Multiple Sequence Alignment for Large Heterogeneous Datasets using SATé, PASTA and UPP

Tandy Warnow and Siavash Mirarab

July 23, 2019

Abstract

The estimation of very large multiple sequence alignments is a challenging problem that requires special techniques in order to achieve high accuracy. Here we describe two software packages–PASTA and UPP–for constructing alignments on large and ultra-large datasets. Both methods have been able to produce highly accurate alignments on 1,000,000 sequences, and trees computed on these alignments are also highly accurate. PASTA provides the best tree accuracy when the input sequences are all full-length, but UPP provides improved accuracy compared to PASTA and other methods when the input contains a large number of fragmentary sequences. Both methods are available in open source form on GitHub.

Keywords: multiple sequence alignment, PASTA, SATé, UPP, ensembles of Hidden Markov Models

1 Introduction

Multiple sequence alignment (MSA) is one of the more complex bioinformatics tasks, and a precursor to many downstream analyses, including protein structure and function prediction, phylogeny estimation, orthology prediction, and even genome assembly. This chapter focuses mainly on the use of multiple sequence alignment for phylogeny estimation, and in particular on the challenge of computing alignments on large, heterogeneous datasets, where standard off-the-shelf methods have low accuracy. Of particular relevance is the challenge of computing multiple sequence alignments of datasets that exhibit sequence length heterogeneity, where all standard methods have particularly poor accuracy.

In 2009, SATé (Simultaneous Alignment and Tree Estimation) was developed to enable the co-estimation of alignments and trees on large challenging datasets [1]. SATé used a combination of divide-and-conquer (where alignments are computed on subsets using standard MSA methods and then merged into

an alignment on the full dataset; see Fig. 1a) and iteration (where each iteration computes a new alignment based on the tree from the prior iteration, and then a new tree is computed on the new alignment) in order to obtain highly accurate alignments on large datasets. SATé-II was developed in 2012 [2] to improve on the accuracy and scalability of SATé; it used a modified decomposition strategy but otherwise had the same structure as SATé. SATé-II was able to run on much larger datasets than SATé, but was still limited to approximately 50,000 sequences. Finally, in 2014, the algorithmic design was changed again to produce PASTA [3, 4]. The objective in the design modification was to enable analyses of even larger datasets, but these changes also improved accuracy. Thus, PASTA, which mainly differs from SATé-II in its merging step (Fig 1b), has the best accuracy and scalability of these three methods.

In 2015, we discovered that PASTA was unable to produce highly accurate alignments when the input dataset has many fragmentary sequences. To address this challenge, we developed UPP (Ultra-large alignments using Phylogeny-aware Profiles [5]), a new technique for alignment estimation that is based on a machine learning technique we developed, called an Ensemble of Profile Hidden Markov Models [6, 7, 8]. UPP uses PASTA to compute a "backbone alignment" of a subset of the input sequences (restricted to just the full-length sequences) and then adds the remaining sequences to the backbone alignment using a computed Ensemble of Profile Hidden Markov Models. UPP provides advantages over PASTA for datasets with fragmentary sequences, but PASTA has advantages over UPP when all the sequences are full-length. Like PASTA, UPP is able to compute highly accurate alignments on ultra-large datasets, including those with 1,000,000 sequences.

This chapter describes, at a very high level, how the PASTA and UPP algorithms operate, and provides some guidance on how to use these methods to obtain the best accuracy[1]. More information on how to run these methods can be obtained from the tutorials for PASTA and UPP available at the GitHub sites for these methods [10, 11].

2 SATé and PASTA

SATé [1], SATé-II [2], and PASTA [4] are methods for computing multiple sequence alignments and trees from unaligned sequences. They all have the same basic algorithmic strategy (Fig. 1), and so can be considered to be members of the same basic paradigm; however, SATé-II was designed to improve on SATé (now called SATé-I) and PASTA was designed to improve on SATé–II, with the result that PASTA dominates the other methods with respect to accuracy, running time, memory usage, and scalability to large datasets. For example, PASTA has been able to compute alignments and trees on up to 1,000,000 sequences, but SATé-I and SATé-II have not been able to analyze datasets of this size. Furthermore, PASTA, which is the method of choice in this family of methods, has

[1]This chapter is an update of [9], a previous article for Methods in Molecular Biology, which focused on using SATé [1, 2] for co-estimation of alignments and trees.

an active user community (e.g., Google group pasta-users@googlegroups.com).

2.1 Iterative divide-and-conquer strategy

Each of these methods has the same basic structure. They begin by computing a quick alignment and tree, for example using the fast maximum likelihood (ML) heuristic FastTree-2 [12] on a fast alignment, such as Clustal-Omega [13], and then they iterate between computing a new alignment using the current tree and computing an ML tree on the new alignment. The number of iterations can be selected by the user, or the user can simply run the method until some stopping criterion is met (e.g., the ML score stops improving). The final alignment/tree pair is then returned.

As noted, each iteration uses the tree from the previous iteration to compute a new alignment, and then a new ML tree is computed on the new alignment. The key to computing the new alignment is divide-and-conquer: the current tree is used to decompose the sequences into disjoint subsets, new alignments are computed on the subsets using a selected "subset aligner", and then the subset alignments are merged together into an alignment on the full dataset (Fig. 1a).

The only difference between SATé-I and SATé-II is that SATé-II enables the user to specify how large the subsets can be, and it modified the decomposition strategy so that the subsets do not exceed the specified maximum size; this change enables SATé-II to analyze larger datasets and results in improved accuracy compared to SATé-I. The major difference between SATé-II and PASTA is how the subset alignments are merged into a single alignment on the full dataset (Fig. 1b). The change in the sub-alignment merging strategy (in addition to other smaller changes, such as using a new method to obtain initial alignments) enables PASTA to analyze larger datasets than SATé-II and also improves its accuracy. In fact, PASTA, can compute alignments on up to 1,000,000 sequences, and neither SATé-I nor SATé-II can analyze datasets of this size. Thus, PASTA strictly dominates SATé-I and SATé-II in terms of accuracy, scalability, and speed.

By design, PASTA is fundamentally a method for enabling a selected MSA method to be run only on subsets of bounded size (where the bound is selected by the user). Furthermore, PASTA provides many choices for the subset aligner, including MAFFT [14], Clustal-Omega, Opal [15], Prank [16], and Muscle [17], and additional subset aligner methods for protein sequences (see Note 7).

The rest of this section is described in terms of how to use the PASTA GUI (which is similar to the SATé GUI). However, the command line version of PASTA enables other options than the GUI, and so the advanced users should not restrict themselves to the GUI.

2.2 PASTA parameters

The PASTA GUI, shown in Figure 2, shows the choices that the user has in running PASTA.

