Systematic Biology (2022), **0**, 0, *pp.* 1–46
doi:10.1093/sysbio/main

ARPIP: Ancestral sequence Reconstruction with insertions and deletions under the Poisson Indel Process

Gholamhossein Jowkar[1,2,3,*], Jūlija Pečerska [1,2], Massimo Maiolo [1,2,4], Manuel Gil [1,2], Maria Anisimova [1,2]

[1] *Zurich University of Applied Sciences, School of Life Sciences and Facility Management, CH-8820, Wädenswil, Switzerland*
[2] *Swiss Institute of Bioinformatics, CH-1015 Lausanne, Switzerland*
[3] *University of Neuchâtel, Institute of biology, CH-2000 Neuchâtel, Switzerland*
[4] *University of Bern, Institute of Pathology, CH-3008 Bern, Switzerland*

**Gholamhossein Jowkar, ZHAW, School of Life Sciences and Facility Management, Applied Computational Genomics Group, Schloss 1, 8820 Wädenswil, Switzerland, E-mail: jowk@zhaw.ch*

Abstract

Modern phylogenetic methods allow inference of ancestral molecular sequences given an alignment and phylogeny relating present day sequences. This provides insight into the evolutionary history of molecules, helping to understand gene function and to study biological processes such as adaptation and convergent evolution across a variety of applications. Here we propose a dynamic programming algorithm for fast joint likelihood-based reconstruction of ancestral sequences under the Poisson Indel Process (PIP). Unlike previous approaches, our method, named ARPIP, enables the reconstruction with insertions and deletions based on an explicit indel model. Consequently, inferred indel events have an explicit biological interpretation. Likelihood computation is achieved in linear time with respect to the number of sequences. Our method consists of two steps, namely finding the most probable indel points and reconstructing ancestral sequences. First, we find the most likely indel points and prune the phylogeny to reflect the insertion and deletion events per site. Second, we infer the ancestral states on the pruned subtree in

a manner similar to FastML. We applied ARPIP on simulated datasets and on real data from the *Betacoronavirus* genus. ARPIP reconstructs both the indel events and substitutions with a high degree of accuracy. Our method fares well when compared to established state-of-the-art methods such as FastML and PAML. Moreover, the method can be extended to explore both optimal and suboptimal reconstructions, include rate heterogeneity through time and more. We believe it will expand the range of novel applications of ancestral sequence reconstruction.

Key words: joint ancestral sequence reconstruction, ancestral sequences, maximum likelihood, indel, phylogeny, dynamic programming, Poisson indel process, SARS-CoV, evolutionary stochastic process.

Phylogenetics is a wide research field with a variety of applications ranging from reconstructing the tree of life to investigating ongoing epidemics. Phylogenetic trees provide insight into unobservable evolutionary events in the past such as adaptation or mass extinction events. Phylogenetic inference can be divided into several interrelated tasks including sequence alignment, phylogeny estimation, detection of selection, and ancestral sequence reconstruction (ASR). ASR aims to infer the likely ancestral sequences for a set of existing homologous sequences.

ASR allows researchers to pursue a wide range of topics from determining the origins of life or epidemics to developing personalised medicine (Pagel, 1999; Liberles, 2007). For example, the functionality of ancient genes can be investigated by reconstructing and synthesising the genetic material inferred by ASR (Thornton, 2004). Such analyses can help us understand the mechanisms underlying adaptation and speciation processes, inspiring new approaches for protein engineering (Chang et al., 2005) and drug design (Zakas et al., 2017). ASR can be used to study epidemiological origins of pathogens, particularly in light of recent coronavirus pandemics (Pagel, 1999; Brintnell

et al., 2021; Starr et al., 2022).

State-of-the-art likelihood-based ASR methods use Markov processes to model character substitutions through time. Such models account for various biases in character substitution, as well as divergence represented by evolutionary time (Yang et al., 1995; Yang, 2007; Pupko et al., 2000). However, Markov models of molecular evolution do not include insertions or deletions (indels) as part of the evolutionary process, meaning that methods relying on these models have to treat gap characters separately. Most of the ASR methods adopt one of two pre-processing approaches. They either treat gaps as missing/ambiguous data or remove gap characters entirely. However, indels represented by gaps carry an important evolutionary signal (Dessimoz and Gil, 2010), and are in fact a major driving force of genomic divergence (Tao et al., 2007). Therefore, methods that model indels explicitly have a clear advantage over methods that do not. Up to this point, most existing frequentist algorithms do not include indel modelling except for two methods, Ancestors (Diallo et al., 2009) and FastML (Ashkenazy et al., 2012). Ancestors has exponential computational complexity and therefore has not been widely adopted by users. FastML handles indels using a heuristic approach called indel-coding. The method relies on the linear time complexity algorithm (Pupko et al., 2000) for joint maximum likelihood (ML) ASR using dynamic programming (DP). FastML makes the analyses of large datasets tractable. Currently, it is provided as a web service (Ashkenazy et al., 2012). While the results of indel-coding can be interpreted from an evolutionary standpoint retrospectively, the approach does not, however, include an explicit evolutionary indel model. All things considered, most methods rely on standard models of sequence evolution without indels which is an issue that can only be resolved by including character and indel evolution in a single model.

Two pioneering mathematical models describing the evolution of indels are TKF91 and TKF92 (Thorne et al., 1991, 1992). However, the computation of marginal likelihood under these models has exponential time complexity, rendering the methods relying on

these models extremely computationally intensive, and making inference under these models unrealistic on large datasets. More recently, Bouchard-Côté and Jordan (2013) proposed the Poisson Indel Process (PIP) model which is based on TKF91. PIP describes insertions by a Poisson process defined on the tree topology, while substitutions and deletions are described by a continuous-time Markov process where deletions are modelled as an absorbing state. The assumption of independence between insertion and substitution/deletion enabled a major computational improvement over the previous models (Bouchard-Côté and Jordan, 2013). In PIP, the insertion rate is also independent of the length of a sequence, which is a realistic assumption based on the data that is most commonly analysed (Bouchard-Côté, 2010, p. 93). In contrast to TKF91, the PIP model allows to compute marginal likelihoods in linear time with respect to the number of sequences, which enables a variety of phylogenetic applications (e.g., Maiolo et al. (2018)).

In this study, we use the PIP model for joint reconstruction of ancestral character states including insertions and deletions. Our method ARPIP (Anscestral Reconstruction under PIP) is implemented in the ML framework, i.e., we use an empirical Bayesian approach with ML estimates. Given a multiple sequence alignment (MSA) and a phylogenetic tree, we first use PIP to infer insertion and deletion points on the tree. Insertion and deletion points are the specific locations on the phylogeny where the events have happened. Next, we extract a subtree rooted at the insertion point and pruned by the deletion points. Finally, we reconstruct ancestral states on the extracted subtree using a modified version of Felsenstein's recursion (Felsenstein, 1981), similar to the FastML algorithm (Pupko et al., 2000). In the following we describe the method in detail, validate it by simulations, and demonstrate its performance in simulations and on a real dataset.

Materials and Methods

ARPIP consists of two main algorithms: indel point inference and ancestral character inference.

The IndelPoints algorithm infers the most likely indel points for each site m of the given alignment (Appendix 2). It traverses the tree in post-order and evaluates a set of possible indel scenarios for each node in the tree. A particular indel scenario defines a homology path $\mathcal{H}$. A homology path contains a single insertion point and a number of deletion points consistent with the input MSA. IndelPoints finds the most likely indel scenario by maximizing the probability of $\mathcal{H}$ given m. The maximisation is simplified by reducing the MSA to gap and non-gap states, and ignoring the substitution history without changing the result of the computation. This allows us to avoid matrix exponentiation, which is computationally expensive but necessary for the full likelihood computation.

Similar to the recursive likelihood computation, we traverse the tree and evaluate all the possible indel scenarios, selecting the best one at each node. At the tree root, we select the best homology path over the whole tree based on the best paths selected in the child nodes. For each site m, we use the inferred homology path to extract a subtree τ_m rooted at the insertion point $\mathcal{I}$ and pruned by deletion points $\mathcal{D}$, which represents the most likely indel history for the given site.

Next, we reconstruct ancestral characters on the pruned subtrees in a manner similar to FastML (Pupko et al., 2000). For each site m, we use DP to reconstruct ancestral characters in two phases. The first phase can be seen as a modification of Felsenstein's peeling recursion for computing marginal likelihoods (Felsenstein, 1981). As in the peeling recursion algorithm, we traverse the tree τ_m in post-order, starting from the leaves upward to the root and propagate partial likelihoods. However, instead of marginalising over internal character states, for each MSA column m we store the likelihood values Lk_v and the corresponding best ancestral character states CS_v for each node v. In the second phase, the algorithm traverses the tree in pre-order and for each node selects the ancestral character A_v with the highest conditional probability.

Preliminaries: the PIP model

The PIP model describes the evolutionary process of substitutions, insertions and deletions along the branches of a phylogenetic tree τ. Here we include the basic description of the process, additional information on the PIP likelihood is available in Appendix 1 and a detailed description of PIP can be found in Bouchard-Côté and Jordan (2013).

Let $\tau = (\mathcal{V}, \mathcal{E}, \mathrm{b})$ represent a rooted binary phylogenetic tree, where set $\mathcal{V}$ is the set of all vertices of the tree, $\mathcal{E}$ is the set of all tree branches ($\mathcal{V} \times \mathcal{V}$) and b refers to the branch lengths in units of time (measured in expected substitutions and deletions per site).

The observed sequences are strings of characters from an alphabet Σ, which can be nucleotides, amino acids or codons. The N observed sequences at the leaves of τ are denoted by set $\mathcal{L} \subset \mathcal{V}$, whereas set $\mathcal{V} \setminus \mathcal{L}$ is the set of $N-1$ internal vertices. The root, the most recent common ancestor of all leaves, is labelled by Ω. The branch length $\mathrm{b}(v)$ associated with node ($v \in \mathcal{V}$) spans from v to its parent vertex $\mathrm{pa}(v)$ (see Fig. 1).

PIP is parameterised by insertion rate λ and deletion rate μ, with the process running over tree topology τ. For every node $v \in \mathcal{V}$, the probability of inserting a single character on edge $e = (\mathrm{pa}(v) \to v)$ is proportional to the branch length and defined by $\iota(v)$ (see Appendix 1). Similarly, the survival probability for a character inserted on edge e is $\beta(v)$ (see Appendix 1). Additionally, we define the pure survival probability $\zeta = \exp(-\mu \mathrm{b}(v))$ associated with node v as if the character was already present at the parent node $\mathrm{pa}(v)$ (Maiolo, 2019). Point substitutions and deletions are modelled by a continuous-time Markov process on $\Sigma_\varepsilon = \Sigma \cup \{\varepsilon\}$, where ε denotes the gap symbol. The generator matrix Q could be any arbitrary reversible substitution model, e.g. WAG for amino acids (Whelan and Goldman, 2001), or K80 for nucleotide data (Kimura, 1980). Accordingly, the extended generator matrix is denoted by Q_ε and the extended quasi-stationary distribution is $\pi_\varepsilon = [\pi, 0]$ (Bouchard-Côté and Jordan, 2013).

Let $\mathcal{G}$ define the site-specific set of all potential deletion points on the tree. $\mathcal{G}$ consists of all leaves with a gap at the respective site, and of all the internal nodes whose

all descendant leaves have a gap at that site. Next, consider the subset $\mathcal{S}$ of leaves that have a non-gap character, $\mathcal{S} = \{v \in \mathcal{L} : m_v \neq \varepsilon\}$. Given the set $\mathcal{S}$, we define the set $\mathcal{A}$ of potential insertion points to include all nodes that are ancestral to all the leaves in $\mathcal{S}$ (see Fig. 1). In general, we compute the probability $p(m)$ of each individual MSA column by marginalizing over all possible homology paths underlying that MSA column (see Appendix 1) based on their homology path probabilities f_v.

Inferring the indel points

We propose a progressive algorithm to infer the most likely indel points (homology path) on the tree under the PIP model. For each site, we progressively find the best partial homology path (constrained by a subtree) and build on the intermediate results to get the most likely indel history on the whole tree. Since we search only for the most likely homology path, we compute a simplified likelihood function which accounts only for insertions and deletions and ignores substitutions.

Under PIP, two mutually exclusive node sets exist on the tree topology τ: the set of nodes where the character has gone extinct and the set of nodes where the character has definitely survived. The first is the set of potential deletion nodes $\mathcal{G}$, defined in the previous section. The second contains all the remaining nodes in the tree $v \in \mathcal{V} \setminus \mathcal{G}$ ($v \notin \mathcal{G}$). A node $v \notin \mathcal{G}$ may also be a potential insertion location, i.e. $v \in \mathcal{A}$ (see Fig. 2 and Appendix 2 for the detailed description). While the set $\mathcal{A}$ may contain multiple nodes, a homology path can only have a single insertion location. Consequently, each node in $\mathcal{A}$ is associated with a single homology path with the highest probability. This implies that when computing the probability of a homology path for node $v \in \mathcal{A}$, it is treated as the only potential insertion location, while all other nodes are treated as regular nodes in the tree. Notably, one cannot simply select the node with the highest insertion probability $\iota(v)$, as the probability of any given homology path also depends on the survival/extinction of the site in the children. Even though we separately describe the treatment of the two node types, all the necessary

computation can be done in a single post-order traversal of the tree.

For each node v in the tree, we first compute f_v, the conditional probability of the deletion/substitution process, assuming that the character exists in v. We compute f_v for the most likely deletion scenario in this subtree rather than marginalise over all possible deletion locations. We also compute p_v, the conditional probability of the homology path assuming that the character was inserted at node v. A character necessarily has to be inserted at one of the nodes $v \in \mathcal{A}$, which means that the probability will be non-zero only for the potential insertion nodes.

In the progressive algorithm we maintain several node sets that are needed to define the most likely homology path per node. Let $\mathcal{I}_v$ denote the set of insertion points for the subtree rooted at v. Then $\mathcal{I}_v = \emptyset$ for $v \notin \mathcal{A}$ and $\mathcal{I}_v = \{v\}$ for $v \in \mathcal{A}$. Similarly, $\mathcal{D}_v$ denotes the set of deletion points for the subtree rooted at v.

For each node v we store the locally optimal homology path $\mathcal{H}_v = \{\mathcal{I}_v, \mathcal{D}_v\}$ for the subtree rooted at v. Once we reach the root node Ω, all possible insertion locations would be considered, and the one with the highest probability is selected among those. At this point, the best homology path $\mathcal{H}_{\arg\max(p_v)}$ (defined by the highest conditional probability p_v) is used to extract the subtree τ_m rooted at $\mathcal{I}_{\arg\max(p_v)}$ and pruned by $\mathcal{D}_{\arg\max(p_v)}$, which represents the best possible indel points for the MSA column m. We will use τ_m to infer ancestral character states for column m. The IndelPoints algorithm is presented in Figure 2 and the pseudocode for the algorithm can be found in the Appendix 2.

Dynamic programming joint ancestral sequence reconstruction

Our method performs ASR in a manner very similar to FastML (Pupko et al., 2000) with two crucial differences. First, we only work on a subtree τ_m of the original tree τ, which limits the reconstruction to the most probable insertion location at this site. This means we do not reconstruct any ancestral states where there were none. Second, to appropriately account for character deletion, the ancestral reconstruction is done using the

PIP substitution rate matrix Q_ϵ.

The joint ASR method under PIP given column m and the pruned rooted phylogenetic subtree τ_{m} consists of two steps. The first step is to compute the partial likelihood values on subtree τ_{m} with the modified version of Felsenstein recursion algorithm, where both likelihood values and their corresponding ancestral character states are stored. The second step is to reconstruct the character states by picking the character with highest conditional probability. The recursive algorithm for joint ASR is shown in Appendix 3 together with the newly defined pseudocode for the procedure.

Results

Three datasets were used to evaluate and illustrate our method. The first dataset was simulated under the PIP. This dataset allows to evaluate the performance of ARPIP under the true model. Given the true simulated trees and MSAs, both the homology path inference and ASR were evaluated.

The second dataset was used to evaluate the performance of ARPIP for sequences with long indels. The data was generated by INDELible (Fletcher and Yang, 2009) with two different settings using the same trees as for the PIP simulations. For this dataset, the ancestral sequences are also known and can be used for evaluation. However, the PIP parameters for this dataset must be inferred. As INDELible does not provide a comprehensive description of indel events on the phylogeny, we used these simulations to evaluate ancestral state inference.

The third dataset is a small coronavirus sample which was extracted from Uniprot (Bateman et al., 2020). With this dataset, we aimed to provide a showcase of the method.

When analysing both INDELible and real-life data, we first have to infer the PIP parameters, i.e. λ and μ, given an MSA and a tree. This computation was done based on Brent's optimization method (Brent, 1973), optimizing one parameter at a time until convergence. For all examples, the protein substitution model used is WAG (Whelan and

Goldman, 2001).

Note that ARPIP was developed for rooted trees. If an unrooted tree is provided, ARPIP uses mid-point rooting method to root the tree. Furthermore, ARPIP can perform ASR without a provided tree. The user can select from established fast methods like neighbor joining, BioNJ, UPGMA, and WPGMA to estimate the tree from the input MSA.

Data simulated under PIP

The simulated sequences are given as input to ARPIP along with the true model parameters so that we only have to estimate the ancestral state values. The simulated dataset contains 100 MSA/tree replicates with their corresponding evolutionary events. Each replicate was simulated using an 8 taxa tree with a topology sampled from the uniform distribution and branch lengths sampled from an exponential distribution with the rate $\rho = 2$, where ρ is a proxy for phylogenetic divergence. One of the simulated trees is shown in Figure 3. On average, the branch lengths of the simulated trees were 0.45 units of time, ranging from minimal branch length of 0 and maximum branch length of 3.23. For the simulations, we set the deletion rate $\mu = 0.1$ and the insertion rate $\lambda = 10$ for PIP.

Analysis of the PIP simulated dataset.— In order to assess the accuracy of ARPIP, we independently evaluated each inference step, IndelPoints and the joint ASR (see Table 1). To asses the accuracy of the IndelPoints algorithm, we also evaluated the inference of insertion and deletion events independently. As this dataset was simulated under the same model we use for inference, we used the true parameter values in the analysis without inferring them ($\mu = 0.1$ and $\lambda = 10$). This way we can evaluate the method without the additional variation of parameter inference, which is done for the other two datasets.

