MUSCLE v5 enables improved estimates of phylogenetic tree confidence by ensemble bootstrapping

Robert C. Edgar[1,‡,*]

[1]Unaffiliated

[‡]Corresponding author: robert@drive5.com

orchid 0000-0001-7355-2541

Abstract

Phylogenetic tree confidence is often estimated from a multiple sequence alignment (MSA) using the Felsenstein bootstrap heuristic. However, this does not account for systematic errors in the MSA, which may cause substantial bias to the inferred phylogeny. Here, I describe the MSA ensemble bootstrap, a new procedure which generates a set of replicate MSAs by varying parameters such as gap penalties and substitution scores. Such an ensemble is called diagnostic if the typical distance between MSAs is comparable to the error rate. Confidence in a prediction derived from an MSA, e.g. a monophyletic clade, is expressed as the fraction of the ensemble where the prediction is reproduced. This approach is implemented in MUSCLE by modifying the Probcons algorithm, which is based on a hidden Markov model (HMM). An ensemble is generated by perturbing HMM parameters and permuting the guide tree. Ensembles generated by this method are shown to be diagnostic on the Balibase benchmark. To enable scaling to large datasets, divide-and-conquer heuristics are introduced. A new benchmark (Balifam) is described with 36 sets of 10000+ proteins. On Balifam, ensembles generated by MUSCLE are shown to align an average of 59% of columns correctly, 13% better than `Clustal-omega` (52% correct) and 26% better than MAFFT (47% correct). The ensemble bootstrap is applied to a previously published tree of RNA viruses, showing that the high reported Felsenstein bootstrap confidence of Ribovirus phylum branching order is an artifact of systematic MSA errors.

Data availability

Muscle source code `https://github.com/rcedgar/muscle`.

Balifam benchmark `https://github.com/rcedgar/balifam`.

Qscore source code `https://github.com/rcedgar/qscore`.

Palmscan source code `https://github.com/rcedgar/palmscan`.

Introduction

Multiple sequence alignments (MSAs) are ubiquitous in molecular biology with applications ranging from protein structure and function prediction to phylogenetic tree estimation. Over the past two decades, work on practical MSA algorithms has focused primarily on the challenges of improving accuracy and scaling to the large datasets generated by next-generation sequencing (NGS). For a recent review see (Chatzou, Magis, et al. 2016).

Scaling to large datasets

The most computationally expensive step in most pre-NGS MSA algorithms is calculation of a distance matrix for the input sequences, which has $O(N^2)$ complexity for N sequences. Recent methods designed to scale to large datasets reduce this cost by divide-and-conquer strategies which split the input into smaller clusters using fast heuristics, then combine clusters by progressive alignment. Such methods include Clustal-omega (Sievers and Higgins 2014), MAFFT-PartTree (Katoh and Toh 2007), PASTA (Mirarab et al. 2015), and MAGUS (Smirnov and Warnow 2020). Here, I describe a new algorithm, `Super5`, which applies a divide-and-conquer approach to the `Probcons` algorithm (Do et al. 2005) and a new protein MSA benchmark, `Balifam`, which assesses accuracy on large datasets by adding homologs to reference alignments from `Balibase` (Thompson, Plewniak, and Poch 1999).

MSA ensemble bootstrap

Downstream analysis based on an MSA, in particular phylogenetic tree inference, typically proceeds on an implicit assumption that the MSA is correct, or that MSA errors can be neglected, or that MSA errors are adequately accounted for by the Felsenstein bootstrap heuristic (Felsenstein 1985). However, systematic errors in the MSA can cause bias in inferred trees, potentially giving high confidence to incorrect tree topologies due to correlated misalignments (Simmons, Mueller, and Webb 2011). The Felsenstein bootstrap generates an ensemble of derived MSAs by sampling of columns from the original MSA with replacement. This procedure is robust if each column in the alignment is well modeled by a stochastic process following the correct branching order, but may not be if misalignments are correlated. Here, I describe an alternative method for generating an MSA ensemble. Alignment parameters such as substitution scores and gap penalties are varied with the goal of generating alternative MSAs with comparable numbers of errors to the MSA produced by default parameters. Ideally, each member of the ensemble will have distinctly different errors, especially systematic errors. The robustness of a downstream analysis can then be assessed on the ensemble as with the Felsenstein bootstrap: the confidence that a given predicted feature is correct is expressed as the fraction of the ensemble where this feature is predicted. This procedure naturally generalizes to other analyses. For example, if an MSA is used to predict protein secondary structure, then the confidence that a given residue is in an alpha helix can be expressed as the fraction of MSAs in the ensemble which yield a helix prediction for this residue. I will refer to this measure as the *MSA ensemble confidence*, or simply *ensemble confidence*.

Pair-wise alignment parameters and systematic errors

The primary parameters of a pair-wise alignment algorithm are its substitution scores and gap penalties. Adjusting gap penalties tends to change the length of the alignment, with higher penalties giving shorter alignments. Any fixed choice of gap penalties may result in systematic error because higher penalties give shorter alignments and thus tend to systematically underestimate the number of insertions and deletions (indels) compared to low-scoring substitutions, while conversely lower gap penalties favor indels over unlikely substitutions. Adjusting substitution scores may have a similar effect. For example, the BLOSUM90 matrix gives negative scores to more substitutions than BLOSUM30, and BLOSUM90 will therefore tend to induce more gaps because the penalty of introducing a gap is relatively lower compared to less likely substitutions (assuming gap penalties are held fixed). These observations suggest generating an ensemble by varying the substitution matrix and gap penalties. Such an ensemble can assess confidence that inferences are robust against variations in the length and gap placement over a collection of plausible alternative alignments.

Guide tree bias

Most MSA algorithms, including consistency-based methods such as `T-Coffee` (Notredame, Higgins, and Heringa 2000) and `Probcons`, build the final MSA using so-called progressive alignment following a guide tree. Phylogenetic trees inferred from such an MSA may be biased towards recapitulating its guide tree (Mukarram Hossain et al. 2015), which can be explained as follows. At each node in the tree, the MSAs of the child sub-trees (A and B) are aligned to each other by profile alignment, i.e. by pair-wise alignment of their columns. Columns from A and B are kept intact; the alignment is achieved by inserting new columns of gaps into A and B. With progressive alignment, the number of errors is necessarily non-decreasing on a path from a leaf to the root of the guide tree, because the alignment of a pair of sequences remains fixed in all subsequent profile alignments after it is first constructed. Errors will tend to accumulate in each profile alignment, especially close to the root where sequence divergence is highest and the alignment is most ambiguous. Suppose the last three sub-trees to be aligned are A, B and C. There are three possible guide tree topologies for the final two profile alignments: $((A, B), C)$, $((A, C), B)$ and $((B, C), A)$. In challenging cases, these topologies will induce distinctly different multiple alignments where the pair that is joined first has fewer errors and fewer intra-pair gap columns compared to the out-group which is joined last. A phylogenetic tree predicted from the MSA may then tend to follow the same topology as the guide tree, especially close to the root, i.e. to "re-discover" groups A, B and C and infer the same branching order as the guide tree, because groups aligned earlier appear to be more similar (fewer gaps, more well-aligned columns). An MSA ensemble for a progressive method should therefore include alignments obtained by varying the guide tree, e.g. by trying all three possible topologies for the final two profile alignments.

