Debiasing FracMinHash and deriving confidence intervals for mutation rates across a wide range of evolutionary distances

Mahmudur Rahman Hera[1], Tessa Pierce[2], and David Koslicki[1,3,4,‡]

[1] Department of Computer Science and Engineering, The Pennsylvania State University
[2] Department of Population Health and Reproduction, University of California, Davis
[3] Department of Biology, The Pennsylvania State University
[4] Huck Institutes of the Life Sciences, The Pennsylvania State University
‡ Corresponding author, dmk333@psu.edu

Abstract. Sketching methods offer computational biologists scalable techniques to analyze data sets that continue to grow in size. MinHash is one such technique to estimate set similarity that has enjoyed recent broad application. However, traditional MinHash has previously been shown to perform poorly when applied to sets of very dissimilar sizes. *FracMinHash* was recently introduced as a modification of MinHash to compensate for this lack of performance when set sizes differ. This approach has been successfully applied to metagenomic taxonomic profiling in the widely used tool `sourmash gather`. While experimental evidence has been encouraging, FracMinHash has not yet been analyzed from a theoretical perspective. In this paper, we perform such an analysis to derive various statistics of FracMinHash, and prove that while FracMinHash is not unbiased (in the sense that its expected value is not equal to the quantity it attempts to estimate), this bias is easily corrected for both the containment and Jaccard index versions. Next, we show how FracMinHash can be used to compute point estimates as well as confidence intervals for evolutionary mutation distance between a pair of sequences by assuming a simple mutation model. We also investigate edge cases where these analyses may fail, to effectively warn the users of FracMinHash indicating the likelihood of such cases. Our analyses show that FracMinHash estimates the containment of a genome in a large metagenome more accurately and more precisely when compared to traditional MinHash, and the point estimates and confidence intervals perform significantly better in estimating mutation distances. A python-based implementation of the algorithms and theorems we derive is freely available at `https://github.com/KoslickiLab/mutation-rate-ci-calculator`. The results presented in this paper can be reproduced using the code at `https://github.com/KoslickiLab/FracMinHash-reproducibles`.

1 Introduction

One strategy scientists use when analyzing large data sets is to create a low-dimensional "sketch" or "fingerprint" of their data that allows fast, but approximate answers to their query of interest. Such sketching-based approaches in recent years have been successfully applied to a variety of genomic and metagenomic analysis tasks, due in large part to such methods incurring low computational burden when applied to large data sets. For example, Mash [25], is a MinHash [7]-based approach that was used to characterize the similarity between all pairs of RefSeq genomes in less than 30 CPU hours. Such efficiency gains are due primarily to sketching-based approaches recording a small subsample (or modification thereof) of the data in such a fashion that some distance metric or similarity measure is approximately preserved, a process called a locality sensitive hashing scheme. In bioinformatics, this has resulted in improvements to error correction [27,24], assembly [9,3,14,12], alignment [18,23], clustering [34,10,26,19], classification [21,20,6], and so on. Importantly, the accuracy and efficiency of sketching approaches can frequently be characterized explicitly, allowing practitioners to balance between efficiency improvements and accuracy. Often, these theoretical guarantees dictate that certain sketching approaches are well suited only to certain kinds of data. For example, MinHash, which is used in many of the aforementioned applications, has been shown to be particularly well-suited to quantify the similarity of sets of roughly the same size, but falters when sets of very different sizes are compared [19]. This motivated the introduction of the containment MinHash which utilized a MinHash sketch of the smaller set, with an additional probabilistic data structure (a Bloom filter [5]) to store the larger set. While this improved speed and accuracy, this approach can become quite inconvenient for large sets due to requiring a bloom filter to be created for the larger of the two sets.

To ameliorate this, an approach called the "FracMinHash" was recently introduced [15,16] that uses a MinHash hash selection approach but allows sketch size to scale naturally with the size of the underlying data, similar to ModHash dynamic scaling [7]. These properties allow both Jaccard and containment estimation between FracMinHash sketches, extending the computational advantages of MinHash sketches beyond similar-sized genome comparisons to sequencing datasets of all types. Most notably, FracMinHash enables large-scale metagenome analyses, including genomic and metagenomic similarity assessment, metagenomic taxonomic classification, streaming database searches, and outbreak detection via genomic surveillance [26,33]. FracMinHash sketching is implemented in a software package called sourmash [8]. Independently, and more recently, the same concept of FracMinHash was introduced by Ekim et al. (2021) but there with the name *universe minimizer*.

While there is ample computational evidence for the superiority of FracMinHash when compared to the classic MinHash, particularly when comparing sets of different sizes, no theoretical characterization about the accuracy and efficiency of the FracMinHash approach has yet been given. In this manuscript, we address this missing characterization of accuracy and efficiency by deriving a number of theoretical guarantees. In particular, we demonstrate that the FracMinHash approach, as originally introduced, requires a slight modification in order to become an unbiased estimator of the containment index (in terms of expected value). After this, we characterize the statistics of this unbiased estimator and derive an asymptotic normality result for FracMinHash. This in turn allows us to derive confidence intervals and hypothesis tests for this estimator when considering a simple mutation model (which is related to the commonly used Average Nucleotide Identity (ANI) score). We demonstrate the accuracy of our ANI estimates exceeds that of current approaches, particularly at high levels of sequence dissimilarity. We also characterize the likelihood of experiencing an edge case when analyzing real data which allows us to provide a level of confidence along with the estimated containment index. Finally, we support the theoretical results with additional experimental evidence and compare our approach to the frequently used Mash distance [25]. Many of these results have already been implemented into the sourmash [8] computational package[1].

A python-based implementation of the theorems we derive is freely available at `https://github.com/KoslickiLab/mutation-rate-ci-calculator`.

A note on naming As Phil Karlton is reported to have said [1]: "There are only two hard things in Computer Science: cache invalidation and naming things." The latter certainly holds true in computational biology as well. As noted above, the concept discussed herein has been defined similarly and independantly by different

[1] https://github.com/sourmash-bio/sourmash/pull/1967,
https://github.com/sourmash-bio/sourmash/pull/2032

authors. Ekim et al. [12] referred to the concept as *universe minimizers*, Irber et al. [17,26,8] called it *scaled MinHash*, and Edgar [11] called them *min code syncmers*. A recent twitter thread [29] involving these authors and others coalesced on the following definitions: *FracHash* is any sketching technique that takes a defined fraction of the hashed elements. As such, Broder's [7] ModHash is one such example of a FracHash. A *FracMinHash* is then a sketch that takes a fraction of the hashed elements, specifically those that hash to a value below some threshold (hence the "min").

2 FracMinHash and its statistics

We begin by formally defining a FracMinHash sketch by slightly modifying the definition in [15]. We aim to compare two sequences by computing the containment index from their corresponding FracMinHash sketches (which we refer to using the term *fractional containment index*, and define formally later). We next derive the statistics of the fractional containment index. A previous work already derived the expectation [15] – here we significantly extend this by: omputing the variance of FracMinHash, proving its asymptotic normality, and analyzing its behavior under a simple mutation model. The latter two results help us to use FracMinHash to accurately determine the average nucleotide identity, and consequently, the evolutionary distance between two sequences.

2.1 Definitions and preliminaries

We recall the definition of FracMinHash given in [15] and reiterate its expected value before extending the statistical analysis of this quantity. Given two arbitrary sets A and B which are subsets of a domain Ω, the containment index $C(A,B)$ is defined as $C(A,B) := \frac{|A \cap B|}{|A|}$. Let h be a perfect hash function $h : \Omega \to [0, H]$ for some $H \in \mathbb{R}$. For a *scale factor* s where $0 \leq s \leq 1$, a FracMinHash sketch of a set A is defined as follows:

$$\mathbf{FRAC}_s(A) = \{\, h(a) \mid a \in A \text{ and } h(a) \leq Hs \}\,. \tag{1}$$

That is, the set of all hashed values in A whose hash value is some fraction s smaller than the maximum hash value H. The scale factor s is an easily tunable parameter that can modify the size of the sketch. Using this FracMinHash sketch, we define the FracMinHash estimate of the containment index $\hat{C}_{\text{frac}}(A,B)$ as follows:

$$\hat{C}_{\text{frac}}(A,B) := \frac{|\mathbf{FRAC}_s(A) \cap \mathbf{FRAC}_s(B)|}{|\mathbf{FRAC}_s(A)|}. \tag{2}$$

Simply speaking, we want to compute $\hat{C}_{\text{frac}}(A,B)$ because the sketches are considerably smaller than the original sets A and B, and we want $\hat{C}_{\text{frac}}(A,B)$ to accurately approximate $C(A,B)$.

For notational simplicity, let us define $X_A := |\mathbf{FRAC}_s(A)|$. We observe that if one views h as a uniformly distributed random variable, we have that X_A is distributed as a binomial random variable: $X_A \sim \text{Binom}(|A|, s)$. In practice, hashing libraries use large enough hash value space (i.e. 2^{64}) and well enough hash functions that the assumptions on h are mostly valid. Furthermore, if $A \cap B \neq \emptyset$ where both A and B are non-empty sets and one is not a subset of the other, then $X_{A \setminus B}$ and $X_{A \cap B}$ are independent when the probability of success, s, is strictly smaller than 1. Using these notations, the expectation of $\hat{C}_{\text{frac}}(A,B)$ is given by Theorem 1, recapitulated from [15] for completeness.

Theorem 1. *For $0 < s < 1$, if A and B are two non-empty sets such that $A \setminus B$ and $A \cap B$ are non-empty, the following holds:*

$$\mathrm{E}\left[\hat{C}_{frac}(A,B)\mathbb{1}_{|\mathbf{FRAC}_s(A)|>0}\right] = \frac{|A \cap B|}{|A|}\left(1 - (1-s)^{|A|}\right).$$

Proof. Please see the appendix for the proof.

In light of Theorem 1, we note that $\hat{C}_{\text{frac}}(A,B)$ is *not* an unbiased estimate of $C(A,B)$: the expected value of $\hat{C}_{\text{frac}}(A,B)$ is not equal to $C(A,B)$. This may explain the observations in [17] that showed the uncorrected version in eq. (2) leads to suboptimal performance for shorts sequences (e.g viruses). However,

for sufficiently large $|A|$ and s, the bias factor $\left(1-(1-s)^{|A|}\right)$ is sufficiently close to 1. Alternatively, if $|A|$ is known (or estimated, eg. by using HyperLogLog [13] or the estimate in Appendix A.5), then

$$C_{\text{frac}}(A,B) := \frac{|\mathbf{FRAC}_s(A) \cap \mathbf{FRAC}_s(B)|}{|\mathbf{FRAC}_s(A)| \left(1-(1-s)^{|A|}\right)} \mathbb{1}_{|\mathbf{FRAC}_s(A)|>0} \tag{3}$$

is an unbiased estimate of the containment index $C(A,B)$. Throughout the rest of the paper, we will refer to the debiased $C_{\text{frac}}(A,B)$ as the *fractional containment index*. We now turn to calculating the statistics of $C_{\text{frac}}(A,B)$.