Among the 100 input sets, 96.69% insertion points and 95.54% deletion points were inferred correctly. In the next step, we computed the accuracy of ASR per site. This

number has been averaged over all existing sites over all MSA replicates. To evaluate the reconstructed ancestral sequences, we used three different metrics. Firstly, we counted the number of full ancestral columns that were inferred correctly, which is 60.08% for this dataset. Secondly, we counted the number of characters that were inferred correctly, which amounts to 88.14% of characters. Thirdly, we counted the number of gap characters themselves that were inferred correctly, which amounts to 99.86%.

Data generated by INDELible

The data simulated by INDELible contains two sets of 100 MSA/tree replicas. Each replica was simulated using an 8 taxa tree from PIP simulations. We used the Zipfian (power law) distribution for the indel model with $a = 1.7$, to generate the samples where $a > 1$ is the exponent characterizing the distribution. Empirical estimates of value a range from 1.5 to 2 (Fletcher and Yang, 2009), which prompted us to select $a = 1.7$. The maximum indel length was set to 5 to avoid MSAs with excessively long gaps. Two different indel rates of 0.01 and 0.05 were used for the simulation with INDELible.

Analysis of the INDELible simulated dataset.— For this dataset, ARPIP inferred the PIP parameters λ and μ as well as ancestral character states. For the two datasets of 100 MSA/tree replicates with indel rates of 0.01 and 0.05, ARPIP correctly inferred 46.58% and 59.49% of the ancestral sites respectively. Further, ARPIP correctly inferred 83.49% and 87.93% of characters including gaps. Finally, over 99.95% of gap characters were inferred correctly for the two datasets (see Table 2).

Coronavirus data

The ongoing *SARS-CoV-2* pandemic strongly affects our lives, causing an immense interest for phylogenetic analyses of the relevant viral molecular sequences. Like in other coronaviruses, the spike protein in *SARS-CoV-2* is important for viral entry into host cells.

It is also one of the major determining factors of host range (Belouzard et al., 2012; Zhou and Zhao, 2020). We therefore used this protein as an example demonstrating ancestral sequence inference.

SARS-CoV-2 is a member of the *Betacoronavirus* genus which also contains the two other recent human coronavirus strains, namely *SARS-CoV* and *MERS-CoV* (Lefkowitz et al., 2018). For our analyses, we selected a small set of available protein sequences from this genus (see Table 3 for the exact sequence list).

Analysis of the coronavirus dataset.— The MSAs of the coronavirus sequences were inferred using ProPIP (Maiolo et al., 2018) and PRANK phylogeny-aware webserver (Löytynoja, 2014). The total length of the reconstructed MSAs was 2002 and 1929 AAs respectively. The phylogenetic trees were reconstructed by ML in PhyML 3.0 (Guindon et al., 2010), using smart model selection on amino acids and SPR tree moves (Lefort et al., 2017) (see Fig. 4). Then, given an MSA and tree we inferred ancestral sequences with ARPIP. The estimated deletion rates for ProPIP and PRANK's MSAs are respectively $\hat{\mu} = 0.242$ and $\hat{\mu} = 0.210$. Figure 5 summarises the resulting ASR by ARPIP comparing to FastML on MSA produced by ProPIP while the results for PRANK can be found in the supplement.

Comparison against the state-of-the-art methods

At this moment, the two most frequently used ML joint ASR approaches are PAML (Yang, 1997) and FastML (Pupko et al., 2000; Ashkenazy et al., 2012), both of which work in linear time with respect to the number of sequences. An important distinction among the methods lies in the way they handle gaps in the alignment. PAML can either ignore gaps in the alignment by removing all columns containing at least one gap character (option used here), or treat all gap characters as ambiguous. For this study, we used PAML on an MSA without any gap characters to compare the accuracy of ancestral

character reconstruction. FastML webservice (Ashkenazy et al., 2012) uses an ad-hoc indel-coding (Simmons and Ochoterena, 2000) approach to account for indels spanning multiple adjacent characters. Indel coding is done as a separate step in the inference process, done independently from the ancestral state reconstruction. We compare the performance of ARPIP and FastML on an MSA with gaps.

On both simulated datasets, the accuracy of ARPIP appears similar to FastML (see Tables 1- 2 and Figs. 6, 8 and 9). For example, both algorithms inferred the ancestral state accurately in certain regions (e.g., Fig. 6a, Fig. 8a and d, and Fig. 9a) and falsely in other regions (e.g., Fig. 6b, Fig. 8b and c, and Fig. 9b). In region c of Figure 6 and region d of Figure 9, FastML inferred a character state even though there is no ancestral character, since the insertion happened at the leaf. In certain regions FastML could not determine which internal node had the information (e.g., Fig. 6d), while ARPIP was capable of determining the character position accurately. ARPIP outperformed FastML in determining the gap position in certain regions (e.g., Fig. 6c and d, Fig. 8b, and Fig. 9c and d). On CoV data, the situation was similar meaning FastML could not detect the insertion location (see Fig. 5a) while in conserved regions the inferred states were almost identical (see Fig. 5b).

We also considered scenarios without indels, allowing the evolutionary process to work only through substitutions. In this case the alignments have no gaps, i.e. no deletion ($\mu = 0$) and all insertions happen at the root of the tree. Under these conditions, all the algorithms perform reasonably as presented in Figure 7.

Discussion and conclusion

In this article, we present a one-of-a-kind approach for fast likelihood-based ancestral sequence reconstruction with insertions and deletions. Unlike previous approaches, our method relies on an explicit model of indel and character evolution and allows us to infer the full history of sequence evolution, including insertion and deletion

points on a phylogeny. The method is implemented in the probabilistic framework and is based on likelihood calculations under the PIP model. Likelihood computations under this model have linear time complexity with respect to the number of sequences, meaning that our method is highly efficient on large datasets.

We show that on PIP simulated datasets, ARPIP correctly infers at least 95% of indel events and at least 88% of ancestral characters. On the INDELible simulated data, ARPIP correctly infers 83% and 87% of ancestral characters including gaps for low and high indel rates respectively. ARPIP also correctly places gaps in over 99.95% cases, showing the credibility of our IndelPoints algorithm. For all datasets, we illustrate the performance of our approach in comparison to FastML on alignments with gaps. In addition, we use gapless alignments to compare ARPIP inferences with those by PAML and FastML, showing that our approach performs just as well for data without gaps.

While indel events represent a major mutational process of gene evolution (Söding and Lupas, 2003), they are rarely accounted for in ASR. ARPIP expands the reconstruction possibilities to include more divergent and gappy sequences allowing us to study a wider range of resurrected ancestral molecules, investigating the functional importance of indels in ancestral proteins. This is particularly valuable for proteins separated by large divergences or within "more flexible" loop regions, as indels frequently occur in regions where amino acid sequences are not well conserved (Taylor et al., 2004a).

While single residue indel modelling may be viewed as a limitation, certain types of genetic material exhibit specifically these kind of indel events more often than others. For example, single nucleotide indels are predominant between recently diverged DNA sequences from various organisms (Tao et al., 2007) and in non-coding DNA sequences (Yamane et al., 2006). While most ASR is done on coding sequences to investigate the properties of reconstructed proteins, it has recently been shown that many trait-associated loci, including some associated with disease, lie outside protein-coding regions (Kellis et al., 2014). ARPIP can be used to reconstruct non-coding sequences with meaningful biological

assumptions, which could be an additional avenue of exploration for disease-related ASR.

ARPIP paves the way for even more new types of indel analyses. The approach can be expanded to analyse the patterns of insertions and deletions by including rate heterogeneity, for example, allowing us to detect lineage-specific patterns through time. We can include site-specific indel rate variation, allowing us to see the difference in indel evolution in different functional regions of proteins such as loops or active sites. Then, we can investigate the occurrence and consequences of indels in specific regions such as indel-tolerant regions of the genome and relation between gene function and indel frequency (Taylor et al., 2004b). Moreover, in the long run our method can be used to extend and potentially improve more sophisticated probabilistic approaches such as (Groussin et al., 2014), which accounts not only for gene-trees but also for species history, therefore including gene gain/loss and horizontal transfer in the inference.

While some other methods have attempted to reconstruct exact indel histories, the only other currently existing method in the frequentist framework can only handle small datasets (Diallo et al., 2007). Even though PIP makes simplifying assumptions like site independence, which only allows us to model single residue indels, the explicit evolutionary model makes indel events interpretable. Moreover, as the method has linear time complexity, we can use this approach as a building block in integrated alignment-tree-ancestor inference (Pečerska et al., 2021), using the indel points under PIP as a starting point for integrating more complex models of indel evolution, e.g. moving on to long indel models.

Like other likelihood based approaches, our method in theory allows us to explore both optimal and suboptimal reconstructions in follow-up analyses. It has been argued that a single reconstruction (i.e. a point estimate) can be inadequate in cases when the likelihood surface is nonconvex and contains multiple local optima (Joy et al., 2016), which can lead to systematic bias (Yang, 2014, p.131). Since ARPIP is in essence an empirical Bayes method, we can extend the method to account for the uncertainty in our estimates

by working with probability profiles of characters and gaps rather than inferences fixed to the optimal estimates (Williams et al., 2006).

Availability of code and experimental data

The proposed algorithm has been implemented based on Bio++ open source library (Guéguen et al., 2013) using C++ programming language. Our code with a brief user manual is freely available at `https://github.com/acg-team/bpp-ARPIP` under GNU GPLv3 licence. The data underlying this article are also available from the Dryad Digital Repository at `https://doi.org/10.5061/dryad.wstqjq2nj`.

Funding

This work was supported by the Swiss National Science Foundation (SNSF) (grant number 31003*A*_176316 to M.A). The funding body did not play any role in the design of the study and collection, analysis, and interpretation of data and in writing the manuscript.

Acknowledgements

We would like to thank A. Bouchard-Côté (University of British Columbia) for providing his code JavaPIP to simulate sequences under the PIP, and our master student J. Peechatt for his preliminary work which helped to develop this method.

References

Ashkenazy, H., O. Penn, A. Doron-Faigenboim, O. Cohen, G. Cannarozzi, O. Zomer, and T. Pupko. 2012. Fastml: a web server for probabilistic reconstruction of ancestral sequences. Nucleic acids research 40:W580–W584.

Bateman, A., M.-J. Martin, S. Orchard, M. Magrane, R. Agivetova, S. Ahmad, E. Alpi,

E. H. Bowler-Barnett, R. Britto, B. Bursteinas, et al. 2020. Uniprot: the universal protein knowledgebase in 2021. Nucleic Acids Research 49:D480D489.

Belouzard, S., J. K. Millet, B. N. Licitra, and G. R. Whittaker. 2012. Mechanisms of coronavirus cell entry mediated by the viral spike protein. Viruses 4:1011–1033.

Bouchard-Côté, A. 2010. Probabilistic Models of Evolution and Language Change. Ph.D. thesis University of California at Berkeley University of California at Berkeley phD thesis.

Bouchard-Côté, A. and M. I. Jordan. 2013. Evolutionary inference via the poisson indel process. Proceedings of the National Academy of Sciences 110:1160–1166.

Brent, R. P. 1973. Algorithms for minimization without derivatives. Englewood Cliffs, NJ, USA Prentice Hall Page 195.

Brintnell, E., M. Gupta, and D. W. Anderson. 2021. Phylogenetic and ancestral sequence reconstruction of sars-cov-2 reveals latent capacity to bind human ace2 receptor. Journal of molecular evolution 89:656–664.

Chang, B. S., J. A. Ugalde, and M. V. Matz. 2005. Applications of ancestral protein reconstruction in understanding protein function: Gfp-like proteins. Pages 652–670 *in* Methods in enzymology vol. 395. Elsevier.

Dessimoz, C. and M. Gil. 2010. Phylogenetic assessment of alignments reveals neglected tree signal in gaps. Genome biology 11:R37.

Diallo, A. B., V. Makarenkov, and M. Blanchette. 2007. Exact and heuristic algorithms for the indel maximum likelihood problem. Journal of Computational Biology 14:446–461.

Diallo, A. B., V. Makarenkov, and M. Blanchette. 2009. Ancestors 1.0: a web server for ancestral sequence reconstruction. Bioinformatics 26:130–131.

Felsenstein, J. 1981. Evolutionary trees from dna sequences: a maximum likelihood approach. Journal of molecular evolution 17:368–376.

Fletcher, W. and Z. Yang. 2009. Indelible: a flexible simulator of biological sequence evolution. Molecular biology and evolution 26:1879–1888.

Groussin, M., J. K. Hobbs, G. J. Szllsi, S. Gribaldo, V. L. Arcus, and M. Gouy. 2014. Toward More Accurate Ancestral Protein GenotypePhenotype Reconstructions with the Use of Species Tree-Aware Gene Trees. Molecular Biology and Evolution 32:13–22.

Guéguen, L., S. Gaillard, B. Boussau, M. Gouy, M. Groussin, N. C. Rochette, T. Bigot, D. Fournier, F. Pouyet, V. Cahais, et al. 2013. Bio++: efficient extensible libraries and tools for computational molecular evolution. Molecular biology and evolution 30:1745–1750.

Guindon, S., J.-F. Dufayard, V. Lefort, M. Anisimova, W. Hordijk, and O. Gascuel. 2010. New algorithms and methods to estimate maximum-likelihood phylogenies: assessing the performance of phyml 3.0. Systematic biology 59:307–321.

Joy, J. B., R. H. Liang, R. M. McCloskey, T. Nguyen, and A. F. Poon. 2016. Ancestral reconstruction. PLoS computational biology 12:e1004763.

Kellis, M., B. Wold, M. P. Snyder, B. E. Bernstein, A. Kundaje, G. K. Marinov, L. D. Ward, E. Birney, G. E. Crawford, J. Dekker, et al. 2014. Defining functional dna elements in the human genome. Proceedings of the National Academy of Sciences 111:6131–6138.

Kimura, M. 1980. A simple method for estimating evolutionary rates of base substitutions through comparative studies of nucleotide sequences. Journal of molecular evolution 16:111–120.

Lefkowitz, E. J., D. M. Dempsey, R. C. Hendrickson, R. J. Orton, S. G. Siddell, and D. B. Smith. 2018. Virus taxonomy: The database of the International Committee on Taxonomy of Viruses (ICTV). Nucleic Acids Research 46:D708–D717.

Lefort, V., J.-E. Longueville, and O. Gascuel. 2017. SMS: Smart Model Selection in PhyML. Molecular Biology and Evolution 34:2422–2424.

Liberles, D. A. 2007. Ancestral sequence reconstruction. Oxford University Press on Demand.

Löytynoja, A. 2014. Phylogeny-aware alignment with prank. Pages 155–170 *in* Multiple sequence alignment methods. Springer.

Maiolo, M. 2019. Progressive Multiple Sequence Alignment with Indel Evolution. Ph.D. thesis University of Lausanne Lausanne phD thesis.

Maiolo, M., X. Zhang, M. Gil, and M. Anisimova. 2018. Progressive multiple sequence alignment with indel evolution. BMC bioinformatics 19:331.

Pagel, M. 1999. Inferring the historical patterns of biological evolution. Nature 401:877–884.

Pečerska, J., M. Gil, and M. Anisimova. 2021. Joint alignment and tree inference. bioRxiv .

Pupko, T., I. Pe, R. Shamir, and D. Graur. 2000. A fast algorithm for joint reconstruction of ancestral amino acid sequences. Molecular biology and evolution 17:890–896.

Simmons, M. P. and H. Ochoterena. 2000. Gaps as characters in sequence-based phylogenetic analyses. Systematic biology 49:369–381.

Söding, J. and A. N. Lupas. 2003. More than the sum of their parts: on the evolution of proteins from peptides. Bioessays 25:837–846.

Starr, T. N., S. K. Zepeda, A. C. Walls, A. J. Greaney, S. Alkhovsky, D. Veesler, and J. D. Bloom. 2022. Ace2 binding is an ancestral and evolvable trait of sarbecoviruses. Nature Pages 1–9.

Tao, S., Y. Fan, W. Wang, G. Ma, L. Liang, and Q. Shi. 2007. Patterns of insertion and deletion in mammalian genomes. Current Genomics 8:370–378.

Taylor, M. S., C. P. Ponting, and R. R. Copley. 2004a. Occurrence and consequences of coding sequence insertions and deletions in mammalian genomes. Genome research 14:555–566.

Taylor, M. S., C. P. Ponting, and R. R. Copley. 2004b. Occurrence and consequences of coding sequence insertions and deletions in mammalian genomes. Genome research 14:555–566.

Thorne, J. L., H. Kishino, and J. Felsenstein. 1991. An evolutionary model for maximum likelihood alignment of dna sequences. Journal of Molecular Evolution 33:114–124.

Thorne, J. L., H. Kishino, and J. Felsenstein. 1992. Inching toward reality: an improved likelihood model of sequence evolution. Journal of molecular evolution 34:3–16.

Thornton, J. W. 2004. Resurrecting ancient genes: experimental analysis of extinct molecules. Nature Reviews Genetics 5:366–375.

Whelan, S. and N. Goldman. 2001. A general empirical model of protein evolution derived from multiple protein families using a maximum-likelihood approach. Molecular biology and evolution 18:691–699.

Williams, P. D., D. D. Pollock, B. P. Blackburne, and R. A. Goldstein. 2006. Assessing the accuracy of ancestral protein reconstruction methods. PLoS computational biology 2:e69.

Yamane, K., K. Yano, and T. Kawahara. 2006. Pattern and rate of indel evolution inferred from whole chloroplast intergenic regions in sugarcane, maize and rice. DNA research 13:197–204.

Yang, Z. 1997. Paml: a program package for phylogenetic analysis by maximum likelihood. Bioinformatics 13:555–556.

Yang, Z. 2007. Paml 4: phylogenetic analysis by maximum likelihood. Molecular biology and evolution 24:1586–1591.

Yang, Z. 2014. Molecular evolution: a statistical approach. Oxford University Press.

Yang, Z., S. Kumar, and M. Nei. 1995. A new method of inference of ancestral nucleotide and amino acid sequences. Genetics 141:1641–1650.