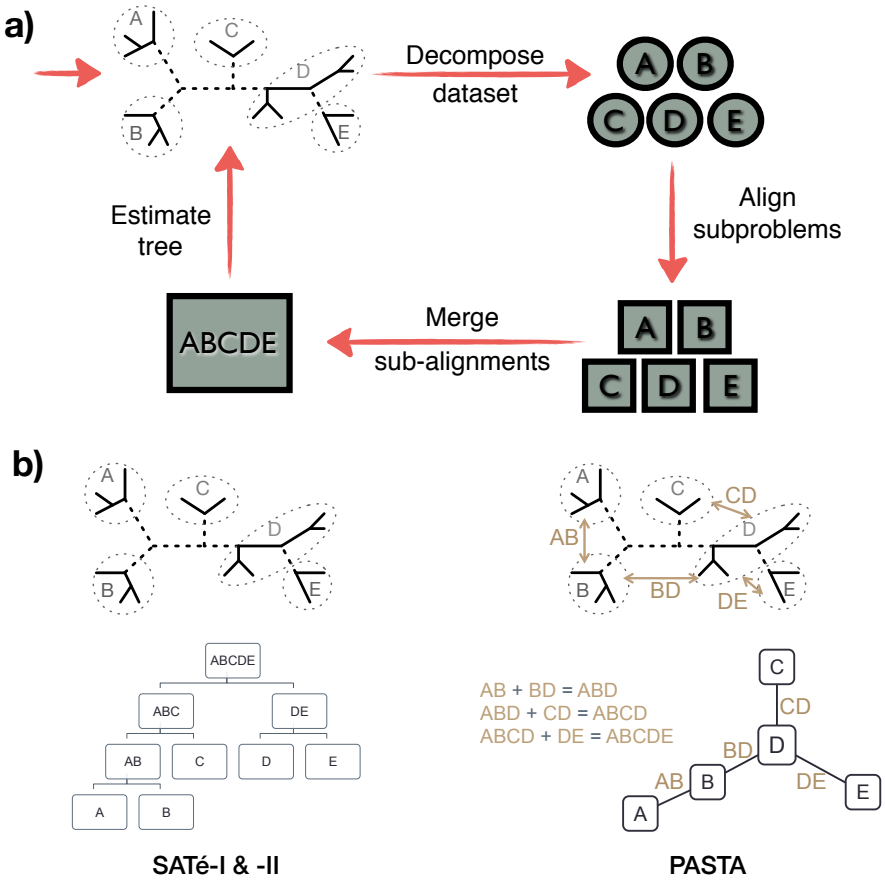

Figure 1: (a) The general divide-and-conquer strategy used in SATé-I, SATé-II, and PASTA. In each iteration, using the current tree, sequences are divided into smaller subsets, each subset is aligned, alignments of subsets are merged, a new tree is inferred, and a new iteration starts. (b) SATé and PASTA differ mainly in how they merge sub-alignments. SATé uses a hierarchical approach, where the hierarchy reflects the tree, and uses external methods like Opal or Muscle to merge alignments. PASTA, on the other hand, uses a spanning tree, computed from the phylogeny, to compute a set of pairwise alignment mergers, which are then combined using transitivity.

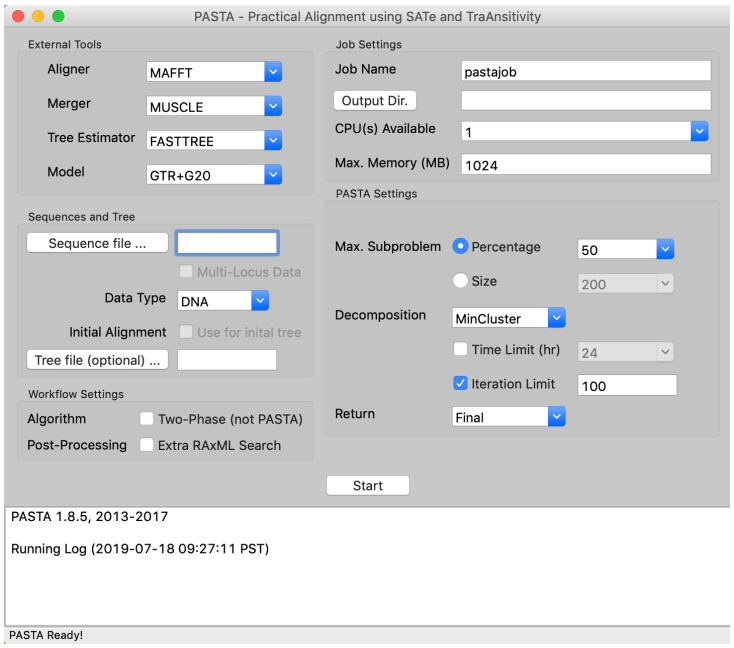

Figure 2: PASTA graphical user interface (GUI). The GUI shows the major algorithmic choices in running PASTA; see Section 2.2. The External Tools determine how subsets are aligned, how these subset alignments are merged, and how trees are computed on the merged alignment in each iteration (which is determined both by the tree estimator method and the sequence evolution model). Sequences and Tree indicate the type of data, and also allow the user to provide an initial alignment and tree. The PASTA Settings specify the maximum size for the subsets, the type of decomposition used in decomposing the dataset into subsets, how many iterations to perform, and whether to return the tree from the last iteration or the tree (from among all the iterations) with the best maximum likelihood score. Finally, Workflow Settings allow the user to just perform a two-phase analysis (first compute an alignment and then a tree) instead of using PASTA, or to run RAxML [18] on the final alignment returned by PASTA.

- **Aligner:** This option specifies the method used to compute alignments on subsets; the default is MAFFT, but other alignment methods are available. See Notes 6 and 7.

- **Merger:** This option allows the user to choose the method to merge pairs of alignments during its approach for combining subset alignments into an alignment on the full dataset; the choice is between Opal and Muscle. The merger technique in PASTA is only used on *pairs* of alignments, and the final alignment is then constructed from these merged pairs using transitivity. See Note 8.

- **Tree Estimator:** This option allows the user to choose between RAxML and FastTree-2, two heuristics for maximum likelihood, for computing trees in each iteration. The default is FastTree-2; see Note 10.

- **Model:** This option specifies the sequence evolution model, but this depends on the data type (RNA, DNA, or protein) as well as the tree estimator (RAxML or FastTree-2); see Notes 11-13.

- **Data Type:** This section allows the user to specify the data type (DNA, RNA, or protein). The default for the data type is DNA, and unless the user specifies otherwise, the analysis will be performed as though the data are DNA. See Note 5.

- **Initial alignment:** This is an optional command that allows the user to provide a pre-computed alignment to PASTA, for use in computing the first tree. See Note 16.

- **Tree file:** This is an optional command that allows the user to provide a pre-computed tree to PASTA, for use in computing the first decomposition and subsequent alignment. See Note 16.

- **Max. Subproblem:** The options here let the user specify the maximum subset size, either as a percentage of the full set of sequences or as a fixed number (i.e., "size"). See Notes 4, 6, and 14.

