Variance in Variants: Propagating Sequence Uncertainty into Downstream Analyses

David Champredon, Devan Becker, Art Poon, Connor Chato, Gopi Gugan

Abstract

Genetic sequencing is subject to many different types of errors, but most analyses treat the resultant sequences as if they are perfect. Since the process of sequencing is very difficult, modern machines rely on significantly larger numbers of reads rather than making each read significantly more accurate. Still, the coverage of such machines is imperfect and leaves uncertainty in many of the base calls. Furthermore, there are circumstances around the sequencing that can induce further problems. In this work, we demonstrate that the uncertainty in sequencing techniques will affect downstream analysis and propose a straightforward (if computationally expensive) method to propagate the uncertainty.

Our method uses a probabilistic matrix representation of individual sequences which incorporates base quality scores and makes various uncertainty propagation methods obvious and easy. With the matrix representation, resampling possible base calls according to quality scores provides a bootstrap- or prior distribution-like first step towards genetic analysis. Analyses based on these re-sampled sequences will include an honest evaluation of the error involved in such analyses.

We demonstrate our resampling method on HIV and SARS-CoV-2 data. The resampling procedures adds computational cost to the analyses, but the large impact on the variance in downstream estimates makes it clear that ignoring this uncertainty leads to invalid conclusions. For HIV data, we show that phylogenetic reconstructions are much more sensitive to sequence error uncertainty than previously believed, and for SARS-CoV-2 data we show that lineage designations via Pangolin are much less certain than the bootstrap support would imply.

TODO

- Ensure I'm using pangolin/pangoLEARN in proper contexts
  - Right now, I'm using them as synonyms (bad)
- Nomenclature:
  - $\mathcal{S}$ is "nucleotide-level uncertainty sequence" or simply "uncertainty matrix"?
    - "Uncertainty matrix" is a misnomer - the entries are the probabilitity of basis call. Is there a better term?
    - Maybe "Call Probability Matrix"?
  - Nucleotide "position", "location", or "locus"?
  - "sequence-level uncertainty sequence" or "genome likelihood"?
    - Genome likelihood is used in other contexts; their formulations are model-based but have the same interpretation.
- Notation:
  - subscript j for arbitrary locations on the genome.
- Style:
  - Ensure I use *i.e.*,, not i.e.
  - I'm really bad at tense.

1 Intro

Extracting DNA/RNA from biological samples is a complex process that involves many steps: extraction of the genetic material of interest (avoiding contamination with foreign/unwanted genetic material); reverse transcription (if RNA); DNA fragmentation of the genome into smaller segments; amplification of the fragmented sequences using PCR; sequencing the fragments (*e.g.,* with fluorescent techniques); putting back the small fragments together by aligning them (de novo) or mapping them to benchmark libraries. Errors can be introduced at each of these steps for various reasons [Beerenwinkel and Zagordi, 2011, O'Rawe et al. [2015]] and some errors can be quantified (*e.g.,* sequencing quality scores from chromatographs).

When the phylogenic tree to infer is based on pathogen sequences infecting hosts, the potential genetic diversity of the infection adds a complexity in phylogeny reconstruction. Typical examples are epidemiological studies reconstructing transmission trees from viral genetic sequences (*e.g.,* HIV, HepC) sampled from infected patients.

The different sources of uncertainty described above impact our observations of the actual genetic sequences. There are standard approaches to deal with identifiable observation errors. Base calls that are ambiguous (from equivocal chromatograph curves or because of genuine polymorphisms) are assigned ambiguity codes (*e.g.,* Y for C or T, R for A or G, etc.). Alignment methods are heuristic methods based on similarity scores that generally do not quantify the uncertainty of alignment. Methods to reconstruct phylogenies usually leave out the uncertainty complexity and settle for sequences composed of the most frequent nucleotides and/or ignore ambiguity codes (with some exceptions, e.g. DePristo et al. [2011]).

In 1998, Ewing and Green [1998] and Richterich [1998] both showed that estimates of the base call quality (called Phred scores) can be an accurate estimate of the number of errors that the machines at the time would make. Modern machines still report these Phred scores, but methods for adjusting/recalibrating these scores for greater accuracy have been proposed [Li et al., 2004, DePristo et al. [2011], Li et al. [2009]] For most analyses, these scores are used to censor the base calls (i.e., label them "N" rather than A, T, C, or G) if the base call error probability is too high or there are too few reads and a given location. It is commonplace to remove the sequence from analysis if the total sequence error probability is too high [see, e.g., Doronina, 2005, Robasky et al. [2014], O'Rawe et al. [2015]]. The error probability is deemed too high based on a strict threshold (e.g. 1% chance of error), but these thresholds aren't necessarily standard across studies.

Some studies have incorporated the error probabilities using genome likelihoods. DePristo et al. [2011] and Gompert and Buerkle [2011] incoporate the adjusted Phred scores into a likelihood framework that allows them to more accurately estimate population-level allele distributions. Fumagalli et al. [2013] present a new estimator for population genetic diversity based on such analyses. Kuo et al. [2018] use genome likelihoods to perform hypothesis tests that evaluate whether a given genome sequence is consistent with a given alternative sequence, after accounting for the errors. As with any parametric model-based approach, genome likelihoods use assumptions about the errors in order to make conclusions about the underlying patterns.