Zakas, P. M., H. C. Brown, K. Knight, S. L. Meeks, H. T. Spencer, E. A. Gaucher, and C. B. Doering. 2017. Enhancing the pharmaceutical properties of protein drugs by ancestral sequence reconstruction. Nature biotechnology 35:35.

Zhou, G. and Q. Zhao. 2020. Perspectives on therapeutic neutralizing antibodies against the novel coronavirus sars-cov-2. International journal of biological sciences 16:1718.

Appendix 1

The PIP description

Insertion Probability.— For every node $v \in V$, the probability of inserting a single character on edge $e = (v \to pa(v))$ is proportional to the branch length $b(v)$:

$$\iota(v) = \frac{1}{||\tau|| + \mu^{-1}} \cdot \begin{cases} b(v) & \text{if } v \neq \Omega \\ \mu^{-1} & \text{if } v = \Omega, \end{cases} \tag{0.1}$$

where $\|\tau\|$ is the sum of all branch lengths (total tree length).

Survival probability.— The survival probability $\beta(v)$ for a character inserted along edge e is given by:

$$\beta(v) = \begin{cases} \left(1-\exp(-\mu b(v))\right)/(\mu b(v)) & \text{if } v \neq \Omega \\ 1 & \text{if } v = \Omega, \end{cases} \tag{0.2}$$

Pure survival probability.— To compute the probability that a character inserted before edge e survives along edge e, we define the pure survival probability $\zeta(v) = \exp(-\mu b(v))$ associated with node v. Likewise, the probability that the character will not survive along the edge is $1 - \zeta(v)$. The pure survival probability $\zeta(v)$ differs from the definition of $\beta(v)$ with regard to the insertion location. While in $\zeta(v)$ the character is already present at $pa(v)$, in $\beta(v)$ the character is inserted in a random location along the edge associated with v. In both cases, however, the character is required to survive until node v (Maiolo, 2019).

MSA column probability.— For each individual MSA column m, the probability $p(m)$ is defined by marginalizing over all possible homology paths that may have created that MSA column:

$$p(m) = \sum_{v \subset V} \iota(v) f_v, \tag{0.3}$$

where f_v denotes the probability of all possible homology paths for MSA column m and the subtree rooted at v. Note that f_v can be zero for some nodes, namely the ones where an insertion is impossible given the data in m.

The probability of all possible homology paths.—

$$f_v = \begin{cases} \mathbb{1}[v \in \mathcal{A}]\beta(v)\sum_{\sigma\in\Sigma}\pi_\varepsilon(\sigma)\tilde{f}_v(\sigma) & \text{if } m \neq m_\emptyset \\ 1-\beta(v)+\beta(v)\sum_{\sigma\in\Sigma}\pi_\varepsilon(\sigma)\tilde{f}_v(\sigma) & \text{o.w.,} \end{cases} \tag{0.4}$$

where $m_\emptyset$ is MSA column full of gaps and

$$\tilde{f}_v(\sigma) = \begin{cases} \mathbb{1}[m_v = \sigma] & \text{if } v \in \mathcal{L} \\ \prod_{w\in\text{child}(v)}\left[\sum_{\sigma'\in\Sigma_\varepsilon}\exp(b(w)Q_\varepsilon)_{(\sigma,\sigma')}\tilde{f}_w(\sigma')\right] & \text{o.w.,} \end{cases} \tag{0.5}$$

and $\mathbb{1}[.]$ is the indicator function. The recursive function $\tilde{f}_v(\sigma)$ denotes the partial likelihood of a single character under the substitution-deletion events.

Appendix 2

Detailed description of IndelPoints algorithm

The goal of the IndelPoints algorithm is to find the combination of insertion and deletion points with the highest conditional probability for a given site. These indel points define a homology path which is the best possible explanation of the homologous characters in the alignment. We infer these indel points using a greedy progressive approach, finding the homology path with the highest conditional probability in each node of the tree and progressively extending the path at other nodes. This way, for a given site we find the best partial homology path at each tree node, and can select the best homology path on the whole tree once we reach the root.

The space of possible homology paths is constrained by the definition of PIP, meaning that per site there can only be a single insertion location and multiple deletions. Moreover, the sets of nodes that are possible insertion and deletion locations are non-overlapping. A homology path will have a non-zero conditional probability if and only

if it explains the full site in the alignment, i.e. it contains one of the nodes $v \in \mathcal{A}$ as an insertion location. This means that in any node that is not a possible insertion location the conditional probability can always be set to 0.

Based on the properties of PIP, we can describe the evaluation separately for possible deletion nodes and for all other nodes in the tree. The set $\mathcal{G}$ of potential deletion locations defines the root nodes of subtrees that went extinct at the leaves. For simplicity we will call these nodes extinction nodes. For each of these nodes, we will find the set of deletion nodes with the highest conditional probability and will set the conditional probability of the full homology path to 0. We will call the complementary set of nodes $v \notin \mathcal{G}$ the set of survival nodes. In these nodes, the character either got inserted along the branch $\mathrm{pa}(v) \to v$ for $v \in \mathcal{A}$, or got inserted in an ancestor of v for $v \notin \mathcal{A}$. In either case the character definitely survived until v.

We traverse the tree in post order, evaluating possible homology paths per tree node. The algorithm is described separately for the two node sets (extinction nodes and survival nodes) however all necessary computation can be done in a single tree traversal. It is important to note that here the values are computed at the highest point of the branch $\mathrm{pa}(v) \to v$ rather than at the bottom of the branch leading to v. To simplify notation, we will use v_{L} and v_{R} to denote the left and right child node of v respectively.

Evaluating extinction nodes.— For each extinction node $v \in \mathcal{G}$ we set the insertion node to none ($\mathcal{I}_v = \emptyset$) and consequently the probability of the best global homology path to 0 ($p_v = 0$).

We need to learn whether the deletion was more likely along the branch leading to v or more likely in the child subtrees. The character could have gone extinct before reaching node v, which is the probability of deletion happening along the branch $\mathrm{pa}(v) \to v$. Moreover, if v is a leaf ($v \in \mathcal{L}$), we can certainly state that it is the current best deletion node ($\mathcal{D}_v = v$) and that f_v at that node represents the certain deletion ($f_v = 1 - \zeta(v)$, extinction probability). On the other hand, if the node v is not a leaf node ($v \notin \mathcal{L}$), we

compare the deletion probability at $\mathrm{pa}(v) \to v$, $1-\zeta(v)$, with the product of survival along $\mathrm{pa}(v) \to v$ and subsequent deletion at or below nodes v_{R} and v_{L}, $f_{v_{\mathrm{R}}} f_{v_{\mathrm{L}}} \zeta(v)$. Among the two, we select the value that represents the best scenario and set f_v and $\mathcal{D}_v$ appropriately ($f_v = 1-\zeta(v)$ and $\mathcal{D}_v = \{v\}$ or $f_v = f_{v_{\mathrm{R}}} f_{v_{\mathrm{L}}} \zeta(v)$ and $\mathcal{D}_v = \{\mathcal{D}_{v_{\mathrm{R}}} \cup \mathcal{D}_{v_{\mathrm{L}}}\}$ respectively).

The pseudocode for extinction node evaluation is shown in Algorithm 1, and the corresponding flowchart is available in the supplemental materials (see Fig. 10).

Algorithm 1: Evaluating extinction nodes.

Input: Phylogenetic tree τ
Output: Homology path with the highest probability $\mathcal{H}(m)$

```
for ∀v ∈ 𝒢 do
    p_v = 0;
    if v ∈ ℒ then
        f_v = 1 − ζ(v);
        ℋ_v = {ℐ_v = ∅, 𝒟_v = {v}};
    else
        if ζ(v) f_{v_R} f_{v_R} < 1 − ζ(v) then
            f_v = 1 − ζ(v);
            ℋ_v = {ℐ_v = ∅, 𝒟_v = {v}};
        else
            f_v = ζ(v) f_{v_R} f_{v_L};
            ℋ_v = {ℐ_v = ∅, 𝒟_v = 𝒟_{v_R} ∪ 𝒟_{v_L}};
        end
    end
end
```

Evaluating survival nodes.— All nodes $v \notin \mathcal{G}$ are definite survival nodes – meaning that the character survived along the branch $\mathrm{pa}(v) \to v$. We distinguish two types of survival nodes, $v \in \mathcal{A}$ (potential insertion nodes) and $v \notin \mathcal{A}$, nodes where an insertion could not have happened. $v \notin \mathcal{A}$ is the simpler one, as the homology path at such nodes will not have an associated insertion node, while a full homology path needs to have an insertion. This means that for all such nodes the probability of a homology path will be 0 ($p_v = 0$) and the set of insertion nodes will be empty ($\mathcal{I}_v = \emptyset$). In case $v \in \mathcal{A}$, we compute the non-zero homology path probability according to the formulas defined for PIP, $p_v = \iota(v)\beta(v)$ for a leaf node and $p_v = \iota(v)\beta(v) f_{v_{\mathrm{R}}} f_{v_{\mathrm{L}}}$ for an internal node. We also set

$\mathcal{I}_v = v$, representing a possible insertion at this node. While the set $\mathcal{A}$ may contain multiple nodes, an instance of a homology path can only have a single insertion location. This implies that when we compute the probability of a homology path for a node $v \in \mathcal{A}$, we essentially assume that it is the only possible insertion location while all other nodes are treated as regular nodes in the tree.

The conditional probability of survival is independent of whether v is a potential insertion node or not. If the node is a leaf ($v \in \mathcal{L}$), the character will have certainly survived, thus $f_v = \zeta(v)$, and the deletion node set is empty $\mathcal{D}_v = \emptyset$. On the other hand, if v is an internal node, we need to account for the survival along branch $\mathrm{pa}(v) \to v$ and propagate any possible deletion nodes that were already selected, thus $f_v = f_{v_\mathrm{R}} f_{v_\mathrm{L}} \zeta(v)$ and $\mathcal{D}_v = \mathcal{D}_{v_\mathrm{R}} \cup \mathcal{D}_{v_\mathrm{L}}$.

Reaching the root of the tree Ω, we will have processed all the nodes in the tree and computed the probabilities of the homology paths conditioned on the insertion point at each node. These probabilities will only be non-zero for nodes $v \in \mathcal{A}$, which allows us to choose the best homology path simply as $\mathcal{H}_{\mathrm{argmax}(p_v)}$ – the homology path corresponding to the node with the highest probability. We then extract the subtree τ_m rooted at $\mathcal{I}_{\arg\max(p_v)}$ and pruned at $\mathcal{D}_{\arg\max(p_v)}$ from the tree τ.

The pseudocode for survival node evaluation is presented in Algorithm 2 and the corresponding flowchart is available in the supplemental materials (see Fig. 11).

Algorithm 2: Evaluating survival nodes

Input: Phylogenetic tree τ
Output: Homology path with the highest conditional probability $\mathcal{H}(m)$

for $\forall v \notin \mathcal{G}$ **do**
  if $v \in \mathcal{L}$ **then**
    $p_v = \mathbb{1}[v \in \mathcal{A}]\iota(v)\beta(v)$;
    $f_v = \zeta(v)$;
    $\mathcal{H}_v = \{\mathcal{I}_v = \mathbb{1}[v \in \mathcal{A}]v, \mathcal{D}_v = \emptyset\}$;
  else
    $p_v = \mathbb{1}[v \in \mathcal{A}]\iota(v)\beta(v)f_{v_\mathrm{R}}f_{v_\mathrm{L}}$;
    $f_v = \zeta(v)f_{v_\mathrm{R}}f_{v_\mathrm{L}}$;
    $\mathcal{H}_v = \{\mathcal{I}_v = \mathbb{1}[v \in \mathcal{A}]v, \mathcal{D}_v = \mathcal{D}_{v_\mathrm{R}} \cup \mathcal{D}_{v_\mathrm{L}}\}$;
  end
end
return $\mathcal{H}_{\arg\max(p_v)}$;

Appendix 3

Detailed DP joint ASR under PIP

Similar to how it is done in FastML (Pupko et al., 2000), we reconstruct the ancestral characters in two consecutive steps. First, we compute the likelihoods using dynamic programming (DP), and then use the precomputed values for joint ASR. However, instead of reconstructing the ancestral states on the whole tree τ, we work on the subtree τ_m extracted using the most likely indel history for the site m under PIP model.

DP likelihood computation.— In the forward phase, as in Pupko's ASR method, we traverse the tree τ_m in post-order. While visiting each internal node v, we compute the likelihood $\mathrm{Lk}_v(i)$ and the corresponding best ancestral character state $\mathrm{CS}_v(i)$ for each character i in the alphabet ($i \in \Sigma$). We assume that the best ASR in a subtree rooted at the node v is independent of the rest of the tree. $\mathrm{Lk}_v(i)$ is the likelihood of the best reconstruction on this subtree conditioned on the parent of node v has the character i at that site. And $\mathrm{CS}_v(i)$ is the character state assigned to node v in this optimal conditional reconstruction.

To compute the likelihood values and the corresponding candidate character states, we have to determine whether a node is a leaf or not. Let j be the observed character at v. If $v \in \mathcal{L}$, we assign $\mathrm{CS}_v(i) = j$ and the likelihood value is $\mathrm{Lk}_v(i) = \mathrm{P}_{ij}(v)$, where $\mathrm{P}(v) = \exp(\mathrm{b}(v) \cdot \mathrm{Q}_\varepsilon)$ is the transition probability matrix along the branch $\mathrm{pa}(v) \to v$. If a node v is not a leaf ($v \notin \mathcal{L}$), we compute the likelihood value and the candidate characters for all the non-root internal nodes based on the values already computed in its children. For each $i \in \Sigma$ the likelihood value is $\mathrm{Lk}_v(i) = \max_j(\mathrm{P}_{ij}(v) \cdot \mathrm{Lk}_{v_\mathrm{L}}(j) \cdot \mathrm{Lk}_{v_\mathrm{R}}(j))$ and the candidate character is the value of j producing the previous maximization $\mathrm{CS}_v(i) = \arg\max_j(\mathrm{P}_{ij}(v) \cdot \mathrm{Lk}_{v_\mathrm{L}}(j) \cdot \mathrm{Lk}_{v_\mathrm{R}}(j))$. We compute there values until all the nodes have been visited except for the root. While visiting the root, the likelihood is computed slightly differently as the transition probabilities are replaced with the alphabet equilibrium frequencies $\pi_{\varepsilon i}$, $\mathrm{Lk}_v(i) = \pi_{\varepsilon i} \cdot \mathrm{Lk}_{v_\mathrm{L}}(i) \cdot \mathrm{Lk}_{v_\mathrm{R}}(i)$. Since the root has no possible ancestors, we define the single candidate character state as $\mathrm{CS}_\Omega = \arg\max_i(\mathrm{Lk}_\Omega)$. The pseudocode of the algorithm is shown in Algorithm 3.

Algorithm 3: Computing the likelihood

Input: Phylogenetic subtree τ_m
MSA column m
Extended substitution model Q_ε
Extended equilibrium frequencies π_ϵ
Output: Likelihood array of node v Lk_v
Candidate character array of node v with ML values CS_v

Ω_{τ_m} = root of subtree τ_m;
$\mathrm{P}(v) = \exp(\mathrm{b}(v) \cdot \mathrm{Q}_\varepsilon)$;
if $v \in \mathcal{L}$ **then**
  j = observed character at v;
  for $\forall i \in \Sigma$ **do**
    $\mathrm{CS}_v(i) = j$;
    $\mathrm{Lk}_v(i) = \mathrm{P}_{ij}(v)$;
  end
else
  v_L = left child node;
  v_R = right child node;
  for $\forall i \in \Sigma$ **do**
    $\mathrm{Lk}_v(i) = \pi_{\varepsilon i} \cdot \mathrm{Lk}_{v_\mathrm{L}}(i) \cdot \mathrm{Lk}_{v_\mathrm{R}}(i)$;
  end
  $\mathrm{CS}_\Omega = \arg\max_i(\mathrm{Lk}_v)$;
  for $i \in \Sigma$ **do**
    for $j \in \Sigma$ **do**
      $S(j)^\dagger = \mathrm{P}_{ij}(v) \cdot \mathrm{Lk}_{v_\mathrm{L}}(j) \cdot \mathrm{Lk}_{v_\mathrm{R}}(j)$;
    end
    $\mathrm{Lk}_v(i) = \max(S)$;
    $\mathrm{CS}_v(i) = \arg\max_j(S)$;
  end
end

S stands for Single Character State selected in this step of the algorithm.

Joint ASR.— After computing the likelihoods of all potential ancestral characters, we traverse the tree in pre-order from the root of the tree τ_m to select the single most likely ancestral character per node A_v. We assign the most likely character state at the root by picking the character i that maximizes the likelihood value $\mathrm{CS}_{\Omega_{\tau_m}} = \arg\max_i(\mathrm{Lk}_v)$. For all other internal nodes in the traversal we assign $\mathrm{CS}_v(\mathrm{A}_{\mathrm{pa}(v)})$ as the best reconstruction (for example if the state reconstructed for the parent of v is

proline ($i = \mathrm{P}$), the ancestral state of v will be $\mathrm{CS}_v(i = \mathrm{P})$). We continue the process until the most likely character is assigned at all the internal nodes. The formula is presented below where Ω_{τ_m} is the root of subtree τ_m:

$$\mathrm{A}_v = \begin{cases} \mathrm{CS}_v(\arg\max(\mathrm{Lk}_v)) & v = \Omega_{\tau_m} \\ \mathrm{CS}_v(\mathrm{A}_{\mathrm{pa}(v)}) & \text{o.w.} \end{cases} \tag{0.6}$$

Metric	Accuracy (%)
Correctly inferred insertion points	96.08 ± 2.84
Correctly inferred deletion points	95.54 ± 2.80
Correctly inferred MSA columns	60.08 ± 9.60
Correctly inferred characters including gaps	88.14 ± 3.91
Correctly inferred gap character	99.86 ± 0.26

Table 1: ARPIP accuracy for inference on PIP simulated data.

Metric	Accuracy (%)	
	indel rate 0.01	**indel rate 0.05**
Correctly inferred MSA columns	46.58 ± 13.22	59.49 ± 10.63
Correctly inferred characters including gaps	83.49 ± 6.04	87.93 ± 4.33
Correctly inferred gap character	99.98 ± 0.16	99.95 ± 0.14

Table 2: ARPIP accuracy for inference on INDELible simulated data.