Diagnostic ensembles

Ideally, all MSAs in an ensemble would have accuracy close to the best possible fixed choice of parameters; i.e., parameters that maximize accuracy over a representative range of practical alignment tasks. MSAs in an ideal ensemble would have similar error rates, but the errors in each MSA would be distinctively different, especially systematic errors. Therefore, in an ideal ensemble, the distance between a typical pair of MSAs (the *dispersion* of the ensemble) would be comparable to the distance between a typical MSA and the correct MSA (Fig. 1). I use the term *diagnostic* for an ensemble which satisfies these criteria to a practically useful approximation, implying that it can be used to diagnose the reliability of downstream analysis such as phylogenetic tree inference. It is not clear *a priori* whether diagnostic ensembles can be constructed in practice by an unsupervised method (i.e., without knowing the correct alignment), but a proposed unsupervised method can be validated by these criteria on a benchmark set of trusted alignments.

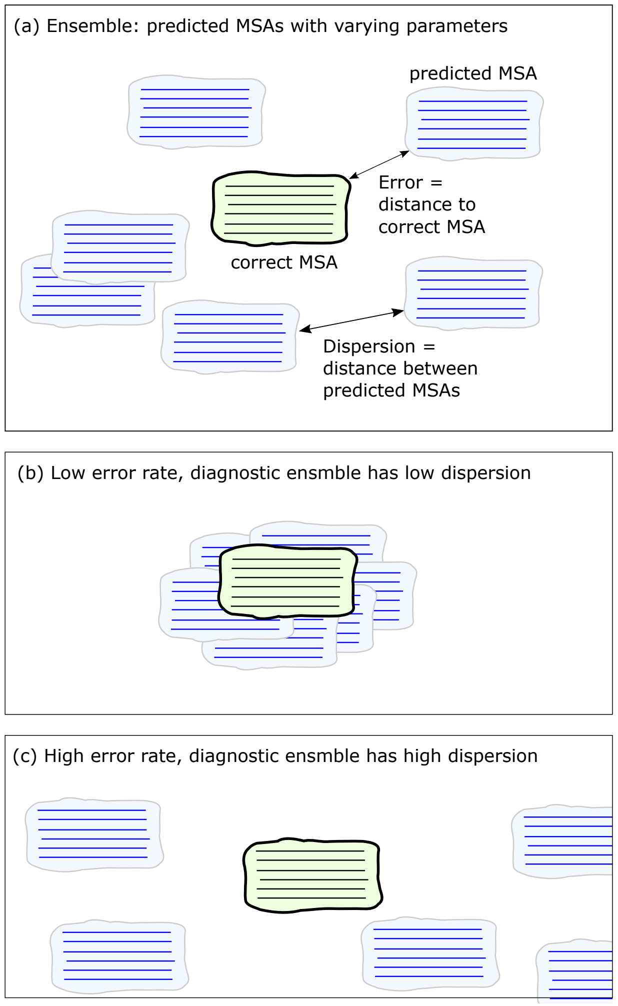
(a) Ensemble: predicted MSAs with varying parameters
predicted MSA
Error = distance to correct MSA
correct MSA
Dispersion = distance between predicted MSAs
(b) Low error rate, diagnostic ensmble has low dispersion
(c) High error rate, diagnostic ensmble has high dispersion

Figure 1. MSA ensemble.

An MSA ensemble is generated by varying parameters such as gap penalties and the substitution matrix. It can be characterized by the typical error rate and the dispersion, i.e. the typical distance between MSAs in the ensemble (top, panel a). In a diagnostic ensemble, the dispersion is comparable to the error rates, so that when error rates are low the MSAs are similar to each other (panel b), and when error rates are high the MSAs are more highly dispersed (panel c).

Methods

Multi-threaded Probcons (MPC)

`Probcons` generalizes the pair-wise posterior decoding alignment algorithm (Holmes and Durbin 1998) to multiple alignment, applying a consistency transformation (Notredame, Higgins, and Heringa 2000) to the posterior probability matrices. Currently, `Probcons` remains among the top-scoring methods on protein alignment benchmarks with small numbers of sequences, but is computationally expensive, scaling to at most a few tens of sequences on a commodity computer. Here, a commodity computer is operationally defined as a `c5a.4xlarge` instance on the Amazon Web Service (`https://aws.amazon.com/`), which has 16 vCPU cores and 32 Gb of RAM, typical cost 0.26 US dollars per hour at the time of writing. I implemented multi-threaded Probcons (`MPC`) by parallelizing calculation of the pair-wise posterior probability matrices and the consistency transformation, noting that in both cases a given pair of sequences can be processed independently of other pairs. Otherwise, `MPC` is essentially equivalent to `Probcons`, except that the guide tree for the final progressive alignment stage is constructed using the biased UPGMA method from MUSCLE v3 (Edgar 2004a).

Hidden Markov model

`MPC` uses a hidden Markov model (HMM) with topology shown in Fig. 2. This topology follows the source code for the final release of `Probcons` (version 1.12, posted March 2005), which differs from the model described in (Do et al. 2005). Columns of an alignment of sequences x and y are emitted by the match state M and by insert states I_x, I_y, J_x and J_y. M emits a column containing an aligned pair of letters. I_x and J_x emit one letter from x, similarly I_x and J_x emit one letter from y. The I states induce short gaps while the J states induce longer gaps. This happens because the $I \to I$ transition probability is lower than $J \to J$, and the $M \to J$ transition probability is higher than $M \to I$. Re-stated in terms of log-odds scores, short gaps have lower open penalty and higher extend penalty, while long gaps have higher open penalty and lower extend penalty. Alignments begin in the start state S and may end in any state other than S. Match state emission probabilities are obtained from the joint-probability form of the BLOSUM62 matrix (Pietrokovski, J. G. Henikoff, and S. Henikoff 1996); insert states emit letters according to the marginal probabilities of BLOSUM62. In `Probcons`, transition probabilities were trained by expectation-maximization on version 2 of the `Balibase` benchmark. For MPC, I chose somewhat arbitrary round numbers (Table 1), guided by the defaults in `Probcons`. These parameters were chosen once and not subsequently tuned or adjusted.

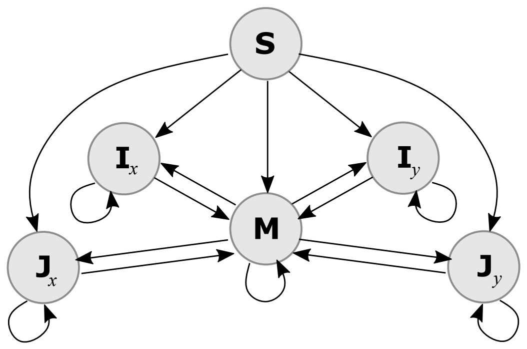

Figure 2. HMM topology.

HMM used by the `MPC` and `Super5` algorithms. Alignments begin in the start state S. Columns of an alignment of sequences x and y are emitted by the match state M and by insert states I_x, I_y, J_x and J_y.

From state	To state	Probability	ProbCons v1.12	Description
S	M	0.60	0.6814756989	First column is match
	I	0.02	0.008008334786	First column opens short gap
	J	0.18	0.1591759622	First column opens long gap
M	M	0.96	0.96008111723	Consecutive match columns
	I	0.012	0.0119511066	Open short gap
	J	0.0080	0.00800833479	Open long gap
I	I	0.40	0.3965826333	Extend short gap
	M	0.60	0.6034173667	Terminate short gap
J	J	0.90	0.8988758326	Continue long gap
	M	0.10	0.10112416730	Terminate long gap

Table 1. Transition probabilities. Transition probabilities for the HMM were set by hand as shown in this table. Values from `Probcons` v1.12 are also shown for comparison.

