24. Length estimation error for 10 bacterial genomes. The 10 bacterial genomes were selected at random from RefSeq and genome-skims were simulated at 1X coverage. The relative error of the estimated length is plotted in log scale on the y-axis.

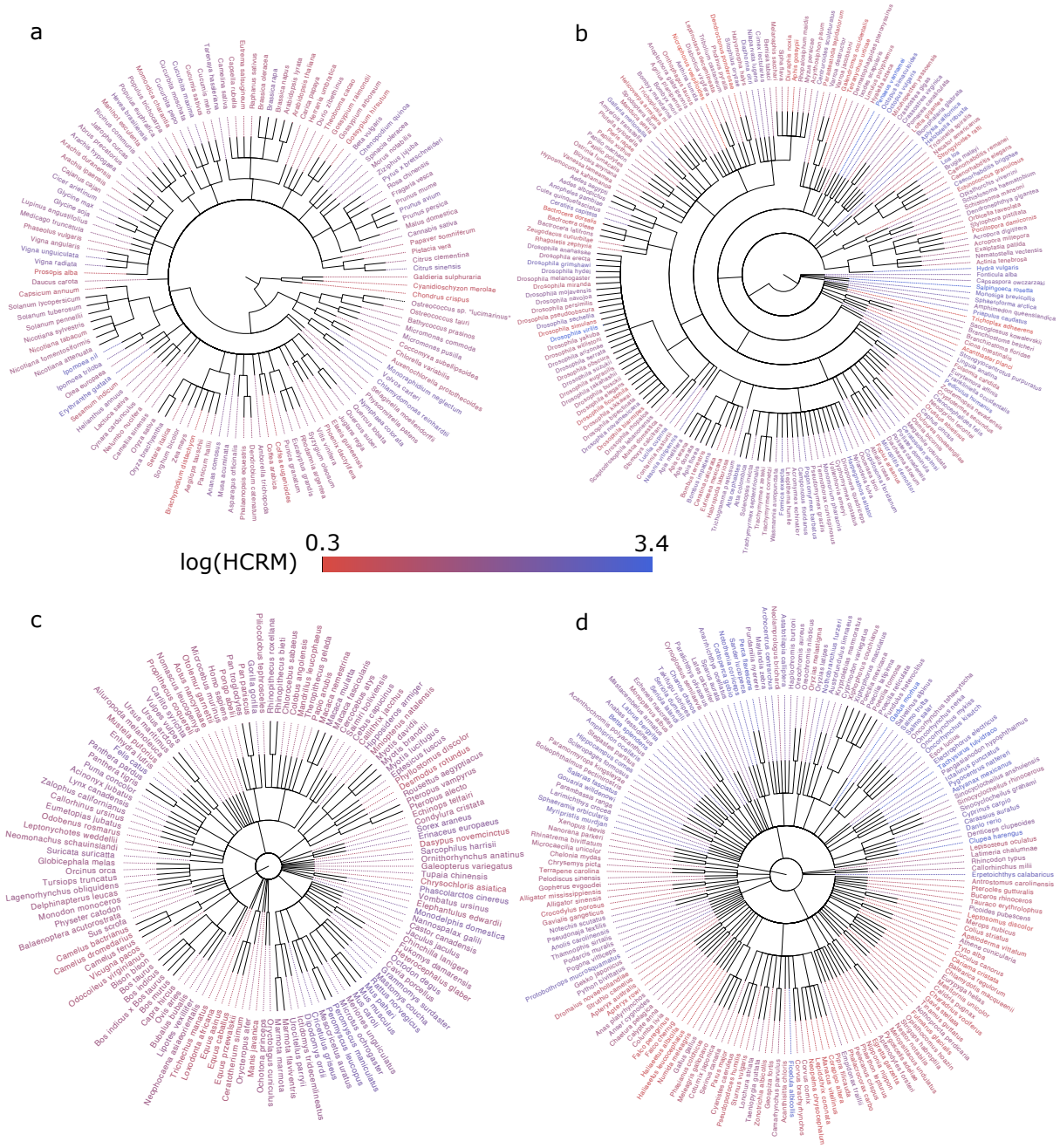


Figure C.25. Whole RefSeq taxonomy with HCRM annotation. Colors are based on logarithm of HCRM values for each genome. (a) Plants. (b) Invertebrates. (c) Mammals. (d) Other vertebrates.