- **Decomposition:** This option specifies both which edges to remove (MinCluster, centroid edge or longest edge) in computing the decomposition of the sequences into subsets (the default is MinCluster) and how many iterations to perform (either a specific number of iterations or a maximum amount of time). The MinCluster decomposition, which minimizes the number of subsets with a bounded size [19], began with version 1.8.0, and it further improves alignment accuracy.

- **Return:** This option allows the user to decide whether to return the tree from the last iteration or the tree with the best maximum likelihood score of all the trees from all the iterations. See Note 15.

The most important considerations in running PASTA are (1) what alignment method to use to compute alignments on subsets, (2) how small to make the subsets, and (3) how many iterations to run. A good default for the subset aligner is MAFFT [14]. When MAFFT is used to align subsets, then limiting the subsets to 200 sequences makes it feasible to run the more computationally intensive variants of MAFFT (such as MAFFT L-INS-i and MAFFT G-INS-i), which improves accuracy; reducing the maximum subset size will tend to reduce the running time while increasing the maximum subset size will tend to increase the running time (see Note 6 for a discussion about the impact on accuracy). Other methods, such as BAli-Phy [20, 21], can also be used to align subsets, as shown in [22].

The number of iterations is also important, and previous studies have shown that accuracy improves substantially in the first few iterations, and then alignment and tree accuracy seem to stabilize. While three iterations seems to be sufficient for high accuracy in most conditions, it seems possible that more iterations could improve accuracy for challenging datasets. However, increasing the number of iterations also increases the running time. Hence, this is an issue that involves a potential trade-off between time and accuracy; see Note 3.

The tree estimation method used in PASTA within each iteration also impacts accuracy, and the default is FastTree-2. Most users will prefer to use other methods than FastTree-2 for the final tree, and PASTA enables the user to perform a final RAxML analysis on the final tree. This is advisable whenever the dataset is not so large that RAxML is infeasible. See Notes 3, 10, and 13.

2.3 PASTA output

PASTA produces both an alignment and a tree. In addition to the final alignment and tree, PASTA outputs alignments and trees generated in each iteration as temporary files. It also outputs a config file recording all the settings used.

For large datasets (many thousands of sequences), PASTA tends to produce very long and gappy alignments because it is conservative in inferring homologies. The gappy alignments (partially a natural consequence of large datasets and partially a consequence of the algorithmic design of PASTA) seem to not hurt PASTA's ability to produce very accurate trees, but the alignments will certainly look strange to some users (see discussion in [23] about the preference among some users for less gappy alignments). Furthermore, whether accurate or not, these gappy alignments can also cause difficulties in some subsequent analyses. For example, phylogenetic inference can become slow given long alignments, and the inclusion of gappy sites may not result in improved phylogenetic accuracy. To speed up the tree estimation stage within each iteration, PASTA alignments are first masked to remove all sites that are at least 99.9% gapped, before trees are computed. This default setting for masking sites inside PASTA can be modified using the `--mask-gappy-sites` option. Removing gappy sites from the final alignment generated by PASTA can be done using the `run_seqtools.py` script, which is packaged with and installed together with PASTA. More aggressive filtering would further reduce the running time for phylogenetic inference,

but could also have negative consequences for accuracy; see discussion in [24].

3 UPP

As we have found, PASTA produces highly accurate alignments and trees, and improves on the accuracy of other methods for ultra-large datasets with high rates of heterogeneity. However, when the input dataset has many fragmentary sequences, then PASTA does not have good accuracy. Furthermore, no standard alignment method has good accuracy when fragments are included. However, UPP [5] is an alternative approach that has good accuracy, and is the focus of this section.

3.1 Ensembles of Profile Hidden Markov Models (HMMs)

UPP builds on PASTA to improve its ability to align datasets with fragmentary sequences using the "ensembles of HMMs" technique, which we now describe in the context of working with multiple sequence alignments.

A profile Hidden Markov Model (HMM) [25] is a probabilistic graphical model that has vertices and directed edges, with a single vertex for the start state, a single vertex for the end state, and additional vertices corresponding to match states, insertion states, and deletion states. With the exception of the insertion states (which can have self-loops), there are no directed cycles in a profile HMM. Each directed edge $e = v \rightarrow w$ in the profile HMM is annotated with a real number p_e where p_e is the probability of moving from v to w. Finally, the insertion states and match states emit letters (e.g., nucleotides or amino acids) from a probability distribution. Thus, when tracing a path through a profile HMM, and selecting the letters to be emitted by the visited match and insertion states, a sequence is produced.

Profile HMMs are a major part of many bioinformatics analyses, and one of the interesting uses is to add sequences into multiple sequence alignments. In what follows, we describe how profile Hidden Markov Models can be used specifically for multiple sequence alignment; see [26] for additional details and discussion.

To add a sequence s into a multiple sequence alignment A, a profile HMM is built for A, and then an optimal path (e.g., a maximum likelihood path) through the model is found for s. Once the path is found, it defines a way of adding s into the alignment A. Note that this addition does *not* define an alignment between A and those letters in s that are mapped to insertion states. Thus, when using HMMs to extend A to include s, some parts of s may remain *unaligned*. Besides finding the best alignment, given the sequence s and a profile HMM, the fit between the profile HMM and s can be calculated in various ways, including finding the overall probability that the profile HMM would generate s. The HMMER3 [27] suite of tools provides a particular implementation of the general profile HMM concept and includes many further optimizations, both for accuracy and speed. HMMER3 includes tools for all these analyses (i.e.,

building profile HMMs from alignments, scoring the fit between a profile HMM and a sequence, and finding the best path through the model for the sequence) [27, 28].

An *ensemble of profile HMMs* is a collection of profile HMMs that are built using a multiple sequence alignment A, with each profile HMM in the set based on just a subset of the sequences in the set. Thus, the match states in each of the profile HMMs in the set correspond to sites in A. Now, given a sequence s, the profile HMM in the collection that has the best fit to s can be found, the best path through the model can be computed, and thus the sequence s can be added to the alignment A. Thus, an ensemble of profile HMMs can also be used to represent the alignment A and then used to add new sequences to A. Here we will show how UPP uses an ensemble of profile HMMs to compute multiple sequence alignments, noting also that ensembles of HMMs have been used for phylogenetic placement [6], taxonomic identification of metagenomic data [7], and classification of protein sequences into families and superfamilies [8].

3.2 UPP's algorithmic protocol

In essence, UPP is a combination of PASTA (which it uses to construct a multiple sequence alignment on a subset of the input sequences) with a way of computing an ensemble of profile HMMs, which it then uses to add the remaining sequences into the PASTA alignment. Here we describe this process as operating in four steps. The first two steps can be omitted if the user wishes to provide UPP with a pre-computed backbone alignment and tree; see Note 16.