2.2 Mean and variance of $C_{\text{frac}}(A,B)$

The expectation of $C_{\text{frac}}(A,B)$ is as follows.

Theorem 2. *For $0 < s < 1$, if A and B are two distinct sets such that $A \setminus B$ and $A \cap B$ are non-empty, the expectation of $C_{frac}(A,B)$ is given by*

$$\mathrm{E}[C_{frac}(A,B)] = \frac{|A \cap B|}{|A|}. \tag{4}$$

Proof. This follows directly from Equation (3) and Theorem 1.

■

We now turn to determining the variance of $C_{\text{frac}}(A,B)$. Ideally, we can do so by using the multivariate probability mass function of $X_{A\cap B}$ and $X_{A\setminus B}$. However, we found that doing so does not result in a closed-form formula. Therefore, we use Taylor expansions to approximate the variance.

Theorem 3. *For $n = |A \cap B|$ and $m = |A \setminus B|$ where both m and n are non-zero, a first order Taylor series approximation gives*

$$\mathrm{Var}\left[\hat{C}_{frac}(A,B)\right] \approx \frac{mn(1-s)}{s(m+n)^3}.$$

Proof. Please see the appendix for the proof.

Using the results of Theorem 3 and Equation (3), we have the variance of $C_{\text{frac}}(A,B)$ as follows.

Corollary 1. *For $n = |A \cap B|$ and $m = |A \setminus B|$ where both m and n are non-zero, a first order Taylor series approximation gives*

$$\mathrm{Var}\left[C_{frac}(A,B)\right] \approx \frac{mn(1-s)}{s(m+n)^3 \left(1-(1-s)^{|A|}\right)^2}.$$

Proceeding in the same fashion, we can obtain series approximations of arbitrarily high order due to the binomial distribution having finite central moments of arbitrary order. However, we found that the higher order expansion derivations are tedious and long, whereas the results obtained using first order approximation are both simple and accurate enough in practice, as our numerical experiments demonstrate.

2.3 Asymptotic normality of $C_{\text{frac}}(A,B)$

In order to estimate the evolutionary distance between two sequences using $C_{\text{frac}}(A,B)$ in the next section, we next prove this quantity's asymptotic normality. We utilize the delta method [2, section 14.1.3] combined with the De Moivre-Laplace theorem, a special case of the Central Limit Theorem (CLT). Indeed, the De Moivre-Laplace theorem guarantees asymptotic normality of $X_{A\cap B}$ and $X_{A\setminus B}$, and since $g(x,y) = \frac{x}{x+y}$ is a function that is twice differentiable, setting $x = X_{A\cap B}$ and $y = X_{A\setminus B}$ satisfies all requirements of using the delta method on $g(x,y)$, which gives us the following result:

Theorem 4. *For $g(x,y) = \frac{x}{x+y}$, $n = |A \cap B|$ and $m = |A \setminus B|$ where both m and n are non-zero,*

$$\sqrt{n+m}\left(g(X_{A\cap B}, X_{A\setminus B}) - g(n,m)\right) \xrightarrow[n,m\to\infty]{\mathscr{D}} \mathscr{N}\left(0, \frac{mn(1-s)}{(m+n)^3 s}\right).$$

Proof. Please see the appendix for the proof.

Before using this asymptotic normality to determine mutation rates, we note that additional statistical quantities can easily be derived. For example, in Appendix A.4 we provide concentration inequalities that demonstrate theoretically how little $C_{\text{frac}}(A,B)$ deviates from its expected value. Appendix A.5 provides a simple way to calculate the number of distinct k-mers from a given sketch. Lastly, Appendix A.6 demonstrates how to compute (and de-bias) a Jaccard estimate from FracMinHash sketches.

3 Statistics of $C_{\text{frac}}(A,B)$ under simple mutation model

In the previous section, we introduced $C_{\text{frac}}(A,B)$, and derived its statistics. In this section, we use these results and connect $C_{\text{frac}}(A,B)$ to a biologically meaningful quantity – the Average Nucleotide Identity (ANI) and mutation rate. We do this by assuming a simple mutation model, where each nucleotide of some sequence S is independently mutated at a fixed rate, p, resulting in the mutated sequence S' which has expected ANI of $1-p$ with S. This model was recently introduced in [4] where it was quantified how this mutation process affects the k-mers in S. We extend the results of [4] to the case where A is the set of k-mers of S, B is the set of k-mers of S', and where the quantity under consideration is $C_{\text{frac}}(A,B)$, and thus show how we can calculate the ANI (or mutation rate) between two sequences with $C_{\text{frac}}(A,B)$.

Before mentioning the details of the mutation model, it is important to note that there are other models of evolution, e.g. TK4 and TK5 models [31], the general time reversible (GTR) model [32] and Sueoka's model [30]. These vary in the number of parameters used, as well as the degree of complexity. In this work, we consider the simple mutation model because (a) the statistics of k-mers under this model are already well explored, and (b) it allows us to connect $C_{\text{frac}}(A,B)$ and mutation rate p directly, which would not be the case if we considered one of these more nuanced models. The mutation model we use, even though simple enough to be mathematically tractable, is more realistic that the Poisson model assumed by Mash [25], which assumes that all k-mers are mutated independently, where in reality, one point mutation can affect up to k number of k-mers. Our experiments reveal that even in case of real genomes, where the lengths of two sequences can be widely dissimilar and clearly the assumptions of the simple mutation model are violated, our approach can accurately determine the mutation rate (and ANI) between two real-world sequences.

We first recall a few important definitions before introducing our findings.

3.1 Preliminaries

Here, we closely follow the exposition contained in [4]. Let $L > 0$ be a natural number that denotes the number of k-mers in some string S. A *k-span* K_i is the range of integers $[i, i+k-1]$ which denotes the set of indices of the sequence S where a k-mer resides. Fix a *mutation rate* p where $0 < p < 1$. The *simple mutation model* considers each position in $i = 1, \ldots, L+k-1$ and with probability p, marks it as *mutated*. A mutation at location i *affects* the k-spans $K_{\max(1,i-k+1)}, \ldots, K_i$. Let N_{mut} be a random variable defined to be the number of affected/mutated k-spans. We use $q = 1-(1-p)^k$ to express the probability that a k-span is mutated. Note that $1-p$ corresponds precisely to the expected average nucleotide identity (ANI) between a sequence S and its mutated counterpart S'.

Given a nonempty sequence S on the alphabet $\{A,C,T,G\}$ and a k-mer size such that each k-mer in S is unique, let A represent the set of all k-mers in S and let $L = |S| - k + 1$. Now, we apply the simple mutation model to S via the following: if for any $i \in [1, \ldots, L+k-1]$, this index is marked as mutated, let ${S'}_i$ be some nucleotide in $\{A,C,T,G\} \setminus \{S_i\}$, and otherwise if the index i is not marked as mutated, let ${S'}_i = S_i$. Let B represent the set of k-mers of S', and we assume that S' does not contain repeated k-mers either. In summary, A denotes the set of k-mers of a sequence S, and B denotes the set of k-mers of a sequence S' derived from S using the simple mutation model with no spurious matches. Note that given a sufficiently large k-mer size, these assumptions will be satisfied, though the k-mer size may be very large (i.e. length

of the sequences under consideration). Even so, violations of these assumptions (i.e. repeats and spurious matches) do not negatively impact our results, as demonstrated by the real-world results in Section 5.

We also recall the definition of an approximate $(1-\alpha)$ confidence interval. Given a distribution and a parameter of interest τ, a $(1-\alpha)$-CI is an interval that contains τ with probability $1-\alpha$. Given $0<\alpha<1$, we define $z_\alpha = \Phi^{-1}(1-\alpha/2)$, where Φ^{-1} is the inverse CDF of the standard Gaussian distribution.

3.2 Expectation and variance of $C_{\text{frac}}(A,B)$

With these notations in hand, we immediately notice that $|A \setminus B| = |B \setminus A| = N_{\text{mut}}$, and $|A \cap B| = L - N_{\text{mut}}$. We note that the results in Theorem 3, Corollary 1 and Theorem 4 above still hold for a fixed N_{mut}. However, assuming a simple mutation model, N_{mut} is not a fixed quantity, rather a random variable Therefore, the analyses so far only connect $C_{\text{frac}}(A,B)$ to a fixed N_{mut}, as we have only considered the randomness from the FracMinHash sketching process so far. To quantify the impact of the mutation rate p on $C_{\text{frac}}(A,B)$, we now consider the randomness introduced by both the FracMinHash sketching process and the mutation process simultaneously.

Let $\mathcal{P} = (\Omega_1, \mathcal{F}_1, \mathbf{P}_1)$ and $\mathcal{S} = (\Omega_2, \mathcal{F}_2, \mathbf{P}_2)$ be the probability spaces corresponding to the mutation and FracMinHash sketching random processes, respectively. Here, $\Omega, \mathcal{F}$ and $\mathbf{P}$ denote the sample space, the sigma-algebra on the sample space, and the probability measure, respectively. We will use the subscript $\mathcal{P}, \mathcal{S}$ to indicate the product probability space, e.g. $\mathrm{E}_{\mathcal{P},\mathcal{S}}[\cdot]$ and $\mathrm{Var}_{\mathcal{P},\mathcal{S}}[\cdot]$. Hence we assume that the mutation process and the process of taking a FracMinHash sketch are independent. Indeed, the hash functions have no relation to the point mutations introduced by the simple mutation model. Before proceeding with the analysis, we make a note that the expectation and variance of N_{mut} under the simple mutation model with no spurious matches have been investigated in [4]. As such, we already know $\mathrm{E}_{\mathcal{P}}[N_{\text{mut}}]$, $\mathrm{Var}_{\mathcal{P}}[N_{\text{mut}}]$ and $\mathrm{E}_{\mathcal{P}}[{N_{\text{mut}}}^2]$, and will use these results directly (see [4, Table 1]).

Theorem 5. *For $0 < s < 1$, if A and B are respectively distinct sets of k-mers of a sequence S and a sequence S' derived from S under the simple mutation model with mutation probability p such that $A \cap B$ is non-empty, then the expectation of $C_{frac}(A,B)$ in the product space $\mathcal{P}, \mathcal{S}$ is given by*

$$E_{\mathcal{P},\mathcal{S}}[C_{frac}(A,B)] = (1-p)^k, \tag{5}$$

where $\mathcal{P} = (\Omega_1, \mathcal{F}_1, \mathbf{P}_1)$ and $\mathcal{S} = (\Omega_2, \mathcal{F}_2, \mathbf{P}_2)$ are the probability spaces corresponding to the mutation and FracMinHash sketching random processes, respectively.