Subgenus	Species	Uniprot acession number
Embecovirus	*Betacoronavirus 1*	A0A191URB2
	China Rattus coronavirus HKU24	A0A0A7UZR7
	Human coronavirus HKU1	U3NAI2
	Murine coronavirus	P11224
	Myodes coronavirus 2JL14	A0A2H4MXV6
Hibecovirus	*Bat Hp-betacoronavirus Zhejiang2013*	A0A088DJY6
Merbecovirus	*Hedgehog coronavirus 1*	A0A4D6G1A4
	Middle East respiratory syndrome-related coronavirus	K9N5Q8
	Pipistrellus bat coronavirus HKU5	A3EXD0
	Tylonycteris bat coronavirus HKU4	A3EX94
Nobecovirus	*Rousettus bat coronavirus GCCDC1*	A0A1B3Q5W5
	Rousettus bat coronavirus HKU9	A3EXG6
Sarbecovirus	*Severe acute respiratory syndrome-related coronavirus*	A0A3Q8AKM0
	Severe acute respiratory syndrome-related coronavirus 2	P0DTC2

Table 3: *Betacoronavirus* sequences used in the analysis.

Figure captions

Figure 1: The phylogenetic tree τ rooted at Ω. $b(v_1)$ represents the branch length from Ω to v_1. The leaves of the tree show a single column of the MSA including gaps as an additional character state. The set $\mathcal{S}$ is defined as all leaves with a character in the given column (not a gap). The set of potential insertion nodes $\mathcal{A}$ contains the nodes ancestral to all nodes in $\mathcal{S}$. Finally, the set of potential deletion nodes $\mathcal{G}$ is defined as all nodes which are either a leaf with a gap in the given column, or a node whose both children are in $\mathcal{G}$.

Figure 2: Overview of the IndelPoints algorithm. The tree is traversed in post-order to infer the most likely homology path progressively using the predefined sets: $\mathcal{L}$ the set of all leaves, $\mathcal{A}$ the set of potential insertion points, and $\mathcal{G}$ the set of potential deletion points. Here, σ represents the character in focus and v is the node visited during the tree traversal.

Figure 3: An example tree from the dataset generated by the PIP simulator.

Figure 4: Illustration of the rooted *Betacoronavirus* phylogenetic tree which was reconstructed by PhyML 3.0 from the ProPIP alignment. Note that the original tree was unrooted which ARPIP used mid-point rooting method to make the tree rooted.

Figure 5: Illustration of a snippet from the *CoV* dataset containing the MSA inferred by ProPIP and the ancestral sequences predicted by ARPIP and FastML. a) The region in which ARPIP infers a very different ancestral history, probably due to inferring the insertion point prior to ancestral character inference. FastML inferred no gap in this column perhaps due to the adjacent (first) column. b) The region in which both algorithms had similar inferences of the ancestral states. Note that FastML algorithm works on an unrooted tree which, compared to ARPIP, resulted in one fewer internal

sequence reconstructed (due to the absence of the root node).

Figure 6: A snippet from the PIP simulated dataset containing the true simulated MSA and ancestors and the ancestral sequences predicted by ARPIP and FastML. a) A region where both ARPIP and FastML accurately inferred the ancestral states. b) A region where both algorithms estimated the ancestral character incorrectly. c) A region where FastML inferred ancestral characters even though there were none in the simulation. d) A region where there was a single ancestral character but FastML inferred its position incorrectly. Note that FastML algorithm works on an unrooted tree which, compared to ARPIP, resulted in one fewer internal sequence reconstructed (due to the absence of the root node).

Figure 7: A gapless snippet from the PIP simulated dataset containing the true simulated MSA and ancestors and the ancestral sequences predicted by ARPIP, PAML and FastML. a) A region where all algorithms accurately inferred the ancestral state. b) A region where all algorithms made mistakes. c) A region where FastML and PAML made incorrect inferences but ARPIP inferred the ancestral state correctly. d) A region where all algorithms accurately inferred the ancestral states except ARPIP. Note that FastML algorithm works on an unrooted tree which, compared to ARPIP, resulted in one fewer internal sequence reconstructed (due to the absence of the root node).

Figure 8: A snippet from the INDELible simulated dataset with indel rate 0.01 **containing the true simulated MSA, ancestors and the ancestral sequences predicted by ARPIP and FastML.** a) A region where both ARPIP and FastML accurately inferred the ancestral states. b) A region that the FastML inferred the gaps incomplete while ARPIP missed the the character state. c) A region where both algorithms estimated most of the ancestral character incorrectly. d) A region where both

methods inferred the ancestral states including gaps positions correctly. Note that FastML algorithm works on an unrooted tree which, compared to ARPIP, resulted in one fewer internal sequence reconstructed (due to the absence of the root node).

Figure 9: A snippet from the INDELible simulated dataset with indel rate 0.05 **containing the true simulated MSA and ancestors and the ancestral sequences predicted by ARPIP and FastML.** a) A region where both ARPIP and FastML accurately inferred the ancestral states. b) A region where both algorithms estimated the indel events correctly but the ancestral character incorrectly. c) A region where FastML missed the gap character but ARPIP inferred it correctly. d) A region where FastML inferred ancestral characters even though there were none in the simulation. Note that FastML algorithm works on an unrooted tree which, compared to ARPIP, resulted in one fewer internal sequence reconstructed (due to the absence of the root node).

Figure 10: Evaluating extinction nodes algorithm, where $\mathcal{H}_v$ defines the homology path of node v, $\mathrm{f_v}$ is the probability of that homology path and p_v is the probability of the best homology path. $\mathcal{I}_v$ is the set of possible insertion points while $\mathcal{D}_v$ is the set of all possible deletion points for node v. $\zeta(v)$ denotes the pure survival probability of node v. Moreover, $\mathcal{G}$ and $\mathcal{L}$ represent the set of potential deletion points and leaves, respectively.

Figure 11: Evaluating survival nodes algorithm. $\mathcal{H}_v$ defines the homology path of node v represents by the set of possible insertion points $\mathcal{I}_v$ and the set of all possible deletion points $\mathcal{D}_v$. Moreover, $\mathrm{f_v}$ is the probability of that homology path while p_v is the probability of the best homology path. $\iota(v)$, $\beta(v)$ and $\zeta(v)$ are respectively insertion, survival and pure survival probabilities at node v. $\mathcal{A}$ represents the set of potential insertion points and $\mathcal{L}$ shows the set of leaves.

Figure 12: Illustration of the rooted *Betacoronavirus* phylogenetic tree reconstructed by PhyML 3.0 from the PRANK alignment. Note that the original tree was unrooted which ARPIP used mid-point rooting method to make the tree rooted.

Figure 13: A snippet from the *CoV* dataset containing the MSA inferred by PRANK and the ancestor sequences predicted by ARPIP and FastML. The tree is obtained by PhyML 3.0, shown in Figure 12. Note that FastML algorithm works on an unrooted tree which, compared to ARPIP, resulted in one fewer internal sequence reconstructed (due to the absence of the root node).

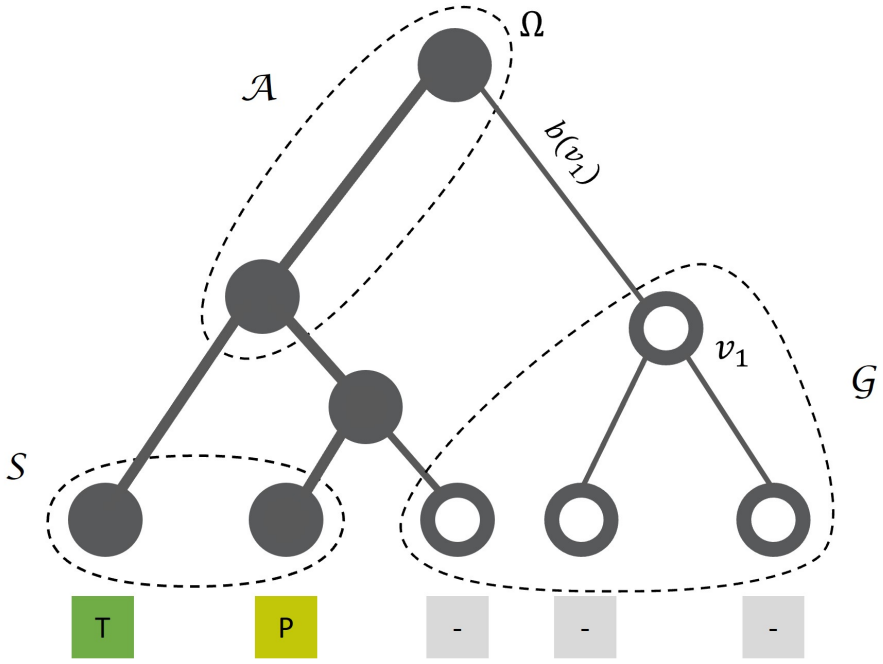

Figure 1: The phylogenetic tree τ rooted at Ω.

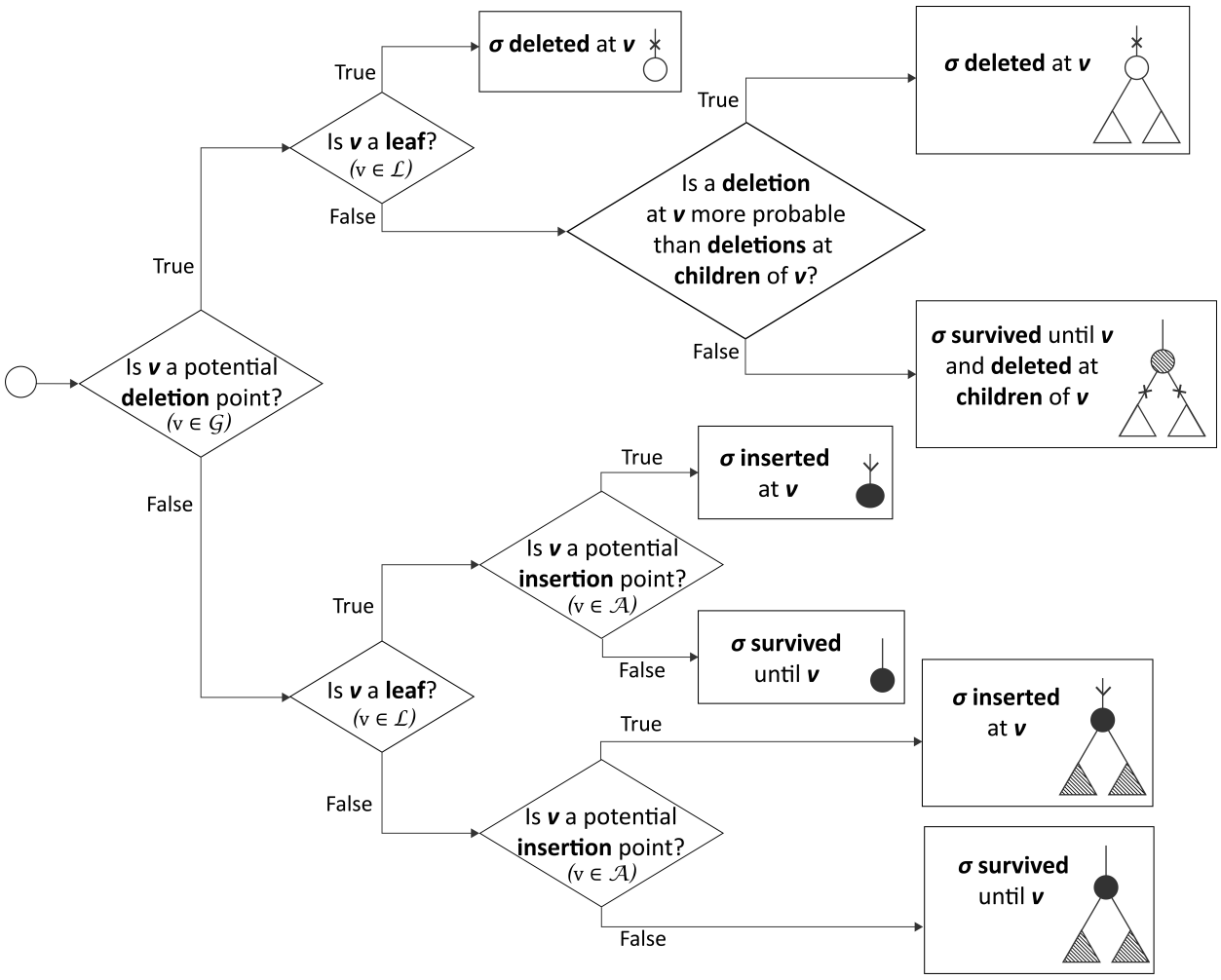

Figure 2: Overview of the IndelPoints algorithm.

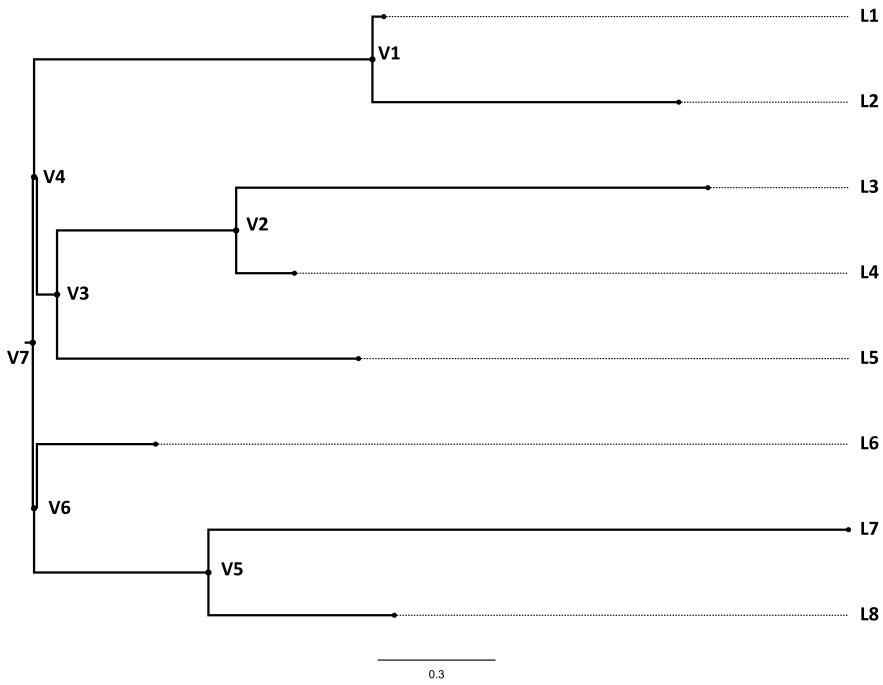

Figure 3: An example tree from the dataset generated by the PIP simulator.

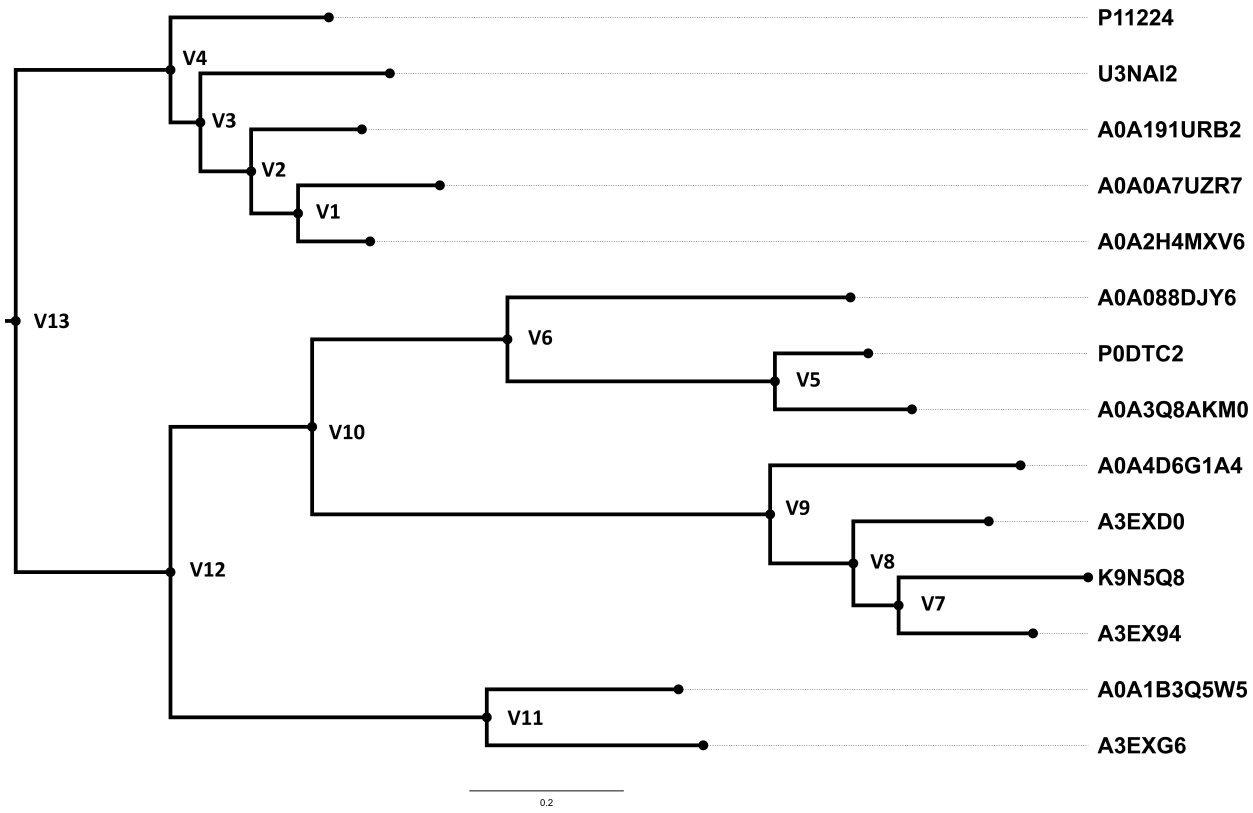

Figure 4: Illustration of the rooted *Betacoronavirus* phylogenetic tree which was reconstructed by PhyML 3.0 from the ProPIP alignment.