- **Step 1:** Given a set S of unaligned sequences, UPP begins by identifying those sequences to be part of "backbone alignment". This is performed first by restricting S to just those sequences with length within 25% of the median sequence length, and then randomly selecting a set of sequences from that set. The number of sequences in the backbone and restrictions on what sequences can be included in the backbone can be modified by using a configuration file or input options (`-B`, `-M`, `-T`, and `-l`).

- **Step 2:** UPP uses PASTA to compute a multiple sequence alignment A' and tree T on S', which are then referred to as the backbone alignment and backbone tree.

- **Step 3:** UPP builds an ensemble of profile HMMs to represent the multiple sequence alignment A' on S': it uses the tree T to break the set of sequences into disjoint subsets of bounded size (using the same centroid edge decomposition as in SATé-II), and then computes a profile HMM for each of the subsets (i.e., for the rows of the alignment A' defined by the sequences in the subset). The set of profile HMMs it creates is the ensemble of profile HMMs used in the next step.

- **Step 4:** The remaining sequences (i.e., the ones that are not in the backbone alignment) are added to A' using the ensemble of profile Hidden

Markov Models computed in Step 3, thus producing a multiple sequence alignment A on S.

As shown in [5], UPP produces more accurate alignments than PASTA and other multiple sequence alignment methods when the input set S has many fragmentary sequences, and trees computed on the alignment are more accurate than trees computed on the other alignments in the presence of fragmentation.

3.3 UPP's parameters

The most important algorithmic options in using UPP are (a) which sequences to put in the backbone subset S', (b) which method to use to compute an alignment on S', and (c) which algorithmic parameters to use for building the ensemble of profile Hidden Markov Models.

For which sequences to put in S', there are two decisions that need to be made: first, which sequences are close enough to full length to be considered, and second, how many of these sequences to use for the backbone alignment. The default UPP operates as follows: it computes the median sequence length of the input sequences and considers any sequence within 25% of this length to be "full-length". Then, UPP selects a random subset of the "full-length" sequences to include in the backbone alignment, with the default setting for the size of this set being the minimum of $\{1000, N\}$, where N is the number of "full-length" sequences. Changing the number of sequences to put in the backbone set can affect accuracy and running time, and is discussed in Note 17.

For how to compute the backbone alignment, the default is to use PASTA, and this is certainly appropriate when S' is large. However, when S' is small enough, then other methods can potentially provide improved accuracy compared to PASTA. For example, BAli-Phy [20, 21] and other statistical methods could be used to compute an alignment A' on S'. Once the backbone alignment is built, a backbone tree is also needed, which can be estimated using fast ML heuristics, such as FastTree-2 (e.g., as outputted by PASTA).

There are several algorithmic options for building the ensemble of profile HMMs on A', which we briefly discuss here. Recall that an ensemble of profile HMMs is a collection of profile HMMs, where each of the profile HMMs is constructed on a subset of the sequences in the backbone alignment. To add a sequence s into the backbone alignment, s is scored with respect to each profile HMM in the collection, and the profile HMM with the best score is selected. Thus, every sequence s that is not in the backbone alignment must be scored against every profile HMM in the collection. Although we observe that typically accuracy is increased by having a large number of profile HMMs, this also increases the running time. Thus, there is a potential trade-off between accuracy and running time. The default in UPP produces 10 profile HMMs, which provides an improvement over a single profile HMM and (obviously) also increases the running time. See Note 18 for additional considerations for this algorithmic setting.

Although modifications to the default settings can result in improved accuracy or speed, the default settings for UPP are sufficient to improve on PASTA if the proportion of fragmentary sequences is large enough. Detailed information on how to adjust the settings of UPP are given in its README file at `https://github.com/smirarab/sepp/blob/master/README.UPP.md`.

4 Discussion and Summary

UPP and PASTA are two methods for large-scale multiple sequence alignment that provide improved accuracy over standard methods when datasets are large and heterogeneous. UPP provides a specific advantage over PASTA when the dataset has fragmentary sequences and PASTA provides advantages when all the sequences are full length. UPP and PASTA are available in open source form in order to encourage further development by the research community. Furthermore, each method is designed to improve scalability of MSA methods, which are run only on subsets of the input sequence set. Therefore, as new MSA methods are developed, PASTA and UPP can be extended to use these new methods.

PASTA is described here as a method for co-estimating alignments and trees, but it is not a statistical co-estimation method in the sense that BAli-Phy [20, 21] and StatAlign [29] are. However, PASTA can run on very large datasets while truly statistical co-estimation methods are limited to fairly small datasets (perhaps 100 sequences). Furthermore, PASTA and UPP have been used with BAli-Phy to compute subset alignments [22], thus enabling BAli-Phy to scale (in some sense) to very large datasets (e.g., up to 10,000 sequences!).

Although the discussion here was largely based on using PASTA within the GUI, the command line version enables additional settings that can provide improved accuracy. This was intentional, as the GUI is the easiest way to become familiar with PASTA, and the GUI version provides the same advantages as the command line version over other methods on large datasets. However, advanced users should use the command line version, which allows the algorithmic settings to be modified in additional ways.

We set out to discuss multiple sequence alignment for the purpose of tree estimation. PASTA is specifically designed to co-estimate alignments and trees (in an iterative fashion), so that the final tree is produced by running a maximum likelihood heuristic (either RAxML or FastTree-2) on the final alignment. Some consideration, therefore, should be made for how to compute trees from these improved alignments. While we focused on maximum likelihood under standard sequence evolution models, other approaches could be used, including Bayesian estimation (e.g., MrBayes [30] and BEAST [31, 32]), distance-based estimation (e.g., FastME [33]), and parsimony analyses (e.g., TNT [34] and PAUP* [35]). Bayesian or maximum likelihood analyses under non-standard sequence evolution models may also be necessary, especially for datasets that span large evolutionary distances where violations of the usual model assumptions (stationarity, time reversibility, and homogeneity) are likely to occur [36, 37, 38]. Divide-and-

conquer phylogeny estimation, where the set of species is divided into smaller, more homogeneous subsets, and then trees on the subsets are computed and combined into a tree on the full dataset (e.g., DACTAL [39], constrained-INC [40, 41], NJMerge [42], and TreeMerge [43]), may provide an improvement in tree accuracy for those datasets that violate the standard model assumptions but are too large for methods that are based on more complex models.

Finally, although UPP was able to produce better alignments than PASTA for datasets with a high number of fragmentary sequences, the construction of trees from such datasets presents additional challenges, even given error-free alignments [44]. One possible direction is to use phylogenetic placement, where an initial tree is built using the full-length sequences and then the fragmentary sequences are added to the tree [6], but other approaches may provide better accuracy. Thus, tree estimation on large heterogeneous datasets will need to be revisited, in order to achieve the goal of accurate inference of large phylogenies.