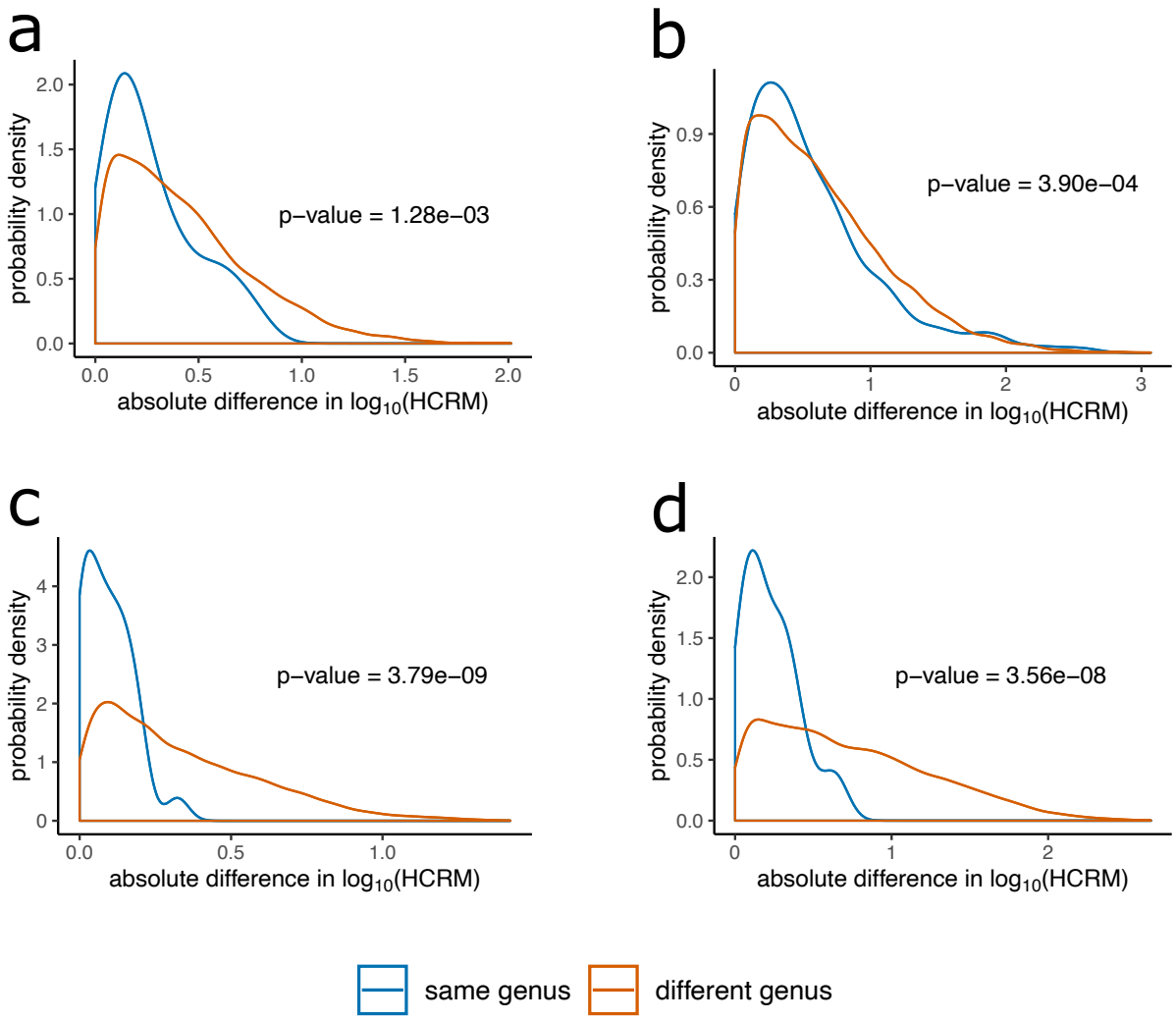


Figure C.26. Distributions of intra-generic versus inter-generic differences in HCRM for pairs of RefSeq species. (a) Plants. (b) Invertebrates. (c) Mammals. (d) Other vertebrates.

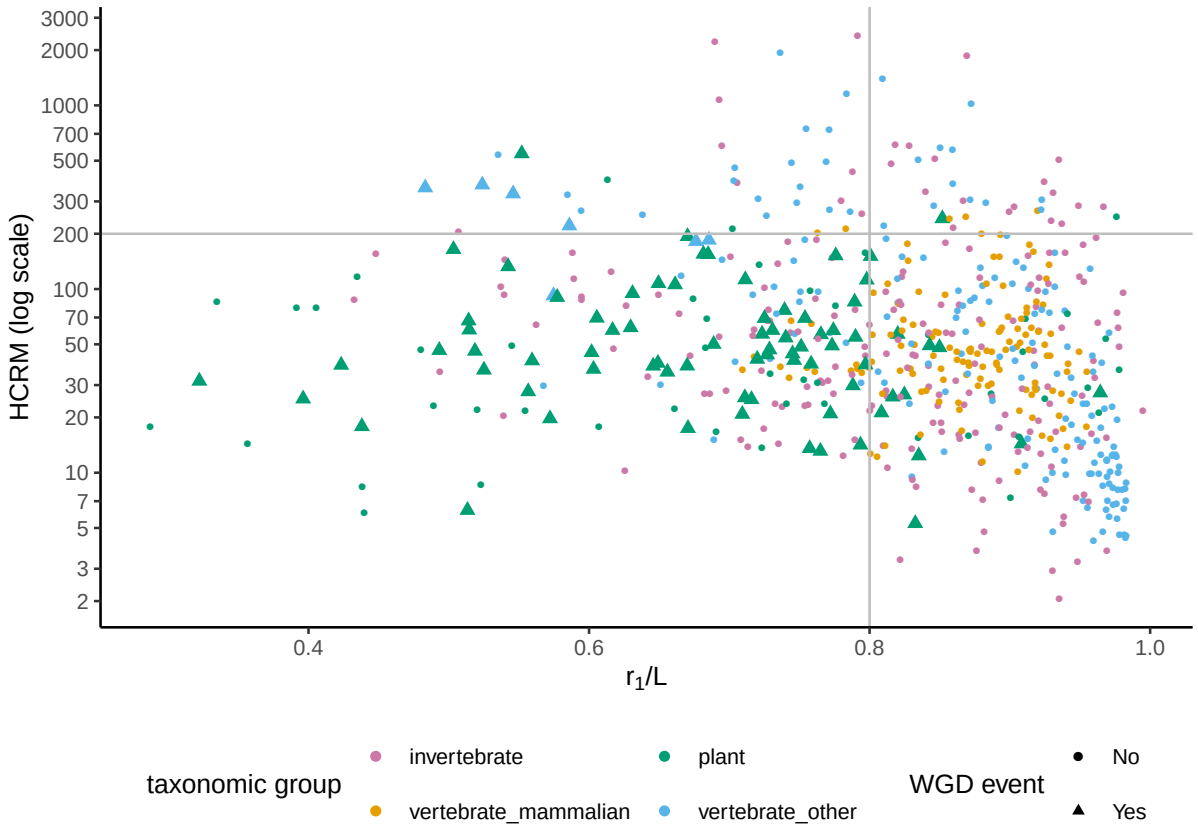


Figure C.27. High copy repeats per million versus uniqueness ratio among genomes with and without known recent WGD events. HRCM values are computed directly from the genome assemblies.