Proof. Please see the appendix for the proof.

Next, we turn to the more challenging task of calculating the variance of $C_{\text{frac}}(A,B)$ in the product space $\mathcal{P}, \mathcal{S}$. In the following, note that $\mathrm{Var}_{\mathcal{P}}(N_{mut})$ is already known (see [4, Theorem 2]).

Theorem 6. *For $0 < s < 1$, if A and B are respectively distinct sets of k-mers of a sequence S and a sequence S' derived from S under the simple mutation model with mutation probability p such that $A \cap B$ is non-empty, then the variance of $C_{frac}(A,B)$ in the product space $\mathcal{P}, \mathcal{S}$ is given by*

$$\operatorname*{Var}_{\mathcal{P},\mathcal{S}}[C_{frac}(A,B)] = \frac{(1-s)}{sL^3\left(1-(1-s)^L\right)^2}(L\mathrm{E}_{\mathcal{P}}[N] - \mathrm{E}_{\mathcal{P}}[N^2]) + \frac{1}{L^2}\operatorname*{Var}_{\mathcal{P}}(N_{mut}) \tag{6}$$

where $\mathcal{P} = (\Omega_1, \mathcal{F}_1, \mathbf{P}_1)$ and $\mathcal{S} = (\Omega_2, \mathcal{F}_2, \mathbf{P}_2)$ are the probability spaces corresponding to the mutation and FracMinHash sketching random processes, respectively.

Proof. Please see the appendix for the proof.

With the results of Theorem 5, we now have a point estimate of the mutation rate p given $C_{\text{frac}}(A,B)$, which is simply $p = 1 - C_{\text{frac}}(A,B)^{1/k}$. Naturally, there will be variability around this point estimate To account for this variability, we now derive a hypothesis test for $C_{\text{frac}}(A,B)$, and later turn it into a confidence interval for the mutation rate p.

3.3 Hypothesis test and confidence interval

We observe that the marginal of $C_{\text{frac}}(A, B)$ with respect to the mutation process is simply $C(A, B) = 1 - \frac{N_{\text{mut}}}{L}$. Using the results in [4], we note that N_{mut} is asymptotically normally distributed when the mutation rate p and k-mer length k are independent of L, and L is sufficiently large. In Theorem 4, we showed that $C_{\text{frac}}(A, B)$ is normally distributed for a fixed N_{mut}. Therefore, considering the randomness from both the FracMinHash sketching and the mutation model independently, $C_{\text{frac}}(A, B)$ is asymptotically normal when all conditions are met. Using the statistics derived in Section 3.2, we obtain the following hypothesis test for $C_{\text{frac}}(A, B)$.

Theorem 7. *Let $0 < s < 1$, let A and B be two distinct sets of k-mers, respectively of a sequence S and a sequence S' derived from S under the simple mutation model with mutation probability p, such that $A \cap B$ is non-empty.*

Also, let $0 < \alpha < 1$,

$$C_{low} = (1-p)^k - z_\alpha \sqrt{\frac{(1-s)}{sL^3\,(1-(1-s)^2)}(L\mathrm{E}_P[N_{mut}] - \mathrm{E}_P[{N_{mut}}^2]) + \frac{1}{L^2}\operatorname*{Var}_P(N_{mut})}$$

and

$$C_{high} = (1-p)^k + z_\alpha \sqrt{\frac{(1-s)}{sL^3\,(1-(1-s)^2)}(L\mathrm{E}_P[N_{mut}] - \mathrm{E}_P[{N_{mut}}^2]) + \frac{1}{L^2}\operatorname*{Var}_P(N_{mut})}.$$

Then, the following holds as $L \to \infty$ and when p and k are independent of L:

$$\Pr[C_{low} \leq C_{frac}(A, B) \leq C_{high}] = 1 - \alpha.$$

Proof. Please see the appendix for the proof.

We can turn this hypothesis test into a confidence interval for the mutation rate p as follows.

Theorem 8. *Let A and B be two distinct sets of k-mers, respectively of a sequence S and a sequence S' derived from S under the simple mutation model with mutation probability p, such that $A \cap B$ is non-empty. Let $\mathrm{E}_{p_{fixed}}[X]$ and $\mathrm{Var}_{p_{fixed}}[X]$ denote the expectation and variance of a given random variable X under the randomness from the mutation process with fixed mutation rate p_{fixed}. Then, for fixed α, s, k and an observed fractional containment index $C_{frac}(A, B)$, there exists an L large enough such that there exists a unique solution $p = p_{low}$ to the equation*

$$C_{frac}(A, B) = (1-p_{low})^k + z_\alpha \sqrt{\frac{(1-s)}{sL^3\,(1-(1-s)^L)^2}(L\mathrm{E}_{p_{low}}[N_{mut}] - \mathrm{E}_{p_{low}}[{N_{mut}}^2]) + \frac{1}{L^2}\operatorname*{Var}_{p_{low}}(N_{mut})},$$

and a unique solution $p = p_{high}$ to the equation

$$C_{frac}(A, B) = (1-p_{high})^k - z_\alpha \sqrt{\frac{(1-s)}{sL^3\,(1-(1-s)^L)^2}(L\mathrm{E}_{p_{high}}[N_{mut}] - \mathrm{E}_{p_{high}}[{N_{mut}}^2]) + \frac{1}{L^2}\operatorname*{Var}_{p_{high}}(N_{mut})},$$

such that the following holds:

$$\lim_{L\to\infty} \Pr[p_{low} \leq p \leq p_{high}] = 1 - \alpha.$$

Proof. Please see the appendix for the proof.

Thus, given an observation of $C_{\text{frac}}(A, B)$, along with the point estimate given in Theorem 5, we now have a statistically significant confidence interval to locate the mutation rate. In Section 5, we demonstrate that on both simulated and real world data, the confidence intervals and hypothesis test allow us to accurately connect the mutation rate (and hence, the ANI) with $C_{\text{frac}}(A, B)$.

4 Setting parameters correctly: likelihood of pathological corner cases

In practice, one disadvantage of sketching techniques is that the size of the sketch (here controlled via the scale factor s) may be too small to distinguish between highly similar or dissimilar sequences. For example, given a small mutation rate p, one may need a very large scale factor, and so sketch, to be able to distinguish between a sequence and the mutated version. Similarly, if the mutation rate p is high and/or a large k size is used, it is possible that FracMinHash may report a containment value of 0, even though the true value is nonzero, yet small. These "corner cases" are precisely the ones where the confidence interval given by Theorem 8 will likely fail. One of these pathological cases shows up when there is nothing common between the two FracMinHash sketches $\mathbf{FRAC}_s(A)$ and $\mathbf{FRAC}_s(B)$. We observe that this occurs when $X_{A\cap B} = 0$. Now $X_{A\cap B}$ is distributed as a binomial distribution $\text{Binom}(n, s)$ where $n = |A \cap B| = L - N_{\text{mut}}$, so the probability of the intersection being empty with respect to the sketching process is:

$$\Pr_{\mathcal{S}}[X_{A\cap B} = 0] = (1 - s)^{L-N_{\text{mut}}}.$$

Ideally, we would be able to directly calculate $\mathrm{E}_{\mathcal{P}}\left[\Pr_{\mathcal{S}}[X_{A\cap B} = 0]\right]$, the expected probability of this corner case happening. The challenge in doing so is that we do not have a closed form representation of the probability mass function (PMF) of N_{mut}. As a workaround, we developed a dynamic programming algorithm (presented in Appendix A.3) to compute $\Pr[N_{\text{mut}} = x]$ given the parameters L and p.

Using this PMF, we can easily compute $\mathrm{E}_{\mathcal{P}}\left[\Pr_{\mathcal{S}}[X_{A\cap B} = 0]\right]$, which is the likelihood of the corner case that we observe nothing common between two sequences purely by chance. The remaining pathological case occurs when $p \neq 0$ and yet $\mathbf{FRAC}_s(A) = \mathbf{FRAC}_s(B)$ (i.e. the sketches are not large enough to distinguish between A and B). Similar to before, we have

$$\Pr_{\mathcal{S}}\left[X_{A\setminus B} = 0, X_{B\setminus A} = 0\right] = \Pr_{\mathcal{S}}\left[X_{A\setminus B} = 0\right]\Pr_{\mathcal{S}}\left[X_{B\setminus A} = 0\right] = (1 - s)^{2N_{\text{mut}}},$$

and hence, by calculating $\mathrm{E}_{\mathcal{P}}\left[(1 - s)^{2N_{\text{mut}}}\right]$ using the PMF of N_{mut}, we can obtain the likelihood of the latter pathological case. Here, $A \setminus B$ and $B \setminus A$ are disjoint sets, allowing us to use the independence of $X_{A\setminus B}$ and $X_{B\setminus A}$. We assume both $A \setminus B$ and $B \setminus A$ are non-empty.

It is important to note the importance to characterize these "corner cases" as without it, a user would be unable to determine if the observed containment index of, say, zero is due to the sequences under consideration being highly diverged, or else the scale factor chosen is much too small. Indeed, these have been implemented[2] into sourmash [8] for precisely this purpose – to help practitioners assess if containment estimates of 0 or 1 are due to parameter settings (eg. scale value too high/low), or else are biologically meaningful.

5 Experiments and results

5.1 FracMinHash accurately estimates the containment index for sets of very different sizes

We first show that FracMinHash can estimate the true containment index better than MinHash when the sizes of two sets are dissimilar. For this experiment, we compared FracMinHash with the popular MinHash implementation tool Mash [25]. We took a Staphylococcus genome from the GAGE dataset [28] and selected a subsequence that covers $C\%$ of the whole genome in terms of number of bases, added this sequence to a metagenome, and calculated the containment of Staphylococcus in this "super metagenome." The metagenome we used is a WGS metagenome sample consisting of approximately 1.3G bases. We used a scale factor of 0.005 for FracMinHash, and we set the number of hash functions for Mash at 4000, since Mash works reasonably well with even only 1000 hash functions to find the containment of Staphylococcus genome in the unaltered metagenome. We picked 0.005 because it generates small enough sketch sizes to be computationally inexpensive, and at the same time ensures that the likelihoods of the corner cases are minimal.

We repeated this setup for different values of C, and compared the containment index calculated by Mash and FracMinHash in Figure 1. We show the mean values for multiple runs with different seeds in the figure, and use the error bars to show the standard deviation. Mash primarily reports MinHash Jaccard index, so we converted the Jaccard into containment by counting the number of distinct k-mers using brute force.