Parameter perturbations

To enable MSA ensembles, I implemented a method for introducing variations (*perturbations*) into HMM parameters, as follows. The function $perturb(P, v), 0 < v < 1$ selects a random real number from the range $(1 - v) \dots (1 + v)$ with uniform distribution; this random number multiplies a transition or emission probability P. Note that *perturb* is uniformly distributed around 1, so for a fixed value of r there is an equal chance of increasing or decreasing P. The argument v is called the *variance*. Two variances are used: one for transition probabilities $v_t = 0.4$ and one for emission probabilities $v_e = 0.3$. All transition and emission parameters are adjusted in this way, then normalized to ensure that probabilities of exclusive alternatives sum to one. The numerical values of the range parameters are quite arbitrary; they were selected once because they seemed intuitively reasonable, and were not subsequently tuned or adjusted. An ensemble is generated by selecting different values of the pseudo-random number generator seed s.

Guide tree permutations

Both `MPC` and `Super5` use progressive alignment. Variant guide trees are constructed as follows. The goal is to identify three large subtrees A, B, C which are joined in all three possible orders $((A, B), C)$, $((A, C), B)$ and $((B, C), A)$. This is achieved by considering all possible bifurcations of the original tree (which is temporarily considered to be unrooted), and identifying the edge which most closely approximates dividing the tree into subtrees with one third and two thirds of the sequences, respectively. The smaller subtree is A. The larger subtree is divided into two equal-sized (or approximately equal-sized) subtrees by a similar search for the best edge, giving B and C. Including the original guide tree, this gives a total of four variant guide trees for generating ensembles.

Pair-wise error (PWE) and pair-wise dispersion (PWD)

Given a pair of sequences, an alignment generated by an algorithm (a *test* alignment) and a trusted (*reference*) alignment, I define the pair-wise error (PWE) and pair-wise dispersion (PWD) as follows (Fig. 3). PWE is the fraction of columns in the reference alignment which are not reproduced in the test alignment (Fig. 3(a)). The pair-wise dispersion (PWD) of two test alignments of the same sequences is defined as the fraction of letters which are aligned differently, considering only the subset of letters annotated as alignable in the reference (Fig. 3(b)). For MSAs, the PWE or PWD is calculated as the mean over all pairs of sequences.

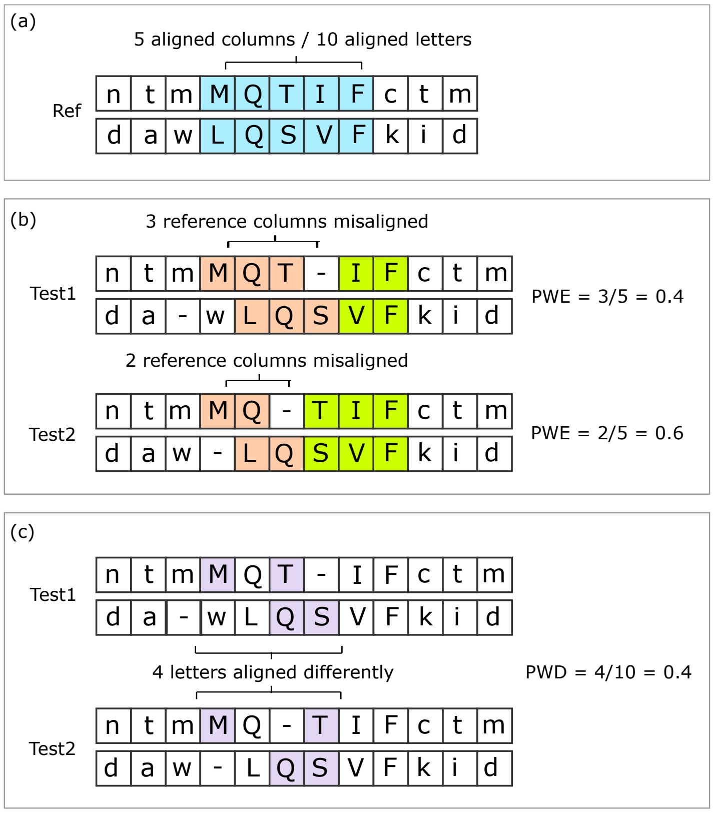

Figure 3. Pair-wise error and pair-wise dispersion.
Upper-case letters in the reference are considered to be reliably aligned; lower-case letters are not counted. Pair-wise error (PWE) is the fraction of columns in the reference alignment which are not reproduced in the test alignment. Pair-wise dispersion (PWD) of two test alignments of the same sequences is defined as the fraction of letters which are aligned differently, considering only the subset of letters annotated as alignable in the reference.

Balibase ensembles

For each of the 386 alignments in `Balibase v3`, I generated an ensemble of 16 MSAs using the `MPC` algorithm with random number seed $s = 0, 1 \ldots 15$. The special case $s = 0$ indicates that the unperturbed HMM and original guide tree are used. Accuracy by the Q and TC metrics was calculated by `qscore` (`https://github/rcedgar/qscore`); PWE and PWD were calculated by the `qscore3` command in `muscle` v5.

Super5 algorithm

The `Super5` algorithm was designed to scale `MPC` by introducing divide-and-conquer heuristics. A sketch of the algorithm workflow is given in Fig. 4.

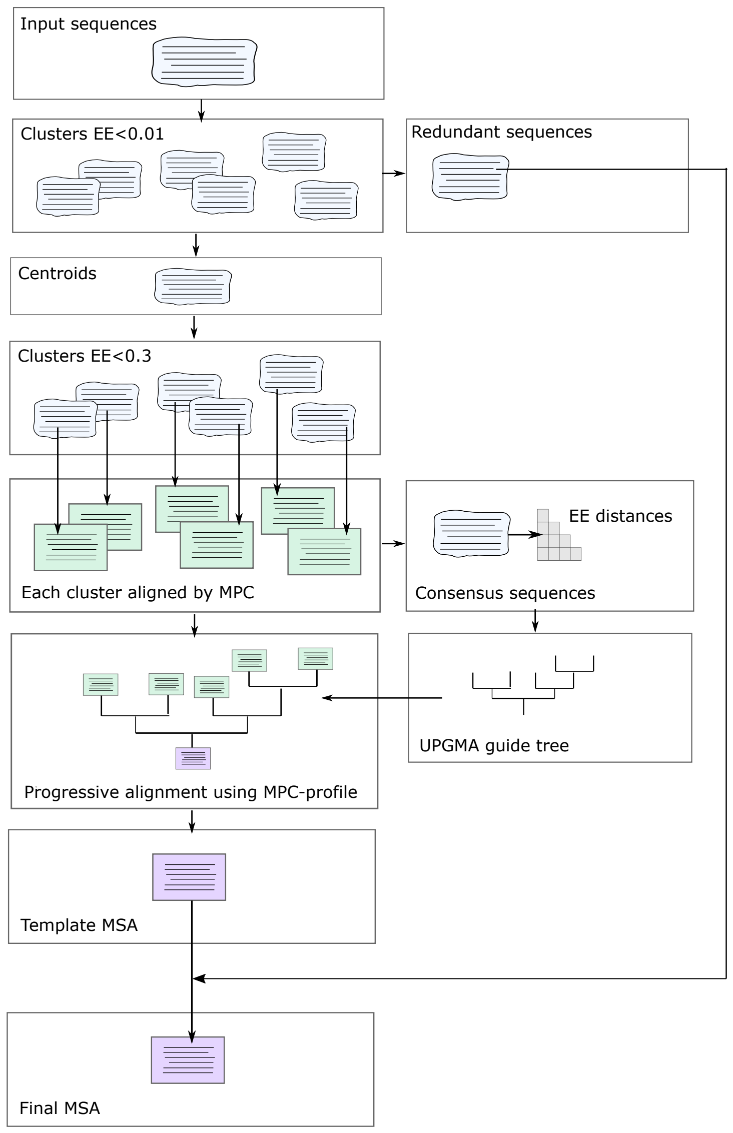

Figure 4. Workflow of the `Super5` algorithm.