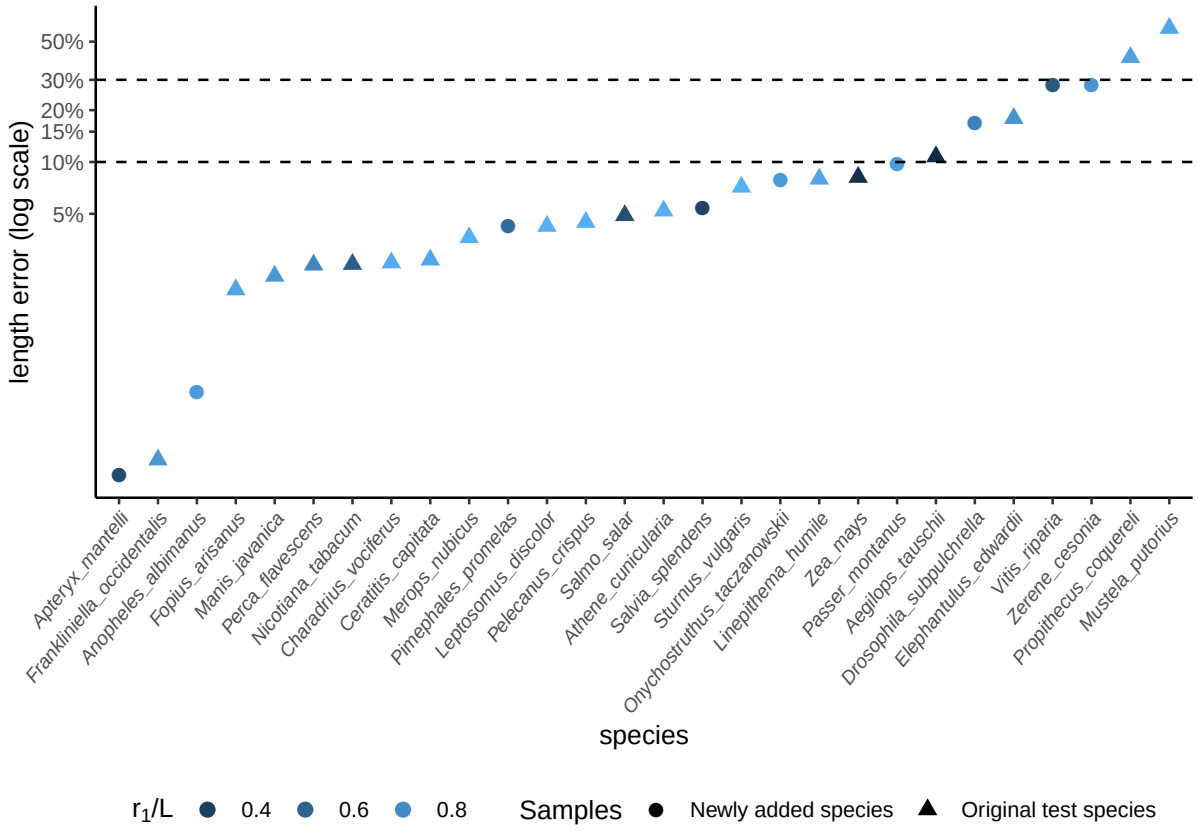


Figure C.28. Estimating genome length using SRA data. RESPECT was test on 10 new samples (chosen at random) made available since the original submission of the manuscript. One of the samples was removed during the preprocessing due to high duplication rate. The results for the remaining 9 samples are plotted along with the original test species. Two newly added samples with high error are *Z. cesonia* and *V. riparia*. RESPECT overestimates their genome length by %28. It could be the case that the assemblies are missing some repetitive sequences (especially *V. riparia* which a has highly repetitive genome), considering that for both species there is a gap between reported total sequence length and total ungapped length.

Table C.1. SRA preprocessing results. The percentage of identified duplicate and contaminant reads are provided. Samples with more than 40% total duplication plus contamination (marked with an asterisk) were discarded.

Species	Common name	Run accession	Genome length (RefSeq)	Duplication	Contamination	Total	RESPECT error	CovEst error
Pelecanus crispus	Dalmatian pelican	SRR959397	1160924693	4%	3%	7%	-4%	-42%
Zea mays	Maize	SRR2960981	2135083061	1%	7%	8%	-8%	-90%
Leptosomus discolor	Cuckoo roller	SRR956935	1136244952	6%	3%	9%	4%	-17%
Merops nubicus	Northern carmine bee-eater	SRR958515	1062961556	6%	3%	9%	4%	-13%
Nicotiana tabacum	Cultivated tobacco	SRR955758	3643471356	7%	3%	10%	3%	-85%
Aegilops tauschii	Tausch's goatgrass	SRR5170323	4327321625	5%	7%	12%	-11%	-91%
Frankliniella occidentalis	Western flower thrips	SRR1300141	274989634	7%	5%	12%	0%	-2%
Mustela putorius	Domestic ferret	SRR085103	2410879678	9%	4%	13%	-60%	-61%
Salmo salar	Atlantic salmon	SRR1264544	2966890203	7%	8%	15%	-5%	-89%
Ceratitis capitata	Mediterranean fruit fly	SRR847379	436490799	10%	7%	17%	-3%	-40%
Athene cunicularia	Burrowing owl	SRR6670174	1157069330	15%	4%	19%	-5%	-6%
Manis javanica	Sunda pangolin	SRR3929782	2547395906	7%	16%	23%	-2%	-58%
Elephantulus edwardii	Cape sengi	SRR387354	3843982861	9%	15%	24%	18%	-90%
Propithecus coquereli	Coquerel's sifaka	SRR1657020	2798152141	7%	20%	27%	-41%	-54%
Perca flavescens	Yellow perch	SRR8482300	877456336	1%	28%	29%	3%	-69%
Fopius arisanus	A Braconid wasp	SRR1560668	153631861	28%	2%	30%	-2%	-43%
Sturnus vulgaris	Common starling	SRR2240710	1036755994	15%	20%	35%	7%	-44%
Charadrius vociferus	Killdeer	SRR944000	1219859583	37%	1%	38%	-3%	-84%
Linepithema humile	Argentine ant	SRR059226	219500750	10%	29%	39%	-8%	-34%
*Cyprinodon variegatus	Sheepshead minnow	SRR1261820	1035184475	24%	19%	43%		
*Bombus impatiens	Eastern bumble bee	SRR1575221	246856484	12%	32%	44%		
*Priapulus caudatus	Penis worm	SRR649590	511738253	18%	27%	45%		
*Diaphorina citri	Citrus psylla	SRR189236	485705082	39%	11%	50%		
*Drosophila miranda	Fruit fly	SRR789694	136728780	49%	12%	61%		
*Musca domestica	Housefly	SRR650115	750403944	12%	50%	62%		
*Stegastes partitus	Bicolor damselfish	SRR649426	800491834	34%	41%	75%		
*Bos mutus	Wild yak	SRR361227	2645161911	64%	12%	76%		
*Populus euphratica	Desert poplar	SRR616245	496032534	94%	4%	98%		
*Carlito syrichta	Philippine tarsier	SRR3502922	3453864774	59%	82%	141%		

Table C.2. List of species with recent WGD events.

Species	UR	HCRM
Amborella trichopoda	0.85	48
Ananas comosus	0.71	113
Arabidopsis lyrata	0.71	26
Arabidopsis thaliana	0.91	14
Asparagus officinalis	0.51	60
Beta vulgaris	0.74	77
Brachypodium distachyon	0.83	5
Brassica napus	0.40	25
Brassica oleracea	0.71	21
Brassica rapa	0.80	112
Cajanus cajan	0.67	38
Cannabis sativa	0.51	67
Carica papaya	0.81	21
Chenopodium quinoa	0.50	165
Chlamydomonas reinhardtii	0.85	242
Chondrus crispus	0.51	6
Cicer arietinum	0.66	106
Citrus clementina	0.75	41
Citrus sinensis	0.68	154
Coccomyxa subellipsoidea	0.96	27
Cucumis melo	0.84	49
Cucumis sativus	0.82	26
Daucus carota	0.66	35

Continued on next page

Table C.2 – *Continued from previous page*

Species	**UR**	**HCRM**
Dendrobium catenatum	0.79	30
Eucalyptus grandis	0.69	50
Fragaria vesca	0.82	57
Glycine max	0.65	107
Glycine soja	0.63	95
Gossypium raimondii	0.76	13
Ipomoea nil	0.55	547
Ipomoea triloba	0.67	193
Jatropha curcas	0.79	55
Juglans regia	0.60	45
Lactuca sativa	0.53	36
Lupinus angustifolius	0.73	60
Malus domestica	0.65	38
Manihot esculenta	0.67	17
Medicago truncatula	0.77	57
Morus notabilis	0.72	57
Musa acuminata	0.79	85
Nelumbo nucifera	0.82	27
Nicotiana attenuata	0.49	46
Nicotiana sylvestris	0.56	41
Nicotiana tabacum	0.57	20
Nicotiana tomentosiformis	0.60	37
Nymphaea colorata	0.78	152
Olea europaea	0.65	40

Continued on next page

Table C.2 – *Continued from previous page*

Species	**UR**	**HCRM**
Oncorhynchus kisutch	0.52	370
Oncorhynchus mykiss	0.59	221
Oncorhynchus nerka	0.68	182
Oncorhynchus tshawytscha	0.57	92
Oryza sativa	0.75	44
Panicum hallii	0.56	28
Papaver somniferum	0.44	18
Phalaenopsis equestris	0.72	25
Phoenix dactylifera	0.77	21
Physcomitrella patens	0.73	47
Populus trichocarpa	0.77	49
Prunus avium	0.68	155
Prunus mume	0.77	60
Prunus persica	0.75	69
Punica granatum	0.72	42
Pyrus x bretschneideri	0.54	133
Quercus lobata	0.63	62
Quercus suber	0.62	60
Ricinus communis	0.73	69
Rosa chinensis	0.61	70
Salmo salar	0.48	356
Salmo trutta	0.55	330
Salvelinus alpinus	0.69	185
Selaginella moellendorffii	0.42	39

Continued on next page

Table C.2 – *Continued from previous page*

Species	**UR**	**HCRM**
Sesamum indicum	0.84	12
Setaria italica	0.76	14
Solanum lycopersicum	0.80	39
Solanum pennellii	0.76	39
Solanum tuberosum	0.74	54
Sorghum bicolor	0.52	46
Syzygium oleosum	0.75	48
Theobroma cacao	0.79	14
Vitis vinifera	0.73	44
Volvox carteri	0.80	151
Zea mays	0.32	32
Ziziphus jujuba	0.58	90

Bibliography

[1] Jurka J. Repbase update: a database and an electronic journal of repetitive elements. Trends Genet. 2000 Sep;16(9):418-20.