Few other authors have considered the uncertainty present in the sequences in downstream analyses. One notable exception is O'Rawe et al. [2015], who suggest methods for propagating sequence-level uncertainty into determining whether two subjects have the same alleles as well as incorporating the sequence-level uncertainty into confidence intervals for allele frequency.

In our work, we present a simple, general framework that can be incorporated into any analysis of genetic sequence data. This framework involves converting the uncertainty scores into a matrix of probabilities, and repeatedly sampling from this matrix and using the resultant samples in downstream analysis. Unlike likelihood-based approaches, we do not make assumptions about the underlying patterns in the data. In doing so, we gain generalizability at the expense of computation time. Our technique is amenable to any appropriate quality score adjustments prior to building the uncertainty matrix.

2 Methods

2.1 Probabilistic Representation of Sequences

Here, we describe two theoretical frameworks to model sequence uncertainty at the *nucleotide level* or at the *sequence level.* In both frameworks, the sequence of nucleotides from a biological sample is not treated as a certain observation, but rather as a collection of possible sequences.

2.1.1 Nucleotide-level uncertainty

To represent the uncertainty at each location along the genome, we introduce following matrix:

$$\mathcal{S} = \begin{array}{c} \\ \texttt{A} \\ \texttt{C} \\ \texttt{G} \\ \texttt{T} \\ \texttt{-} \end{array} \begin{array}{c} \begin{array}{cccc} 1 & 2 & \dots & \ell \end{array} \\ \begin{pmatrix} \mathcal{S}_{A,1} & \mathcal{S}_{A,2} & \dots & \mathcal{S}_{A,\ell} \\ \mathcal{S}_{C,1} & \mathcal{S}_{C,2} & \dots & \mathcal{S}_{C,\ell} \\ \mathcal{S}_{G,1} & \mathcal{S}_{G,2} & \dots & \mathcal{S}_{G,\ell} \\ \mathcal{S}_{T,1} & \mathcal{S}_{T,2} & \dots & \mathcal{S}_{T,\ell} \\ \mathcal{S}_{x,1} & \mathcal{S}_{x,2} & \dots & \mathcal{S}_{x,\ell} \end{pmatrix} \end{array} \tag{1}$$

Each column represents the nucleotide position, each row one of the four nucleotide `A`,`C`,`G`,`T` as well as an empty position "`-`"" that symbolizes a genuine deletion (not caused by missing data). Hence, $\mathcal{S}$ is a $5 \times \ell$ matrix. Its elements represent the probability that a nucleotide is at given position:

$$\mathcal{S}_{\texttt{n},j} = \mathbb{P}(\text{nucleotide } \texttt{n} \text{ is at position } j) \tag{2}$$

with the special case for a deletion:

$$\mathcal{S}_{\texttt{-},j} = \mathbb{P}(\text{empty position } j) \tag{3}$$

Note that we have for all $1 \leq j \leq \ell$:

$$\sum_{n \in \{\texttt{A,C,G,T,-}\}} \mathcal{S}_{n,j} = 1 \tag{4}$$

Also, the sequence length is stochastic if $\mathcal{S}_{\texttt{-},i} > 0$ for at least one i. The probability that the sequence has the maximum length ℓ is $\prod_{i=1}^{\ell}(1 - \mathcal{S}_{\texttt{-},i})$. We call the matrix $\mathcal{S}$ the *nucleotide-level probabilistic sequence* of a biological sample. The nucleotide (or deletion) drawn at each position is independent from all the others, so there are 5^ℓ possible different sequences for a given probabilistic nucleotide sequence, but these sequences are *not* equally probable.

2.1.2 Sequence-level uncertainty

Out of the 5^ℓ possible sequences, the nucleotide uncertainty may assign a positive probability to sequences that are not biologically possible. As an alternative representation and to reduce the space of possible sequences, let's assume we have enough information (either directly observed from data or simulated) to generate a set of sequences $\mathcal{B} = (\mathcal{B}_i)_{i \in \{1 \dots m\}}$ of all m biologically possible sequences. Note that the $\mathcal{B}_i$ do not have necessarily the same length. The observed genetic sequence, s, is a sample from a specified distribution a:

$$\mathbb{P}(s = \mathcal{B}_i) = a(i) \tag{5}$$

We call the set $\mathcal{B}$ the *sequence-level probabilistic sequence*. Note that, because a is a distribution, we must have $\sum_{i=1}^{m} a(i) = 1$.

If we have the following nucleotide-level probabilistic sequence:

$$\mathcal{S} = \begin{array}{c} \\ \texttt{A} \\ \texttt{C} \\ \texttt{G} \\ \texttt{T} \\ \texttt{-} \end{array} \begin{array}{c} \begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array} \\ \begin{pmatrix} 0.9 & 0.05 & 0.99 & 0 & 0 & 0.6 \\ 0 & 0.8 & 0 & 0 & 0.1 & 0.1 \\ 0.1 & 0.15 & 0 & 0.3 & 0.9 & 0 \\ 0 & 0 & 0.01 & 0.7 & 0 & 0.3 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix} \end{array}$$

then there are $2 \times 3 \times 2^3 \times 3 = 144$ possible sequences. The most likely is the one having the highest nucleotides probabilities: `ACATGA` with probability 0.2694 ($0.9 \times 0.8 \times 0.99 \times 0.7 \times 0.9 \times 0.6$).