5 Notes

We now give some high-level advice on using PASTA and UPP. The reader will benefit from consulting the GitHub sites for these methods (and in particular the tutorials and READMEs at those sites). PASTA users should also read the Notes section in [9] for advice about using PASTA (which is built on the SATé codebase, so that much of the advice for SATé is relevant to PASTA).

Notes for PASTA. PASTA utilizes FASTA-formatted sequence files and Newick-formatted tree files. See the PASTA README for details about allowed characters in the input data.

1. If you have a MAC, then installing PASTA by downloading the MAC application .dmg from the GitHub site is easy, but it only allows you to use the GUI (which is not always the most up-to-date version of PASTA). If you prefer to use the command line or do not have a MAC, you will need to install PASTA using some other process. The PASTA GitHub site provides details on how to do these installations, and the PASTA users group can help with installation issues.

2. PASTA has been mainly developed and tested for Linux and MAC; as a result, Windows users will generally have more difficulty and will need to rely on virtualization (through virtual images or docker images provided on the website).

3. PASTA uses iteration as well as divide-and-conquer to improve alignment accuracy compared to standard MSA methods. The main algorithmic parameters (i.e., how small to make the subsets, how to compute subset alignments, how to compute trees on the alignments and which sequence evolution models to use) impact the accuracy that can be obtained in each iteration, but also impact running time. In general, our recommendation

is to use the best method you can afford to run that still allows PASTA to perform at least three iterations (and more iterations, when time permits). This will allow the alignment produced by PASTA to have very good accuracy, and a final tree can then be computed on the PASTA alignment using more computationally intensive tree estimation methods. Much of the discussion below about how to set the algorithmic parameters reflects this point. Similarly, if desired, the final alignment/tree pair produced by PASTA can be given as input to PASTA (see Sequences and Tree in the PASTA GUI, in Figure 2), if additional PASTA iterations using more computationally intensive approaches are desired.

4. PASTA and SATé were designed to enable improved accuracy on large datasets, but they have also been used to compute alignments on small datasets (e.g., the avian datasets in [45] with fewer than 50 taxa). The PASTA default setting automatically adjusts the subset size appropriately for small datasets. For example, on sufficiently small datasets, PASTA may set the maximum subset size to as much as 50% of the number of sequences in the input.

5. PASTA (in command line mode) does not automatically detect the data type (DNA, RNA, or proteins), and the default setting is DNA. Therefore, if your data are not DNA sequences, you should make sure to specify the type explicitly, as otherwise the behavior of PASTA can be unpredictable (and the resultant alignment and tree may have poor accuracy).

6. As mentioned above, MAFFT is the default technique for aligning subsets, and works well for both proteins and nucleotides. However, when aligning proteins, other subset alignment methods can also have good accuracy, and are enabled in [46]. As mentioned earlier, when MAFFT is used to align subsets, limiting the subsets to 200 sequences makes it feasible to run the most accurate (but also most computationally intensive) variants of MAFFT, such as MAFFT L-INS-i and MAFFT G-INS-i. Changing the subset size will change the final alignment. Our studies (published and unpublished) have revealed inconsistent trends regarding the impact of the subset size parameter. However, at this time, based on the preponderance of the evidence, we suggest using the default settings, which puts the alignment subset size at 200, when using MAFFT as the subset aligner. The interested user may wish to explore the impact of changing alignment subset size, for those datasets that are small enough to allow such exploratory data analysis.

7. For protein alignment, PASTA enables the use of additional subset aligners MAFFT-G-INS-i, MAFFT-homologs, CONTRAlign (version 1) [47], and PROBCONS [48, 49]. To use MAFFT-homologs and CONTRAlign (available only in command line), the user must take additional steps during installation, as detailed in the most up to data README file. If you wish to use MAFFT-Homologs as the subset aligner, you should use the version of PASTA available at [46].

8. PASTA allows two methods for merging pairs of alignments–Opal and Muscle. The choice between the two methods does not have a large impact on accuracy, provided that the subsets are not too small (because when the subsets are very small, then the returned alignment is largely based on the technique used to merge pairs of alignments).

9. Although the default setting for PASTA sets the number of iterations to three, additional iterations could lead to improved accuracy under some conditions. In general, using additional iterations has shown some improvement in tree and alignment accuracy, but the optimal number of iterations is an under-explored topic. We therefore recommend that the user consider enabling additional iterations, when time permits, and explore the set of alignment/tree pairs that are returned in these iterations.

10. PASTA has two methods for tree estimation that are used in each iteration. The default is FastTree-2, but RAxML is also allowed. In our experience, RAxML is much more computationally intensive than FastTree-2, making FastTree-2 a better choice on large datasets, since many iterations can be run if FastTree-2 is used instead of RAxML (see previous Note). When the number of sequences is small enough, then adding a final RAxML run (with the post-processing command in the GUI) is recommended, since RAxML generally produces better ML scores and can, in some conditions, improve the tree accuracy (although there are many conditions where the improvement in ML score does not correspond to an improvement in tree topology accuracy [50]). However, when the number of sequences is large then we do not recommend having PASTA automatically perform a RAxML analysis on the final alignment, as this can be too computationally intensive. Instead, for very large datasets, we recommend the following approach: let PASTA perform its iterations using FastTree-2, save the final PASTA alignment, and then separately compute a tree on the final PASTA alignment using the preferred software (e.g., RAxML, or potentially some other method) and selected sequence evolution model. In this way, PASTA can be used to produce a highly accurate alignment, and then the best tree accuracy (and associated numeric parameters) can be obtained using a separate tree estimation phase.

11. The set of possible sequence evolution models depends on the tree estimation method (RAxML or FastTree-2) and the type of data (nucleotides or proteins). When PASTA is used with FastTree-2, only a very limited number of models are available (described in the notes below). If the user wishes to select a model for PASTA, then they should obtain a preliminary alignment and then use external software (e.g., ProtTest [51] for protein datasets and ModelTest [52] or PLTB [53] for nucleotide datasets). However, an alternative approach can also be used: the user can run PASTA using the default model, then use the resultant alignment with the external software to select a substitution model for a final round of tree inference

or potentially another iteration of PASTA (using the new model). See discussion in [26] about selecting models for phylogeny estimation.

12. To select a nucleotide sequence evolution model within PASTA, FastTree-2 and RAxML both enable the Generalized Time Reversible (GTR, [54]) model, and each can be used with a selected model for rate variation across sites (with different models depending on what tree estimation method is selected). In addition, FastTree-2 enables the use of the Jukes-Cantor (JC, [55]) model (with two models for rate variation across sites); however, we do not recommend using the JC model unless the data seem to fit the JC model well. FastTree-2 only enables two types of rate variation across sites (CAT and G20, which is an approximation of gamma distributed rates with 20 categories), but RAxML enables rate variation models that include invariable sites. The choice of rate variation model can also impact accuracy, but the more complex models are also more computationally intensive. However, our studies suggest that using simple sequence evolution models within the iterative process may not reduce the alignment accuracy substantially, and a new tree can be estimated on the final alignment using more complex models.