[2] Venter JC, Adams MD, Myers EW, Li PW, Mural RJ, Sutton GG, et al. The Sequence of the Human Genome. Science. 2001;291(5507):1304-51. Available from: https://www.science.org/doi/abs/10.1126/science.1058040.

[3] Bartel DP. MicroRNAs: target recognition and regulatory functions. Cell. 2009 Jan;136(2):215-33.

[4] Dunham I, et al. An integrated encyclopedia of DNA elements in the human genome. Nature. 2012 Sep;489(7414):57-74.

[5] Birney E, et al. Identification and analysis of functional elements in 1human genome by the ENCODE pilot project. Nature. 2007 Jun;447(7146):799-816.

[6] Ingolia NT, Ghaemmaghami S, Newman JR, Weissman JS. Genome-wide analysis in vivo of translation with nucleotide resolution using ribosome profiling. Science. 2009 Apr;324(5924):218-23.

[7] Ingolia NT, Brar GA, Rouskin S, McGeachy AM, Weissman JS. The ribosome profiling strategy for monitoring translation in vivo by deep sequencing of ribosome-protected mRNA fragments. Nat Protoc. 2012 Jul;7(8):1534-50.

[8] Pinkel D, Segraves R, Sudar D, Clark S, Poole I, Kowbel D, et al. High resolution analysis of DNA copy number variation using comparative genomic hybridization to microarrays. Nat Genet. 1998 Oct;20(2):207-11.

[9] Feuk L, Carson AR, Scherer SW. Structural variation in the human genome. Nat Rev Genet. 2006 Feb;7(2):85-97.

[10] Rosenberg KV, Dokter AM, Blancher PJ, Sauer JR, Smith AC, Smith PA, et al. Decline of the North American avifauna. Science. 2019 sep:eaaw1313. Available from: http://www.sciencemag.org/lookup/doi/10.1126/science.aaw1313.

[11] Lewin HA, Robinson GE, Kress WJ, Baker WJ, Coddington J, Crandall KA, et al. Earth BioGenome Project: Sequencing life for the future of life. Proceedings of the National

Academy of Sciences. 2018 apr;115(17):4325-33. Available from: http://www.pnas.org/lookup/doi/10.1073/pnas.1720115115.

[12] Coissac E, Hollingsworth PM, Lavergne S, Taberlet P. From barcodes to genomes: extending the concept of DNA barcoding. Molecular Ecology. 2016;25(7):1423-8. Available from: http://dx.doi.org/10.1111/mec.13549.

[13] Straub SCK, Parks M, Weitemier K, Fishbein M, Cronn RC, Liston A. Navigating the tip of the genomic iceberg: Next-generation sequencing for plant systematics. American Journal of Botany. 2012 feb;99(2):349-64. Available from: http://www.amjbot.org/cgi/doi/10.3732/ajb.1100335.

[14] France Génomique - Mutualisation des compétences et des équipements français pour l'analyse génomique et la bio-informatique;. Accessed 16 October 2018. https://www.france-genomique.org/.

[15] Norwegian Barcode of Life (NorBOL);. Accessed 16 October 2018. http://www.norbol.org/en/.

[16] DNAmark;. Accessed 16 October 2018. http://dnamark.ku.dk/english/.

[17] Tonti-Filippini J, Nevill PG, Dixon K, Small I. What can we do with 1000 plastid genomes? Plant Journal. 2017;90(4):808-18.

[18] Guenther MG, Levine SS, Boyer LA, Jaenisch R, Young RA. A chromatin landmark and transcription initiation at most promoters in human cells. Cell. 2007;130(1):77-88.

[19] Zarrei M, MacDonald JR, Merico D, Scherer SW. A copy number variation map of the human genome. Nature Reviews Genetics. 2015;16(3):172-83.

[20] Wang YH. On the number of successes in independent trials. Statistica Sinica. 1993:295-312.

[21] Hong Y. On computing the distribution function for the Poisson binomial distribution. Computational Statistics & Data Analysis. 2013;59:41-51.

[22] Deshpande V, Luebeck J, Bakhtiari M, Nguyen NPD, Turner KM, Schwab R, et al. Reconstructing and characterizing focal amplifications in cancer using AmpliconArchitect. bioRxiv. 2018. Available from: https://www.biorxiv.org/content/early/2018/10/30/457333.

[23] Ernst J, Kellis M. Discovery and characterization of chromatin states for systematic annotation of the human genome. Nature biotechnology. 2010;28(8):817-25.

[24] aCGH copy number data;. https://wiki.nci.nih.gov/display/TCGA/aCGH+copy+number+data.

[25] Turner KM, Deshpande V, Beyter D, Koga T, Rusert J, Lee C, et al. Extrachromosomal oncogene amplification drives tumour evolution and genetic heterogeneity. Nature. 2017 mar;543(7643):122-5. Available from: http://www.nature.com/articles/nature21356.

[26] Hebert PDN, Cywinska A, Ball SL, deWaard JR. Biological identifications through DNA barcodes. Proceedings of the Royal Society B: Biological Sciences. 2003;270(1512):313-21. Available from: http://rspb.royalsocietypublishing.org/cgi/doi/10.1098/rspb.2002.2218.

[27] Savolainen V, Cowan RS, Vogler AP, Roderick GK, Lane R. Towards writing the encyclopaedia of life: an introduction to DNA barcoding. Philosophical Transactions of the Royal Society B: Biological Sciences. 2005;360(1462):1805-11. Available from: http://rstb.royalsocietypublishing.org/cgi/doi/10.1098/rstb.2005.1730.

[28] Taberlet P, Coissac E, Pompanon F, Brochmann C, Willerslev E. Towards next-generation biodiversity assessment using DNA metabarcoding. Molecular Ecology. 2012 4;21(8):2045-50. Available from: http://doi.wiley.com/10.1111/j.1365-294X.2012.05470.x.

[29] Seifert KA, Samson RA, deWaard JR, Houbraken J, Levesque CA, Moncalvo JM, et al. Prospects for fungus identification using CO1 DNA barcodes, with Penicillium as a test case. Proceedings of the National Academy of Sciences. 2007;104(10):3901-6. Available from: http://www.pnas.org/cgi/doi/10.1073/pnas.0611691104.