If there is a positive probability of deletion for at least one position, then the sequence has a variable length. Let's take the same example as above, but adding one possible empty position:

$$\mathcal{S} = \begin{array}{c} \\ \texttt{A} \\ \texttt{C} \\ \texttt{G} \\ \texttt{T} \\ \texttt{-} \end{array} \begin{array}{c} \begin{array}{cccccc} {\scriptstyle 1} & {\scriptstyle 2} & {\scriptstyle 3} & {\scriptstyle 4} & {\scriptstyle 5} & {\scriptstyle 6} \end{array} \\ \begin{pmatrix} 0.9 & 0.05 & 0.99 & 0 & 0 & 0.6 \\ 0 & 0.8 & 0 & 0 & 0.1 & 0.1 \\ 0.1 & 0.15 & 0 & 0.2 & 0.9 & 0 \\ 0 & 0 & 0.01 & 0.7 & 0 & 0.3 \\ 0 & 0 & 0 & 0.1 & 0 & 0 \end{pmatrix} \end{array}$$

Like above, there is still a 0.2694 probability that the sequence is ACATGA, but now there is a chance that position 4 is deleted. For example, with probability 0.038 the sequence is ACA-GA.

Below is an example for a sequence-level probabilistic sequence $\mathcal{B}$:

sequence	a
ACATGA	0.60
ACATCA	0.12
AGATCA	0.15
ACAGA	0.05
GCATGA	0.08

% Sampling from $\mathcal{B}$, we will have for example ACATCA 12% of the time. Large genomes, such as SARS-CoV-2, will result in vanishly small probabilities for all sequences, and thus work on the log scale is often preferred.

2.2 Deletions and Insertions

By construction, the nucleotide-level probabilistic sequence must be defined with its longest possible length. Deletions are naturally modelled with our representation but insertions have to be modelled using deletion probability.

Consider the following nucleotide-level probabilistic sequence:

$$\mathcal{S} = \begin{array}{c} \\ \texttt{A} \\ \texttt{C} \\ \texttt{G} \\ \texttt{T} \\ \texttt{-} \end{array} \begin{array}{c} \begin{array}{cccccc} {\scriptstyle 1} & {\scriptstyle 2} & {\scriptstyle 3} & {\scriptstyle 4} & {\scriptstyle 5} & {\scriptstyle 6} \end{array} \\ \begin{pmatrix} 0 & 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0.99 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0.01 & 0 & 1 \\ 0 & 0.01 & 0 & 0.99 & 0 & 0 \end{pmatrix} \end{array} \tag{6}$$

The low deletion probability for position 2 is straightforward to interpret: about 1% of the time, nucleotide G at position 2 is deleted. The high deletion probability for position 4 means there is a 1% chance of a T insertion at this position. Table 1 illustrates this.

Table 1: Representation of insertions and deletions from $\mathcal{S}$ defined in (6)

sequence	**frequency**
CGAAT	common, 98% of the time
CAAT	rare (1% frequency) G deletion at position 2,
CGATAT	rare (1% frequency) T insertion at position 4
CATAT	very rare (0.01% frequency) deletion and insertion

The representation of deletions and insertions with a sequence-level probabilistic sequence (not nucleotide-level) is straightforward because in this framework the sequences are explicitly written out, so are their deletions/insertions.

2.3 Constructing The Uncertainty Matrix

Fragment sequencing error is an error that is quantified with quality (or "Phred") score attributed to each base call from sequencing instrument. The quality score Q is directly related to the error probability: $\epsilon = 10^{-Q/10}$. Ewing and Green [1998] (where Q typically ranges between 1 and 60). The FASTQ file format is the standard representation for combining sequence and observation error. Hence, the uncertainty associated to the base call is quantified by defining the probability that the observed nucleotide is the correct one:

$$\mathbb{P}(\text{nucleotide} = X \mid \text{observed nucleotide} = X) = 1 - \epsilon \tag{7}$$

Unfortunately, this base-call probability relates to only one *focal* nucleotide and we have no information on the probability for the three other possible nucleotides. Hence, we must make a modelling choice regarding the distribution of the remaining probabilities.

2.3.1 SAM Files

Massive parallel sequencing platforms (*e.g.,* Illumina, Oxford Nanopores, etc.) provide a large number of short reads sequences of the biological sample of interest. The length of those short reads are typically much smaller than the genome sequenced, so they have to be aligned and stitch together in order to re-assemble the full genome sequence. The short reads are typically stored in FASTQ files where the observation error of each nucleotide (estimated by the sequencing platform itself)) is indicated by its Phred score. The alignment and assembly of the short reads is performed by software (internal to the sequencing platform or not *((check this. Examples?))*) and generates a SAM file *((ref))* that efficiently stores the alignment information. The assembly of the short reads in the SAM file can be represented in as an array where the columns are the nucleotide positions. The short reads are "stacked" vertically according to the alignment. The number of short reads stacked for a given nucleotide gives the "coverage" of that position. See Figure 1 for an illustration of this SAM file representation.