`Super5` applies a divide-and-conquer strategy to `MPC`, enabling scaling to larger datasets.

MPC profile alignment (MPC-p)

The final stage of `Probcons` performs progressive alignment where a pair of profiles X and Y is aligned by maximizing the total posterior probability under the constraint that columns in each profile are held fixed. This is achieved by calculating a matrix M as follows,

$$M_{ij} = \sum_{x \in X} \sum_{y \in Y} P(x_i \leftrightarrow y_j), \tag{1}$$

where x is a sequence in MSA X, y is a sequence in MSA Y, and $P(x_i \leftrightarrow y_j)$ is the posterior probability that the letter of x in column i of X aligns to the letter of y in column j of Y. In `MPC`, this calculation is parallelized by observing that the contributions to M_{ij} from different pairs of sequences in Eq. 2 are independent, and can therefore be calculated on separate threads. The profile alignment is determined by using dynamic programming to maximize the sum of M_{ij} over alignment columns. I call this method for aligning a pair of profiles `MPC-p`.

MPC profile alignment with subsampling (MPC-ps)

`MPC-p` has complexity $O(N_X N_Y)$ in the number of sequences N_X and N_Y in X and Y respectively. When N_X and/or N_Y are large, this cost can be prohibitive. `Super5` implements a faster approximation to `MPC-p` by selecting random subsets X^* and Y^* of the sequences in X and Y respectively. The subsampled matrix M^* is then computed as follows,

$$M^*_{ij} = \sum_{x \in X^*} \sum_{y \in Y^*} P(x_i \leftrightarrow y_j). \tag{2}$$

The maximum posterior alignment is then calculated using M^* rather than M. I call this method `MPC-ps`. By default, `Super5` aligns profiles using `MPC-p` if $N_X N_Y \leq 2000$, otherwise a random subset of 2000 sequence pairs is selected and `MPC-ps` is used.

Expected-error distance (EE)

As shown in (Holmes and Durbin 1998), the expected number of errors per column in the posterior decoding alignment A of sequences x and y can be calculated as

$$EE = 1 - \frac{1}{|A|} \sum_{i,j \in A} P(x_i \leftrightarrow y_j). \tag{3}$$

Sequences that can be more accurately aligned (according to the HMM) have smaller EE, which can be considered as a distance measure defined on pairs of sequences.

Redundancy reduction

The first step of `Super5` aims to identify clusters of two or more highly similar sequences in the input data. For each cluster, a representative sequence is identified. Representatives are propagated for subsequent processing while the remaining sequences are set aside and added back into the representative alignment in the final step. This strategy is effective in reducing computational cost if the number of representatives is substantially smaller than the number of input sequences, which is often the case in practice. Clusters are constructed using a greedy list removal strategy similar to the `UCLUST` algorithm (Edgar 2010). Input sequences are sorted by decreasing length, with the goal of ensuring that shorter fragments do not become representatives. A k-mer index on the representatives is used to prioritize sequence comparisons by U-sorting (Edgar 2010) with a maximum of 16 rejections. If an input sequence matches a representative with $EE < 0.01$, it is assigned to the corresponding cluster, otherwise it becomes a new representative. This strategy reduces the $O(N^2)$ cost of all-vs-all comparison of N input sequences to an effective cost of $O(NR)$, where R is the number of representatives. This clustering method is called `UCLUST-EE`.

Coarse clustering

The next step divides representatives into clusters small enough to be tractable for `MPC`, i.e. a few hundred sequences. A first-draft set of clusters is obtained by `UCLUST-EE` with $EE < 0.3$. Clusters which are bigger than the maximum size (default 500) are sub-divided by `UCLUST-EE` with $EE < 0.1$. Any remaining clusters which are still too large are sub-divided at random.

Intra-cluster alignment and consensus

Each coarse cluster is aligned by `MPC`, and the consensus sequence for each MSA is calculated by taking the highest-frequency symbol from each column, deleting any positions where this symbol is a gap.

Guide tree construction

An all-vs-all EE distance matrix is calculated from the consensus sequences, and a biased UPGMA tree (Edgar 2004a) constructed from the distance matrix.

Representative MSA

MSAs for coarse clusters are combined by progressive alignment following the guide tree. Profile alignment is performed by `MPC-p` up to a size threshold (2000 pairs by default), otherwise by `MPC-ps` where the threshold number of sequence pairs is selected at random. This yields an MSA of all representative sequences.

Final MSA

The final MSA is constructed by re-introducing the non-representative sequences set aside in the first step, using the previously constructed pair-wise alignments to their corresponding representatives. These pair-wise alignments imply transitive alignments of non-representative sequences to the representative MSA, which are used to construct the final MSA.

Benchmark datasets

I used the following benchmarks for comparative validation: `Balibase v3` and `Prefab v4` (Edgar 2004b) for comparison with previously published results, and `Balifam`, a new benchmark described below for evaluation on larger datasets.

Balifam benchmark

`Balifam` was constructed from reference alignments in `Balibase v3` by adding homologs identified by `PFAM` (Bateman et al. 2004) (Fig. 5). Each reference sequence in `Balibase` was aligned to `PFAM`. For each `PFAM` domain identified in a given `Balibase` reference alignment, the subset of columns in the reference which aligned to that domain were extracted, giving a *seed* alignment. For each domain, the seed alignment with largest number of sequences was chosen. (This step was necessary because `Balibase` is highly redundant, often re-using the same sequences in different sets). To each seed alignment, I added homologs from the corresponding `PFAM` "full" alignment. The `PFAM` alignments *per se* were not used; this procedure served only to increase the size of the datasets. Random subsets of size 100, 1000 and 10000 respectively were selected and added to the seed alignments, yielding 59 sets in `Balifam-100`, 56 sets in `Balifam-1000` and 36 sets in `Balifam-10000`. Alignment accuracy is assessed on the subset of sequences in the seed alignment.

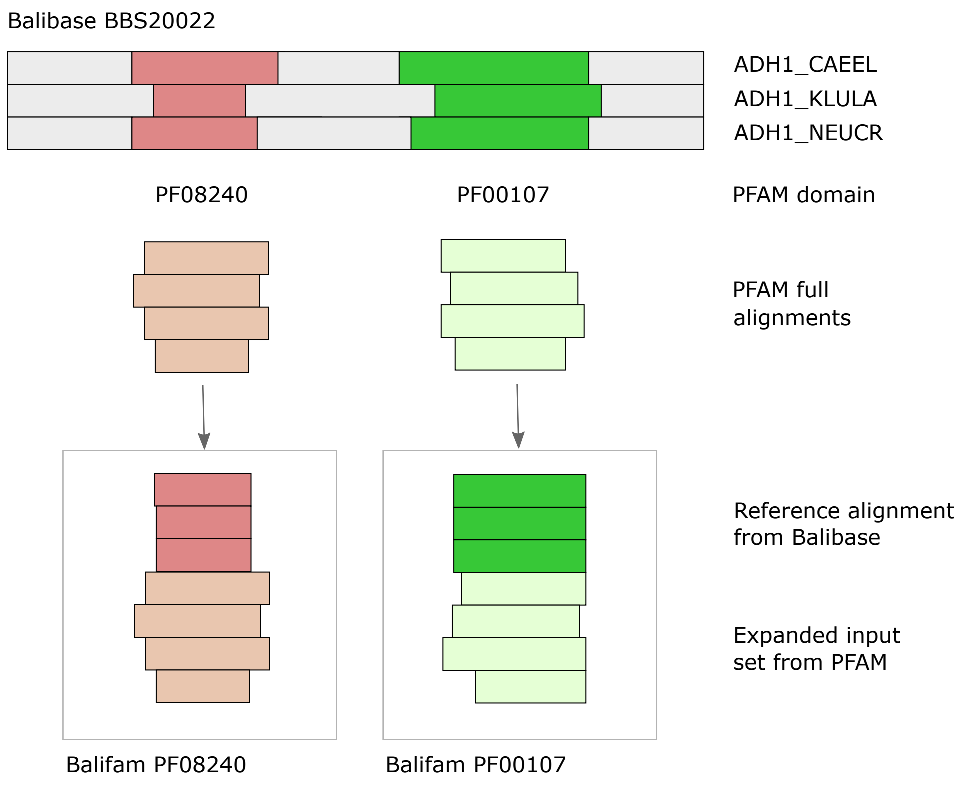

Figure 5. Construction of `Balifam`.