[30] Vences M, Thomas M, van der Meijden A, Chiari Y, Vieites DR. Comparative performance of the 16S rRNA gene in DNA barcoding of amphibians. Frontiers in zoology. 2005;2:5.

[31] Ardura A, Linde AR, Moreira JC, Garcia-Vazquez E. DNA barcoding for conservation and management of Amazonian commercial fish. Biological Conservation. 2010;143(6):1438-43.

[32] Hollingsworth PM, Forrest LL, Spouge JL, Hajibabaei M, Ratnasingham S, van der Bank M, et al. A DNA barcode for land plants. Proceedings of the National Academy of Sciences. 2009 8;106(31):12794-7. Available from: http://www.pnas.org/content/106/31/12794.fullhttp://www.pnas.org/cgi/doi/10.1073/pnas.0905845106.

[33] Schoch CL, Seifert KA, Huhndorf S, Robert V, Spouge JL, Levesque CA, et al. Nuclear ribosomal internal transcribed spacer (ITS) region as a universal DNA barcode marker for Fungi. Proceedings of the National Academy of Sciences. 2012 4;109(16):6241-6. Available from: http://www.pnas.org/cgi/doi/10.1073/pnas.1117018109.

[34] Zhang Ds, Zhou Yd, Wang Cs, Rouse G. A new species of Ophryotrocha (Annelida, Eunicida, Dorvilleidae) from hydrothermal vents on the Southwest Indian Ridge. ZooKeys. 2017 8;687:1-9. Available from: https://zookeys.pensoft.net/articles.php?id=13046.

[35] Hedin MC, Maddison WP. A Combined Molecular Approach to Phylogeny of the Jumping Spider Subfamily Dendryphantinae (Araneae: Salticidae). Molecular Phylogenetics and Evolution. 2001 3;18(3):386-403. Available from: http://linkinghub.elsevier.com/retrieve/pii/S1055790300908832.

[36] Taylor KH, Rouse GW, Messing CG. Systematics of Himerometra (Echinodermata: Crinoidea: Himerometridae) based on morphology and molecular data. Zoological Journal of the Linnean Society. 2017 10;181(2):342-56. Available from: http://dx.doi.org/10.1093/zoolinnean/zlx009.

[37] Ratnasingham S, Hebert PDN. BOLD : The Barcode of Life Data System (www.barcodinglife.org). Molecular Ecology Notes. 2007;7(April 2016):355-64.

[38] Steinke D, Vences M, Salzburger W, Meyer A. TaxI: a software tool for DNA barcoding using distance methods. Philosophical Transactions of the Royal Society B: Biological Sciences. 2005;360(1462):1975-80. Available from: http://rstb.royalsocietypublishing.org/cgi/doi/10.1098/rstb.2005.1729.

[39] Mirarab S, Nguyen N, Warnow T. SEPP: SATé-Enabled Phylogenetic Placement. Pacific Symposium On Biocomputing. 2012:247-58. Available from: http://www.ncbi.nlm.nih.gov/pubmed/22174280.

[40] Berger SA, D K , Stamatakis A, Krompass D. Performance, Accuracy, and Web Server for Evolutionary Placement of Short Sequence Reads under Maximum Likelihood. Systematic Biology. 2011 5;60(3):291-302. Available from: http://sysbio.oxfordjournals.org/cgi/content/abstract/60/3/291http://sysbio.oxfordjournals.org/content/60/3/291.abstracthttp://sysbio.oxfordjournals.org/content/60/3/291.full.pdfhttp://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=3078422&tool=pmc.

[41] Matsen FA, Kodner RB, Armbrust EV. pplacer: linear time maximum-likelihood and Bayesian phylogenetic placement of sequences onto a fixed reference tree. BMC bioinformatics. 2010 1;11(1):538. Available from: http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=3098090&tool=pmcentrez&rendertype=abstract.

[42] Hickerson MJ, Meyer CP, Moritz C, Hedin M. DNA Barcoding Will Often Fail to Discover New Animal Species over Broad Parameter Space. Systematic Biology. 2006 10;55(5):729-39. Available from: http://academic.oup.com/sysbio/article/55/5/729/1663614/DNA-Barcoding-Will-Often-Fail-to-Discover-New.

[43] Quicke DLJ, Alex Smith M, Janzen DH, Hallwachs W, Fernandez-Triana J, Laurenne NM, et al. Utility of the DNA barcoding gene fragment for parasitic wasp phylogeny (Hymenoptera: Ichneumonoidea): Data release and new measure of taxonomic congruence. Molecular Ecology Resources. 2012 7;12(4):676-85. Available from: http://doi.wiley.com/10.1111/j.1755-0998.2012.03143.x.

[44] Blaisdell BE. A measure of the similarity of sets of sequences not requiring sequence alignment. Proceedings of the National Academy of Sciences of the United States of America. 1986 7;83(14):5155-9. Available from: http://www.ncbi.nlm.nih.gov/pubmed/3460087http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=PMC323909.

[45] Vinga S, Almeida J. Alignment-free sequence comparison–a review. Bioinformatics. 2003 3;19(4):513-23. Available from: http://www.ncbi.nlm.nih.gov/pubmed/12611807https://academic.oup.com/bioinformatics/article-lookup/doi/10.1093/bioinformatics/btg005.

[46] Zielezinski A, Vinga S, Almeida J, Karlowski WM. Alignment-free sequence comparison: benefits, applications, and tools. Genome Biology. 2017 dec;18(1):186. Available from: http://genomebiology.biomedcentral.com/articles/10.1186/s13059-017-1319-7.

[47] Haubold B, Pfaffelhuber P, Domazet-Lošo M, Wiehe T. Estimating Mutation Distances from Unaligned Genomes. Journal of Computational Biology. 2009 oct;16(10):1487-500. Available from: http://www.liebertpub.com/doi/10.1089/cmb.2009.0106.

[48] Morgenstern B, Zhu B, Horwege S, Leimeister CA. Estimating evolutionary distances between genomic sequences from spaced-word matches. Algorithms for Molecular Biology. 2015 dec;10(1):5. Available from: http://www.almob.org/content/10/1/5.

[49] Reinert G, Chew D, Sun F, Waterman MS. Alignment-free sequence comparison (I): statistics and power. Journal of computational biology : a journal of computational molecular cell biology. 2009 12;16(12):1615-34. Available from: http://www.ncbi.nlm.nih.gov/pubmed/20001252http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=PMC2818754.

[50] Thorne JL, Kishino H. Freeing phylogenies from artifacts of alignment. Molecular biology and evolution. 1992 11;9(6):1148-62. Available from: http://www.ncbi.nlm.nih.gov/pubmed/1435239.

[51] Höhl M, Ragan MA. Is multiple-sequence alignment required for accurate inference of phylogeny? Systematic Biology. 2007;56(2):206-21.

[52] Fan H, Ives AR, Surget-Groba Y, Cannon CH. An assembly and alignment-free method of phylogeny reconstruction from next-generation sequencing data. BMC Genomics. 2015 7;16(1):522. Available from: https://doi.org/10.1186/s12864-015-1647-5.

[53] Daskalakis C, Roch S. Alignment-free phylogenetic reconstruction: Sample complexity via a branching process analysis. Annals of Applied Probability. 2013;23(2):693-721.