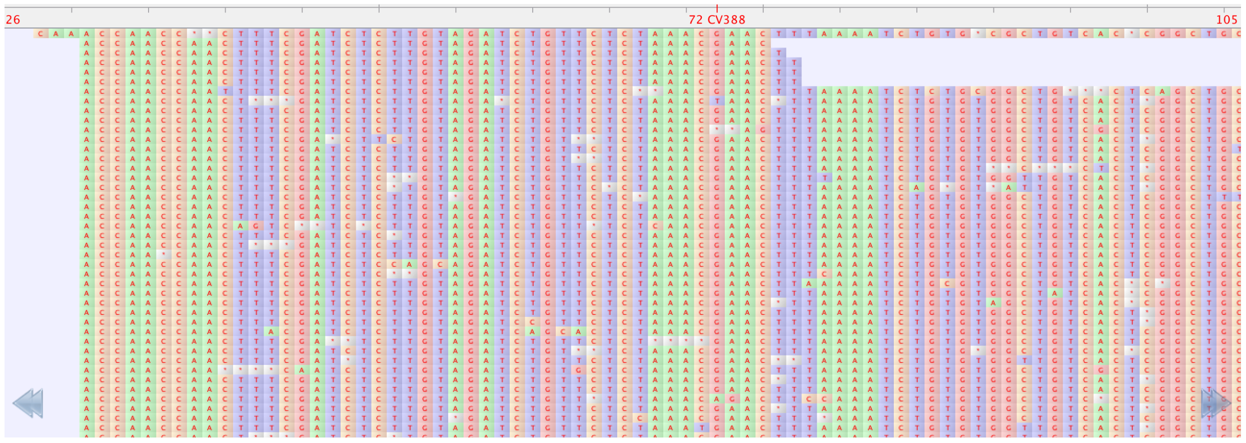

Figure 1: **SAM file graphical representation.** The software Tablet *((ref))* was used. Each base call is colour coded so that base calls that disagree with the consesnsu are obvious. The 72th nucleotide in this alignment has a coverage of 388 reads.

The nucleotide-level probabilistic sequence can be constructed from this alignment. The algorithm that we suggest is as follows.

We begin by initializing an empty $5 \times \ell$ matrix $\mathcal{S}'$. Each label in Figure 1 has an associated probability score. These labels denote a row in $\mathcal{S}$, and their location on the genome denotes the columns, say $(\texttt{A}, j)$. At this entry, we add the corresponding quality score to whatever was there before.

By this construction, the sum of each column represents the coverage at that location. Dividing each entry of $\mathcal{S}'$ by the sum of that column results in $\mathcal{S}$. For our proposed propagation methods, it is convenient to work with $\mathcal{S}'$ until probabilities are necessary.

It is important to note that many NGS platforms use paired reads where the same genome is read in both directions. In these situations, the two reads are not independent and should not be treated as such. Our implementation involves taking the average of the qualtiy scores. If the quality scores were 90% `A` for one read and 95% `A` in the second read, then 0.925 is added to the `A` row in $\mathcal{S}'$. If the two reads were 60% `A` and 55% `C` in the two reads, then the average of 0.6 and (1-0.55)/3 would be added to the `A` row, the average of (1-0.55)/6 and 0.55 would be added to the `C` row, and the average of (1-0.55)/3 and (1-0.6)/3 would be added to the other rows. This is equivalent to simply determining which reads are paired and dividing all values in their column by two before adding it to $\mathcal{S}'$. This is advantageous as it allows us to parallelize the reading of SAM files without needing to know which read it is paired with.

2.3.2 Consensus sequence FASTQ files

For most published genome sequences, the short read files are not available. It is common to find the so-called consensus sequence, which is the sequence that represents the most commonly called base at each location, along with a single quality score for each location. The methods for obtaining Phred scores when converting a SAM file to a consensus FASTQ differ by software [Li et al., 2004, Keith et al. [2002], Li et al. [2008]], but generally involve a computation that includes all of the Phred scores of the bases that agree with the consensus (*e.g.,* if the short reads have `A` with Phred of 30, `A` with a Phred of 31, and `C` with phred of 15, then the Phred scores of 30 and 31 are combined in software-defined way, usually incorporating extra information on top of the Phred score).

As a first simplifying step, we ignore insertions and deletions. Given a base call and its associated quality score at each position, we can assume that the other bases are all equally likely with probability $\epsilon/3$. For example, let's assume the output sequence after fragment sequencing and alignment is `ACATG` and its associated quality scores are respectively $Q = 60, 30, 50, 10, 40$. The probabilistic sequence is:

$$S = \begin{pmatrix} 1-10^{-6} & 10^{-3}/3 & 1-10^{-5} & 10^{-1}/3 & 10^{-4}/3 \\ 10^{-6}/3 & 1-10^{-3} & 10^{-5}/3 & 10^{-1}/3 & 10^{-4}/3 \\ 10^{-6}/3 & 10^{-3}/3 & 10^{-5}/3 & 10^{-1}/3 & 1-10^{-4} \\ 10^{-6}/3 & 10^{-3}/3 & 10^{-5}/3 & 1-10^{-1} & 10^{-4}/3 \\ 0 & 0 & 0 & 0 & 0 \end{pmatrix} \tag{8}$$

Usually, the genetic sequence `ACATG` would be considered as certain and quality scores discarded. In contrast, within the probabilistic sequence framework the probability the sequence is `ACATG` is only 0.899 $(= (1-10^{-6}) \times (1-10^{-3}) \times (1-10^{-5}) \times (1-10^{-1}) \times (1-10^{-4}))$.