[2] https://github.com/sourmash-bio/sourmash/pull/1860

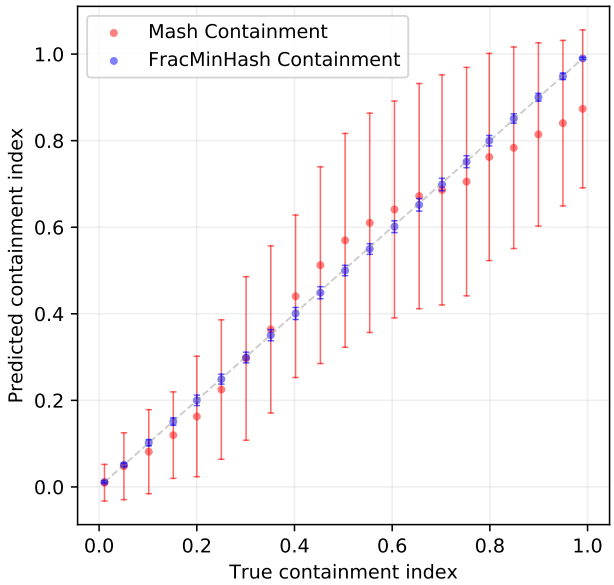


Fig. 1: True versus estimated containment index for both Mash and the FracMinHash approach for two sets with dissimilar sizes. The containment index of a Staphylococcus genome was computed when $C\%$ of this genome is inserted into an assembled metagenome. Error bars indicate standard deviation over hash seed values.

Figure 1 illustrates that while Mash and FracMinHash both faithfully estimate the true containment index, the FracMinHash approach more accurately estimates the containment index as this index increases in value. In addition, the estimate is more precise as demonstrated by the size of the error bars on the estimates. This is likely due to the fact that while Mash and FracMinHash both use a sketch of size 4,000 for the Staphylococcus genome, Mash uses the same fixed value of 4,000 when forming a sketch for the metagenome, while FracMinHash selects a sketch size that scales with the size of the metagenome. This can be seen most starkly when the metagenome is significantly larger than the query genome.

5.2 FracMinHash gives accurate confidence intervals around mutation rates

Next, we show that the confidence interval from Theorem 8 for the mutation rate p is statistically sound and works well in practice. To do so, we performed 10,000 simulations of sequences of length $L = 10$k, 100k and 1M that underwent the simple mutation model with $p = 0.001$, 0.1 and 0.2. We then used a scale factor of $s = 0.1$ when calculating p_{low} and p_{high} for a 95% confidence interval and repeated this for k-mer sizes of 21, 51 and 100. Table 1 records the percentage of experiments that resulted in $p_{\text{low}} \leq p \leq p_{\text{high}}$ and demonstrates that the confidence intervals indeed are approximately 95%. We also performed the same experiment for other scale factors that also result in minimal likelihood of the corner cases discussed in Section 4. The results are similar, but for the sake of brevity these tables are included in the appendix.

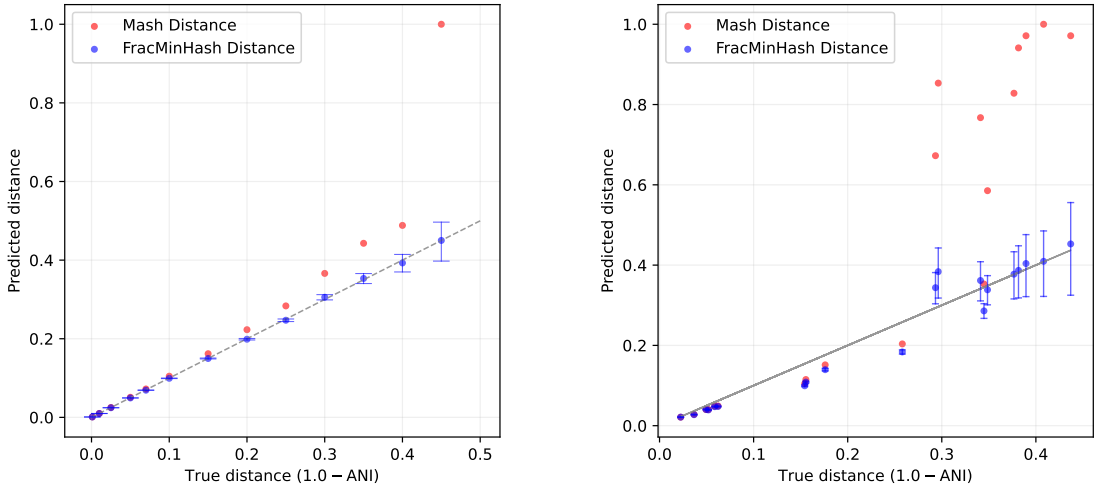
5.3 FracMinHash more accurately estimates mutation distance

On simulated data We finally compare the Mash estimate and FracMinHash estimate (given as a confidence interval) of mutation rates. For this experiment, we simulated point mutations in the aforementioned Staphylococcus genome at a mutation rate p, and then calculated the distance of the original Staphylococcus genome with this mutated genome using both Mash and the interval given by Theorem 9. The results are shown in Figure 2a. This plot shows that Mash overestimates the mutation rate by a noticeable degree, with increasing inaccuracy as the mutation distance increases. This is likely due to the Mash distance assuming

	$L = 10$ K			$L = 100$ K			$L = 1$ M		
	$p = 0.001$	0.1	0.2	0.001	0.1	0.2	0.001	0.1	0.2
$k = 21$	95.7	94.9	95.0	95.2	95.0	95.3	95.0	94.8	95.1
$k = 51$	95.2	94.6	N/A	95.2	95.5	N/A	95.0	94.8	N/A
$k = 100$	95.1	N/A	N/A	95.2	N/A	N/A	95.1	94.7	N/A

Table 1: The percentage of experiments that resulted in the true mutation rate falling within the 95% confidence interval given in Theorem 8 when using various mutation rates across multiple k-mer sizes and L values. A scale factor of $s = 0.1$ was used. The results show an average over 10,000 simulations for each setting. N/A entries indicate that the parameters are not particularly interesting, either because $E[N_{\text{mut}}] \approx L$ in these cases, or because the scale factor is too small to differentiate between the two FracMinHash sketches.

a Poisson model for how mutations affect k-mer occurrences, which has been shown to be violated when considering a point mutation model. In contrast, the point estimate given by Theorem 9 is fairly close to the true mutation rate, and the confidence interval accurately entails the true mutation rate.



(a) Estimates of evolutionary distances between original and mutated Staphylococcus genome

(b) Estimates of evolutionary distances between pairs of real bacterial genomes

Fig. 2: Mash distances and FracMinHash estimates of evolutionary distance (given in terms of one minus the average nucleotide identity: ANI) when (a) introducing point mutations to a Staphylococcus genome at a known rate, and (b) between pairs of real bacterial genomes. Error bars indicate the confidence intervals surrounding the FracMinHash estimate calculated using Theorem 9.

On real data Finally, we conclude this section by presenting pairwise mutation distances between a collection of real genomes using both Mash and the interval in Theorem 9. To make a meaningful comparison, it is important to compute the true mutation distance (or equivalently, the average nucleotide identity) between a pair of genomes. For this purpose, we used OrthoANI [22], a fast ANI calculation tool. From amongst 199K bacterial genomes downloaded from NCBI, we randomly filtered out pairs of genomes so that the pairwise ANI ranges from 0.5 to 1. For visual clarity, we kept at most 3 pairs of genomes for any ANI interval of width 5%. We used 4000 hash functions to run Mash, and set $L = (|A| + |B|)/2$ for the confidence intervals in Theorem 9, where $|A|$ and $|B|$ denote the numbers of distinct kmers in the two genomes in a pair. The results are presented in Figure 2b.

Clearly, Mash keeps overestimating the mutation distance, particularly for moderate to high distances. In contrast, the confidence intervals given by Theorem 9 perform significantly better. It is noticeable that the confidence intervals are not as accurate as in case of a simulated genome (presented in Figure 2a). This is natural because when we introduce point mutations, the resulting pair of genomes do not vary in length. On the other hand, in this real setup, the sizes of the genomes are very dissimilar, have repeats, and very easily violate the simplifying assumptions of the simple mutation model. Nonetheless, these results demonstrate the usefulness of the proposed approach even when the model assumptions are violated.

6 Conclusions

In contrast to classic MinHash, which uses a fixed sketch size, FracMinHash automatically scales the size of the sketch based on the size of the input data. This has the advantage of facilitating accurate comparison of sets of very different sizes, extending sketch-based comparisons to metagenomic datasets, including streaming-based analyses and large-scale database search. Given that a user has control over what percentage of the data to keep in the sketch (in terms of s), reasonable estimates can be made about sketch sizes *a priori*, and trade-offs employed to prevent large sketch sizes while maintaining sufficient resolution for search. One particularly attractive feature of FracMinHash is its analytical tractability: as we have demonstrated, it is relatively straightforward to characterize the performance of FracMinHash, derive its statistics, and study how it interacts with a simple mutation model. Given these advantages, it seems reasonable to favor FracMinHash in situations where sets of differing sizes are being compared, or else when fast and accurate estimates of mutation rates are desired (particularly for moderate to high mutation rates).

References

1. Naming things is hard. `https://web.archive.org/web/20210416191614/https://www.karlton.org/2017/12/naming-things-hard/`. Accessed: Dec. 3, 2021.
2. Alan Agresti. *Categorical data analysis*. John Wiley & Sons, 2002.
3. Inanç Birol, Shaun D Jackman, Cydney B Nielsen, Jenny Q Qian, Richard Varhol, Greg Stazyk, Ryan D Morin, Yongjun Zhao, Martin Hirst, Jacqueline E Schein, et al. De novo transcriptome assembly with abyss. *Bioinformatics*, 25(21):2872–2877, 2009.
4. Antonio Blanca, Robert S Harris, David Koslicki, and Paul Medvedev. The statistics of k-mers from a sequence undergoing a simple mutation process without spurious matches. *Journal of Computational Biology*, 29(2):155–168, 2022.
5. Burton H. Bloom. Space/time trade-offs in hash coding with allowable errors. *Communications of the ACM*, 13(7):422–426, 1970. `doi:10.1145/362686.362692`.
6. Florian P Breitwieser, DN Baker, and Steven L Salzberg. Krakenuniq: confident and fast metagenomics classification using unique k-mer counts. *Genome biology*, 19(1):1–10, 2018.
7. Andrei Z Broder. On the resemblance and containment of documents. In *Proceedings. Compression and Complexity of SEQUENCES 1997 (Cat. No. 97TB100171)*, pages 21–29. IEEE, 1997.
8. C Titus Brown and Luiz Irber. sourmash: a library for minhash sketching of dna. *Journal of Open Source Software*, 1(5):27, 2016.
9. Chen-Shan Chin and Asif Khalak. Human genome assembly in 100 minutes. *bioRxiv*, page 705616, 2019.
10. Michael R Crusoe, Hussien F Alameldin, Sherine Awad, Elmar Boucher, Adam Caldwell, Reed Cartwright, Amanda Charbonneau, Bede Constantinides, Greg Edvenson, Scott Fay, et al. The khmer software package: enabling efficient nucleotide sequence analysis. *F1000Research*, 4, 2015.
11. Robert Edgar. Syncmers are more sensitive than minimizers for selecting conserved k-mers in biological sequences. *PeerJ*, 9:e10805, 2021.
12. Barış Ekim, Bonnie Berger, and Rayan Chikhi. Minimizer-space de bruijn graphs: Whole-genome assembly of long reads in minutes on a personal computer. *Cell Systems*, 2021.
13. Philippe Flajolet, Éric Fusy, Olivier Gandouet, and Frédéric Meunier. Hyperloglog: the analysis of a near-optimal cardinality estimation algorithm. 2007.
14. Priyanka Ghosh and Ananth Kalyanaraman. Fastetch: a fast sketch-based assembler for genomes. *IEEE/ACM transactions on computational biology and bioinformatics*, 16(4):1091–1106, 2017.
15. Luiz Carlos Irber, Phillip T Brooks, Taylor E Reiter, N Tessa Pierce-Ward, Mahmudur Rahman Hera, David Koslicki, and C. Titus Brown. Lightweight compositional analysis of metagenomes with fracminhash and minimum metagenome covers. *bioRxiv*, 2022. URL: `https://www.biorxiv.org/content/early/`

2022/01/12/2022.01.11.475838, arXiv:https://www.biorxiv.org/content/early/2022/01/12/2022.01.11.475838.full.pdf, doi:10.1101/2022.01.11.475838.