13. For protein alignment, the two tree estimation methods, RAxML and FastTree-2, offer very different sequence evolution models. Specifically, FastTree-2 only offers two protein substitution models (JTT and WAG), each with two site variation models, and RAxML offers 11 protein substitution models, each with four site variation models. Thus, RAxML allows a larger set of protein sequence evolution models than FastTree-2, making RAxML a better method for computing trees than FastTree-2 for proteins. However, here too the benefit from using RAxML within the iterative process may be offset by the extra time used to compute trees with RAxML. Hence, we would suggest instead that FastTree-2 be used as the tree estimation method within the iterative procedure, even for protein sequences. Then, after the PASTA alignment is computed, the user can compute a new tree on the alignment using RAxML or some other software, under the best fitting model.

14. If you wish to use BAli-Phy to align subsets within PASTA, the maximum subset size and the running time for each subset alignment need to be set so that BAli-Phy is able to converge on each subset. This is discussed in [56], and the software for PASTA using BAli-Phy is available at [57]. However, see [58] for a study comparing BAli-Phy and other alignment methods on protein benchmark datasets, which showed differences between performance on simulated and biological datasets.

15. The choice of which alignment/tree pair to return (i.e., whether to return the pair produced in the final iteration, or the pair that has the best maximum likelihood score) is an interesting one. In general, we expect little difference in accuracy between the two options, and so the choice may

not make much difference in practice. In addition, there is no theoretical basis on which to select the pair that has the best maximum likelihood score [2], since the alignment is allowed to change. For these reasons, and also because the ML score is calculated within PASTA on masked versions of the computed alignments, the default in PASTA is the final alignment/tree pair.

Notes for UPP.

16. The user can provide UPP with a pre-computed backbone alignment and tree (referred to in the UPP tutorial as a "custom seed alignment and tree"); this is a natural approach when using alignments and trees obtained from external sources (such as PFAM [59]) or when alignments and trees have been estimated using additional information (such as secondary or tertiary structure) or by specialized methods not available within UPP.

17. UPP uses PASTA to compute its backbone alignment and tree, but the selection of which sequences are put into the backbone set can be controlled by the user. In the default mode, UPP operates as follows: it computes the median sequence length of the input sequences and considers any sequence within 25% of this length to be "full-length"; the user can modify this approach as needed using options `-M` and `-T`. Once that set of full-length sequences is determined, the user can specify how many of the sequences to include in the backbone alignment using the `-B` option. The default is to take the minimum of $\{1000, N\}$, where N is the number of "full-length" sequences. However, another option is to include all of the full-length sequences (even when this is more than 1000); in our experience, this improves accuracy but may also increase the running time. Furthermore, reducing the number of sequences, even to as low as just 100 (the UPP-fast version), produces a reduction in accuracy (but sometimes only a small reduction, which depends on the heterogeneity in the input dataset) and a dramatic reduction in running time. Hence, there is a potential tradeoff between accuracy and running time, that needs to be considered in building the ensemble.

18. The default mode for UPP is to create an ensemble of HMMs that has ten (10) profile HMMs. However, changes to this number can be considered with a potential for improved accuracy. In particular, when the input set is highly heterogeneous (as represented by low average sequence similarity), then using a larger number of profile HMMs can improve accuracy; however, the benefit in increasing the number of profile HMMs is reduced when the dataset has high average sequence similarity. Furthermore, increasing the number of profile HMMs automatically increases the running time (as it scales linearly with this number).

Finally, we add a point relevant to both methods.

19. Errors in the input unaligned sequence data have the potential to reduce the accuracy of the alignment. One way to detect such errors is to use automated methods such as TreeShrink [60]. TresShrink looks for extremely long branches in the phylogeny to detect potential errors in the data. Thus, TreeShrink can be combined with PASTA in a natural way: remove sequences on long branches from the PASTA alignment and tree (implemented in the `--treeshrink-filter` option), recompute the PASTA alignment, and add back those potentially problematic sequences using UPP. In addition, UPP allows for sequences that are on very long branches to be removed from the backbone set (see `-l`).

6 Additional resources

6.1 Websites for PASTA

- PASTA is available in open source form on GitHub at [10] and a protein version is available at [46]. The software is developed under the GNU Public License (GPL).
- A version of PASTA for use with BAli-Phy is available at [57].
- All questions and inquires should be addressed to our user email group: pasta-users@googlegroups.com, with posts available at `https://groups.google.com/forum/#!forum/pasta-users`.
- A PASTA tutorial is available at `https://github.com/smirarab/pasta/blob/master/pasta-doc/pasta-tutorial.md`.

6.2 Websites for UPP

- The UPP software is available in open source form on GitHub at [11], and is part of the SEPP [6] distribution (which has code for various methods that use ensembles of profile HMMs). UPP is available as Python code.
- A tutorial on UPP is available at `https://github.com/smirarab/sepp/blob/master/tutorial/upp-tutorial.md`
- The UPP users group forum is available at `https://groups.google.com/forum/#!forum/ensemble-of-hmms`.

7 Acknowledgments

This paper was supported by NSF grant ABI-1458652 to TW and NSF grant IIS-1845967 to SM.

References

[1] K. Liu, S. Raghavan, S. Nelesen, C. R. Linder, and T. Warnow, "Rapid and accurate large-scale coestimation of sequence alignments and phylogenetic trees," *Science*, vol. 324, no. 5934, pp. 1561–1564, 2009.

[2] K. Liu, T. Warnow, M. T. Holder, S. M. Nelesen, J. Yu, A. P. Stamatakis, and C. R. Linder, "SATé-II: very fast and accurate simultaneous estimation of multiple sequence alignments and phylogenetic trees," *Syst Biol*, vol. 61, pp. 90–106, Jan. 2012.

[3] S. Mirarab, N. Nguyen, and T. Warnow, "PASTA: ultra-large multiple sequence alignment," in *International Conference on Research in Computational Molecular Biology*, pp. 177–191, Springer, 2014.

[4] S. Mirarab, N. Nguyen, L.-S. Wang, S. Guo, J. Kim, and T. Warnow, "PASTA: ultra-large multiple sequence alignment of nucleotide and amino acid sequences," *J. Computational Biology*, vol. 22, pp. 377–386, 2015.

[5] N. Nguyen, S. Mirarab, K. Kumar, and T. Warnow, "Ultra-large alignments using phylogeny aware profiles," *Genome Biology*, vol. 16, no. 124, 2015. A preliminary version appeared in the Proceedings RECOMB 2015.

[6] S. Mirarab, N. Nguyen, and T. Warnow, "SEPP: SATé-enabled phylogenetic placement," in *Pacific Symposium on Biocomputing*, pp. 247–58, 2012.

[7] N. Nguyen, S. Mirarab, B. Liu, M. Pop, and T. Warnow, "TIPP: taxonomic identification and phylogenetic profiling," *Bioinformatics*, vol. 30, no. 24, pp. 3548–3555, 2014.