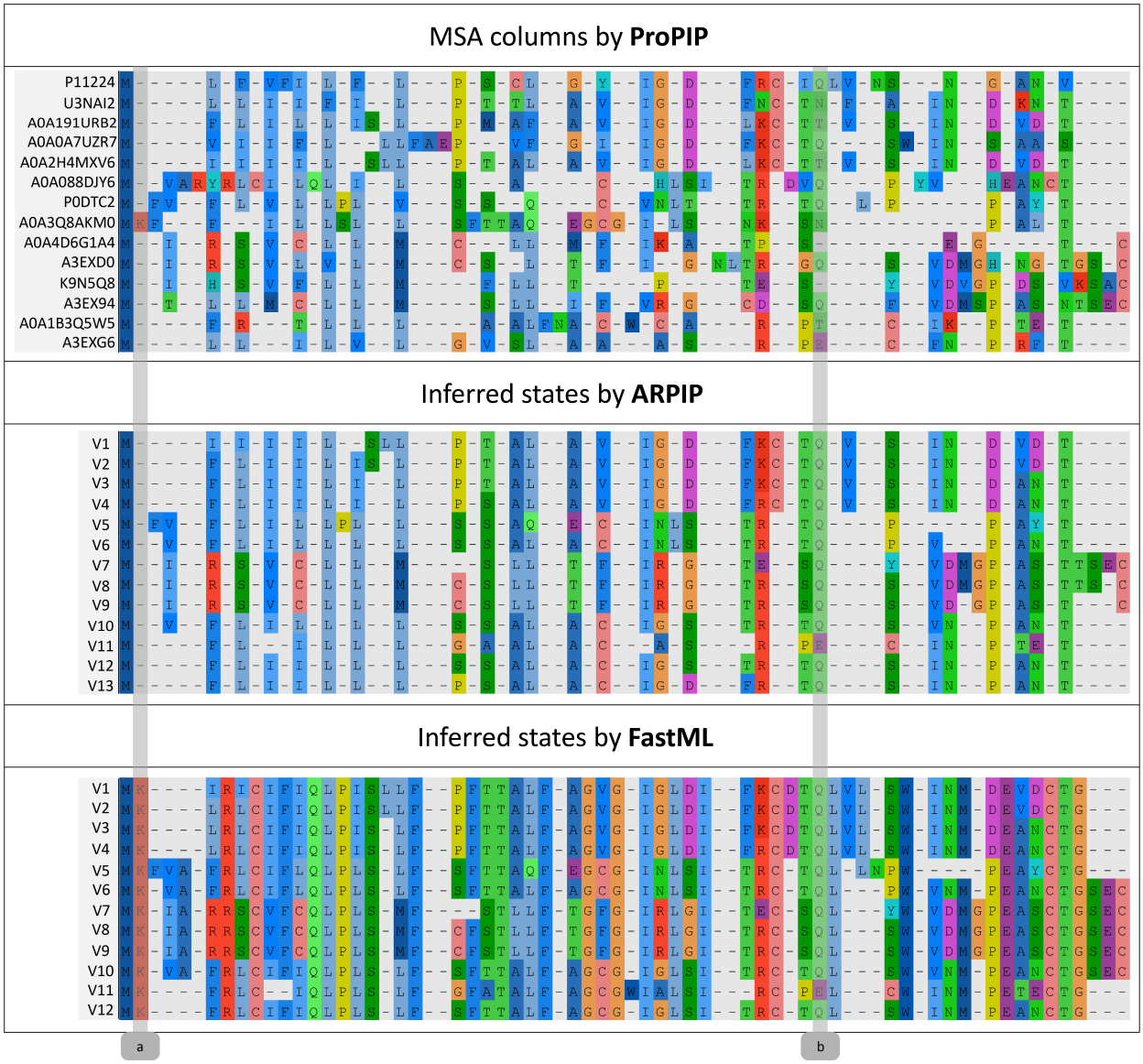

Figure 5: Illustration of a snippet from the *CoV* dataset containing the MSA inferred by ProPIP and the ancestral sequences predicted by ARPIP and FastML.

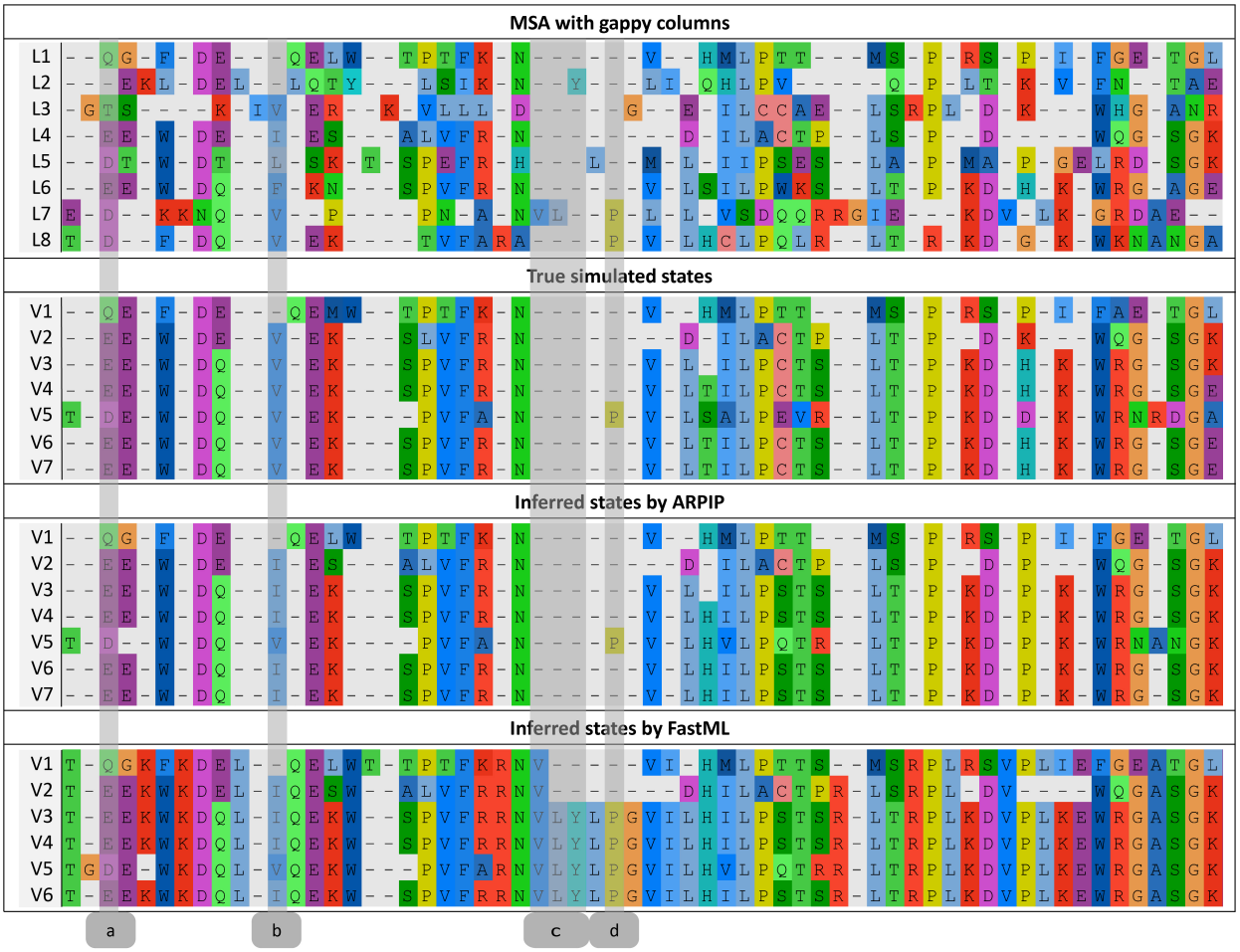

Figure 6: A snippet from the PIP simulated dataset containing the true simulated MSA and ancestors and the ancestral sequences predicted by ARPIP and FastML.

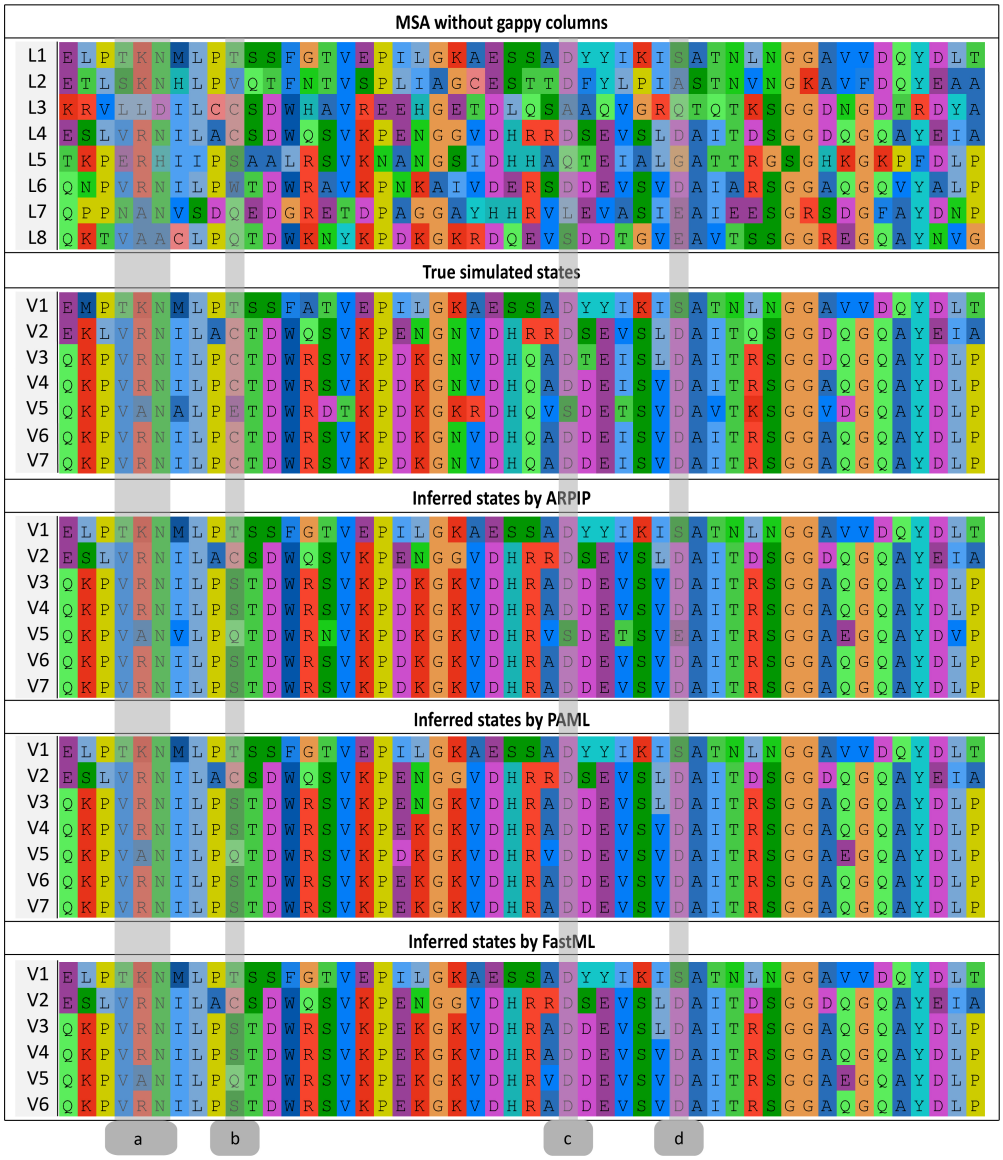

Figure 7: A gapless snippet from the PIP simulated dataset containing the true simulated MSA and ancestors and the ancestral sequences predicted by ARPIP, PAML and FastML.

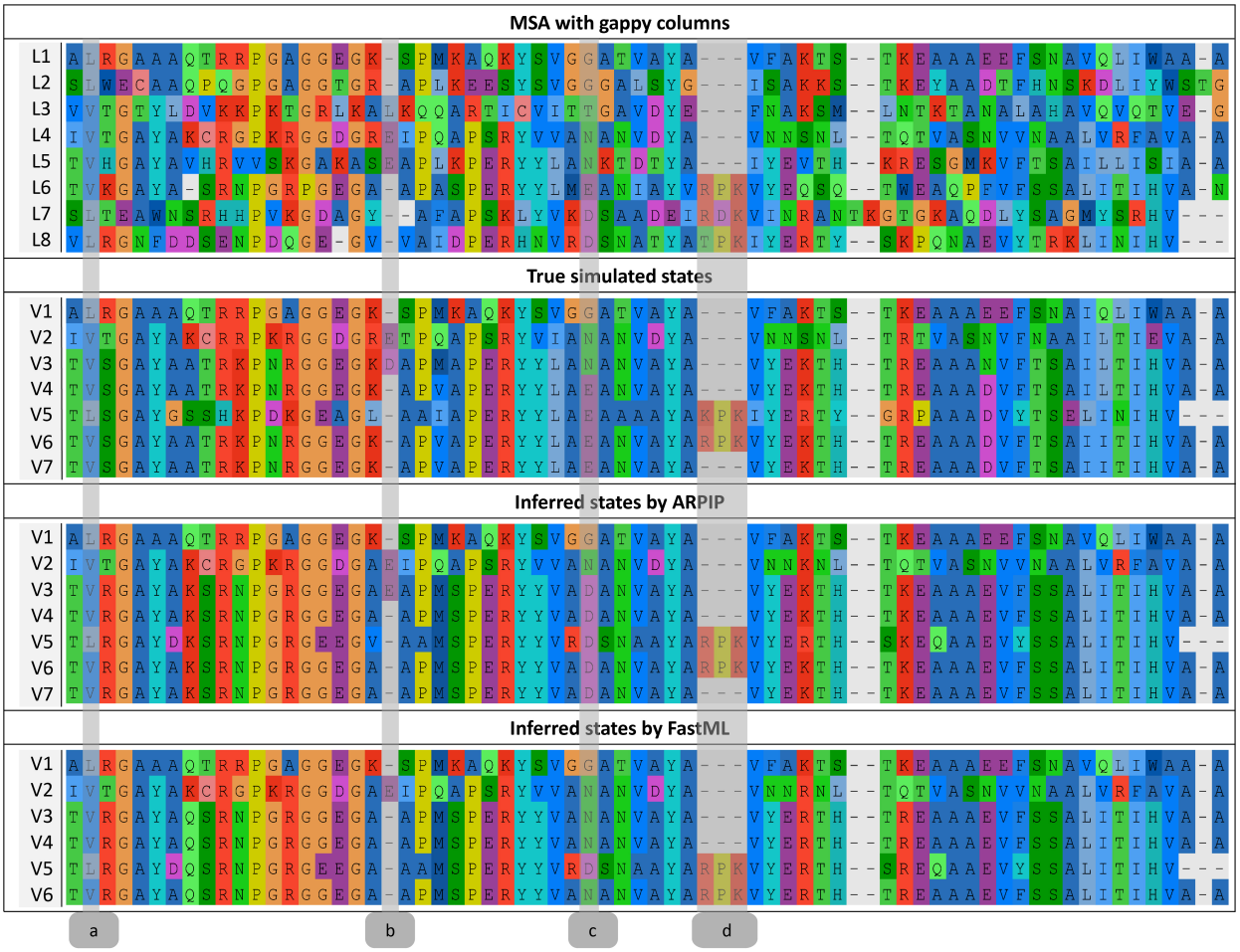

Figure 8: A snippet from the INDELible simulated dataset with indel rate 0.01 containing the true simulated MSA, ancestors and the ancestral sequences predicted by ARPIP and FastML.

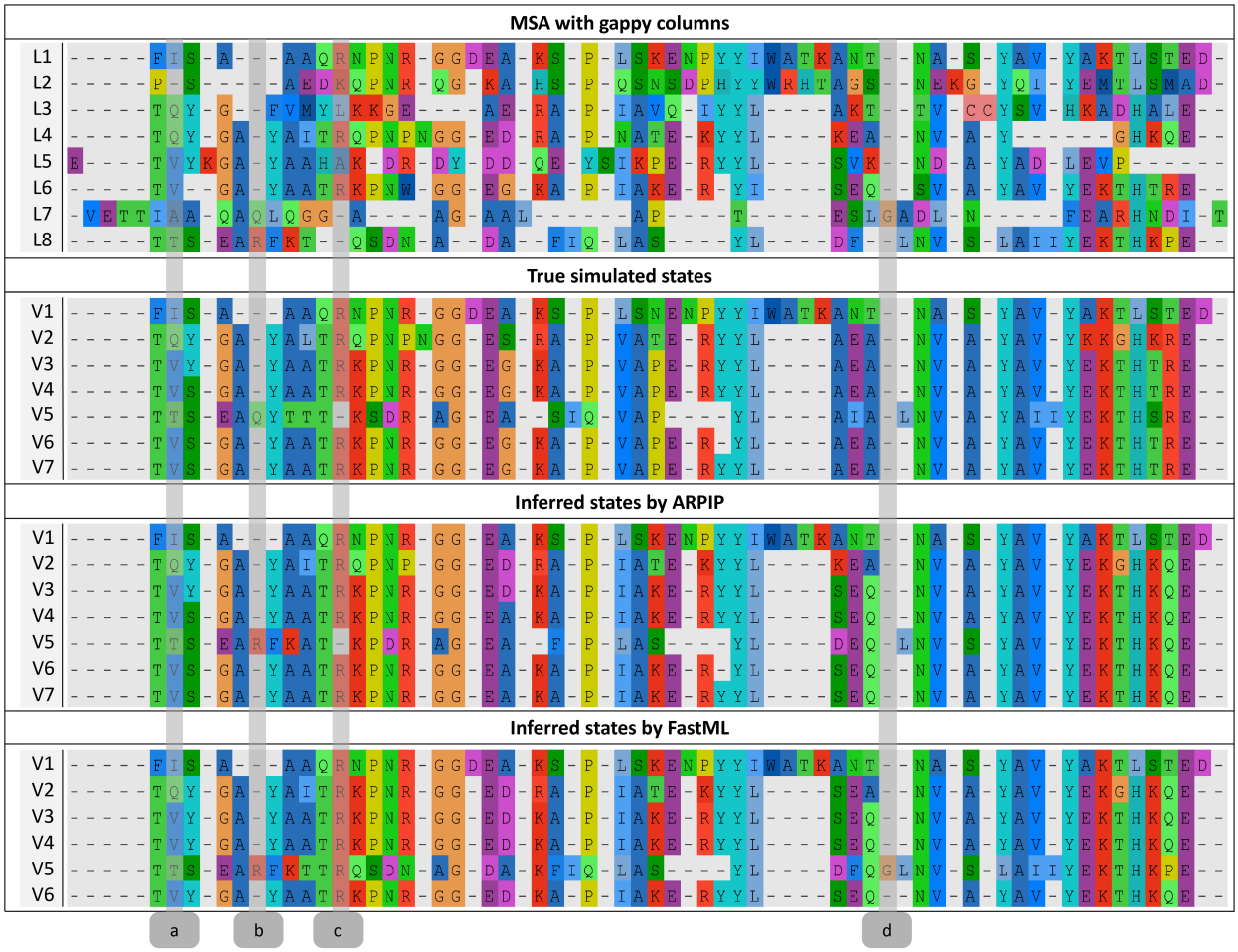

Figure 9: A snippet from the INDELible simulated dataset with indel rate 0.05 containing the true simulated MSA and ancestors and the ancestral sequences predicted by ARPIP and FastML.