16. Luiz Carlos Irber Jr. *Decentralizing Indices for Genomic Data*. PhD thesis, University of California, Davis, 2020.
17. Luiz Carlos Irber Jr. *Decentralizing Indices for Genomic Data*. PhD thesis, University of California, Davis, 2020.
18. Chirag Jain, Luis M Rodriguez-R, Adam M Phillippy, Konstantinos T Konstantinidis, and Srinivas Aluru. High throughput ani analysis of 90k prokaryotic genomes reveals clear species boundaries. *Nature communications*, 9(1):1–8, 2018.
19. David Koslicki and Hooman Zabeti. Improving minhash via the containment index with applications to metagenomic analysis. *Applied Mathematics and Computation*, 354:206–215, 2019.
20. Nathan LaPierre, Mohammed Alser, Eleazar Eskin, David Koslicki, and Serghei Mangul. Metalign: efficient alignment-based metagenomic profiling via containment min hash. *Genome biology*, 21(1):1–15, 2020.
21. Nathan LaPierre, Serghei Mangul, Mohammed Alser, Igor Mandric, Nicholas C Wu, David Koslicki, and Eleazar Eskin. Micop: microbial community profiling method for detecting viral and fungal organisms in metagenomic samples. *BMC genomics*, 20(5):1–10, 2019.
22. Imchang Lee, Yeong Ouk Kim, Sang-Cheol Park, and Jongsik Chun. Orthoani: an improved algorithm and software for calculating average nucleotide identity. *International journal of systematic and evolutionary microbiology*, 66(2):1100–1103, 2016.
23. Heng Li. Minimap2: pairwise alignment for nucleotide sequences. *Bioinformatics*, 34(18):3094–3100, 2018.
24. Giles Miclotte, Mahdi Heydari, Piet Demeester, Pieter Audenaert, and Jan Fostier. Jabba: Hybrid error correction for long sequencing reads using maximal exact matches. In *International Workshop on Algorithms in Bioinformatics*, pages 175–188. Springer, 2015.
25. Brian D Ondov, Todd J Treangen, Páll Melsted, Adam B Mallonee, Nicholas H Bergman, Sergey Koren, and Adam M Phillippy. Mash: fast genome and metagenome distance estimation using minhash. *Genome biology*, 17(1):1–14, 2016.
26. N Tessa Pierce, Luiz Irber, Taylor Reiter, Phillip Brooks, and C Titus Brown. Large-scale sequence comparisons with sourmash. *F1000Research*, 8, 2019.
27. Kristoffer Sahlin and Paul Medvedev. Error correction enables use of oxford nanopore technology for reference-free transcriptome analysis. *Nature Communications*, 12(1):1–13, 2021.
28. Steven L Salzberg, Adam M Phillippy, Aleksey Zimin, Daniela Puiu, Tanja Magoc, Sergey Koren, Todd J Treangen, Michael C Schatz, Arthur L Delcher, Michael Roberts, et al. Gage: A critical evaluation of genome assemblies and assembly algorithms. *Genome research*, 22(3):557–567, 2012.
29. @shenwei356 (Wei Shen) et al. "I prefer Scaled MinHash, simple, fast and context independent." https://twitter.com/ctitusbrown/status/1465360407518928909. *Twitter*, Nov. 21, 2021.
30. Noboru Sueoka. Intrastrand parity rules of dna base composition and usage biases of synonymous codons. *Journal of molecular evolution*, 40(3):318–325, 1995.
31. Naoyuki Takahata and Motoo Kimura. A model of evolutionary base substitutions and its application with special reference to rapid change of pseudogenes. *Genetics*, 98(3):641–657, 1981.
32. Simon Tavaré. Line-of-descent and genealogical processes, and their applications in population genetics models. *Theoretical population biology*, 26(2):119–164, 1984.
33. Adrian Viehweger, Christian Blumenscheit, Norman Lippmann, Kelly L Wyres, Christian Brandt, Jörg B Hans, Martin Hölzer, Luiz Irber, Sören Gatermann, Christoph Lübbert, et al. Context-aware genomic surveillance reveals hidden transmission of a carbapenemase-producing klebsiella pneumoniae. *Microbial genomics*, 7(12), 2021.
34. Qingpeng Zhang, Jason Pell, Rosangela Canino-Koning, Adina Chuang Howe, and C Titus Brown. These are not the k-mers you are looking for: efficient online k-mer counting using a probabilistic data structure. *PloS one*, 9(7):e101271, 2014.

A Appendix

A.1 Verification of Theorem 8 using simulations

Similar to Table 1, we repeated the experiment for the same settings except with two different scale factors. The results are shown in this section.

	L = 10 K			L = 100 K			L = 1 M		
p =	0.001	0.1	0.2	0.001	0.1	0.2	0.001	0.1	0.2
k = 21	95.4	95.3	94.7	95.0	95.2	95.06	95.0	95.0	94.6
k = 51	95.4	94.8	N/A	94.8	94.6	N/A	94.9	95.1	94.4
k = 100	94.7	N/A	N/A	94.6	N/A	N/A	95.4	93.7	N/A

Table S1: The percentage of experiments that resulted in the true mutation rate falling within the 95% confidence interval given in Theorem 8 when using various mutation rates across multiple k-mer sizes and L values. A scale factor of 0.2 was used. The results show an average over 10,000 simulations for each setting. N/A entries indicate that the parameters are not particularly interesting, either because $E[N_{\text{mut}}] \approx L$ in these cases, or because the scale factor is too small to differentiate the two FracMinHash sketches.

	L = 10 K			L = 100 K			L = 1 M		
p =	0.001	0.1	0.2	0.001	0.1	0.2	0.001	0.1	0.2
k = 21	96.3	95.0	96.0	95.1	95.0	95.3	95.0	95.2	94.9
k = 51	94.9	94.5	N/A	94.7	95.3	N/A	94.7	95.0	N/A
k = 100	95.2	N/A	N/A	95.2	N/A	N/A	94.5	N/A	N/A

Table S2: The percentage of experiments that resulted in the true mutation rate falling within the 95% confidence interval given in Theorem 8 when using various mutation rates across multiple k-mer sizes and L values. A scale factor of 0.05 was used. The results show an average over 10,000 simulations for each setting. N/A entries indicate that the parameters are not particularly interesting, either because $E[N_{\text{mut}}] \approx L$ in these cases, or because the scale factor is too small to differentiate the two FracMinHash sketches.

A.2 Theorems and proofs

Theorem 1. *For $0 < s < 1$, if A and B are two non-empty sets such that $A \setminus B$ and $A \cap B$ are non-empty, the following holds:*

$$\mathrm{E}\left[\hat{C}_{frac}(A,B)\mathbb{1}_{|\mathbf{FRAC}_s(A)|>0}\right] = \frac{|A \cap B|}{|A|}\left(1-(1-s)^{|A|}\right).$$

Proof. Using the notation introduced previously, observe that

$$\hat{C}_{\text{frac}}(A,B)\mathbb{1}_{|\mathbf{FRAC}_s(A)|>0} = \frac{X_{A\cap B}}{X_{A\cap B}+X_{A\setminus B}}\mathbb{1}_{X_{A\cap B}+X_{A\setminus B}>0},$$

and that the random variables $X_{A\cap B}$ and $X_{A\setminus B}$ are independent (which follows directly from the fact that $A \cap B$ and $A \setminus B$ are non-empty, distinct sets). We will use the following fact from standard calculus:

$$\int_0^1 xt^{x+y-1}\,dt = \frac{x}{x+y}\mathbb{1}_{x,y>0}. \tag{7}$$

Then using the moment generating function of the binomial distribution, we have

$$\mathrm{E}\left[t^{X_{A\cap B}}\right] = (1-s+st)^{|A\cap B|} \tag{8}$$

$$\mathrm{E}\left[t^{X_{A\setminus B}}\right] = (1-s+st)^{|A\setminus B|} \tag{9}$$

We also know by continuity that

$$\mathrm{E}\left[X_{A\cap B}\, t^{X_{A\cap B}-1}\right] = \frac{d}{dt}(1-s+st)^{|A\cap B|} \tag{10}$$

$$= |A\cap B| s(1-s+st)^{|A\cap B|-1}. \tag{11}$$

Using these observations, we can then finally calculate that

$$\mathrm{E}\left[\frac{X_{A\cap B}}{X_{A\cap B}+X_{A\setminus B}}\mathbb{1}_{X_{A\cap B}+X_{A\setminus B}>0},\right] = \mathrm{E}\left[\int_0^1 X_{A\cap B}\, t^{X_{A\cap B}+X_{A\setminus B}-1}\, dt\right] \tag{12}$$

$$= \int_0^1 \mathrm{E}\left[X_{A\cap B}\, t^{X_{A\cap B}+X_{A\setminus B}-1}\, dt\right] \tag{13}$$

$$= \int_0^1 \mathrm{E}\left[X_{A\cap B}\, t^{X_{A\cap B}-1}\right] \mathrm{E}\left[t^{X_{A\setminus B}}\right]\, dt \tag{14}$$

$$= |A\cap B| s \int_0^1 (1-s+st)^{|A\cap B|+|A\setminus B|-1}\, dt \tag{15}$$

$$= \left.\frac{|A\cap B| s(1-s+st)^{|A|}}{|A|s}\right|_{t=0}^{t=1} \tag{16}$$

$$= \frac{|A\cap B|}{|A|}\left(1-(1-s)^{|A|}\right), \tag{17}$$

where Fubini's theorem is used in Equation (13) and independence in Equation (14). ■