[54] Dai Q, Yang Y, Wang T. Markov model plus k-word distributions: a synergy that produces novel statistical measures for sequence comparison. Bioinformatics. 2008 10;24(20):2296-302. Available from: http://www.ncbi.nlm.nih.gov/pubmed/18710871https://academic.oup.com/bioinformatics/article-lookup/doi/10.1093/bioinformatics/btn436.

[55] Yang K, Zhang L. Performance comparison between k-tuple distance and four model-based distances in phylogenetic tree reconstruction. Nucleic Acids Research. 2008 1;36(5):e33-3. Available from: http://www.ncbi.nlm.nih.gov/pubmed/18296485http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=PMC2275138https://academic.oup.com/nar/article-lookup/doi/10.1093/nar/gkn075.

[56] Qi J, Luo H, Hao B. CVTree: a phylogenetic tree reconstruction tool based on whole genomes. Nucleic Acids Research. 2004 7;32(Web Server):W45-7. Available from: http://www.ncbi.nlm.nih.gov/pubmed/15215347http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=PMC441500https://academic.oup.com/nar/article-lookup/doi/10.1093/nar/gkh362.

[57] Ulitsky I, Burstein D, Tuller T, Chor B. The Average Common Substring Approach to Phylogenomic Reconstruction. Journal of Computational Biology. 2006 3;13(2):336-50. Available from: http://www.ncbi.nlm.nih.gov/pubmed/16597244http://www.liebertonline.com/doi/abs/10.1089/cmb.2006.13.336.

[58] Yi H, Jin L. Co-phylog: an assembly-free phylogenomic approach for closely related organisms. Nucleic Acids Research. 2013 4;41(7):e75-5. Available from: http://www.ncbi.nlm.nih.gov/pubmed/23335788http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=PMC3627563https://academic.oup.com/nar/article-lookup/doi/10.1093/nar/gkt003.

[59] Roychowdhury T, Vishnoi A, Bhattacharya A. Next-Generation Anchor Based Phylogeny (NexABP): Constructing phylogeny from Next-generation sequencing data. Scientific Reports. 2013 12;3(1):2634. Available from: http://www.ncbi.nlm.nih.gov/pubmed/24022334http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=PMC3769656http://www.nature.com/articles/srep02634.

[60] Haubold B. Alignment-free phylogenetics and population genetics. Briefings in Bioinformatics. 2014 5;15(3):407-18. Available from: http://www.ncbi.nlm.nih.gov/pubmed/24291823https://academic.oup.com/bib/article-lookup/doi/10.1093/bib/bbt083.

[61] Morgenstern B, Schöbel S, Leimeister CA. Phylogeny reconstruction based on the length distribution of k-mismatch common substrings. Algorithms for Molecular Biology. 2017 dec;12(1):27. Available from: https://almob.biomedcentral.com/articles/10.1186/s13015-017-0118-8.

[62] Leimeister CA, Sohrabi-Jahromi S, Morgenstern B, Valencia A. Fast and accurate phylogeny reconstruction using filtered spaced-word matches. Bioinformatics. 2017 jan;33(7):btw776. Available from: https://academic.oup.com/bioinformatics/article-lookup/doi/10.1093/bioinformatics/btw776.

[63] Leimeister CA, Boden M, Horwege S, Lindner S, Morgenstern B. Fast alignment-free sequence comparison using spaced-word frequencies. Bioinformatics. 2014 jul;30(14):1991-9. Available from: http://www.ncbi.nlm.nih.gov/pubmed/24700317http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=PMC4080745https://academic.oup.com/bioinformatics/article-lookup/doi/10.1093/bioinformatics/btu177.

[64] Leimeister CA, Morgenstern B. Kmacs: the k-mismatch average common substring approach to alignment-free sequence comparison. Bioinformatics (Oxford, England). 2014 jul;30(14):2000-8. Available from: http://www.ncbi.nlm.nih.gov/pubmed/24828656http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=PMC4080746.

[65] Ondov BD, Treangen TJ, Melsted P, Mallonee AB, Bergman NH, Koren S, et al. Mash: fast genome and metagenome distance estimation using MinHash. Genome Biology. 2016 12;17(1):132. Available from: http://download.springer.com/static/pdf/329/art%253A10.1186%252Fs13059-016-0997-x.pdf?originUrl=http%3A%2F%2Fgenomebiology.biomedcentral.com%2Farticle%2F10.1186%2Fs13059-016-0997-x&token2=exp=1490224184~acl=%2Fstatic%2Fpdf%2F329%2Fart%25253A10.1186%25252F.

[66] Benoit G, Peterlongo P, Mariadassou M, Drezen E, Schbath S, Lavenier D, et al. Multiple comparative metagenomics using multiset k-mer counting. PeerJ Computer Science. 2016 nov;2:e94. Available from: https://peerj.com/articles/cs-94.

[67] Domazet-Lošo M, Haubold B. Alignment-free detection of local similarity among viral and bacterial genomes. Bioinformatics. 2011 6;27(11):1466-72. Available from: https://academic.oup.com/bioinformatics/article-lookup/doi/10.1093/bioinformatics/btr176.

[68] Marçais G, Kingsford C. A fast, lock-free approach for efficient parallel counting of occurrences of k-mers. Bioinformatics. 2011 3;27(6):764-70. Available from: https://academic.oup.com/bioinformatics/article-lookup/doi/10.1093/bioinformatics/btr011.

[69] Lefort V, Desper R, Gascuel O. FastME 2.0: A Comprehensive, Accurate, and Fast Distance-Based Phylogeny Inference Program: Table 1. Molecular Biology and Evolution. 2015 oct;32(10):2798-800. Available from: https://academic.oup.com/mbe/article-lookup/doi/10.1093/molbev/msv150.

[70] Jukes TH, Cantor CR. Evolution of protein molecules. In: In Mammalian protein metabolism, Vol. III (1969), pp. 21-132. vol. III; 1969. p. 21-132. Available from: http://www.citeulike.org/group/1390/article/768582.

[71] Robinson D, Foulds L. Comparison of weighted labelled trees. Lecture Notes in Mathematics. 1979.

[72] Miller DE, Staber C, Zeitlinger J, Hawley RS. Highly Contiguous Genome Assemblies of 15 Drosophila Species Generated Using Nanopore Sequencing. G3: Genes, Genomes, Genetics. 2018 oct;8(10):3131-41. Available from: http://www.ncbi.nlm.nih.gov/pubmed/30087105http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=PMC6169393http://g3journal.org/lookup/doi/10.1534/g3.118.200160.

[73] Chandler JA, Lang JM, Bhatnagar S, Eisen JA, Kopp A. Bacterial communities of diverse Drosophila species: ecological context of a host-microbe model system. PLoS genetics. 2011 sep;7(9):e1002272. Available from: http://www.ncbi.nlm.nih.gov/pubmed/21966276http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=PMC3178584.

[74] Broderick NA, Lemaitre B. Gut-associated microbes of Drosophila melanogaster. Gut microbes. 2012;3(4):307-21. Available from: http://www.ncbi.nlm.nih.gov/pubmed/22572876http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=PMC3463489.