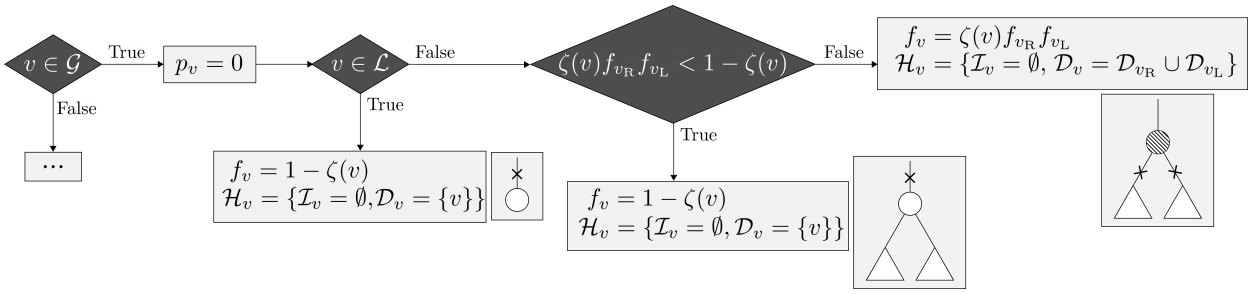

Figure 10: Evaluating extinction nodes algorithm.

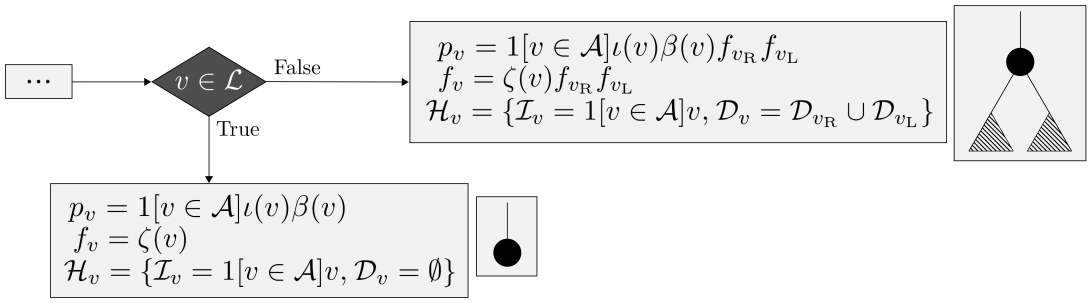

Figure 11: Evaluating survival nodes algorithm.

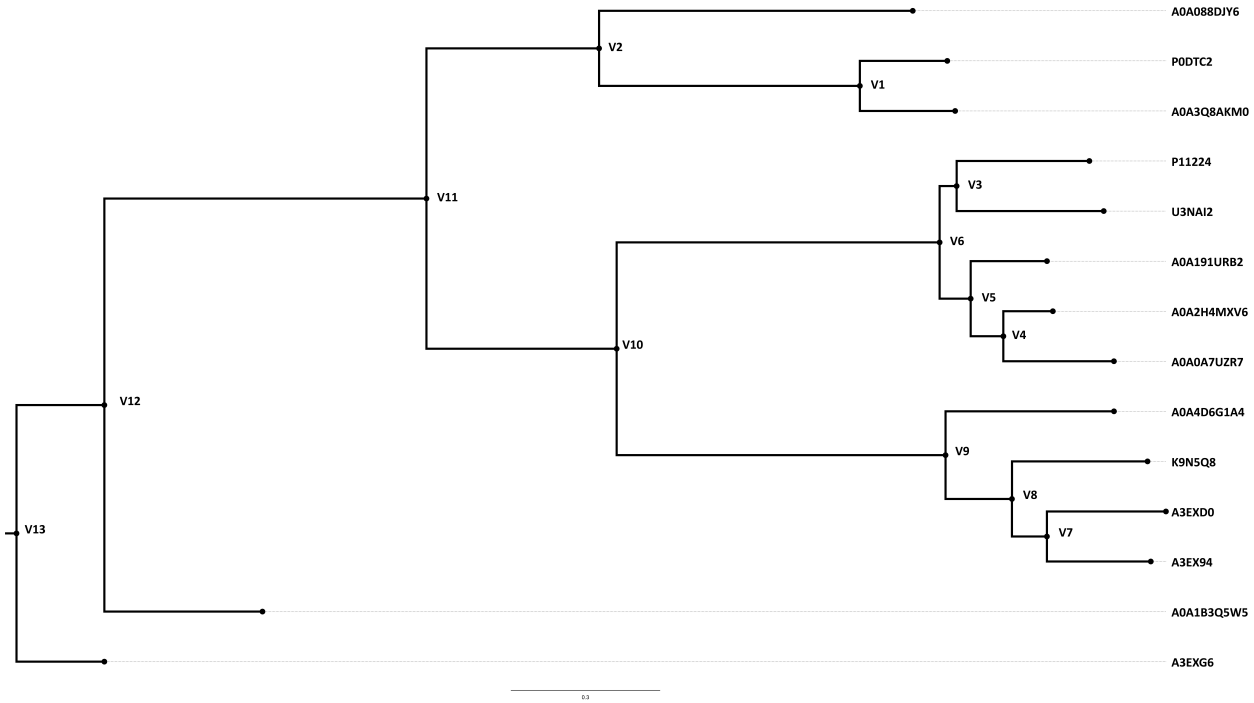

Figure 12: Illustration of the rooted *Betacoronavirus* phylogenetic tree reconstructed by PhyML 3.0 from the PRANK alignment.

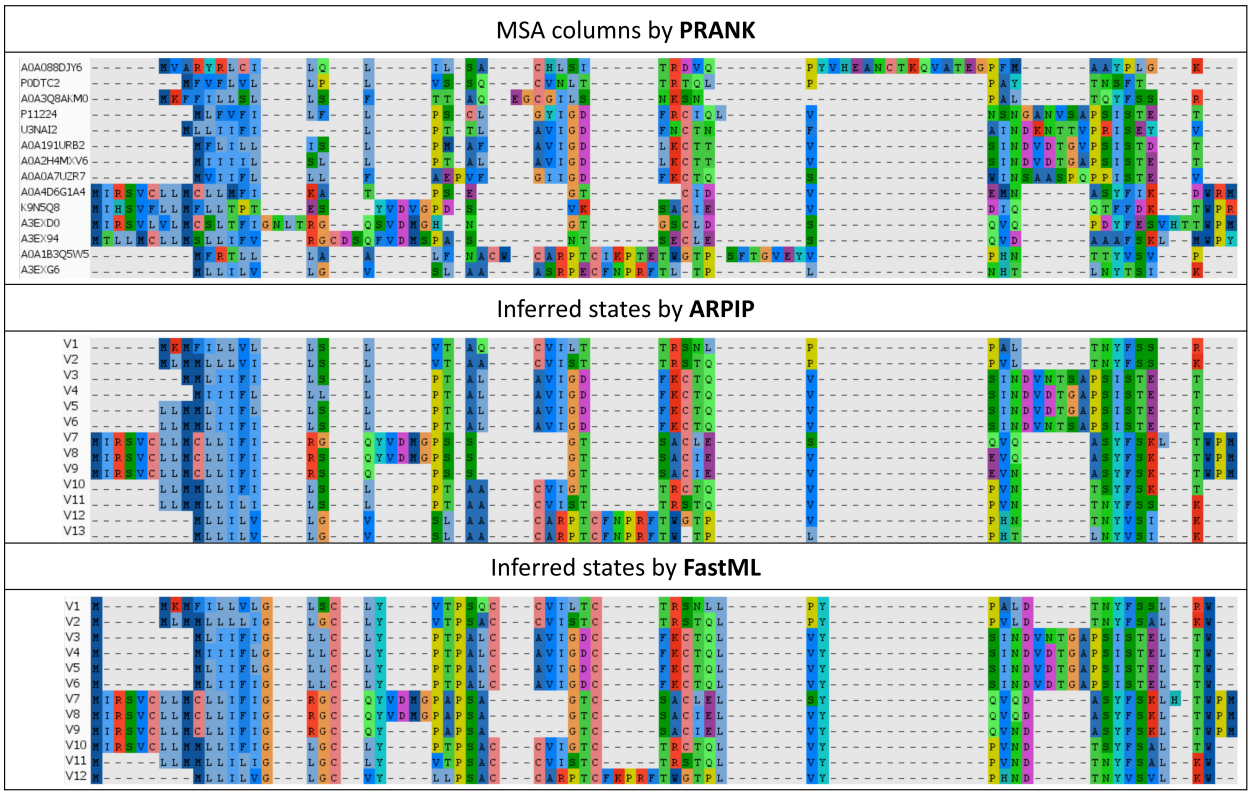

Figure 13: A snippet from the *CoV* dataset containing the MSA inferred by PRANK and the ancestor sequences predicted by ARPIP and FastML.

Systematic Biology (2022), **0**, 0, *pp.* 1–**??**
doi:10.1093/sysbio/output

ARPIP: Ancestral sequence Reconstruction with insertions and deletions under the Poisson Indel Process

Gholamhossein Jowkar[1,2,3,*], Jūlija Pečerska [1,2], Massimo Maiolo [1,2,4], Manuel Gil [1,2], Maria Anisimova [1,2]

[1] *Zurich University of Applied Sciences, School of Life Sciences and Facility Management, CH-8820, Wädenswil, Switzerland*
[2] *Swiss Institute of Bioinformatics, CH-1015 Lausanne, Switzerland*
[3] *University of Neuchâtel, Institute of biology, CH-2000 Neuchâtel, Switzerland*
[4] *University of Bern, Institute of Pathology, CH-3008 Bern, Switzerland*

**Gholamhossein Jowkar, ZHAW, School of Life Sciences and Facility Management, Applied Computational Genomics Group, Schloss 1, 8820 Wädenswil, Switzerland, E-mail: jowk@zhaw.ch*

Abstract

Modern phylogenetic methods allow inference of ancestral molecular sequences given an alignment and phylogeny relating present day sequences. This provides insight into the evolutionary history of molecules, helping to understand gene function and to study biological processes such as adaptation and convergent evolution across a variety of applications. Here we propose a dynamic programming algorithm for fast joint likelihood-based reconstruction of ancestral sequences under the Poisson Indel Process (PIP). Unlike previous approaches, our method, named ARPIP, enables the reconstruction with insertions and deletions based on an explicit indel model. Consequently, inferred indel events have an explicit biological interpretation. Likelihood computation is achieved in linear time with respect to the number of sequences. Our method consists of two steps, namely finding the most probable indel points and reconstructing ancestral sequences. First, we find the most likely indel points and prune the phylogeny to reflect the insertion and deletion events per site. Second, we infer the ancestral states on the pruned subtree in

a manner similar to FastML. We applied ARPIP on simulated datasets and on real data from the *Betacoronavirus* genus. ARPIP reconstructs both the indel events and substitutions with a high degree of accuracy. Our method fares well when compared to established state-of-the-art methods such as FastML and PAML. Moreover, the method can be extended to explore both optimal and suboptimal reconstructions, include rate heterogeneity through time and more. We believe it will expand the range of novel applications of ancestral sequence reconstruction.

Key words: joint ancestral sequence reconstruction, ancestral sequences, maximum likelihood, indel, phylogeny, dynamic programming, Poisson indel process, SARS-CoV, evolutionary stochastic process.

Phylogenetics is a wide research field with a variety of applications ranging from reconstructing the tree of life to investigating ongoing epidemics. Phylogenetic trees provide insight into unobservable evolutionary events in the past such as adaptation or mass extinction events. Phylogenetic inference can be divided into several interrelated tasks including sequence alignment, phylogeny estimation, detection of selection, and ancestral sequence reconstruction (ASR). ASR aims to infer the likely ancestral sequences for a set of existing homologous sequences.

ASR allows researchers to pursue a wide range of topics from determining the origins of life or epidemics to developing personalised medicine (Pagel, 1999; Liberles, 2007). For example, the functionality of ancient genes can be investigated by reconstructing and synthesising the genetic material inferred by ASR (Thornton, 2004). Such analyses can help us understand the mechanisms underlying adaptation and speciation processes, inspiring new approaches for protein engineering (Chang et al., 2005) and drug design (Zakas et al., 2017). ASR can be used to study epidemiological origins of pathogens, particularly in light of recent coronavirus pandemics (Pagel, 1999; Brintnell

et al., 2021; Starr et al., 2022).

State-of-the-art likelihood-based ASR methods use Markov processes to model character substitutions through time. Such models account for various biases in character substitution, as well as divergence represented by evolutionary time (Yang et al., 1995; Yang, 2007; Pupko et al., 2000). However, Markov models of molecular evolution do not include insertions or deletions (indels) as part of the evolutionary process, meaning that methods relying on these models have to treat gap characters separately. Most of the ASR methods adopt one of two pre-processing approaches. They either treat gaps as missing/ambiguous data or remove gap characters entirely. However, indels represented by gaps carry an important evolutionary signal (Dessimoz and Gil, 2010), and are in fact a major driving force of genomic divergence (Tao et al., 2007). Therefore, methods that model indels explicitly have a clear advantage over methods that do not. Up to this point, most existing frequentist algorithms do not include indel modelling except for two methods, Ancestors (Diallo et al., 2009) and FastML (Ashkenazy et al., 2012). Ancestors has exponential computational complexity and therefore has not been widely adopted by users. FastML handles indels using a heuristic approach called indel-coding. The method relies on the linear time complexity algorithm (Pupko et al., 2000) for joint maximum likelihood (ML) ASR using dynamic programming (DP). FastML makes the analyses of large datasets tractable. Currently, it is provided as a web service (Ashkenazy et al., 2012). While the results of indel-coding can be interpreted from an evolutionary standpoint retrospectively, the approach does not, however, include an explicit evolutionary indel model. All things considered, most methods rely on standard models of sequence evolution without indels which is an issue that can only be resolved by including character and indel evolution in a single model.

Two pioneering mathematical models describing the evolution of indels are TKF91 and TKF92 (Thorne et al., 1991, 1992). However, the computation of marginal likelihood under these models has exponential time complexity, rendering the methods relying on

these models extremely computationally intensive, and making inference under these models unrealistic on large datasets. More recently, Bouchard-Côté and Jordan (2013) proposed the Poisson Indel Process (PIP) model which is based on TKF91. PIP describes insertions by a Poisson process defined on the tree topology, while substitutions and deletions are described by a continuous-time Markov process where deletions are modelled as an absorbing state. The assumption of independence between insertion and substitution/deletion enabled a major computational improvement over the previous models (Bouchard-Côté and Jordan, 2013). In PIP, the insertion rate is also independent of the length of a sequence, which is a realistic assumption based on the data that is most commonly analysed (Bouchard-Côté, 2010, p. 93). In contrast to TKF91, the PIP model allows to compute marginal likelihoods in linear time with respect to the number of sequences, which enables a variety of phylogenetic applications (e.g., Maiolo et al. (2018)).

In this study, we use the PIP model for joint reconstruction of ancestral character states including insertions and deletions. Our method ARPIP (Anscestral Reconstruction under PIP) is implemented in the ML framework, i.e., we use an empirical Bayesian approach with ML estimates. Given a multiple sequence alignment (MSA) and a phylogenetic tree, we first use PIP to infer insertion and deletion points on the tree. Insertion and deletion points are the specific locations on the phylogeny where the events have happened. Next, we extract a subtree rooted at the insertion point and pruned by the deletion points. Finally, we reconstruct ancestral states on the extracted subtree using a modified version of Felsenstein's recursion (Felsenstein, 1981), similar to the FastML algorithm (Pupko et al., 2000). In the following we describe the method in detail, validate it by simulations, and demonstrate its performance in simulations and on a real dataset.

Materials and Methods

ARPIP consists of two main algorithms: indel point inference and ancestral character inference.

The IndelPoints algorithm infers the most likely indel points for each site m of the given alignment (Appendix 2). It traverses the tree in post-order and evaluates a set of possible indel scenarios for each node in the tree. A particular indel scenario defines a homology path $\mathcal{H}$. A homology path contains a single insertion point and a number of deletion points consistent with the input MSA. IndelPoints finds the most likely indel scenario by maximizing the probability of $\mathcal{H}$ given m. The maximisation is simplified by reducing the MSA to gap and non-gap states, and ignoring the substitution history without changing the result of the computation. This allows us to avoid matrix exponentiation, which is computationally expensive but necessary for the full likelihood computation.

Similar to the recursive likelihood computation, we traverse the tree and evaluate all the possible indel scenarios, selecting the best one at each node. At the tree root, we select the best homology path over the whole tree based on the best paths selected in the child nodes. For each site m, we use the inferred homology path to extract a subtree τ_m rooted at the insertion point $\mathcal{I}$ and pruned by deletion points $\mathcal{D}$, which represents the most likely indel history for the given site.