Theorem 3. *For $n = |A\cap B|$ and $m = |A\setminus B|$ where both m and n are non-zero, a first order Taylor series approximation gives*

$$\mathrm{Var}\left[\hat{C}_{frac}(A,B)\right] \approx \frac{mn(1-s)}{s(m+n)^3}.$$

Proof. Let $g(x,y) = \frac{x}{x+y}$, $\mu_x = ns$, $\mu_y = ms$ and use subscripts to denote partial derivatives:

$$g_x(x,y) = \frac{y}{(x+y)^2}$$

$$g_y(x,y) = \frac{-x}{(x+y)^2}$$

We then have the first order Taylor series:

$$\begin{aligned}\mathrm{Var}\left(g\left(X_{A\cap B}, X_{A\setminus B}\right)\right) &= g_x^2(\mu_x,\mu_y)\,\mathrm{Var}(X_{A\cap B}) \\ &+ 2g_x(\mu_x,\mu_y)g_y(\mu_x,\mu_y)\mathrm{E}[X_{A\cap B}-\mu_x]\mathrm{E}[X_{A\setminus B}-\mu_y] \\ &+ g_y^2(\mu_x,\mu_y)\,\mathrm{Var}(X_{A\setminus B}) \qquad (18)\\ &= \frac{m^2}{s^2(m+n)^4}ns(1-s) + \frac{n^2}{s^2(m+n)^4}ms(1-s) \\ &= \frac{mn(1-s)}{(m+n)^3 s},\end{aligned}$$

with the middle term of eq. (18) factoring due to independence.

■

Theorem 4. *For $g(x,y) = \frac{x}{x+y}$, $n = |A \cap B|$ and $m = |A \setminus B|$ where both m and n are non-zero,*

$$\sqrt{n+m}\left(g(X_{A\cap B}, X_{A\setminus B}) - g(n,m)\right) \xrightarrow[n,m\to\infty]{\mathscr{D}} \mathscr{N}\left(0, \frac{mn(1-s)}{(m+n)^3 s}\right).$$

Proof. The covariance matrix is calculated as

$$\Sigma = \begin{bmatrix} ns(1-s) & 0 \\ 0 & ms(1-s) \end{bmatrix}.$$

Using the same notation as in Theorem 3, let

$$\phi = \begin{bmatrix} g_x(\mu_x, \mu_y) \\ g_y(\mu_x, \mu_y) \end{bmatrix} = \begin{bmatrix} \frac{m}{s(n+m)^2} \\ \frac{-n}{s(n+m)^2} \end{bmatrix}.$$

The delta method then uses the first order Taylor series from Theorem 3 to obtain that $\sqrt{n+m}\left(g(X_{A\cap B}, X_{A\setminus B}) - g(n,m)\right)$ converges in distribution to a centered normal with variance

$$\phi' \Sigma \phi = \frac{mn(1-s)}{(m+n)^3 s}.$$

■

Theorem 5. *For $0 < s < 1$, if A and B are respectively distinct sets of k-mers of a sequence S and a sequence S' derived from S under the simple mutation model with mutation probability p such that $A \cap B$ is non-empty, then the expectation of $C_{frac}(A,B)$ in the product space $\mathcal{P}, \mathcal{S}$ is given by*

$$E_{\mathcal{P},\mathcal{S}}[C_{frac}(A,B)] = (1-p)^k, \tag{5}$$

where $\mathcal{P} = (\Omega_1, \mathcal{F}_1, \mathbf{P}_1)$ and $\mathcal{S} = (\Omega_2, \mathcal{F}_2, \mathbf{P}_2)$ are the probability spaces corresponding to the mutation and FracMinHash sketching random processes, respectively.

Proof.

$$\begin{aligned} E_{\mathcal{P},\mathcal{S}}[C_{\text{frac}}(A,B)] &= \int_{\mathcal{P},\mathcal{S}} C_{\text{frac}}(A,B)\, d_{\mu_1\times\mu_2} = \int_P \int_S C_{\text{frac}}(A,B)\, d\mu_2\, d\mu_1 \\ &= \mathrm{E}_P\left[\mathrm{E}_S\left[C_{\text{frac}}(A,B)\right]\right] = \mathrm{E}_P\left[1 - \frac{N_{\text{mut}}}{L}\right] \\ &= 1 - \frac{Lq}{L} = 1 - (1-(1-p)^k) \\ &= (1-p)^k. \end{aligned}$$

Here, we used Fubini's theorem in the second step. We also used the expectation of N_{mut} from [4], where $q = 1-(1-p)^k$. ■

Theorem 6. *For $0 < s < 1$, if A and B are respectively distinct sets of k-mers of a sequence S and a sequence S' derived from S under the simple mutation model with mutation probability p such that $A \cap B$ is non-empty, then the variance of $C_{frac}(A,B)$ in the product space $\mathcal{P}, \mathcal{S}$ is given by*

$$\operatorname*{Var}_{\mathcal{P},\mathcal{S}}[C_{frac}(A,B)] = \frac{(1-s)}{sL^3\left(1-(1-s)^L\right)^2}(L\mathrm{E}_{\mathcal{P}}[N] - \mathrm{E}_{\mathcal{P}}[N^2]) + \frac{1}{L^2}\operatorname*{Var}_{\mathcal{P}}(N_{mut}) \tag{6}$$

where $\mathcal{P} = (\Omega_1, \mathcal{F}_1, \mathbf{P}_1)$ and $\mathcal{S} = (\Omega_2, \mathcal{F}_2, \mathbf{P}_2)$ are the probability spaces corresponding to the mutation and FracMinHash sketching random processes, respectively.

Proof. First, we calculate the second moment of $C_{\text{frac}}(A,B)$ in the product space as follows:

$$\begin{aligned}
\mathrm{E}_{\mathcal{P},\mathcal{S}}[C_{\text{frac}}(A,B)^2] &= \int_{\mathcal{P},\mathcal{S}} C_{\text{frac}}(A,B)^2 \, d_{\mu_1\times\mu_2} = \int_{\mathcal{P}}\int_{S} C_{\text{frac}}(A,B)^2 \, d\mu_2 \, d\mu_1 \\
&= \int_{\mathcal{P}} \left[\frac{mn(1-s)}{s(m+n)^3\left(1-(1-s)^L\right)^2} + \left(\frac{L-N_{mut}}{L}\right)^2 \right] d\mu_1 \\
&= \mathrm{E}_{\mathcal{P}} \left[\frac{N(L-N)(1-s)}{sL^3\left(1-(1-s)^L\right)^2} + \frac{1}{L^2}(L^2-2LN+N^2) \right] \\
&= \frac{(1-s)}{sL^3\left(1-(1-s)^L\right)^2}(L\mathrm{E}_{\mathcal{P}}[N]-\mathrm{E}_{\mathcal{P}}[N^2]) \\
&\quad + \frac{1}{L^2}(L^2-2L\mathrm{E}_{\mathcal{P}}[N]+\mathrm{E}_{\mathcal{P}}[N^2])
\end{aligned}$$

Therefore, we calculate the variance in the product space as follows.

$$\begin{aligned}
\operatorname*{Var}_{\mathcal{P},\mathcal{S}}(C_{\text{frac}}(A,B)) &= \mathrm{E}_{\mathcal{P},\mathcal{S}}[C_{\text{frac}}(A,B)^2] - E_{\mathcal{P},\mathcal{S}}[C_{\text{frac}}(A,B)]^2 \\
&= \frac{(1-s)}{sL^3\left(1-(1-s)^L\right)^2}(L\mathrm{E}_{\mathcal{P}}[N]-\mathrm{E}_{\mathcal{P}}[N^2]) \\
&\quad + \frac{1}{L^2}(L^2-2L\mathrm{E}_{\mathcal{P}}[N]+\mathrm{E}_{\mathcal{P}}[N^2]) \\
&\quad - \frac{1}{L^2}(L-\mathrm{E}_{\mathcal{P}}[N])^2 \\
&= \frac{(1-s)}{sL^3\left(1-(1-s)^L\right)^2}(L\mathrm{E}_{\mathcal{P}}[N]-\mathrm{E}_{\mathcal{P}}[N^2]) \\
&\quad + \frac{1}{L^2}(L^2-2L\mathrm{E}_{\mathcal{P}}[N]+\mathrm{E}_{\mathcal{P}}[N^2]) \\
&\quad - \frac{1}{L^2}(L^2-2L\mathrm{E}_{\mathcal{P}}[N]+\mathrm{E}_{\mathcal{P}}[N]^2) \\
&= \frac{(1-s)}{sL^3\left(1-(1-s)^L\right)^2}(L\mathrm{E}_{\mathcal{P}}[N]-\mathrm{E}_{\mathcal{P}}[N^2]) + \frac{1}{L^2}\operatorname*{Var}_{\mathcal{P}}(N_{mut})
\end{aligned}$$

■

Theorem 7. *Let $0 < s < 1$, let A and B be two distinct sets of k-mers, respectively of a sequence S and a sequence S' derived from S under the simple mutation model with mutation probability p, such that $A \cap B$ is non-empty.*

Also, let $0 < \alpha < 1$,

$$C_{low} = (1-p)^k - z_\alpha \sqrt{\frac{(1-s)}{sL^3\left(1-(1-s)^2\right)}(L\mathrm{E}_P[N_{mut}]-\mathrm{E}_P[{N_{mut}}^2]) + \frac{1}{L^2}\operatorname*{Var}_P(N_{mut})}$$

and

$$C_{high} = (1-p)^k + z_\alpha \sqrt{\frac{(1-s)}{sL^3\left(1-(1-s)^2\right)}(L\mathrm{E}_P[N_{mut}]-\mathrm{E}_P[{N_{mut}}^2]) + \frac{1}{L^2}\operatorname*{Var}_P(N_{mut})}.$$

Then, the following holds as $L \to \infty$ and when p and k are independent of L:

$$\Pr[C_{low} \le C_{frac}(A,B) \le C_{high}] = 1-\alpha.$$

Proof. As discussed in Section 3.3, $C_{\text{frac}}(A,B)$ is asymptotically normal when the required conditions are met. Therefore, the hypothesis test for a random variable following the Gaussian distribution holds for $C_{\text{frac}}(A,B)$. Using the expectation and the variance proved in Theorems 5 and 6, we have the results stated in the theorem.