[75] Petkau K, Fast D, Duggal A, Foley E. Comparative evaluation of the genomes of three common Drosophila-associated bacteria. Biology open. 2016 sep;5(9):1305-16. Available from: http://www.ncbi.nlm.nih.gov/pubmed/27493201http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=PMC5051641.

[76] Hinchliff CE, Smith SA, Allman JF, Burleigh JG, Chaudhary R, Coghill LM, et al. Synthesis of phylogeny and taxonomy into a comprehensive tree of life. Proceedings of the National Academy of Sciences of the United States of America. 2015 oct;112(41):12764-9. Available from: http://www.ncbi.nlm.nih.gov/pubmed/26385966http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=PMC4611642.

[77] Fofanov Y, Luo Y, Katili C, Wang J, Belosludtsev Y, Powdrill T, et al. How independent are the appearances of n-mers in different genomes? Bioinformatics. 2004 10;20(15):2421-8. Available from: https://academic.oup.com/bioinformatics/article-lookup/doi/10.1093/bioinformatics/bth266.

[78] Gascuel O. BIONJ: an improved version of the NJ algorithm based on a simple model of sequence data. Molecular Biology and Evolution. 1997 jul;14(7):685-95. Available from: http://www.ncbi.nlm.nih.gov/pubmed/9254330https://academic.oup.com/mbe/article-lookup/doi/10.1093/oxfordjournals.molbev.a025808.

[79] Andong National University. Cotesia vestalis isolate:ANU101 Genome sequencing. 2015 March In: BioProject [Internet] Bethesda, MD: National Library of Medicine (US), National Center for Biotechnology Information; 2011- Available from: http://wwwncbinlmnihgov/bioproject/PRJNA271135 NCBI:BioProject: PRJNA271135.

[80] Huang W, Li L, Myers JR, Marth GT. ART: a next-generation sequencing read simulator. Bioinformatics. 2012 2;28(4):593-4. Available from: https://academic.oup.com/bioinformatics/article-lookup/doi/10.1093/bioinformatics/btr708.

[81] Yin C, Shen G, Guo D, Wang S, Ma X, Xiao H, et al. InsectBase: a resource for insect genomes and transcriptomes. Nucleic Acids Research. 2016 1;44(D1):D801-7. Available from: http://www.ncbi.nlm.nih.gov/pubmed/26578584http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=PMC4702856https://academic.oup.com/nar/article-lookup/doi/10.1093/nar/gkv1204.

[82] Jarvis ED, Mirarab S, Aberer AJ, Li B, Houde P, Li C, et al. Whole-genome analyses resolve early branches in the tree of life of modern birds. Science. 2014 12;346(6215):1320-31. Available from: http://www.sciencemag.org/content/346/6215/1320.abstract.

[83] Jarvis ED, Mirarab S, Aberer AJ, Li B, Houde P, Li C, et al. Phylogenomic analyses data of the avian phylogenomics project. GigaScience. 2015;4(1):4.

[84] Stowers Institute for Medical Research. Sequencing and assembly of 14 Drosophila species. 2017 December In: BioProject [Internet] Bethesda, MD: National Library of Medicine (US), National Center for Biotechnology Information; 2011- Available from: http://wwwncbinlmnihgov/bioproject/PRJNA427774 NCBI:BioProject: PRJNA427774.

[85] Miller DE, Staber C, Zeitlinger J, Hawley RS. Highly Contiguous Genome Assemblies of 15 Drosophila Species Generated Using Nanopore Sequencing [assembled genomes]. github;. Available from: https://github.com/danrdanny/Drosophila15GenomesProject/.

[86] Chen S, Zhou Y, Chen Y, Gu J. fastp: an ultra-fast all-in-one FASTQ preprocessor. Bioinformatics. 2018 sep;34(17):i884-90. Available from: https://academic.oup.com/bioinformatics/article/34/17/i884/5093234.

[87] Morgulis A, Coulouris G, Raytselis Y, Madden TL, Agarwala R, Schäffer AA. Database indexing for production MegaBLAST searches. Bioinformatics (Oxford, England). 2008 aug;24(16):1757-64. Available from: http://www.ncbi.nlm.nih.gov/pubmed/18567917http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=PMC2696921.

[88] Langmead B, Salzberg SL. Fast gapped-read alignment with Bowtie 2. Nature Methods. 2012 apr;9(4):357-9. Available from: http://www.ncbi.nlm.nih.gov/pubmed/22388286http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=PMC3322381http://www.nature.com/articles/nmeth.1923.

[89] Edgar RC. MUSCLE: multiple sequence alignment with high accuracy and high throughput. Nucleic acids research. 2004;32(5):1792-7. Available from: http://www.ncbi.nlm.nih.gov/pubmed/15034147http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=PMC390337.

[90] Guindon S, Dufayard JF, Lefort V, Anisimova M, Hordijk W, Gascuel O. New Algorithms and Methods to Estimate Maximum-Likelihood Phylogenies: Assessing the Performance of PhyML 3.0. Systematic Biology. 2010 mar;59(3):307-21. Available from: http://www.ncbi.nlm.nih.gov/pubmed/20525638https://academic.oup.com/sysbio/article/59/3/307/1702850.

[91] Maddison WP. Gene Trees in Species Trees. Systematic Biology. 1997 sep;46(3):523-36. Available from: http://sysbio.oxfordjournals.org/cgi/content/abstract/46/3/523http://www.jstor.org/stable/2413694?origin=crossrefhttp://sysbio.oxfordjournals.org/content/46/3/523.short.

[92] Dasarathy G, Nowak R, Roch S. Data requirement for phylogenetic inference from multiple loci: a new distance method. IEEE/ACM Transactions on Computational Biology and Bioinformatics (TCBB). 2015;12(2):422-32.

[93] Bresler G, Bresler M, Tse D. Optimal assembly for high throughput shotgun sequencing. BMC bioinformatics. 2013;14 Suppl 5(Suppl 5):S18. Available from: http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=3706340{&}tool=pmcentrez{&}rendertype=abstract.

[94] Shomorony I, Kim SH, Courtade TA, Tse DNC. Information-optimal genome assembly via sparse read-overlap graphs. Bioinformatics. 2016;32(17):i494-502. Available from: +http://dx.doi.org/10.1093/bioinformatics/btw450.

[95] Altschul SF, Gish W, Miller W, Myers EW, Lipman DJ. Basic local alignment search tool. Journal of Molecular Biology. 1990 oct;215(3):403-10. Available from: http://www.ncbi.nlm.nih.gov/pubmed/2231712http://linkinghub.elsevier.com/retrieve/pii/S0022283605803602.

[96] Wood DE, Salzberg SL. Kraken: ultrafast metagenomic sequence classification using exact alignments. Genome Biology. 2014;15(3):R46. Available from: http://genomebiology.biomedcentral.com/articles/10.1186/gb-2014-15-3-r46.

[97] Earl D, Nguyen N, Hickey G, Harris RS, Fitzgerald S, Beal K, et al. Alignathon: a competitive assessment of whole-genome alignment methods. Genome research. 2014 dec;24(12):2077-89. Available from: http://www.ncbi.nlm.nih.gov/pubmed/25273068http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=PMC4248324.