Insertions and deletions ("indels' ') can be included in the uniform framework. Here, we propose that the nucleotides probabilities are defined conditional on an indel but other models are possible. For a given position, the error probability is $\epsilon = 10^{-Q/10}$ (Q is the quality score) and we assume the probability a deletion happens at this position is d. Conditional on not being deleted, the probability to have the base called is $(1-d)(1-\epsilon)$ and the other three nucleotides can occur with probability $(1-d)\epsilon/3$. Hence, if we assume the base call is `A`, the column of the nucleotide-level probabilistic sequence for that position is

$$\begin{pmatrix} (1-d)(1-\epsilon) \\ (1-d)\,\epsilon/3 \\ (1-d)\,\epsilon/3 \\ (1-d)\,\epsilon/3 \\ d \end{pmatrix} \tag{9}$$

2.3.3 Consensus sequence FASTA files

- Analysis will not draw any firm conclusions, but...
    - stress testing for your analysis, e.g. could resample genomes at different overall error rates, then simulate error rates from overall rate at each location, then see how much it F's up your analysis.

In the absense of any measured uncertainty we are constrained with what we can do. Any uncertainty that we impose upon the data will be a principled assumption. However, imposing our assumptions can still be very useful as a "stress test" of our analysis; we can assume different levels/types of uncertainty and quantify the stability of our results.

The error probability at any given nucleotide location can be simulated as a beta distribution, i.e.

$$\epsilon_j \sim \text{Beta}(\alpha, \beta)$$

The called base at position j will be assigned $1 - \epsilon$ as the probability, then the remaining bases (not includeing a gap) will be assigned $\epsilon_j/3$. To incorporate gaps, another probability d can be generated as the "gap probability". With these defiend, the nucleotide-level uncertainty sequence at the jth column (supposing the base call at location j was `A`) can be generated as:

$$\mathcal{S} = \begin{array}{c} \\ \texttt{A} \\ \texttt{C} \\ \texttt{G} \\ \texttt{T} \\ \texttt{-} \end{array} \begin{array}{c} {\scriptstyle j} \\ \left(\begin{array}{c} (1-d/4)(1-\epsilon_j) \\ (1-d/4)\epsilon_j/3 \\ (1-d/4)\epsilon_j/3 \\ (1-d/4)\epsilon_j/3 \\ d \end{array} \right) \end{array}$$

This uncertainty sequence is completely fabricated, but downstream analyses can be evaluated by choosing values of α, β, and d based on previous studies and propagating these uncertainties into downstream analysis.

2.4 Propagation of uncertainty via resampling

- Works in any analysis
- provides something like bootstrapped variance or an extra prior sampling step

The most general way to propagate uncertainty is through resampling. Given $\mathcal{S}$ and assuming that locations are independent (or pre-defining biologically-reasonable sequences and calculating their probabilities), uncertainty can be propagated by re-sampling the sequences according to their probability and re-running the analysis on each set of sampled sequences (I don't have a citation for this, but I believe Peter Piper sampled a set of sampled sequences).

Re-sampling can either be done simply based on the nucleotide-level uncertainty as a multinomial distribution. If the jth column of $\mathcal{S}$ is (0.5, 0.2,0.2,0.09.0.01), then we could sample `A` with 50% probability, `C` with 20%, etc. However, this disregards the coverage at each location, thereby disregarding the variance of the error probabilities.

To incorporate the variance of the errors, a Dirichlet-Multinomial sampling scheme can be derived by assuming that the error probabilities follow a Dirichlet prior and the bases follow a multinomial likelihood. For each base at each location, the entry for $\mathcal{S}'$ is defined as the sum of the probabilities (one minus the error probabilities). Thes values can be used as the parameters in the Dirichlet prior, from which base probabilities can be sampled.

For example, if the jth column of $\mathcal{S}'$ is (1200.45, 200.09, 100.74, 121.11, 0), then possible base probabilities could be (0.743, 0.126, 0.062, 0.070, 0), (0.739, 0.121, 0.066, 0.074, 0), or (0.744, 0.128, 0.056, 0.071, 0). In contrast, if the jth column of $\mathcal{S}'$ were (12.4, 1.2, 0.4, 1.7, 0), then some samples from the Dirichlet distribution might be (0.759, 0.059, 0.029, 0.153, 0), (0.916, 0.065, 0.009, 0.009, 0), or (0.779, 0.117, 0, 0.104, 0). The lower coverage is incorporated into the analysis by the increase in the variance of the Dirichlet distribution.

In a maximum likelihood framework, this is similar to bootstrapping. In fact, the ultimate effect of this would be to decrease the bootstrap confidence to a level that is more in line with the actual uncertainty in the data.

In a Bayesian framework, this could be incorporated as prior distribution on each nucleotide. For large collections of large sequences, this increases the dimensionality dramatically. In the next section, we provide a proposal for reducing the number of sequences that must be sampled.