`Balifam` is constructed by using `PFAM` to identify domains in `Balibase` reference alignments. Here, three of the 58 sequences in reference set `BBS20022` are shown as examples. These sequences contain two domains, `PF08240` Alcohol dehydrogenase GroES-like domain and `PF00107` Zinc-binding dehydrogenase. Columns from the reference alignment matching each domain are extracted and combined with unaligned sequences from the corresponding `PFAM` full alignments.

Comparative validation

I chose to validate `MPC` and `Super5` against `Probcons` v1.12, `Clustal-omega` and `MAFFT`. I aimed to include `MAGUS` also, but an issue with installing the software was unresolved at the time this work was completed (`https://github.com/vlasmirnov/MAGUS/issues/12#`). Versions and command-line options for the other methods are given in Table 2.

Algorithm	Short name	Version	Command line options
Clustal-omega	clustalo	clustalo-1.2.4-Ubuntu-x86_64	-threads=16
MUSCLE v3	muscle3	muscle3.8.31_i86linux64	(none)
MAFFT	mafft	v7.453	--anysymbol --thread -1 --auto
Probcons	probcons	v1.12	(none)

Table 2. Methods tested in comparative validation.

RdRP alignment and tree

To illustrate a practical application of the MSA ensemble bootstrap, I analyzed a phylogenetic tree of RNA viruses described in (Wolf et al. 2018), which has been influential in a recent major revision of RNA virus taxonomy to introduce the new realm *Ribovaria* with sub-divisions into phyla (Walker et al. 2019). This tree was estimated from an MSA of 4627 RNA-dependent RNA polymerase (RdRP) sequences. I generated an MSA ensemble from these sequences using `Super5`, and generated maximum-likelihood trees using `IQ-TREE` (Nguyen et al. 2015) from each MSA in the ensemble. MSA accuracy was assessed by using the `Palmscan` algorithm (Babaian and Edgar 2021) to identify alignment columns containing the six essential catalytic residues of the polymerase palm domain (`D...D...G...GDD`, which I call the *super-motif*). Each RdRP sequence was scanned individually to identify the positions of the super-motif residues. For each residue, the consensus column in the MSA was identified as the column where the largest number of sequences has this residue. A sequence having this residue in a different column is said to *disagree* with the consensus, indicating a misalignment. MSA quality was assessed by measuring the fraction of sequences and fraction of super-motif letters which disagree, respectively.

Results

Alignment benchmarks

Results on the alignment accuracy benchmarks are shown in Tables 3 and 4. Tests were run on AWS `c5a.4xlarge` instances. Methods were excluded on `Balifam-1000` and `Balifam-10000` if they required $>$ 32Gb RAM or took more than 24 hours to complete any single set in the benchmark. On `Balifam-10000`, `Super5` aligns 59% of columns correctly, which is a 13% improvement over `Clustal-omega` (52% columns correct) and a 26% improvement over `MAFFT` (47% columns correct). This difference is significant by the Wilcoxon test: `Super5` $>$ `Clustal-omega` with $p = 1.2 \times 10^{-4}$ and `Super5` $>$ `MAFFT` with $p = 1.4 \times 10^{-7}$.

Benchmark	Method	Q	TC	Total time	Max memory
Balibase	clustalo	0.8397	0.5620	04:37	939 Mb
	muscle3	0.8433	0.5317	08:49	74 Mb
	super5	0.8698	0.5926	41:26	1.6 Gb
	mafft	0.8708	0.6076	07:18	3.3 Gb
	probcons	0.8831	0.6189	3:53:34	529 Mb
	mpc	0.8843	0.6178	14:51	1.9 Gb
Prefab	muscle3	0.6496	0.6496	12:27	30 Mb
	clustalo	0.6699	0.6699	05:49	80 Mb
	probcons	0.6843	0.6843	11:43:47	94 Mb
	super5	0.6848	0.6848	1:03:51	1.0 Gb
	mpc	0.6863	0.6863	34:00	718 Mb
	mafft	0.6955	0.6955	03:06	300 Mb
Balifam100	muscle3	0.8457	0.5916	05:37	56 Mb
	clustalo	0.8523	0.5726	00:53	37 Mb
	probcons	0.8828	0.6580	2:22:55	907 Mb
	mafft	0.8840	0.6528	00:54	197 Mb
	mpc	0.8878	0.6461	09:16	914 Mb
	super5	0.8899	0.6380	06:54	418 Mb
Balifam1000	mafft	0.8122	0.4732	00:25	401 Mb
	clustalo	0.8454	0.5249	13:17	76 Mb
	super5	0.8817	0.5907	59:08	783 Mb
Balifam10000	clustalo	0.7803	0.4457	1:37:30	697 Mb
	mafft	0.8072	0.4416	22:01	1.3 Gb
	super5	0.8902	0.6191	5:20:48	3.3 Gb

Table 3. Alignment benchmark results. Alignment accuracy, total elapsed time and maximum memory use for the tested methods.

```
           Balibase         mpc    probcons       mafft      super5     muscle3    clustalo
====================  ----------  ----------  ----------  ----------  ----------  ----------
     mpc  0.8843  |            .     +0.0355   +1.55e-06   +2.95e-15    +4.5e-38   +1.27e-30
probcons  0.8831  |      -0.0355           .   +0.000379   +2.19e-12   +8.82e-37   +4.78e-27
   mafft  0.8708  |    -1.55e-06   -0.000379           .    (+0.203)   +8.82e-26   +8.44e-26
  super5  0.8698  |    -2.95e-15   -2.19e-12    (-0.203)           .   +1.44e-22   +2.29e-19
 muscle3  0.8433  |     -4.5e-38   -8.82e-37   -8.82e-26   -1.44e-22           .    (-0.694)
clustalo  0.8397  |    -1.27e-30   -4.78e-27   -8.44e-26   -2.29e-19    (+0.694)           .

             Prefab       mafft         mpc      super5    probcons    clustalo     muscle3
====================  ----------  ----------  ----------  ----------  ----------  ----------
   mafft  0.6955  |            .   +1.05e-05   +2.55e-08   +7.18e-08   +2.41e-27   +2.99e-63
     mpc  0.6863  |    -1.05e-05           .    (+0.248)    +0.00716   +4.12e-10   +7.41e-46
  super5  0.6848  |    -2.55e-08    (-0.248)           .    (+0.932)   +2.62e-08   +1.49e-37
probcons  0.6843  |    -7.18e-08    -0.00716    (+0.932)           .   +2.14e-09   +5.61e-40
clustalo  0.6699  |    -2.41e-27   -4.12e-10   -2.62e-08   -2.14e-09           .   +8.52e-14
 muscle3  0.6496  |    -2.99e-63   -7.41e-46   -1.49e-37   -5.61e-40   -8.52e-14           .

         Balifam100      super5         mpc       mafft    probcons    clustalo     muscle3
====================  ----------  ----------  ----------  ----------  ----------  ----------
  super5  0.8899  |            .    (-0.232)    (-0.950)    (-0.424)   +0.000346   +0.000161
     mpc  0.8878  |     (+0.232)           .    (+0.209)    (+0.101)   +1.46e-06   +1.44e-07
   mafft  0.8840  |     (+0.950)    (-0.209)           .    (-0.981)    +3.9e-06   +2.07e-06
probcons  0.8828  |     (+0.424)    (-0.101)    (+0.981)           .   +4.43e-05   +1.48e-05
clustalo  0.8523  |    -0.000346   -1.46e-06    -3.9e-06   -4.43e-05           .    (+0.361)
 muscle3  0.8457  |    -0.000161   -1.44e-07   -2.07e-06   -1.48e-05    (-0.361)           .

        Balifam1000      super5    clustalo       mafft
====================  ----------  ----------  ----------
  super5  0.8817  |            .   +0.000122   +1.46e-07
clustalo  0.8454  |    -0.000122           .    +0.00287
   mafft  0.8122  |    -1.46e-07    -0.00287           .

       Balifam10000      super5       mafft    clustalo
====================  ----------  ----------  ----------
  super5  0.8902  |            .   +6.89e-07   +6.28e-06
   mafft  0.8072  |    -6.89e-07           .    (+0.187)
clustalo  0.7803  |    -6.28e-06    (-0.187)           .
```