Next, we reconstruct ancestral characters on the pruned subtrees in a manner similar to FastML (Pupko et al., 2000). For each site m, we use DP to reconstruct ancestral characters in two phases. The first phase can be seen as a modification of Felsenstein's peeling recursion for computing marginal likelihoods (Felsenstein, 1981). As in the peeling recursion algorithm, we traverse the tree τ_m in post-order, starting from the leaves upward to the root and propagate partial likelihoods. However, instead of marginalising over internal character states, for each MSA column m we store the likelihood values Lk_v and the corresponding best ancestral character states CS_v for each node v. In the second phase, the algorithm traverses the tree in pre-order and for each node selects the ancestral character A_v with the highest conditional probability.

Preliminaries: the PIP model

The PIP model describes the evolutionary process of substitutions, insertions and deletions along the branches of a phylogenetic tree τ. Here we include the basic description of the process, additional information on the PIP likelihood is available in Appendix 1 and a detailed description of PIP can be found in Bouchard-Côté and Jordan (2013).

Let $\tau = (\mathcal{V}, \mathcal{E}, \mathrm{b})$ represent a rooted binary phylogenetic tree, where set $\mathcal{V}$ is the set of all vertices of the tree, $\mathcal{E}$ is the set of all tree branches ($\mathcal{V} \times \mathcal{V}$) and b refers to the branch lengths in units of time (measured in expected substitutions and deletions per site).

The observed sequences are strings of characters from an alphabet Σ, which can be nucleotides, amino acids or codons. The N observed sequences at the leaves of τ are denoted by set $\mathcal{L} \subset \mathcal{V}$, whereas set $\mathcal{V} \setminus \mathcal{L}$ is the set of $N-1$ internal vertices. The root, the most recent common ancestor of all leaves, is labelled by Ω. The branch length $\mathrm{b}(v)$ associated with node ($v \in \mathcal{V}$) spans from v to its parent vertex $\mathrm{pa}(v)$ (see Fig. 1).

PIP is parameterised by insertion rate λ and deletion rate μ, with the process running over tree topology τ. For every node $v \in \mathcal{V}$, the probability of inserting a single character on edge $e = (\mathrm{pa}(v) \to v)$ is proportional to the branch length and defined by $\iota(v)$ (see Appendix 1). Similarly, the survival probability for a character inserted on edge e is $\beta(v)$ (see Appendix 1). Additionally, we define the pure survival probability $\zeta = \exp(-\mu \mathrm{b}(v))$ associated with node v as if the character was already present at the parent node $\mathrm{pa}(v)$ (Maiolo, 2019). Point substitutions and deletions are modelled by a continuous-time Markov process on $\Sigma_\varepsilon = \Sigma \cup \{\varepsilon\}$, where ε denotes the gap symbol. The generator matrix Q could be any arbitrary reversible substitution model, e.g. WAG for amino acids (Whelan and Goldman, 2001), or K80 for nucleotide data (Kimura, 1980). Accordingly, the extended generator matrix is denoted by Q_ε and the extended quasi-stationary distribution is $\pi_\varepsilon = [\pi, 0]$ (Bouchard-Côté and Jordan, 2013).

Let $\mathcal{G}$ define the site-specific set of all potential deletion points on the tree. $\mathcal{G}$ consists of all leaves with a gap at the respective site, and of all the internal nodes whose

all descendant leaves have a gap at that site. Next, consider the subset $\mathcal{S}$ of leaves that have a non-gap character, $\mathcal{S} = \{v \in \mathcal{L} : m_v \neq \varepsilon\}$. Given the set $\mathcal{S}$, we define the set $\mathcal{A}$ of potential insertion points to include all nodes that are ancestral to all the leaves in $\mathcal{S}$ (see Fig. 1). In general, we compute the probability $p(m)$ of each individual MSA column by marginalizing over all possible homology paths underlying that MSA column (see Appendix 1) based on their homology path probabilities f_v.

Inferring the indel points

We propose a progressive algorithm to infer the most likely indel points (homology path) on the tree under the PIP model. For each site, we progressively find the best partial homology path (constrained by a subtree) and build on the intermediate results to get the most likely indel history on the whole tree. Since we search only for the most likely homology path, we compute a simplified likelihood function which accounts only for insertions and deletions and ignores substitutions.

Under PIP, two mutually exclusive node sets exist on the tree topology τ: the set of nodes where the character has gone extinct and the set of nodes where the character has definitely survived. The first is the set of potential deletion nodes $\mathcal{G}$, defined in the previous section. The second contains all the remaining nodes in the tree $v \in \mathcal{V} \setminus \mathcal{G}$ ($v \notin \mathcal{G}$). A node $v \notin \mathcal{G}$ may also be a potential insertion location, i.e. $v \in \mathcal{A}$ (see Fig. 2 and Appendix 2 for the detailed description). While the set $\mathcal{A}$ may contain multiple nodes, a homology path can only have a single insertion location. Consequently, each node in $\mathcal{A}$ is associated with a single homology path with the highest probability. This implies that when computing the probability of a homology path for node $v \in \mathcal{A}$, it is treated as the only potential insertion location, while all other nodes are treated as regular nodes in the tree. Notably, one cannot simply select the node with the highest insertion probability $\iota(v)$, as the probability of any given homology path also depends on the survival/extinction of the site in the children. Even though we separately describe the treatment of the two node types, all the necessary

computation can be done in a single post-order traversal of the tree.

For each node v in the tree, we first compute f_v, the conditional probability of the deletion/substitution process, assuming that the character exists in v. We compute f_v for the most likely deletion scenario in this subtree rather than marginalise over all possible deletion locations. We also compute p_v, the conditional probability of the homology path assuming that the character was inserted at node v. A character necessarily has to be inserted at one of the nodes $v \in \mathcal{A}$, which means that the probability will be non-zero only for the potential insertion nodes.

In the progressive algorithm we maintain several node sets that are needed to define the most likely homology path per node. Let $\mathcal{I}_v$ denote the set of insertion points for the subtree rooted at v. Then $\mathcal{I}_v = \emptyset$ for $v \notin \mathcal{A}$ and $\mathcal{I}_v = \{v\}$ for $v \in \mathcal{A}$. Similarly, $\mathcal{D}_v$ denotes the set of deletion points for the subtree rooted at v.

For each node v we store the locally optimal homology path $\mathcal{H}_v = \{\mathcal{I}_v, \mathcal{D}_v\}$ for the subtree rooted at v. Once we reach the root node Ω, all possible insertion locations would be considered, and the one with the highest probability is selected among those. At this point, the best homology path $\mathcal{H}_{\arg\max(p_v)}$ (defined by the highest conditional probability p_v) is used to extract the subtree τ_m rooted at $\mathcal{I}_{\arg\max(p_v)}$ and pruned by $\mathcal{D}_{\arg\max(p_v)}$, which represents the best possible indel points for the MSA column m. We will use τ_m to infer ancestral character states for column m. The IndelPoints algorithm is presented in Figure 2 and the pseudocode for the algorithm can be found in the Appendix 2.

Dynamic programming joint ancestral sequence reconstruction

Our method performs ASR in a manner very similar to FastML (Pupko et al., 2000) with two crucial differences. First, we only work on a subtree τ_m of the original tree τ, which limits the reconstruction to the most probable insertion location at this site. This means we do not reconstruct any ancestral states where there were none. Second, to appropriately account for character deletion, the ancestral reconstruction is done using the

PIP substitution rate matrix Q_ϵ.

The joint ASR method under PIP given column m and the pruned rooted phylogenetic subtree τ_{m} consists of two steps. The first step is to compute the partial likelihood values on subtree τ_{m} with the modified version of Felsenstein recursion algorithm, where both likelihood values and their corresponding ancestral character states are stored. The second step is to reconstruct the character states by picking the character with highest conditional probability. The recursive algorithm for joint ASR is shown in Appendix 3 together with the newly defined pseudocode for the procedure.

Results

Three datasets were used to evaluate and illustrate our method. The first dataset was simulated under the PIP. This dataset allows to evaluate the performance of ARPIP under the true model. Given the true simulated trees and MSAs, both the homology path inference and ASR were evaluated.

The second dataset was used to evaluate the performance of ARPIP for sequences with long indels. The data was generated by INDELible (Fletcher and Yang, 2009) with two different settings using the same trees as for the PIP simulations. For this dataset, the ancestral sequences are also known and can be used for evaluation. However, the PIP parameters for this dataset must be inferred. As INDELible does not provide a comprehensive description of indel events on the phylogeny, we used these simulations to evaluate ancestral state inference.

The third dataset is a small coronavirus sample which was extracted from Uniprot (Bateman et al., 2020). With this dataset, we aimed to provide a showcase of the method.

When analysing both INDELible and real-life data, we first have to infer the PIP parameters, i.e. λ and μ, given an MSA and a tree. This computation was done based on Brent's optimization method (Brent, 1973), optimizing one parameter at a time until convergence. For all examples, the protein substitution model used is WAG (Whelan and

Goldman, 2001).

Note that ARPIP was developed for rooted trees. If an unrooted tree is provided, ARPIP uses mid-point rooting method to root the tree. Furthermore, ARPIP can perform ASR without a provided tree. The user can select from established fast methods like neighbor joining, BioNJ, UPGMA, and WPGMA to estimate the tree from the input MSA.

Data simulated under PIP

The simulated sequences are given as input to ARPIP along with the true model parameters so that we only have to estimate the ancestral state values. The simulated dataset contains 100 MSA/tree replicates with their corresponding evolutionary events. Each replicate was simulated using an 8 taxa tree with a topology sampled from the uniform distribution and branch lengths sampled from an exponential distribution with the rate $\rho = 2$, where ρ is a proxy for phylogenetic divergence. One of the simulated trees is shown in Figure 3. On average, the branch lengths of the simulated trees were 0.45 units of time, ranging from minimal branch length of 0 and maximum branch length of 3.23. For the simulations, we set the deletion rate $\mu = 0.1$ and the insertion rate $\lambda = 10$ for PIP.

Analysis of the PIP simulated dataset.— In order to assess the accuracy of ARPIP, we independently evaluated each inference step, IndelPoints and the joint ASR (see Table 1). To asses the accuracy of the IndelPoints algorithm, we also evaluated the inference of insertion and deletion events independently. As this dataset was simulated under the same model we use for inference, we used the true parameter values in the analysis without inferring them ($\mu = 0.1$ and $\lambda = 10$). This way we can evaluate the method without the additional variation of parameter inference, which is done for the other two datasets.

Among the 100 input sets, 96.69% insertion points and 95.54% deletion points were inferred correctly. In the next step, we computed the accuracy of ASR per site. This

number has been averaged over all existing sites over all MSA replicates. To evaluate the reconstructed ancestral sequences, we used three different metrics. Firstly, we counted the number of full ancestral columns that were inferred correctly, which is 60.08% for this dataset. Secondly, we counted the number of characters that were inferred correctly, which amounts to 88.14% of characters. Thirdly, we counted the number of gap characters themselves that were inferred correctly, which amounts to 99.86%.

Data generated by INDELible

The data simulated by INDELible contains two sets of 100 MSA/tree replicas. Each replica was simulated using an 8 taxa tree from PIP simulations. We used the Zipfian (power law) distribution for the indel model with $a = 1.7$, to generate the samples where $a > 1$ is the exponent characterizing the distribution. Empirical estimates of value a range from 1.5 to 2 (Fletcher and Yang, 2009), which prompted us to select $a = 1.7$. The maximum indel length was set to 5 to avoid MSAs with excessively long gaps. Two different indel rates of 0.01 and 0.05 were used for the simulation with INDELible.

Analysis of the INDELible simulated dataset.— For this dataset, ARPIP inferred the PIP parameters λ and μ as well as ancestral character states. For the two datasets of 100 MSA/tree replicates with indel rates of 0.01 and 0.05, ARPIP correctly inferred 46.58% and 59.49% of the ancestral sites respectively. Further, ARPIP correctly inferred 83.49% and 87.93% of characters including gaps. Finally, over 99.95% of gap characters were inferred correctly for the two datasets (see Table 2).

Coronavirus data

The ongoing *SARS-CoV-2* pandemic strongly affects our lives, causing an immense interest for phylogenetic analyses of the relevant viral molecular sequences. Like in other coronaviruses, the spike protein in *SARS-CoV-2* is important for viral entry into host cells.

It is also one of the major determining factors of host range (Belouzard et al., 2012; Zhou and Zhao, 2020). We therefore used this protein as an example demonstrating ancestral sequence inference.

SARS-CoV-2 is a member of the *Betacoronavirus* genus which also contains the two other recent human coronavirus strains, namely *SARS-CoV* and *MERS-CoV* (Lefkowitz et al., 2018). For our analyses, we selected a small set of available protein sequences from this genus (see Table 3 for the exact sequence list).

Analysis of the coronavirus dataset.— The MSAs of the coronavirus sequences were inferred using ProPIP (Maiolo et al., 2018) and PRANK phylogeny-aware webserver (Löytynoja, 2014). The total length of the reconstructed MSAs was 2002 and 1929 AAs respectively. The phylogenetic trees were reconstructed by ML in PhyML 3.0 (Guindon et al., 2010), using smart model selection on amino acids and SPR tree moves (Lefort et al., 2017) (see Fig. 4). Then, given an MSA and tree we inferred ancestral sequences with ARPIP. The estimated deletion rates for ProPIP and PRANK's MSAs are respectively $\hat{\mu} = 0.242$ and $\hat{\mu} = 0.210$. Figure 5 summarises the resulting ASR by ARPIP comparing to FastML on MSA produced by ProPIP while the results for PRANK can be found in the supplement (Figs. 12-13).

Comparison against the state-of-the-art methods

At this moment, the two most frequently used ML joint ASR approaches are PAML (Yang, 1997) and FastML (Pupko et al., 2000; Ashkenazy et al., 2012), both of which work in linear time with respect to the number of sequences. An important distinction among the methods lies in the way they handle gaps in the alignment. PAML can either ignore gaps in the alignment by removing all columns containing at least one gap character (option used here), or treat all gap characters as ambiguous. For this study, we used PAML on an MSA without any gap characters to compare the accuracy of ancestral

character reconstruction. FastML webservice (Ashkenazy et al., 2012) uses an ad-hoc indel-coding (Simmons and Ochoterena, 2000) approach to account for indels spanning multiple adjacent characters. Indel coding is done as a separate step in the inference process, done independently from the ancestral state reconstruction. We compare the performance of ARPIP and FastML on an MSA with gaps.

On both simulated datasets, the accuracy of ARPIP appears similar to FastML (see Tables 1- 2 and Figs. 6, 8 and 9). For example, both algorithms inferred the ancestral state accurately in certain regions (e.g., Fig. 6a, Fig. 8a and d, and Fig. 9a) and falsely in other regions (e.g., Fig. 6b, Fig. 8b and c, and Fig. 9b). In region c of Figure 6 and region d of Figure 9, FastML inferred a character state even though there is no ancestral character, since the insertion happened at the leaf. In certain regions FastML could not determine which internal node had the information (e.g., Fig. 6d), while ARPIP was capable of determining the character position accurately. ARPIP outperformed FastML in determining the gap position in certain regions (e.g., Fig. 6c and d, Fig. 8b, and Fig. 9c and d). On CoV data, the situation was similar meaning FastML could not detect the insertion location (see Fig. 5a) while in conserved regions the inferred states were almost identical (see Fig. 5b).

We also considered scenarios without indels, allowing the evolutionary process to work only through substitutions. In this case the alignments have no gaps, i.e. no deletion ($\mu = 0$) and all insertions happen at the root of the tree. Under these conditions, all the algorithms perform reasonably as presented in Figure 7.

Discussion and conclusion

In this article, we present a one-of-a-kind approach for fast likelihood-based ancestral sequence reconstruction with insertions and deletions. Unlike previous approaches, our method relies on an explicit model of indel and character evolution and allows us to infer the full history of sequence evolution, including insertion and deletion

points on a phylogeny. The method is implemented in the probabilistic framework and is based on likelihood calculations under the PIP model. Likelihood computations under this model have linear time complexity with respect to the number of sequences, meaning that our method is highly efficient on large datasets.

We show that on PIP simulated datasets, ARPIP correctly infers at least 95% of indel events and at least 88% of ancestral characters. On the INDELible simulated data, ARPIP correctly infers 83% and 87% of ancestral characters including gaps for low and high indel rates respectively. ARPIP also correctly places gaps in over 99.95% cases, showing the credibility of our IndelPoints algorithm. For all datasets, we illustrate the performance of our approach in comparison to FastML on alignments with gaps. In addition, we use gapless alignments to compare ARPIP inferences with those by PAML and FastML, showing that our approach performs just as well for data without gaps.

While indel events represent a major mutational process of gene evolution (Söding and Lupas, 2003), they are rarely accounted for in ASR. ARPIP expands the reconstruction possibilities to include more divergent and gappy sequences allowing us to study a wider range of resurrected ancestral molecules, investigating the functional importance of indels in ancestral proteins. This is particularly valuable for proteins separated by large divergences or within "more flexible" loop regions, as indels frequently occur in regions where amino acid sequences are not well conserved (Taylor et al., 2004a).

While single residue indel modelling may be viewed as a limitation, certain types of genetic material exhibit specifically these kind of indel events more often than others. For example, single nucleotide indels are predominant between recently diverged DNA sequences from various organisms (Tao et al., 2007) and in non-coding DNA sequences (Yamane et al., 2006). While most ASR is done on coding sequences to investigate the properties of reconstructed proteins, it has recently been shown that many trait-associated loci, including some associated with disease, lie outside protein-coding regions (Kellis et al., 2014). ARPIP can be used to reconstruct non-coding sequences with meaningful biological

assumptions, which could be an additional avenue of exploration for disease-related ASR.

ARPIP paves the way for even more new types of indel analyses. The approach can be expanded to analyse the patterns of insertions and deletions by including rate heterogeneity, for example, allowing us to detect lineage-specific patterns through time. We can include site-specific indel rate variation, allowing us to see the difference in indel evolution in different functional regions of proteins such as loops or active sites. Then, we can investigate the occurrence and consequences of indels in specific regions such as indel-tolerant regions of the genome and relation between gene function and indel frequency (Taylor et al., 2004b). Moreover, in the long run our method can be used to extend and potentially improve more sophisticated probabilistic approaches such as (Groussin et al., 2014), which accounts not only for gene-trees but also for species history, therefore including gene gain/loss and horizontal transfer in the inference.