2.5 Reducing computational burden with sequence-level uncertainty

Suppose we wish to use a single sequence to estimate some parameter θ (for instance, the p-value for whether a sequence is a particular variant or the lineage assignment for grafting to a pre-computed tree) and we are interested in the effect that uncertainty has on this estimate. Computational burden can be partially alleviated by incorporating sequence-level uncertainty.

To start with, the most likely sequences must be calculated. This could be done by calculating all of the sequences, comparing their likelihoods (labelled a_i), and putting them in order ($a_{(i)}$), but this is not necessary. Instead, the most likely sequence is already known (the consensus sequence). The second most likely sequence can be found by substituting the base call that was least likely (but still called) with the second most likely base call.

The calculation of the likelihood will be identical in these two cases except for the one substitution. Instead of calculating the entire likelihood (which can be numerically unstable since there are nearly 30,000 base pairs), the difference in the likelihood can be calculated.

Consider the two vectors M and m, where M_j represents the largest value in the jth column of $\mathcal{S}$ and m_j represents the second largest value. The likelihood of the consensus sequence is $a_{(0)} = \prod_{j=1}^{N} M_j$. Let k be the index of the smallest value in M, then the second most likely sequence has likelihood $a_{(1)} = \prod_{j\neq k} M_j m_k$. The change in likelihood is $a_{(0)} - a_{(1)} = \prod_{j\neq k}(M_k - m_k)$.

This process does not necessarily continue in a straightforward manner. At some point, it will be uncertain whether making a different substitution will have a larger effect on the likelihood than adding a substitution than one made in another sequence. The differences in likelihood derived above provide a very fast comparison of likelihoods, though, so it is not expensive to calculate both possible situations. Because of this speed-up, it is computationally feasible to simply calculate the likelihood for all substitutions involving the d least likely base calls that made it into the conseq and then sort them out later. There are $d * 2^d$ possible combinations of substitutions for d bases, which means that $d = 9$ will provide the 4,608 most likely sequences/

Suppose we are using these sequences to estimate some parameter θ. The sequences can be ordered according to their uncertainty, with the most likely sequence having uncertainty $a_{(1)}$, the second-most likely sequence $a_{(i)}$, etc. The analysis can be performed using the most likely sequence, resulting in an estimate $\theta_{(0)}$. The analysis can be re-run with the second-most likely sequence, $\theta_{(1)}$. Given just these two estimates, a weighted estimate of θ given the 2 most likely sequences could be found as:

DEVAN! BAD! $a_{(0)}$ has already been used! Maybe use prime.

$$\hat{\theta}_{(1)} \approx \theta_{(0)}\tilde{a}_{(0)} + \theta_{(1)}\tilde{a}_{(1)}, \text{ where } \tilde{a}_{(j)} = a_{(j)}/\sum_{i=1}^{2} a_{(i)}$$

The third-most likely sequence can be analysed, resulting in $\hat{\theta}_{(2)}$. This process can be repeated until new estimates do not have any impact on the results (conversion criteria could be set, but a graphical representation of $\theta_{(i)}$ would be more informative). If convergence is not clearly met, the d can be increased until enough sequences are obtained.

With this algorithm, the computational burden is separated into calculating the sequence liklihood for the most likely sequences and running the analysis enough times to be confident convergence is acheived.

For analyses that require a set of sequences, the likelihoods for the set of sequences can be calculated as the product of the likelihoods (or, for computational stability, the sum of the log-likelihoods) of the individual sequences. The algorithm above can be adjusted accordingly.

3 Applications

3.1 HIV

Copy from David, re-word as necessary.

3.2 SARS-CoV-2

In this section, we demonstrate the re-sampling method on lineage assignment of SARS-CoV-2. Sequences are sampled from $\mathcal{S}'$ and then assigned a lineage based on the current state-of-the-art in phylogenetic analyis.

3.2.1 PangoLEARN

We use the lineage designations described in Rambaut et al. [2020] and assign our sequences to lineages using the pangoLEARN tool (pangolin version 2.3.2, pangoLEARN version 2021-02-21) that the authors have made available (github.com/cov-lineages/pangolin). This tool uses a decision tree model to determine which lineage a given sequence is most likely to belong to. The tool also reports a bootstrap support probability as a measurement of the tool's confidence in its assignment. We demonstrate that even the best available tools are underestimating the variance and therefore producing overconfident conclusions.

3.2.2 Data

The data for this application were downloaded from NCBI's SRA web interface (https://www.ncbi.nlm.nih.gov/sra/?term=txid2697049). Search results were filtered to only include runs that had SAM files. To select which runs to download, a selection of 5-10 files from each of 20 non-sequential results pages were chosen. Once collecting the run accession numbers from the search results, an R script was run to download the relevant files and check that all information was complete. 23 out of 300 files were labelled incomplete due to having too few reads (possibly because the download timed out) or not containing a CIGAR string. The GISAID accession numbers for the seqeunces we used are provided in the appendix.

There was no particular reason for choosing any given file, but the resulting data should not be viewed as a random sample. Each result page likely includes several runs from the same study, and runs were chosen arbitrarily within each result page. We were not attempting a completely random sampling strategy, we simply wanted a collection of runs on which to demonstrate our methods.