[8] N. Nguyen, M. Nute, S. Mirarab, and T. Warnow, "HIPPI: highly accurate protein family classification with ensembles of hidden Markov models," *BMC Bioinformatics*, vol. 17 (Suppl 10), p. 765, 2016.

[9] K. Liu and T. Warnow, "Large-scale multiple sequence alignment and tree estimation using SATé," in *Multiple Sequence Alignment Methods*, pp. 219–244, Springer, 2014.

[10] S. Mirarab, "Github site for PASTA software." `https://github.com/smirarab/pasta`. Accessed July 13, 2019.

[11] S. Mirarab, "Github site for Ensemble of HMM methods (SEPP, TIPP, UPP) software." `https://github.com/smirarab/sepp`. Accessed July 13, 2019.

[12] M. N. Price, P. S. Dehal, and A. P. Arkin, "FastTree 2 – approximately maximum-likelihood trees for large alignments," *PLoS ONE*, vol. 5, no. 3, p. e9490. doi:10.1371/journal.pone.0009490, 2010.

[13] F. Sievers, A. Wilm, D. Dineen, T. J. Gibson, K. Karplus, W. Li, R. Lopez, H. McWilliam, M. Remmert, J. Soding, J. D. Thompson, and D. G. Higgins, "Fast, scalable generation of high-quality protein multiple sequence alignments using Clustal Omega," *Mol. Syst. Biol.*, vol. 7, 2011.

[14] K. Katoh and H. Toh, "Recent developments in the MAFFT multiple sequence alignment program," *Brief Bioinf*, vol. 9, no. 4, pp. 286–298, 2008.

[15] T. Wheeler and J. Kececioglu, "Multiple alignment by aligning alignments," in *Proceedings of the 15th ISCB Conference on Intelligent Systems for Molecular Biology*, pp. 559–568, 2007.

[16] A. Löytynoja and N. Goldman, "An algorithm for progressive multiple alignment of sequences with insertions," *Proc Nat Acac Sci*, vol. 102, pp. 10557–10562, 2005.

[17] R. C. Edgar, "MUSCLE: a multiple sequence alignment method with reduced time and space complexity," *BMC Bioinformatics*, vol. 5, no. 113, p. 113, 2004.

[18] A. Stamatakis, "RAxML-VI-HPC: Maximum likelihood-based phylogenetic analyses with thousands of taxa and mixed models," *Bioinformatics*, vol. 22, pp. 2688–2690, 2006.

[19] M. Balaban, N. Moshiri, U. Mai, and S. Mirarab, "TreeCluster: clustering biological sequences using phylogenetic trees," *bioRxiv*, 2019. doi:10.1101/591388.

[20] M. A. Suchard and B. D. Redelings, "BAli-Phy: simultaneous Bayesian inference of alignment and phylogeny," *Bioinformatics*, vol. 22, pp. 2047–2048, 2006.

[21] B. D. Redelings and M. A. Suchard, "Incorporating indel information into phylogeny estimation for rapidly emerging pathogens," *BMC Evol Biol*, vol. 7, p. 40, 2007.

[22] M. Nute and T. Warnow, "Scaling statistical multiple sequence alignment to large datasets," *BMC Genomics*, vol. 17, p. 764, Nov 2016.

[23] A. Löytynoja and N. Goldman, "Phylogeny-aware gap placement prevents errors in sequence alignment and evolutionary analysis," *Science*, vol. 320, no. 5883, pp. 1632–1635, 2008.

[24] G. Tan, M. Muffato, C. Ledergerber, J. Herrero, N. Goldman, M. Gil, and C. Dessimoz, "Current methods for automated filtering of multiple sequence alignments frequently worsen single-gene phylogenetic inference," *Systematic Biology*, vol. 64, no. 5, pp. 778–791, 2015.

[25] R. Durbin, S. Eddy, A. Krogh, and G. Mitchison, *Biological Sequence Analysis*. Cambridge University Press, 1998.

[26] T. Warnow, *Computational phylogenetics: An introduction to designing methods for phylogeny estimation*. Cambridge: Cambridge University Press, 2018.

[27] S. R. Eddy, "A new generation of homology search tools based on probabilistic inference," *Genome Inform*, vol. 23, pp. 205–211, 2009.

[28] R. D. Finn, J. Clements, and S. R. Eddy, "HMMER web server: interactive sequence similarity searching," *Nucleic Acids Research*, vol. 39, pp. W29–W37, 2011.

[29] Á. Novák, I. Miklós, R. Lyngsoe, and J. Hein, "StatAlign: an extendable software package for joint Bayesian estimation of alignments and evolutionary trees," *Bioinformatics*, vol. 24, pp. 2403–2404, 2008.

[30] J. Huelsenbeck and R. Ronquist, "MrBayes: Bayesian inference of phylogeny.," *Bioinf*, vol. 17, pp. 754–755, 2001.

[31] A. Drummond and A. Rambaut, "BEAST: Bayesian evolutionary analysis by sampling trees," *BMC Evolutionary Biology*, vol. 7, p. 214, 2007.

[32] R. Bouckaert, J. Heled, D. Kühnert, T. Vaughan, C.-H. Wu, D. Xie, M. A. Suchard, A. Rambaut, and A. J. Drummond, "BEAST 2: a software platform for bayesian evolutionary analysis," *PLoS computational biology*, vol. 10, no. 4, p. e1003537, 2014.

[33] V. Lefort, R. Desper, and O. Gascuel, "FastME 2.0: a comprehensive, accurate, and fast distance-based phylogeny inference program," *Molecular Biology and Evolution*, vol. 32, no. 10, pp. 2798–2800, 2015.

[34] P. Goloboff, J. Farris, and K. Nixon, "TNT, a free program for phylogenetic analysis," *Cladistics*, vol. 24, pp. 1–13, 2008.

[35] D. L. Swofford, "PAUP*: Phylogenetic analysis using parsimony (and other methods)," 1996. Sinauer Associates, Sunderland, Massachusetts, Version 4.0.

[36] S. Naser-Khdour, B. Q. Minh, W. Zhang, E. Stone, and R. Lanfear, "The prevalence and impact of model violations in phylogenetics," *BioRxiv*, 2019. doi:10.1101/460121.

[37] S. M. Crotty, B. Q. Minh, N. G. Bean, B. R. Holland, J. Tuke, L. S. Jermiin, and A. v. Haeseler, "GHOST: Recovering historical signal from heterotachously-evolved sequence alignments," *bioRxiv*, 2019. doi:10.1101/174789.

[38] L. S. Jermiin, R. A. Catullo, and B. R. Holland, "A new phylogenetic protocol: Dealing with model misspecification and confirmation bias in molecular phylogenetics," *bioRxiv*, 2018. doi:10.1101/400648.