■

Theorem 8. *Let A and B be two distinct sets of k-mers, respectively of a sequence S and a sequence S' derived from S under the simple mutation model with mutation probability p, such that $A \cap B$ is non-empty. Let $\mathrm{E}_{p_{fixed}}[X]$ and $\mathrm{Var}_{p_{fixed}}[X]$ denote the expectation and variance of a given random variable X under the randomness from the mutation process with fixed mutation rate p_{fixed}. Then, for fixed α, s, k and an observed fractional containment index $C_{frac}(A, B)$, there exists an L large enough such that there exists a unique solution $p = p_{low}$ to the equation*

$$C_{frac}(A,B) = (1-p_{low})^k + z_\alpha \sqrt{\frac{(1-s)}{sL^3\left(1-(1-s)^L\right)^2}\left(L\mathrm{E}_{p_{low}}[N_{mut}] - \mathrm{E}_{p_{low}}[{N_{mut}}^2]\right) + \frac{1}{L^2}\operatorname*{Var}_{p_{low}}(N_{mut})},$$

and a unique solution $p = p_{high}$ to the equation

$$C_{frac}(A,B) = (1-p_{high})^k - z_\alpha \sqrt{\frac{(1-s)}{sL^3\left(1-(1-s)^L\right)^2}\left(L\mathrm{E}_{p_{high}}[N_{mut}] - \mathrm{E}_{p_{high}}[{N_{mut}}^2]\right) + \frac{1}{L^2}\operatorname*{Var}_{p_{high}}(N_{mut})},$$

such that the following holds:

$$\lim_{L\to\infty} \Pr[p_{low} \le p \le p_{high}] = 1-\alpha.$$

Proof. Given the results in Theorem 7, we only need to prove that p_{low} and p_{high} are well defined. It suffices to show that

$$(1-p_{\mathrm{low}})^k + z_\alpha \sqrt{\frac{(1-s)}{sL^3\left(1-(1-s)^L\right)^2}\left(L\mathrm{E}_{p_{\mathrm{low}}}[N_{\mathrm{mut}}] - \mathrm{E}_{p_{\mathrm{low}}}[{N_{\mathrm{mut}}}^2]\right) + \frac{1}{L^2}\operatorname*{Var}_{p_{\mathrm{low}}}(N_{\mathrm{mut}})}$$

and

$$(1-p_{\mathrm{high}})^k - z_\alpha \sqrt{\frac{(1-s)}{sL^3\left(1-(1-s)^2\right)}\left(L\mathrm{E}_{p_{\mathrm{high}}}[N_{\mathrm{mut}}] - \mathrm{E}_{p_{\mathrm{high}}}[{N_{\mathrm{mut}}}^2]\right) + \frac{1}{L^2}\operatorname*{Var}_{p_{\mathrm{high}}}(N_{\mathrm{mut}})}$$

are strictly monotonic in p_{low} and p_{high}, respectively under the Stated conditions.

Let us first investigate the function of p_{low}. For simplicity, we will write p instead of p_{low}, z instead of z_α and N instead of N_{mut}. We observe the following:

$$
\begin{aligned}
&\frac{\partial}{\partial p}\left[(1-p)^k + z_\alpha\sqrt{\frac{(1-s)}{sL^3\big(1-(1-s)^L\big)^2}(L\mathrm{E}_p[N]-\mathrm{E}_p[N^2]) + \frac{1}{L^2}\operatorname*{Var}_p(N)}\right]\\
&= -k(1-p)^{-1+k} - \big(\big(\frac{1}{L^2}\big(-kL\big(-2k+\big(1-(1-p)^k\big)\big(-1+2k+\frac{2}{p}\big)\big)(1-p)^{-1+k}+\\
&L\big(k\big(-1+2k+\frac{2}{p}\big)(1-p)^{-1+k} - \frac{2\big(1-(1-p)^k\big)}{p^2}\big)(1-p)^k - 2(-1+k)k^2(1-p)^{-1+2k}-\\
&\frac{4(1-p)^k\big(-1+(1-p)^k+\big(1+(-1+k)(1-p)^k\big)p\big)}{p^3}-\\
&\frac{2k(1-p)^{-1+k}\big(-1+(1-p)^k+\big(1+(-1+k)(1-p)^k\big)p\big)}{p^2}+\\
&\frac{2(1-p)^k\big(1-k(1-p)^{-1+k}+(-1+k)(1-p)^k-(-1+k)k(1-p)^{-1+k}p\big)}{p^2}\big)+\\
&\frac{1}{L^3\big(1-(1-s)^L\big)^2 s}\big(kL^2(1-p)^{-1+k}+kL\big(-2k+\big(1-(1-p)^k\big)\big(-1+2k+\frac{2}{p}\big)\big)(1-p)^{-1+k}-\\
&2\mathrm{k}L^2\big(1-(1-p)^k\big)(1-p)^{-1+k} - L\big(k\big(-1+2k+\frac{2}{p}\big)(1-p)^{-1+k} - \frac{2\big(1-(1-p)^k\big)}{p^2}\big)(1-p)^k+\\
&2(-1+k)k^2(1-p)^{-1+2k} + \frac{4(1-p)^k\big(-1+(1-p)^k+\big(1+(-1+k)(1-p)^k\big)p\big)}{p^3}+\\
&\frac{2k(1-p)^{-1+k}\big(-1+(1-p)^k+\big(1+(-1+k)(1-p)^k\big)p\big)}{p^2}-\\
&\frac{2(1-p)^k\big(1-k(1-p)^{-1+k}+(-1+k)(1-p)^k-(-1+k)k(1-p)^{-1+k}p\big)}{p^2}\big)(1-s)\big)z\big)/2\sqrt{f},
\end{aligned}
$$

where

$$
\begin{aligned}
&f = \frac{L\big(-2k+\big(1-(1-p)^k\big)\big(-1+2k+\frac{2}{p}\big)\big)(1-p)^k+(-1+k)k(1-p)^{2k}}{L^2}+\\
&\frac{2(1-p)^k\big(-1+(1-p)^k+\big(1+(-1+k)(1-p)^k\big)p\big)}{L^2p^2} + \frac{1}{L^3\big(1-(1-s)^L\big)^2 s}\big(L^2\big(1-(1-p)^k\big)\\
&-L^2\big(1-(1-p)^k\big)^2 - L\big(-2k+\big(1-(1-p)^k\big)\big(-1+2k+\frac{2}{p}\big)\big)(1-p)^k-\\
&(-1+k)k(1-p)^{2k} - \frac{2(1-p)^k\big(-1+(1-p)^k+\big(1+(-1+k)(1-p)^k\big)p\big)}{p^2}\big)(1-s).
\end{aligned}
$$

After a tedious, but straightforward (due to the polynomial and rational terms) series expansion of this derivative about $L=\infty$, we obtain that the derivative is

$$
-k(1-p)^{k-1} + O(L^{-1/2})
$$

Therefore, as L approaches ∞, the derivative is always negative, which gives us that the function $(1-p_{\text{low}})^k + z_\alpha\sqrt{\frac{(1-s)}{sL^3(1-(1-s)^2)}(L\mathrm{E}_{p_{\text{low}}}[N]-\mathrm{E}_{p_{\text{low}}}[N^2]) + \frac{1}{L^2}\operatorname{Var}_{p_{\text{low}}}(N_{mut})}$ is monotonically decreasing in p_{low} in the asymptotic case.

The proof that $(1-p_{\text{high}})^k - z_\alpha\sqrt{\frac{(1-s)}{sL^3(1-(1-s)^2)}(L\mathrm{E}_{p_{\text{high}}}[N]-\mathrm{E}_{p_{\text{high}}}[N^2]) + \frac{1}{L^2}\operatorname{Var}_{p_{\text{high}}}(N_{mut})}$ is monotonically decreasing in p_{high} proceeds in an entirely analogous manner.

■

A.3 Dynamic Programming algorithm to compute the PMF of N_{mut}

Here, we will continue to use the notations of the simple mutation model for this algorithm, namely the parameters L, k and p. Let a string $\mathscr{S}$ of length l undergo the simple mutation process. For ease of understanding, we will represent the mutations introduced to $\mathscr{S}$ using a binary string $\mathscr{B}$ of length l, where $\mathscr{B}[i] = 1$ if position i in $\mathscr{S}$ was mutated, and 0 otherwise. Therefore, each 1 in this binary string comes from a point mutation, occurring with a probability of p, and each 0 with a probability of $1-p$. Note that there are $l-k+1$ k-mers in $\mathscr{S}$. If we could account for all such binary string $\mathscr{B}$'s that result in a total of x mutated k-mers, we can accumulate the probabilities associated with each of these strings and compute $\Pr[N_{\text{mut}} = x]$ by letting $l = L+k-1$ (which is the length of S and S'). We do this efficiently by defining the following indicator variable:

$$I[i] = \begin{cases} 1 & \text{if k-span } K_i \text{ in } \mathscr{S} \text{ is a mutated } k\text{-mer} \\ 0 & \text{otherwise} \end{cases}$$

for $i = 1$ up to $l-k+1$, and making use of the following subproblems:

$$\mathbf{P}(l,x,z) = \Pr\left[\left(\sum_{i=1}^{l-k+1} I[i]\right) = x, \forall j \text{ s.t. } l-z+1 \le j \le l,\ \mathscr{B}[j] = 0\right] \tag{19}$$

where $0 \le z < k$, $l \ge k$, $0 \le x \le l-k+1$. Put another way, $\mathbf{P}(l,x,z)$ is the probability of having x mutated k-mers in a string of length l with z trailing zeros. Here, $l \ge k$ is required to make sure there is at least one k-mer. Equation (19) covers the cases where a string can have at most $k-1$ trailing zeros. For the rest of the cases, we define the following subproblem:

$$\mathbf{P}(l,x,k) = \Pr\left[\left(\sum_{i=1}^{l-k+1} I[i]\right) = x, \forall j \text{ s.t. } l-\mu+1 \le j \le l,\ \mathscr{B}[j] = 0, \mu \ge k\right] \tag{20}$$

where $l \ge k$, $0 \le x \le l-k+1$. Put another way, $\mathbf{P}(l,x,k)$ is the probability of having x mutated k-mers in a string of length l with k or more trailing zeros.

The base cases of these subproblems are when the string has a length of k, and there can only be one k-mer. This k-mer will be non-mutated when the corresponding binary string has k zeros, giving us a probability of $\mathbf{P}(l=k,0,k) = (1-p)^k$. On the other hand, if we have $z < k$ trailing zeros, all we need is a 1 preceding these zeros for the k-mer to be mutated, giving us $\mathbf{P}(l=k,1,z) = p(1-p)^z$ for $0 \le z < k$. It is straightforward to verify that summing these probabilities indeed gives us 1.