[98] Edgar RC, Asimenos G, Batzoglou S, Sidow A. Evolver: a whole-genome sequence evolution simulator;. Available from: https://www.drive5.com/evolver/.

[99] Jain C, Rodriguez-R LM, Phillippy AM, Konstantinidis KT, Aluru S. High-throughput ANI Analysis of 90K Prokaryotic Genomes Reveals Clear Species Boundaries. bioRxiv. 2017 nov:225342. Available from: https://www.biorxiv.org/content/early/2017/11/27/225342.

[100] ;. Accessed 16 October 2018. Available from: http://hgdownload.soe.ucsc.edu/goldenPath/dm6/multiz27way/.

[101] ;. Accessed 16 October 2018. Available from: http://hgdownload.soe.ucsc.edu/goldenPath/droYak2/vsDm3/.

[102] ;. Accessed 16 October 2018. Available from: https://genome.ucsc.edu/index.html.

[103] Tavaré S. Some Probabilistic and Statistical Problems in the Analysis of DNA Sequences. Lectures on Mathematics in the Life Sciences. 1986;17:57-86.

[104] Erdos P, Steel M, Szekely L, Warnow T. A few logs suffice to build (almost) all trees: Part II. Theoretical Computer Science. 1999;221(1-2):77-118.

[105] Brondizio E, Settele J, Diaz S, Ngo H. Global assessment report on biodiversity and ecosystem services of the Intergovernmental Science-Policy Platform on Biodiversity and Ecosystem Services. IPBES Secretariat, Bonn. 2019.

[106] Liu S, Li Y, Lu J, Su X, Tang M, Zhang R, et al. SOAP Barcode: revealing arthropod biodiversity through assembly of Illumina shotgun sequences of PCR amplicons. Methods in Ecology and Evolution. 2013 12;4(12):1142-50. Available from: http://doi.wiley.com/10.1111/2041-210X.12120.

[107] Coissac E, Hollingsworth PM, Lavergne S, Taberlet P. From barcodes to genomes: Extending the concept of DNA barcoding; 2016.

[108] Bohmann K, Mirarab S, Bafna V, Gilbert MTP. Beyond DNA barcoding: The unrealized potential of genome skim data in sample identification. Molecular Ecology. 2020 jul;29(14):2521-34. Available from: https://onlinelibrary.wiley.com/doi/abs/10.1111/mec.15507.

[109] Sarmashghi S, Bohmann K, P Gilbert MT, Bafna V, Mirarab S. Skmer: assembly-free and alignment-free sample identification using genome skims. Genome Biol. 2019 02;20(1):34.

[110] Balaban M, Sarmashghi S, Mirarab S. APPLES: Scalable Distance-based Phylogenetic Placement with or without Alignments. Systematic Biology. 2019.

[111] Rachtman E, Bafna V, Mirarab S. CONSULT: accurate contamination removal using locality-sensitive hashing. NAR Genomics and Bioinformatics. 2021 08;3(3). Lqab071. Available from: https://doi.org/10.1093/nargab/lqab071.

[112] Li X, Waterman MS. Estimating the repeat structure and length of DNA sequences using L-tuples. Genome research. 2003 aug;13(8):1916-22. Available from: http://www.ncbi.nlm.nih.gov/pubmed/12902383http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=PMC403783.

[113] Williams D, Trimble WL, Shilts M, Meyer F, Ochman H. Rapid quantification of sequence repeats to resolve the size, structure and contents of bacterial genomes. BMC Genomics. 2013.

[114] Hozza M, Vinař T, Brejová B. How Big is that Genome? Estimating Genome Size and Coverage from k-mer Abundance Spectra. In: String Processing and Information Retrieval. Cham: Springer International Publishing; 2015. p. 199-209.

[115] Melsted P, Pritchard JK. Efficient counting of k-mers in DNA sequences using a bloom filter. BMC Bioinformatics. 2011.

[116] Melsted P, Halldórsson BV. KmerStream: Streaming algorithms for k-mer abundance estimation. Bioinformatics. 2014.

[117] Wahba G. Spline models for observational data. SIAM; 1990.

[118] Hastie TJ, Tibshirani RJ. Generalized additive models. vol. 43. CRC press; 1990.

[119] Leinonen R, Sugawara H, Shumway M, Collaboration INSD. The sequence read archive. Nucleic acids research. 2010;39(suppl_1):D19-21.

[120] Bushnell B. BBMap;. https://sourceforge.net/projects/bbmap/.

[121] Wood DE, Lu J, Langmead B. Improved metagenomic analysis with Kraken 2. Genome biology. 2019;20(1):257.

[122] Rachtman E, Balaban M, Bafna V, Mirarab S. The impact of contaminants on the accuracy of genome skimming and the effectiveness of exclusion read filters. Molecular Ecology Resources. 2020 may;20(3):1755-0998.13135. Available from: http://biorxiv.org/content/early/2019/11/05/831941.abstracthttps://onlinelibrary.wiley.com/doi/abs/10.1111/1755-0998.13135.

[123] Lien S, Koop BF, Sandve SR, Miller JR, Kent MP, Nome T, et al. The Atlantic salmon genome provides insights into rediploidization. Nature. 2016;533(7602):200-5.

[124] Van de Peer Y, Mizrachi E, Marchal K. The evolutionary significance of polyploidy. Nature Reviews Genetics. 2017;18(7):411.

[125] Initiative OTPT, et al. One thousand plant transcriptomes and the phylogenomics of green plants. Nature. 2019;574(7780):679.

[126] R Core Team. R: A Language and Environment for Statistical Computing. Vienna, Austria; 2019. Available from: https://www.R-project.org/.

[127] Lawson CL, Hanson RJ. Solving least squares problems. SIAM; 1995.

[128] Virtanen P, Gommers R, Oliphant TE, Haberland M, Reddy T, Cournapeau D, et al. SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. Nature Methods. 2020;17:261-72.

[129] Gurobi Optimization, LLC. Gurobi Optimizer Reference Manual; 2020. http://www.gurobi.com.

[130] Wood SN. Fast stable restricted maximum likelihood and marginal likelihood estimation of semiparametric generalized linear models. Journal of the Royal Statistical Society (B). 2011;73(1):3-36.

[131] SRA Toolkit Development Team. SRA-Tools;. http://ncbi.github.io/sra-tools/.

[132] DeGroot MH, Schervish MJ. Probability and statistics. Pearson Education; 2012.

[133] Wolfram Alpha LLC. Wolfram|Alpha;. Accessed: Dec. 09, 2020. https://www.wolframalpha.com/widgets/view.jsp?id=74e8bb60ad4e38d6a1b0dc865d7197ff.

[134] Meyer CD. Matrix analysis and applied linear algebra. vol. 71. Siam; 2000.

[135] Petersen KB, Pedersen MS. The Matrix Cookbook. Technical University of Denmark; 2012. Version 20121115. http://www2.compute.dtu.dk/pubdb/pubs/3274-full.html.

[136] Gautschi W. On inverses of Vandermonde and confluent Vandermonde matrices. Numerische Mathematik. 1962 Dec;4(1):117-23. Available from: https://doi.org/10.1007/BF01386302.

[137] Robbins H. A remark on Stirling's formula. The American mathematical monthly. 1955;62(1):26-9.