Table 4. Pair-wise method comparisons of alignment accuracy. The relative accuracy of each pair of methods was assessed on each benchmark using the Wilcoxon signed rank test. Numbers in the tables are *p*-values according to the Wilcoxon test. A plus sign indicates that the method named at the start of the row is better than the method named at the top of the column; a minus sign indicates that the row method is worse. *P*-values > 0.05 are shown in parentheses to indicate that they are conventionally considered to be not significant. Methods are shown in order of decreasing Q.

Balibase ensembles

Fig. 7 is a scatterplot of PWE vs. PWD on the `Balibase` ensembles. This shows that dispersions are comparable to error rates, and the method therefore consistently generates diagnostic ensembles. Results for two typical example datasets are shown in Fig. 8; these ensembles are seen to be highly diagnostic.

s	*Q*	*TC*
0	0.884	0.619
1	0.875	0.610
2	0.880	0.612
3	0.881	0.615
4	0.877	0.613
5	0.885	0.618
6	0.883	0.615
7	0.883	0.622
8	0.875	0.608
9	0.876	0.611
10	0.882	0.620
11	0.882	0.615
12	0.878	0.613
13	0.880	0.620
14	0.884	0.622
15	0.880	0.617
16	0.875	0.603

Table 5. `MPC` alignment accuracy with different random number seeds.
Columns are s, random number seed ($s = 0$ means no perturbations); Q letter pair accuracy, TC column accuracy. This shows that perturbing HMM parameters and the guide tree gives similar average accuracy to default parameters, with variations of around 1%.

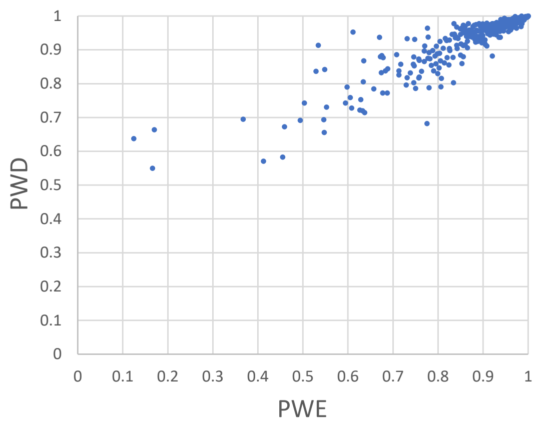

Figure 7. Correlation between dispersion and error.
Scatterplot of PWE vs. PWD on `Balibase` ensembles generated by `MPC`.

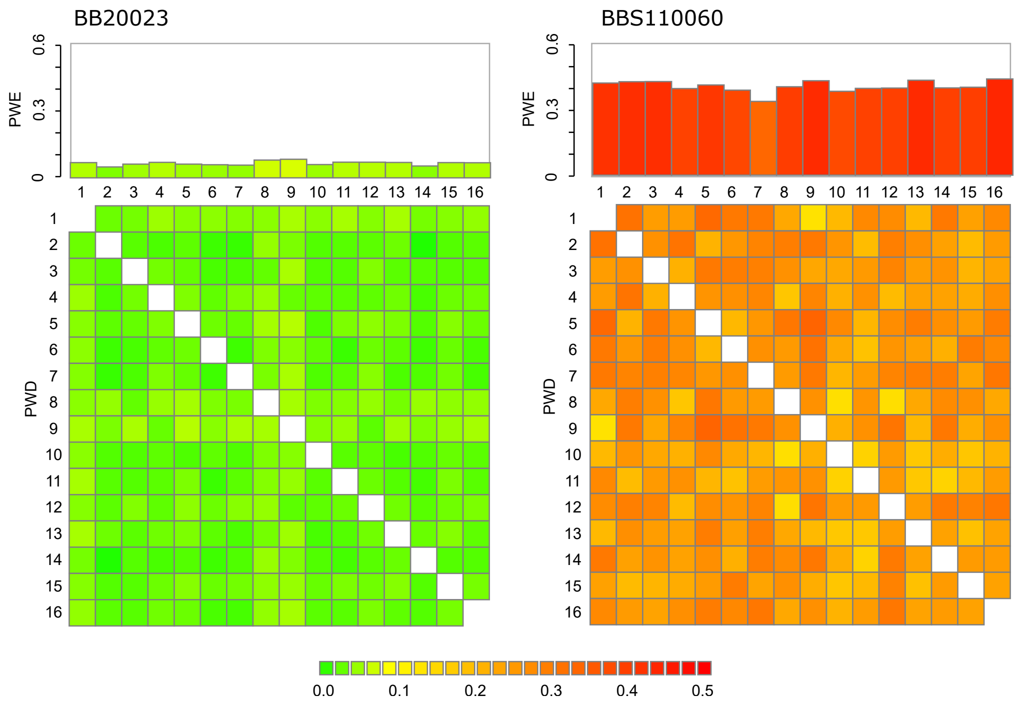

Figure 8. `MPC` **ensembles for two** `Balibase` **datasets.**

Typical example datasets `BB20023` (left, high accuracy) and `BBS110060` (right, low accuracy). Histograms at the top show PWE for each MSA in the ensemble. The matrix below is the dispersion (PWD) of each pair of MSAs. Note that on both these datasets, (1) the error rate of each MSA is similar, (2) the pair-wise dispersions between all pairs are similar, and (3) dispersions are similar to the error rates. These ensembles are therefore diagnostic (see main text for definition).

RdRP MSA and tree

Results for the `Palmscan` analysis of super-motif misalignments are shown in Table 5. By this standard, all `Super5` variants are more accurate than `MAFFT`, `Clustal-omega` and the original alignment in (Wolf et al. 2018). Accuracy is very robust again HMM perturbations, but consistently degrades slightly when the guide tree is permuted. Despite these small increases in error, all of the `Super5` MSAs are more accurate than the MSAs generated by `MAFFT` and `Clustal-omega`, and also more accurate than the MSA reported in (Wolf et al. 2018). Maximum-likelihood trees inferred from the `Super5` ensemble MSAs are shown in Fig. 9, colored by phylum. Fig. 10 shows the phylum topology of the trees in Fig. 9, demonstrating that the branching order is highly variable across the ensemble. I interpret these results as showing that the high bootstrap values reported in (Wolf et al. 2018) for branching order near the root of the tree are artifacts of the progressive alignment method used to build the MSA. A more realistic assessment is given by considering Fig. 10, where the branching order of phyla is inconsistent and all top-level branches therefore have ensemble confidence ≪0.5.