We next turn to using the smaller subproblems to solve the larger ones. The core idea is that if we append a 1 at the end of a binary string, then the number of mutated k-mers will increase by one, and there are no trailing zeros in the resulting string. On the other hand, if we append a 0 at the end of the string, then the number of mutated k-mers will stay the same if the total number of trailing zeros is k or more. Appending a 0 at the end will increase the number of mutated k-mers by one if the total number of trailing zeros is less than k. In both of these latter scenarios, the number of trailing zeros will increase by one. These observations lead to the following recurrence relation:

$$\mathbf{P}(l,x,z) = \begin{cases} \left(\sum_{z'=0}^{k-1} \mathbf{P}(l-1,x-1,z') + \mathbf{P}(l-1,x-1,k)\right) \times p & \text{if } z = 0 \\ \mathbf{P}(l-1,x-1,z-1) \times (1-p) & \text{if } 1 \le z < k \\ \left(\mathbf{P}(l-1,x,k-1) + \mathbf{P}(l-1,x,k)\right) \times (1-p) & \text{if } z = k. \end{cases} \tag{21}$$

For our parameters L, k and p, we would need to solve the subproblems for $l = L+k-1$. Finally, we would compute $\Pr[N_{\text{mut}} = x]$, $x = 0$ up to L as follows:

$$\Pr[N_{\text{mut}} = x] = \mathbf{P}(L+k-1,x,k) + \sum_{z'=0}^{k-1} \mathbf{P}(L+k-1,x,z'). \tag{22}$$

These base cases and recurrence relations give us Algorithm 1 to compute the PMF of N_{mut}. The loop at Step 5 of the algorithm iterates L times. The inner loop at Step 6 iterates at most L times. It is straightforward

Algorithm 1 : PMF − Nmut

Input:

L, total number of k-mers
k, length of a k-mer
p, mutation rate

Initialization:

$\mathbf{P}(l, x, z) = 0$ for $l = k$ up to $L + k - 1$, $x = 0$ up to L, $z = 0$ up to k

Steps:

1: $\mathbf{P}(k, 0, k) = (1 - p)^k$
2: **for** $z = 0, \ldots, k - 1$ **do**
3: $\quad p(k, 1, z) = \mathbf{P}(1 - p)^z$
4: **end for**

5: **for** $l = k + 1, \ldots, L + k - 1$ **do**
6: $\quad$ **for** $x = 0, \ldots, l - k + 1$ **do**
7: $\quad\quad \mathbf{P}(l, x, 0) = \Big(\sum_{z'=0}^{k-1} \mathbf{P}(l - 1, x - 1, z') + \mathbf{P}(l - 1, x - 1, k)\Big) \times p$
8: $\quad\quad$ **for** $z = 1, \ldots, k - 1$ **do**
9: $\quad\quad\quad \mathbf{P}(l, x, z) = \mathbf{P}(l - 1, x - 1, z - 1) \times (1 - p)$
10: $\quad\quad$ **end for**
11: $\quad\quad \mathbf{P}(l, x, k) = \Big(\mathbf{P}(l - 1, x, k - 1) + \mathbf{P}(l - 1, x, k)\Big) \times (1 - p)$
12: $\quad$ **end for**
13: **end for**

14: **for** $x = 0, \ldots, L$ **do**
15: $\quad \mathrm{PMF}[x] = \mathbf{P}(L + k - 1, x, k) + \sum_{z'=0}^{k-1} \mathbf{P}(L + k - 1, x, z')$
16: **end for**

Output:

PMF, where $\mathrm{PMF}[x] = \Pr[N_{\mathrm{mut}} = x]$

to count that Steps 7 − 11 take $O(k)$ number of operations. These observations give us a running time of $O(L^2 k)$. Note that k is usually in the magnitude of 20 to 50. Considering $k << L$, we have an $O(L^2)$ algorithm to compute the PMF of N_{mut}.

A.4 Theoretical guarantees to accurately estimate containment index

In this section, we present theoretical evidence that $C_{\mathrm{frac}}(A, B)$ is able to estimate the true containment index $C(A, B)$ with high accuracy. Let the elements in $A \cup B$ be e_i for $i = 1$ to N. We define an indicator variable Y_i associated with an element e_i as follows:

$$Y_i = \begin{cases} 1 & \text{if } \; e_i \in \mathbf{FRAC}_s(A) \cap \mathbf{FRAC}_s(B) \\ 0 & \text{otherwise} \end{cases}.$$

Let Y be the number of elements in $\mathbf{FRAC}_s(A) \cap \mathbf{FRAC}_s(B)$. Naturally, $Y = \sum_{i=1}^{N} Y_i$. The probability of Y_i being 1 is $\frac{|A \cap B|s}{|A \cup B|}$. Therefore, we have:

$$E[Y] = \sum_{i=1}^{N} \Pr[Y_i = 1] = |A \cap B|s.$$

Let us make a simplifying assumption that the exact cardinality of the set A is known. Let us define Y' as $Y' = \frac{Y}{|A|s}$. Therefore, $E[Y'] = |A \cap B|/|A| = C(A, B)$. If we use Y' as the estimator to measure $C(A, B)$, then we have

$$\Pr\Big[\Big|\frac{Y' - C(A,B)}{C(A,B)}\Big| \geq \delta\Big] = \Pr\Big[\Big|\frac{Y - |A \cap B|s}{|A \cap B|s}\Big| \geq \delta\Big] \leq 2e^{-\delta^2 |A \cap B| s/3},$$

where we used Chernoff bound for a sum of Bernoulli random variables in the last step. The results are trivial, stating that when the two sets have more in common, or when we work with a larger scale factor, the estimate Y' performs better. This is expected, and conforms to the concept of using a scale factor. $C_{\text{frac}}(A, B)$ estimates $C(A, B)$ slightly differently than Y', and further investigations are required to narrow down the theoretical guarantees of $C_{\text{frac}}(A, B)$ estimating $C(A, B)$.

A.5 Estimating number of distinct *k*-mers from FracMinHash

In this section, we detail a simple method to estimate the total number of distinct k-mers in a given set from its FracMinHash. This can be useful for applications such as the de-biasing in eq. (3) when the set under consideration is small. We have already observed that for $X_A := |\mathbf{FRAC}_s(A)|$ the size of the sketch, X_A is distributed as a binomial random variable: $X_A \sim \text{Binom}(|A|, s)$. Hence $E[X_A/s] = |A|$, so a point estimate of the number of distinct k-mers $|A|$ can be had by dividing the sketch size by the scale factor. As the underlying distribution is binomial, we can easily obtain a Chernoff bound for the probability of deviating from this expected value by some relative error δ:

$$P\left(\left|\frac{X_a/s - |A|}{|A|}\right| < \delta\right) > 1 - 2e^{-\delta^2 |A| s/3}.$$

So, for example, if using a scale factor of $s = 1/1000$, if you want to be at least 95% sure that the estimate X_A/s is off by less than $\delta = 5\%$, this would require that $|A| \geq 4.4 \times 10^6$.

A.6 Jaccard calculated using FracMinHash sketches has bias

The theoretical analyses of $C_{\text{frac}}(A, B)$ presented in this work reveal the bias in containment index when computed from two FracMinHash sketches. Similarly, a bias in the Jaccard index computed from two FracMinHash sketches can also be proved. Please note that a similar confidence interval can *not* be derived for the Jaccard index as we were able to for the containment index. This is primarily because we found that the Jaccard index cannot be proved to be asymptotically Normal. Nonetheless, the following analysis proves that Jaccard version has a bias associated with it as well. Let us define

$$\hat{J}_{\text{frac}}(A, B) := \frac{|\mathbf{FRAC}_s(A) \cap \mathbf{FRAC}_s(B)|}{|\mathbf{FRAC}_s(A) \cup \mathbf{FRAC}_s(B)|} \tag{23}$$

and investigate how well $\hat{J}_{\text{frac}}(A, B)$ approximates the Jaccard index

$$J(A, B) := \frac{|A \cap B|}{|A \cup B|}. \tag{24}$$

Using the same notations introduced previously, we have the following theorem.

Theorem 9. *For $0 < s < 1$, if A and B are two non-empty sets such that $A \setminus B$ and $A \cap B$ are non-empty and $B \not\subset A$ as well as $A \not\subset B$, the following holds:*

$$\mathrm{E}\left[\hat{J}_{frac}(A, B) \mathbb{1}_{|\mathbf{FRAC}_s(A) \cup \mathbf{FRAC}_s(B)| > 0}\right] = \frac{|A \cap B|}{|A \cup B|}\left(1 - (1 - s)^{|A \cup B|}\right).$$

Proof. We observe that

$$\hat{J}_{\text{frac}}(A,B)\mathbb{1}_{|\mathbf{FRAC}_s(A)\cup\mathbf{FRAC}_s(B)|>0} = \frac{X_{A\cap B}}{X_{A\cap B}+X_{A\ominus B}}\mathbb{1}_{X_{A\cap B}+X_{A\ominus B}>0},$$

and that the random variables $X_{A\cap B}$ and $X_{A\ominus B}$ are independent assuming the conditions of the theorem. Here, $A\ominus B = (A\setminus B)\cup(B\setminus A)$. From standard calculus, we have:

$$\int_0^1 xt^{x+y-1}\,dt = \frac{x}{x+y}\mathbb{1}_{x+y>0}. \tag{25}$$

Then using the moment generating function of the binomial distribution, we have

$$\mathrm{E}\left[t^{X_{A\cap B}}\right] = (1-s+st)^{|A\cap B|} \tag{26}$$

$$\mathrm{E}\left[t^{X_{A\ominus B}}\right] = (1-s+st)^{|A\ominus B|}. \tag{27}$$

We also know by continuity that

$$\mathrm{E}\left[X_{A\cap B}\,t^{X_{A\cap B}-1}\right] = \frac{d}{dt}(1-s+st)^{|A\cap B|} \tag{28}$$

$$= |A\cap B|s(1-s+st)^{|A\cap B|-1}. \tag{29}$$

Using these observations, we can then finally calculate that

$$\mathrm{E}\left[\frac{X_{A\cap B}}{X_{A\cap B}+X_{A\ominus B}}\mathbb{1}_{X_{A\cap B}+X_{A\ominus B}>0},\right] = \mathrm{E}\left[\int_0^1 X_{A\cap B}\,t^{X_{A\cap B}+X_{A\ominus B}-1}\,dt\right] \tag{30}$$

$$= \int_0^1 \mathrm{E}\left[X_{A\cap B}\,t^{X_{A\cap B}+X_{A\ominus B}-1}\,dt\right] \tag{31}$$

$$= \int_0^1 \mathrm{E}\left[X_{A\cap B}\,t^{X_{A\cap B}-1}\right]\mathrm{E}\left[t^{X_{A\ominus B}}\right]\,dt \tag{32}$$

$$= |A\cap B|\int_0^1 (1-s+st)^{|A\cap B|+|A\ominus B|-1}\,dt \tag{33}$$

$$= \left.\frac{|A\cap B|(1-s+st)^{|A\cup B|}}{|A\cup B|}\right|_{t=0}^{t=1} \tag{34}$$

$$= \frac{|A\cap B|}{|A\cup B|}\left(1-(1-s)^{|A\cup B|}\right), \tag{35}$$

where Fubini's theorem is used in eq. (31) and independence in eq. (32).