While some other methods have attempted to reconstruct exact indel histories, the only other currently existing method in the frequentist framework can only handle small datasets (Diallo et al., 2007). Even though PIP makes simplifying assumptions like site independence, which only allows us to model single residue indels, the explicit evolutionary model makes indel events interpretable. Moreover, as the method has linear time complexity, we can use this approach as a building block in integrated alignment-tree-ancestor inference (Pečerska et al., 2021), using the indel points under PIP as a starting point for integrating more complex models of indel evolution, e.g. moving on to long indel models.

Like other likelihood based approaches, our method in theory allows us to explore both optimal and suboptimal reconstructions in follow-up analyses. It has been argued that a single reconstruction (i.e. a point estimate) can be inadequate in cases when the likelihood surface is nonconvex and contains multiple local optima (Joy et al., 2016), which can lead to systematic bias (Yang, 2014, p.131). Since ARPIP is in essence an empirical Bayes method, we can extend the method to account for the uncertainty in our estimates

by working with probability profiles of characters and gaps rather than inferences fixed to the optimal estimates (Williams et al., 2006).

Availability of code and experimental data

The proposed algorithm has been implemented based on Bio++ open source library (Guéguen et al., 2013) using C++ programming language. Our code with a brief user manual is freely available at `https://github.com/acg-team/bpp-ARPIP` under GNU GPLv3 licence. The data underlying this article are also available from the Dryad Digital Repository at `https://doi.org/10.5061/dryad.wstqjq2nj`.

Funding

This work was supported by the Swiss National Science Foundation (SNSF) (grant number 31003*A*_176316 to M.A). The funding body did not play any role in the design of the study and collection, analysis, and interpretation of data and in writing the manuscript.

Acknowledgements

We would like to thank A. Bouchard-Côté (University of British Columbia) for providing his code JavaPIP to simulate sequences under the PIP, and our master student J. Peechatt for his preliminary work which helped to develop this method.

References

Ashkenazy, H., O. Penn, A. Doron-Faigenboim, O. Cohen, G. Cannarozzi, O. Zomer, and T. Pupko. 2012. Fastml: a web server for probabilistic reconstruction of ancestral sequences. Nucleic acids research 40:W580–W584.

Bateman, A., M.-J. Martin, S. Orchard, M. Magrane, R. Agivetova, S. Ahmad, E. Alpi,

E. H. Bowler-Barnett, R. Britto, B. Bursteinas, et al. 2020. Uniprot: the universal protein knowledgebase in 2021. Nucleic Acids Research 49:D480–D489.

Belouzard, S., J. K. Millet, B. N. Licitra, and G. R. Whittaker. 2012. Mechanisms of coronavirus cell entry mediated by the viral spike protein. Viruses 4:1011–1033.

Bouchard-Côté, A. 2010. Probabilistic Models of Evolution and Language Change. Ph.D. thesis University of California at Berkeley University of California at Berkeley phD thesis.

Bouchard-Côté, A. and M. I. Jordan. 2013. Evolutionary inference via the poisson indel process. Proceedings of the National Academy of Sciences 110:1160–1166.

Brent, R. P. 1973. Algorithms for minimization without derivatives. Englewood Cliffs, NJ, USA Prentice Hall Page 195.

Brintnell, E., M. Gupta, and D. W. Anderson. 2021. Phylogenetic and ancestral sequence reconstruction of sars-cov-2 reveals latent capacity to bind human ace2 receptor. Journal of molecular evolution 89:656–664.

Chang, B. S., J. A. Ugalde, and M. V. Matz. 2005. Applications of ancestral protein reconstruction in understanding protein function: Gfp-like proteins. Pages 652–670 *in* Methods in enzymology vol. 395. Elsevier.

Dessimoz, C. and M. Gil. 2010. Phylogenetic assessment of alignments reveals neglected tree signal in gaps. Genome biology 11:R37.

Diallo, A. B., V. Makarenkov, and M. Blanchette. 2007. Exact and heuristic algorithms for the indel maximum likelihood problem. Journal of Computational Biology 14:446–461.

Diallo, A. B., V. Makarenkov, and M. Blanchette. 2009. Ancestors 1.0: a web server for ancestral sequence reconstruction. Bioinformatics 26:130–131.

Felsenstein, J. 1981. Evolutionary trees from dna sequences: a maximum likelihood approach. Journal of molecular evolution 17:368–376.

Fletcher, W. and Z. Yang. 2009. Indelible: a flexible simulator of biological sequence evolution. Molecular biology and evolution 26:1879–1888.

Groussin, M., J. K. Hobbs, G. J. Szöllősi, S. Gribaldo, V. L. Arcus, and M. Gouy. 2014. Toward More Accurate Ancestral Protein Genotype–Phenotype Reconstructions with the Use of Species Tree-Aware Gene Trees. Molecular Biology and Evolution 32:13–22.

Guéguen, L., S. Gaillard, B. Boussau, M. Gouy, M. Groussin, N. C. Rochette, T. Bigot, D. Fournier, F. Pouyet, V. Cahais, et al. 2013. Bio++: efficient extensible libraries and tools for computational molecular evolution. Molecular biology and evolution 30:1745–1750.

Guindon, S., J.-F. Dufayard, V. Lefort, M. Anisimova, W. Hordijk, and O. Gascuel. 2010. New algorithms and methods to estimate maximum-likelihood phylogenies: assessing the performance of phyml 3.0. Systematic biology 59:307–321.

Joy, J. B., R. H. Liang, R. M. McCloskey, T. Nguyen, and A. F. Poon. 2016. Ancestral reconstruction. PLoS computational biology 12:e1004763.

Kellis, M., B. Wold, M. P. Snyder, B. E. Bernstein, A. Kundaje, G. K. Marinov, L. D. Ward, E. Birney, G. E. Crawford, J. Dekker, et al. 2014. Defining functional dna elements in the human genome. Proceedings of the National Academy of Sciences 111:6131–6138.

Kimura, M. 1980. A simple method for estimating evolutionary rates of base substitutions through comparative studies of nucleotide sequences. Journal of molecular evolution 16:111–120.

Lefkowitz, E. J., D. M. Dempsey, R. C. Hendrickson, R. J. Orton, S. G. Siddell, and D. B. Smith. 2018. Virus taxonomy: The database of the International Committee on Taxonomy of Viruses (ICTV). Nucleic Acids Research 46:D708–D717.

Lefort, V., J.-E. Longueville, and O. Gascuel. 2017. SMS: Smart Model Selection in PhyML. Molecular Biology and Evolution 34:2422–2424.

Liberles, D. A. 2007. Ancestral sequence reconstruction. Oxford University Press on Demand.

Löytynoja, A. 2014. Phylogeny-aware alignment with prank. Pages 155–170 *in* Multiple sequence alignment methods. Springer.

Maiolo, M. 2019. Progressive Multiple Sequence Alignment with Indel Evolution. Ph.D. thesis University of Lausanne Lausanne phD thesis.

Maiolo, M., X. Zhang, M. Gil, and M. Anisimova. 2018. Progressive multiple sequence alignment with indel evolution. BMC bioinformatics 19:331.

Pagel, M. 1999. Inferring the historical patterns of biological evolution. Nature 401:877–884.

Pečerska, J., M. Gil, and M. Anisimova. 2021. Joint alignment and tree inference. bioRxiv .

Pupko, T., I. Pe, R. Shamir, and D. Graur. 2000. A fast algorithm for joint reconstruction of ancestral amino acid sequences. Molecular biology and evolution 17:890–896.

Simmons, M. P. and H. Ochoterena. 2000. Gaps as characters in sequence-based phylogenetic analyses. Systematic biology 49:369–381.

Söding, J. and A. N. Lupas. 2003. More than the sum of their parts: on the evolution of proteins from peptides. Bioessays 25:837–846.

Starr, T. N., S. K. Zepeda, A. C. Walls, A. J. Greaney, S. Alkhovsky, D. Veesler, and J. D. Bloom. 2022. Ace2 binding is an ancestral and evolvable trait of sarbecoviruses. Nature Pages 1–9.

Tao, S., Y. Fan, W. Wang, G. Ma, L. Liang, and Q. Shi. 2007. Patterns of insertion and deletion in mammalian genomes. Current Genomics 8:370–378.

Taylor, M. S., C. P. Ponting, and R. R. Copley. 2004a. Occurrence and consequences of coding sequence insertions and deletions in mammalian genomes. Genome research 14:555–566.

Taylor, M. S., C. P. Ponting, and R. R. Copley. 2004b. Occurrence and consequences of coding sequence insertions and deletions in mammalian genomes. Genome research 14:555–566.

Thorne, J. L., H. Kishino, and J. Felsenstein. 1991. An evolutionary model for maximum likelihood alignment of dna sequences. Journal of Molecular Evolution 33:114–124.

Thorne, J. L., H. Kishino, and J. Felsenstein. 1992. Inching toward reality: an improved likelihood model of sequence evolution. Journal of molecular evolution 34:3–16.

Thornton, J. W. 2004. Resurrecting ancient genes: experimental analysis of extinct molecules. Nature Reviews Genetics 5:366–375.

Whelan, S. and N. Goldman. 2001. A general empirical model of protein evolution derived from multiple protein families using a maximum-likelihood approach. Molecular biology and evolution 18:691–699.

Williams, P. D., D. D. Pollock, B. P. Blackburne, and R. A. Goldstein. 2006. Assessing the accuracy of ancestral protein reconstruction methods. PLoS computational biology 2:e69.

Yamane, K., K. Yano, and T. Kawahara. 2006. Pattern and rate of indel evolution inferred from whole chloroplast intergenic regions in sugarcane, maize and rice. DNA research 13:197–204.

Yang, Z. 1997. Paml: a program package for phylogenetic analysis by maximum likelihood. Bioinformatics 13:555–556.

Yang, Z. 2007. Paml 4: phylogenetic analysis by maximum likelihood. Molecular biology and evolution 24:1586–1591.

Yang, Z. 2014. Molecular evolution: a statistical approach. Oxford University Press.

Yang, Z., S. Kumar, and M. Nei. 1995. A new method of inference of ancestral nucleotide and amino acid sequences. Genetics 141:1641–1650.

Zakas, P. M., H. C. Brown, K. Knight, S. L. Meeks, H. T. Spencer, E. A. Gaucher, and C. B. Doering. 2017. Enhancing the pharmaceutical properties of protein drugs by ancestral sequence reconstruction. Nature biotechnology 35:35.

Zhou, G. and Q. Zhao. 2020. Perspectives on therapeutic neutralizing antibodies against the novel coronavirus sars-cov-2. International journal of biological sciences 16:1718.

Figure captions

Figure 1: The phylogenetic tree τ rooted at Ω. $b(v_1)$ represents the branch length from Ω to v_1. The leaves of the tree show a single column of the MSA including gaps as an additional character state. The set $\mathcal{S}$ is defined as all leaves with a character in the given column (not a gap). The set of potential insertion nodes $\mathcal{A}$ contains the nodes ancestral to all nodes in $\mathcal{S}$. Finally, the set of potential deletion nodes $\mathcal{G}$ is defined as all nodes which are either a leaf with a gap in the given column, or a node whose both children are in $\mathcal{G}$.

Figure 2: Overview of the IndelPoints algorithm. The tree is traversed in post-order to infer the most likely homology path progressively using the predefined sets: $\mathcal{L}$ the set of all leaves, $\mathcal{A}$ the set of potential insertion points, and $\mathcal{G}$ the set of potential deletion points. Here, σ represents the character in focus and v is the node visited during the tree traversal.

Figure 3: An example tree from the dataset generated by the PIP simulator.

Figure 4: Illustration of the rooted *Betacoronavirus* phylogenetic tree which was reconstructed by PhyML 3.0 from the ProPIP alignment. Note that the original tree was unrooted which ARPIP used mid-point rooting method to make the tree rooted.

Figure 5: Illustration of a snippet from the *CoV* dataset containing the MSA inferred by ProPIP and the ancestral sequences predicted by ARPIP and FastML. a) The region in which ARPIP infers a very different ancestral history, probably due to inferring the insertion point prior to ancestral character inference. FastML inferred no gap in this column perhaps due to the adjacent (first) column. b) The region in which both algorithms had similar inferences of the ancestral states. Note that FastML algorithm works on an unrooted tree which, compared to ARPIP, resulted in one fewer internal

sequence reconstructed (due to the absence of the root node).

Figure 6: A snippet from the PIP simulated dataset containing the true simulated MSA and ancestors and the ancestral sequences predicted by ARPIP and FastML. a) A region where both ARPIP and FastML accurately inferred the ancestral states. b) A region where both algorithms estimated the ancestral character incorrectly. c) A region where FastML inferred ancestral characters even though there were none in the simulation. d) A region where there was a single ancestral character but FastML inferred its position incorrectly. Note that FastML algorithm works on an unrooted tree which, compared to ARPIP, resulted in one fewer internal sequence reconstructed (due to the absence of the root node).

Figure 7: A gapless snippet from the PIP simulated dataset containing the true simulated MSA and ancestors and the ancestral sequences predicted by ARPIP, PAML and FastML. a) A region where all algorithms accurately inferred the ancestral state. b) A region where all algorithms made mistakes. c) A region where FastML and PAML made incorrect inferences but ARPIP inferred the ancestral state correctly. d) A region where all algorithms accurately inferred the ancestral states except ARPIP. Note that FastML algorithm works on an unrooted tree which, compared to ARPIP, resulted in one fewer internal sequence reconstructed (due to the absence of the root node).

Figure 8: A snippet from the INDELible simulated dataset with indel rate 0.01 **containing the true simulated MSA, ancestors and the ancestral sequences predicted by ARPIP and FastML.** a) A region where both ARPIP and FastML accurately inferred the ancestral states. b) A region that the FastML inferred the gaps incomplete while ARPIP missed the the character state. c) A region where both algorithms estimated most of the ancestral character incorrectly. d) A region where both

methods inferred the ancestral states including gaps positions correctly. Note that FastML algorithm works on an unrooted tree which, compared to ARPIP, resulted in one fewer internal sequence reconstructed (due to the absence of the root node).

Figure 9: A snippet from the INDELible simulated dataset with indel rate 0.05 **containing the true simulated MSA and ancestors and the ancestral sequences predicted by ARPIP and FastML.** a) A region where both ARPIP and FastML accurately inferred the ancestral states. b) A region where both algorithms estimated the indel events correctly but the ancestral character incorrectly. c) A region where FastML missed the gap character but ARPIP inferred it correctly. d) A region where FastML inferred ancestral characters even though there were none in the simulation. Note that FastML algorithm works on an unrooted tree which, compared to ARPIP, resulted in one fewer internal sequence reconstructed (due to the absence of the root node).

Figure 10: Evaluating extinction nodes algorithm, where $\mathcal{H}_v$ defines the homology path of node v, $\mathrm{f_v}$ is the probability of that homology path and p_v is the probability of the best homology path. $\mathcal{I}_v$ is the set of possible insertion points while $\mathcal{D}_v$ is the set of all possible deletion points for node v. $\zeta(v)$ denotes the pure survival probability of node v. Moreover, $\mathcal{G}$ and $\mathcal{L}$ represent the set of potential deletion points and leaves, respectively.

Figure 11: Evaluating survival nodes algorithm. $\mathcal{H}_v$ defines the homology path of node v represents by the set of possible insertion points $\mathcal{I}_v$ and the set of all possible deletion points $\mathcal{D}_v$. Moreover, $\mathrm{f_v}$ is the probability of that homology path while p_v is the probability of the best homology path. $\iota(v)$, $\beta(v)$ and $\zeta(v)$ are respectively insertion, survival and pure survival probabilities at node v. $\mathcal{A}$ represents the set of potential insertion points and $\mathcal{L}$ shows the set of leaves.

Figure 12: Illustration of the rooted *Betacoronavirus* phylogenetic tree reconstructed by PhyML 3.0 from the PRANK alignment. Note that the original tree was unrooted which ARPIP used mid-point rooting method to make the tree rooted.

Figure 13: A snippet from the *CoV* dataset containing the MSA inferred by PRANK and the ancestor sequences predicted by ARPIP and FastML. The tree is obtained by PhyML 3.0, shown in Figure 12. Note that FastML algorithm works on an unrooted tree which, compared to ARPIP, resulted in one fewer internal sequence reconstructed (due to the absence of the root node).

Metric	Accuracy (%)
Correctly inferred insertion points	96.08 ± 2.84
Correctly inferred deletion points	95.54 ± 2.80
Correctly inferred MSA columns	60.08 ± 9.60
Correctly inferred characters including gap	88.14 ± 3.91
Correctly inferred gap character	99.86 ± 0.26

Table 1: ARPIP accuracy for inference on PIP simulated data.

Metric	Accuracy (%)	
	indel rate 0.01	indel rate 0.05
Correctly inferred MSA columns	46.58 ± 13.22	59.49 ± 10.63
Correctly inferred characters including gap	83.49 ± 6.04	87.93 ± 4.33
Correctly inferred gap character	99.98 ± 0.16	99.95 ± 0.14

Table 2: ARPIP accuracy for inference on INDELible simulated data.

Subgenus	Species	Uniprot accession number
Embecovirus	*Betacoronavirus 1*	A0A191URB2
	China Rattus coronavirus HKU24	A0A0A7UZR7
	Human coronavirus HKU1	U3NAI2
	Murine coronavirus	P11224
	Myodes coronavirus 2JL14	A0A2H4MXV6
Hibecovirus	*Bat Hp-betacoronavirus Zhejiang2013*	A0A088DJY6
Merbecovirus	*Hedgehog coronavirus 1*	A0A4D6G1A4
	Middle East respiratory syndrome-related coronavirus	K9N5Q8
	Pipistrellus bat coronavirus HKU5	A3EXD0
	Tylonycteris bat coronavirus HKU4	A3EX94
Nobecovirus	*Rousettus bat coronavirus GCCDC1*	A0A1B3Q5W5
	Rousettus bat coronavirus HKU9	A3EXG6
Sarbecovirus	*Severe acute respiratory syndrome-related coronavirus*	A0A3Q8AKM0
	Severe acute respiratory syndrome-related coronavirus 2	P0DTC2

Table 3: *Betacoronavirus* sequences used in the analysis.