3.2.3 Re-sampling the call probability matrix

Since pangoLEARN is a pre-trained model, the analysis is not computationally burdensome. Sampling 10,000 different sequences from a call probability matrix is reasonable, even on a mid-range consumer laptop.

For this analysis we use the most basic resampling strategy. We simply draw probabilities from the Dirichlet distribution with parameters defined by $\mathcal{S}'$, sample base calls from the multinomial distribution, then use pangoLEARN to determine the lineage assignment.

Locations with 0 reads were discarded. This is a somewhat arbitrary choice since random draws from the Dirichlet distribution when the sum of the parameters (which, in this case, corresponds to the coverage) is small are essentially draws from the uniform distribution.

In 3.2.3, each row represents resamples from one SAM file. Each bar represents one lineage assignment from pangolin. The width of the bars represents the sum of the weights (genome likelihoods) of sampled sequences. Bars are subdivided to visualize these weights.

Initial analyses used pangolin version 2.0.8, which used multinomial logistic regression, rather than decision trees, to assign lineages. This procedure resulted in bootstrap support values that varied from 0.6 to 1. After upgrading to pangolin version 2.3.2, which uses a decision tree model, all bootstrap probabilities are reported as 1.

3.2.4 Sequential re-analysis with genome likelihoods

TODO (Analysis not yet run - not sure how this will compare to resampling.)

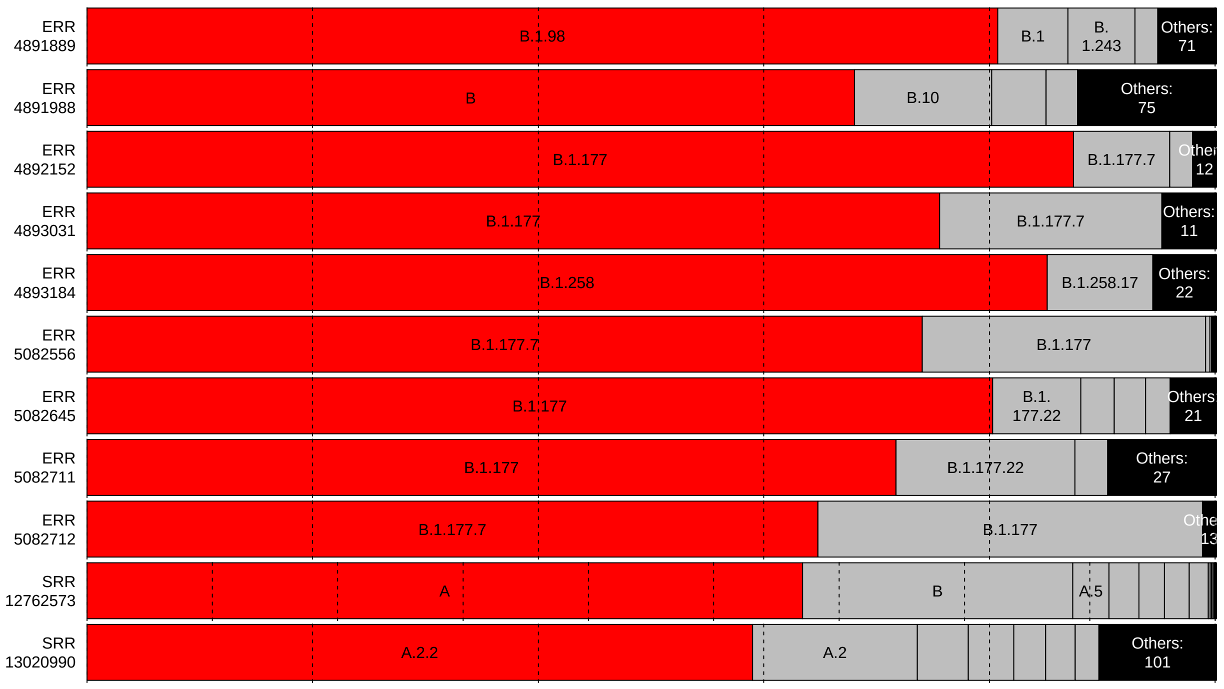

Figure 2: Distribution of called lineages from pangolin. Red bars indicate the lineage of the most probable sequence.

4 Conclusions

The short run files produced by next generation sequencing platforms include valuable information about the quality of base calls which can easily be propagated into analyses. Re-sampling allows a more honest appraisal of the variance in the estimates (or provides a reasonable prior distribution in a Bayesian setting), while comparing results for the most likely sequences provide a measure of robustness to sequence uncertainty.

Our proposed methods can result in a drastic increase in compuatational expense. Even the method base on ordering the sequences by likelihood can result in re-running the analysis numerous times. However, we have demonstrated that the uncertainty in the sequences themseleves can lead to major changes to the interpretations of the results. The so-called "consensus sequence" is simply the most likely sequence, and the reported uncertainty is not merely an academic curiousity.

Our analysis focused on phylogenetic trees based on HIV sequences as well as designation of lineages according to the Pangolin model. The importance of incorporating sequence uncertainty is not confined to these applications; any analysis involving sequenced genomes would benefit from some method of incoporating the uncertainty or including some measure of robustness.