Method	*Guide tree*	*Seed*	*SD*	*LD*
Super5	(default)	0	8.99%	2.11%
		1	8.93%	2.08%
		2	8.95%	2.10%
	ABC	0	13.69%	3.12%
		1	13.58%	3.11%
		2	13.32%	3.02%
	ACB	0	13.47%	3.05%
		1	13.58%	3.11%
		2	13.54%	3.10%
	BCA	0	13.47%	3.05%
		1	13.58%	3.11%
		2	13.32%	3.02%
mafft	n/a	n/a	15.6%	4.98%
clustalo	n/a	n/a	55.28%	13.46%
Wolf2018	n/a	n/a	26.65%	5.14%

Table 5. Rates of RdRP super-motif misalignment according to `Palmscan`.
SD is sequence disagreement, i.e. fraction of sequences where at least one super-motif residue is not in its consensus column; *LD* is letter disagreement, i.e. fraction of super-motif residues placed outside their consensus column.

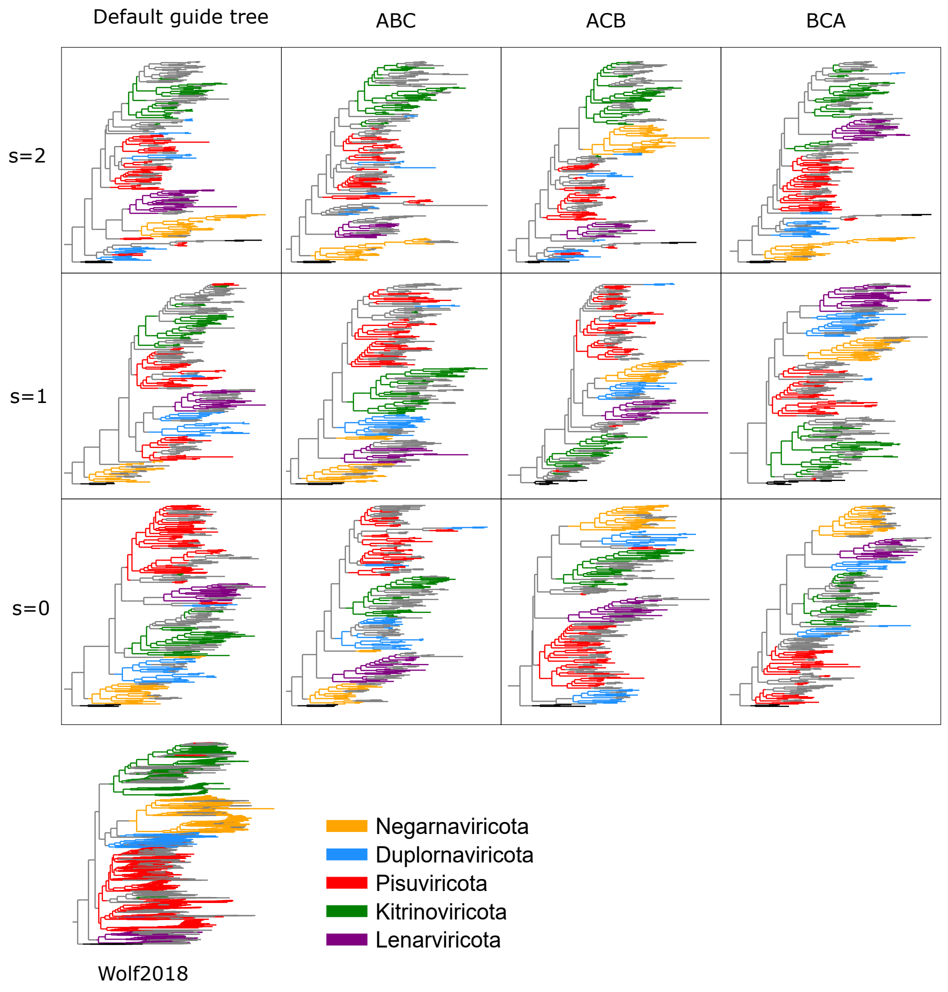

Figure 9. ML trees for `Super5` RdRP ensemble.

Trees are colored by phylum. The tree from (Wolf et al. 2018) is shown for comparison. Note that the branching order of phyla is inconsistent; see also Fig. 10.

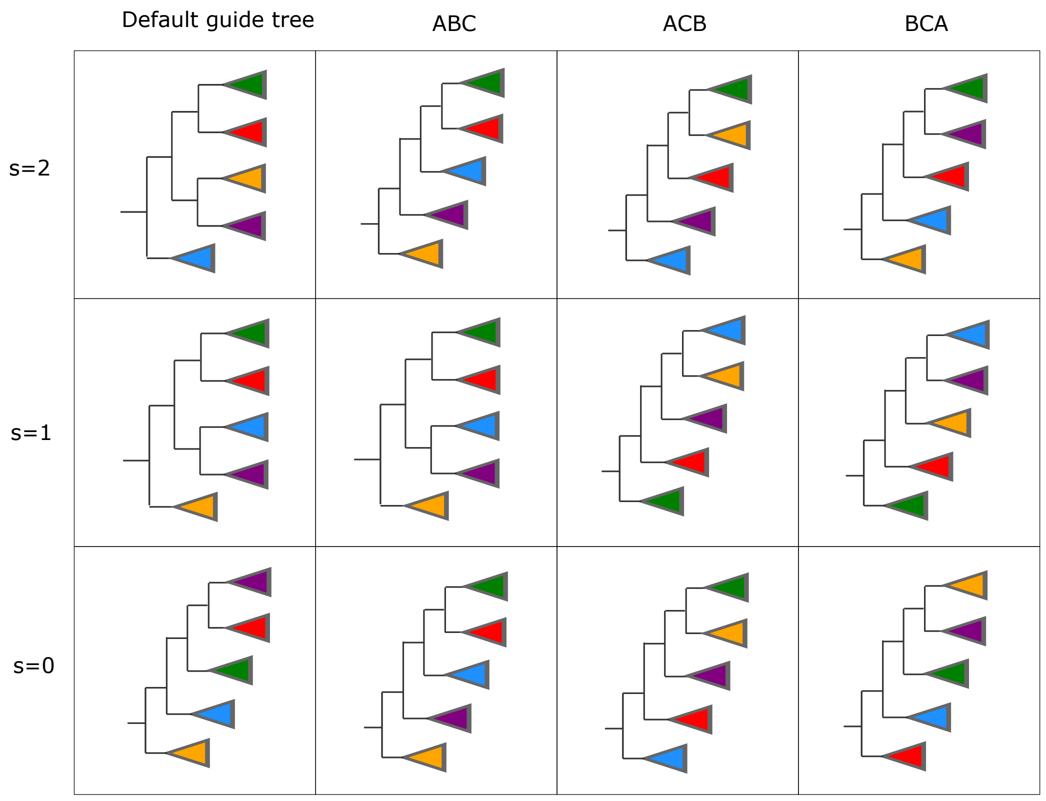

Figure 10. Phylum branching orders in RdRP ensemble trees.
Phylum branching order for each tree in Fig. 9. Note that the branching order is highly variable between replicates, with no apparent consensus for any subgroup. Phyla are colored as in Fig. 9.