[39] S. Nelesen, K. Liu, L.-S. Wang, C. R. Linder, and T. Warnow, "DACTAL: divide-and-conquer trees (almost) without alignments," *Bioinformatics*, vol. 28, pp. pages i274–i282, 2012.

[40] Q. Zhang, S. Rao, and T. Warnow, "Constrained incremental tree building: new absolute fast converging phylogeny estimation methods with improved scalability and accuracy," *Algorithms for Molecular Biology*, vol. 14, no. 1, p. 2, 2019.

[41] T. Le, A. Sy, E. K. Molloy, Q. R. Zhang, S. Rao, and T. Warnow, "Using INC within divide-and-conquer phylogeny estimation," in *International Conference on Algorithms for Computational Biology*, pp. 167–178, Springer, 2019.

[42] E. K. Molloy and T. Warnow, "NJMerge: a generic technique for scaling phylogeny estimation methods and its application to species trees," in *RECOMB International conference on Comparative Genomics*, pp. 260–276, Springer, 2018.

[43] E. K. Molloy and T. Warnow, "TreeMerge: a new method for improving the scalability of species tree estimation methods," *Bioinformatics*, vol. 35, no. 14, pp. i417–i426, 2019.

[44] E. Sayyari, J. B. Whitfield, and S. Mirarab, "Fragmentary gene sequences negatively impact gene tree and species tree reconstruction," *Molecular biology and evolution*, vol. 34, no. 12, pp. 3279–3291, 2017.

[45] E. Jarvis, S. Mirarab, A. J. Aberer, B. Li, P. Houde, C. Li, S. Ho, B. C. Faircloth, B. Nabholz, J. T. Howard, A. Suh, C. C. Weber, R. R. da Fonseca, J. Li, F. Zhang, H. Li, L. Zhou, N. Narula, L. Liu, G. Ganapathy, B. Boussau, M. S. Bayzid, V. Zavidovych, S. Subramanian, T. Gabaldón, S. Capella-Gutiérrez, J. Huerta-Cepas, B. Rekepalli, K. Munch, M. Schierup, B. Lindow, W. C. Warren, D. Ray, R. E. Green, M. W. Bruford, X. Zhan, A. Dixon, S. Li, N. Li, Y. Huang, E. P. Derryberry, M. F. Bertelsen, F. H. Sheldon, R. T. Brumfield, C. V. Mello, P. V. Lovell, M. Wirthlin, M. P. C. Schneider, F. Prosdocimi, J. A. Samaniego, A. M. V. Velazquez, A. Alfaro-Núnez, P. F. Campos, B. Petersen, T. Sicheritz-Ponten, A. Pas, T. Bailey, P. Scofield, M. Bunce, D. M. Lambert, Q. Zhou, P. Perelman, A. C. Driskell, B. Shapiro, Z. Xiong, Y. Zeng, S. Liu, Z. Li, B. Liu, K. Wu, J. Xiao, X. Yinqi, Q. Zheng, Y. Zhang, H. Yang, J. Wang, L. Smeds, F. E. Rheindt, M. Braun, J. Fjeldsa, L. Orlando, F. K. Barker, K. A. Jonsson, W. Johnson, K.-P. Koepfli, S. O'Brien, D. Haussler, O. A. Ryder, C. Rahbek, E. Willerslev, G. R. Graves, T. C. Glenn, J. McCormack, D. Burt, H. Ellegren, P. Alstrom, S. V. Edwards, A. Stamatakis, D. P. Mindell, J. Cracraft, E. L. Braun, T. Warnow, W. Jun, M. T. P. Gilbert, and G. Zhang, "Whole-genome analyses resolve early branches in the tree of life of modern birds," *Science*, vol. 346, no. 6215, pp. 1320–1331, 2014.

[46] K. Collins, "PASTA for proteins github site." `https://github.com/kodicollins/pasta-databases`.

[47] C. B. Do, S. S. Gross, and S. Batzoglou, "CONTRAlign: discriminative training for protein sequence alignment," in *Proceedings of the Tenth Annual International Conference on Computational Molecular Biology (RECOMB 2006)*, pp. 160–174, Springer Berlin Heidelberg, 2006.

[48] C. B. Do, M. S. Mahabhashyam, M. Brudno, and S. Batzoglou, "ProbCons: probabilistic consistency-based multiple sequence alignment," *Genome Research*, vol. 15, no. 2, pp. 330–340, 2005.

[49] C. B. Do, M. S. Mahabhashyam, M. Brudno, and S. Batzoglou, "ProbCons: probabilistic consistency-based multiple sequence alignment of amino acid sequences," 2006. Software available at `http://probcons.stanford.edu/download.html`.

[50] K. Liu, C. Linder, and T. Warnow, "RAxML and FastTree: comparing two methods for large-scale maximum likelihood phylogeny estimation," *PLOS ONE*, vol. 6, no. 11, p. e27731, 2012.

[51] F. Abascal, R. Zardoya, and D. Posada, "ProtTest: selection of best-fit models of protein evolution," *Bioinformatics*, vol. 21, no. 9, pp. 2104–2105, 2005.

[52] D. Posada and K. Crandall, "Modeltest: testing the model of DNA substitution," *Bioinformatics*, vol. 14(9), pp. 817–818, 1998.

[53] M. Hoff, S. Orf, B. Riehm, D. Darriba, and A. Stamatakis, "Does the choice of nucleotide substitution models matter topologically?," *BMC Bioinformatics*, vol. 17, p. 143, 2016.

[54] S. Tavaré, "Some probabilistic and statistical problems in the analysis of DNA sequences," in *Lectures on Mathematics in the Life Sciences*, vol. 17, pp. 57–86, American Mathematical Society, 1986.

[55] T. H. Jukes and C. R. Cantor, "Evolution of protein molecules," *Mammalian Protein Metabolism*, pp. 21–132, 1969.

[56] M. Nute and T. Warnow, "Scaling statistical multiple sequence alignment to large datasets," *BMC Bioinformatics*, 2016. Special issue for RECOMB-CG 2016.

[57] M. Nute, "Github site for PASTA+BAli-Phy." `https://github.com/mgnute/pasta`. Accessed July 18, 2019.

[58] M. Nute, E. Saleh, and T. Warnow, "Evaluating statistical multiple sequence alignment in comparison to other alignment methods on protein data sets," *Systematic biology*, vol. 68, no. 3, pp. 396–411, 2018.

[59] A. Bateman, E. Birney, L. Cerruti, R. Durbin, L. Etwiller, S. R. Eddy, S. Griffiths-Jones, K. L. Howe, M. Marshall, and E. L. Sonnhammer, "The Pfam protein families database," *Nucleic Acids Res.*, vol. 30, pp. 276–280, 2002.

[60] U. Mai and S. Mirarab, "TreeShrink: fast and accurate detection of outlier long branches in collections of phylogenetic trees," *BMC Genomics*, vol. 19, p. 272, may 2018.