4.1 For Pangolin

TODO

4.2 For phylogenetics in general

TODO

4.3 For analysis of genetic data

- Our method does not preclude tertiary analyses to test for systematic errors or deviations from a Mendelian inheritance pattern assumption.

- Our method allows for adjustments for Phred or more sophisticated genome likelihoods.

Bibliography

Niko Beerenwinkel and Osvaldo Zagordi. Ultra-deep sequencing for the analysis of viral populations. *Current Opinion in Virology*, 1(5):413–418, November 2011. ISSN 1879-6257. doi: 10.1016/j.coviro.2011.07.008.

Mark A DePristo, Eric Banks, Ryan Poplin, Kiran V Garimella, Jared R Maguire, Christopher Hartl, Anthony A Philippakis, Guillermo del Angel, Manuel A Rivas, Matt Hanna, Aaron McKenna, Tim J Fennell, Andrew M Kernytsky, Andrey Y Sivachenko, Kristian Cibulskis, Stacey B Gabriel, David Altshuler, and Mark J Daly. A framework for variation discovery and genotyping using next-generation DNA sequencing data. *Nature Genetics*, 43(5):491–498, May 2011. ISSN 1061-4036, 1546-1718. doi: 10.1038/ng.806.

N. V. Doronina. Phylogenetic position and emended description of the genus Methylovorus. *INTERNATIONAL JOURNAL OF SYSTEMATIC AND EVOLUTIONARY MICROBIOLOGY*, 55(2):903–906, March 2005. ISSN 1466-5026, 1466-5034. doi: 10.1099/ijs.0.63111-0.

Brent Ewing and Phil Green. Base-Calling of Automated Sequencer Traces Using *Phred.* II. Error Probabilities. *Genome Research*, 8(3):186–194, March 1998. ISSN 1088-9051, 1549-5469. doi: 10.1101/gr.8.3.186.

Matteo Fumagalli, Filipe G Vieira, Thorfinn Sand Korneliussen, Tyler Linderoth, Emilia Huerta-Sánchez, Anders Albrechtsen, and Rasmus Nielsen. Quantifying Population Genetic Differentiation from Next-Generation Sequencing Data. *Genetics*, 195(3):979–992, November 2013. ISSN 1943-2631. doi: 10.1534/genetics.113.154740.

Zachariah Gompert and C. Alex Buerkle. A Hierarchical Bayesian Model for Next-Generation Population Genomics. *Genetics*, 187(3):903–917, March 2011. ISSN 0016-6731, 1943-2631. doi: 10.1534/genetics.110.124693.

Jonathan M. Keith, Peter Adams, Darryn Bryant, Dirk P. Kroese, Keith R. Mitchelson, Duncan A. E. Coachran, and Gita H. Lala. A simulated annealing algorithm for finding consensus sequences. *Bioinformatics*, 18(11):1494–1499, November 2002. ISSN 1367-4803. doi: 10.1093/bioinformatics/18.11.1494.

Tony Kuo, Martin C. Frith, Jun Sese, and Paul Horton. EAGLE: Explicit Alternative Genome Likelihood Evaluator. *BMC Medical Genomics*, 11(2):28, April 2018. ISSN 1755-8794. doi: 10.1186/s12920-018-0342-1.

Heng Li, Jue Ruan, and Richard Durbin. Mapping short DNA sequencing reads and calling variants using mapping quality scores. *Genome Research*, 18(11):1851–1858, January 2008. ISSN 1088-9051, 1549-5469. doi: 10.1101/gr.078212.108.

Ming Li, Magnus Nordborg, and Lei M. Li. Adjust quality scores from alignment and improve sequencing accuracy. *Nucleic Acids Research*, 32(17):5183–5191, 2004. ISSN 0305-1048. doi: 10.1093/nar/gkh850.

Ruiqiang Li, Yingrui Li, Xiaodong Fang, Huanming Yang, Jian Wang, Karsten Kristiansen, and Jun Wang. SNP detection for massively parallel whole-genome resequencing. *Genome Research*, 19(6):1124–1132, January 2009. ISSN 1088-9051, 1549-5469. doi: 10.1101/gr.088013.108.

Jason A. O'Rawe, Scott Ferson, and Gholson J. Lyon. Accounting for uncertainty in DNA sequencing data. *Trends in Genetics*, 31(2):61–66, February 2015. ISSN 0168-9525. doi: 10.1016/j.tig.2014.12.002.

Andrew Rambaut, Edward C. Holmes, Áine O'Toole, Verity Hill, John T. McCrone, Christopher Ruis, Louis du Plessis, and Oliver G. Pybus. A dynamic nomenclature proposal for SARS-CoV-2 lineages to assist genomic epidemiology. *Nature Microbiology*, July 2020. ISSN 2058-5276. doi: 10.1038/s41564-020-0770-5.

Peter Richterich. Estimation of Errors in "Raw" DNA Sequences: A Validation Study. *Genome Research*, 8(3):251–259, January 1998. ISSN 1088-9051, 1549-5469. doi: 10.1101/gr.8.3.251.

Kimberly Robasky, Nathan E. Lewis, and George M. Church. The role of replicates for error mitigation in next-generation sequencing. *Nature Reviews Genetics*, 15(1):56–62, January 2014. ISSN 1471-0064. doi: 10.1038/nrg3655.