Discussion

Related prior work

It has long been recognized that errors and biases in multiple sequence alignments may have substantial effects on inferred phylogenies. I will give a few representative examples. In (Morrison and Ellis 1997) the authors tried several different multiple alignment methods with varying parameters such as gap penalties and compared inferred protist trees, concluding "many of the literature disagreements concerning the phylogeny of the *Apicomplexa* are probably based on differences in sequence alignment strategies rather than differences in data or tree-building methods." (Wheeler 1995) investigated the effects of varying gap penalties and transition-transversion ratio on an arthropod tree, noting: "Transversion-transition ratios and alignment gap costs are generally not directly measurable. These values are statements of process, and they can only be inferred appropriately from a predetermined phylogenetic pattern. The disturbing circularity of the interaction between the specification of values *a priori* and their inference *a posteriori* is a general and central problem in molecular phylogenetic analysis." More recently, (Wong, Suchard, and Huelsenbeck 2008) investigated trees for seven yeast species, noting that "The statistical methods applied to the analysis of genomic data do not account for uncertainty in the sequence alignment. Indeed, the alignment is treated as an observation, and all of the subsequent inferences depend on the alignment being correct", concluding that "different alignment methods [result] in different conclusions". A modified Felsenstein bootstrap to account for sequence input order bias was proposed by (Chatzou, Floden, et al. 2018), who found that "Maximum Likelihood (ML) trees estimated from these MSAs [were] unstable with over 38% of the branches being sensitive to the sequence input order."

Validation of the MSA ensemble bootstrap

The MSA ensemble bootstrap proposed here subsumes and automates the *ad hoc* approaches of prior work into a single framework which enables a quantitative estimate of uncertainty due to alignment errors. The central question confronting the design of an automated ensemble procedure is, how much should algorithm parameters be varied? For pair-wise parameters such as gap penalties and substitution scores, this is answered by requiring *diagnosticity*, i.e. that variations between alignments in the ensemble should be comparable to the error rate of a typical alignment. This implies that alignments in the ensemble have similar errors but distinctly different errors. If an inference is robust within such an ensemble, we can have high confidence that it is correct, or at least not an artifact of MSA errors. The guide tree for progressive alignment may introduce systematic errors which persist independently of perturbations applied to pair-wise parameters; this is accounted for by introducing permutations into the joining order. Empirically, a simple procedure with two fixed parameters is shown to generate diagnostic ensembles on trusted alignments in `Balibase`. To the extent that `Balibase` alignments are typical, this validates that diagnostic ensembles can be generated in practice. Notably, perturbing HMM parameters causes no noticeable degradation in accuracy which might otherwise have been expected. This reinforces the intuitive justification for the ensemble bootstrap because in general there is no reason to prefer one alignment over another; in particular, the HMM with default parameters is not noticeably better than an HMM with perturbed parameters. In my view, the price of a small degradation of accuracy would be well worth it to get the greatly improved estimate of confidence, but perhaps unexpectedly it seems that no such price must be paid.

References

Babaian, Artem and Robert C Edgar (2021). "Ribovirus classification by a polymerase barcode sequence". In: *bioRxiv*.

Bateman, Alex et al. (2004). "The Pfam protein families database". In: *Nucleic acids research* 32.suppl_1, pp. D138–D141.

Chatzou, Maria, Evan W Floden, et al. (2018). "Generalized bootstrap supports for phylogenetic analyses of protein sequences incorporating alignment uncertainty". In: *Systematic Biology* 67.6, pp. 997–1009.

Chatzou, Maria, Cedrik Magis, et al. (2016). "Multiple sequence alignment modeling: methods and applications". In: *Briefings in bioinformatics* 17.6, pp. 1009–1023.

Do, Chuong B et al. (2005). "ProbCons: Probabilistic consistency-based multiple sequence alignment". In: *Genome research* 15.2, pp. 330–340.

Edgar, Robert C (2004a). "MUSCLE: a multiple sequence alignment method with reduced time and space complexity". In: *BMC bioinformatics* 5.1, pp. 1–19.

– (2004b). "MUSCLE: multiple sequence alignment with high accuracy and high throughput". In: *Nucleic acids research* 32.5, pp. 1792–1797.

– (2010). "Search and clustering orders of magnitude faster than BLAST". In: *Bioinformatics* 26.19, pp. 2460–2461.

Felsenstein, Joseph (1985). "Confidence limits on phylogenies: an approach using the bootstrap". In: *evolution* 39.4, pp. 783–791.

Holmes, Ian and Richard Durbin (1998). "Dynamic programming alignment accuracy". In: *Journal of computational biology* 5.3, pp. 493–504.

Katoh, Kazutaka and Hiroyuki Toh (2007). "PartTree: an algorithm to build an approximate tree from a large number of unaligned sequences". In: *Bioinformatics* 23.3, pp. 372–374.

Mirarab, Siavash et al. (2015). "PASTA: ultra-large multiple sequence alignment for nucleotide and amino-acid sequences". In: *Journal of Computational Biology* 22.5, pp. 377–386.

Morrison, David A and John T Ellis (1997). "Effects of nucleotide sequence alignment on phylogeny estimation: a case study of 18S rDNAs of Apicomplexa." In: *Molecular biology and evolution* 14.4, pp. 428–441.

Mukarram Hossain, AS et al. (2015). "Evidence of statistical inconsistency of phylogenetic methods in the presence of multiple sequence alignment uncertainty". In: *Genome biology and evolution* 7.8, pp. 2102–2116.

Nguyen, Lam-Tung et al. (2015). "IQ-TREE: a fast and effective stochastic algorithm for estimating maximum-likelihood phylogenies". In: *Molecular biology and evolution* 32.1, pp. 268–274.

Notredame, Cédric, Desmond G Higgins, and Jaap Heringa (2000). "T-Coffee: A novel method for fast and accurate multiple sequence alignment". In: *Journal of molecular biology* 302.1, pp. 205–217.

Pietrokovski, Shmuel, Jorja G Henikoff, and Steven Henikoff (1996). "The Blocks database—a system for protein classification". In: *Nucleic acids research* 24.1, pp. 197–200.

Sievers, Fabian and Desmond G Higgins (2014). "Clustal omega". In: *Current protocols in bioinformatics* 48.1, pp. 3–13.

Simmons, Mark P, Kai F Mueller, and Colleen T Webb (2011). "The deterministic effects of alignment bias in phylogenetic inference". In: *Cladistics* 27.4, pp. 402–416.

Smirnov, Vladimir and Tandy Warnow (2020). "MAGUS: Multiple Sequence Alignment using Graph Clustering". In: *Bioinformatics*.

Thompson, Julie D., Frédéric Plewniak, and Olivier Poch (1999). "BAliBASE: a benchmark alignment database for the evaluation of multiple alignment programs." In: *Bioinformatics (Oxford, England)* 15.1, pp. 87–88.

Walker, Peter J et al. (2019). "Changes to virus taxonomy and the International Code of Virus Classification and Nomenclature ratified by the International Committee on Taxonomy of Viruses (2019)". In: *Archives of virology* 164.9, pp. 2417–2429.

Wheeler, Ward C (1995). "Sequence alignment, parameter sensitivity, and the phylogenetic analysis of molecular data". In: *Systematic Biology* 44.3, pp. 321–331.

Wolf, Yuri I et al. (2018). "Origins and evolution of the global RNA virome". In: *MBio* 9.6, e02329–18.

Wong, Karen M, Marc A Suchard, and John P Huelsenbeck (2008). "Alignment uncertainty and genomic analysis". In: *Science* 319.5862, pp. 473–476.