# Strengths and Limits of Multiple Sequence Alignment and Filtering Methods

Vincent Ranwez, Nathalie N. Chantret

# Chapter 2.2 Strengths and Limits of Multiple Sequence Alignment and Filtering Methods

**Vincent Ranwez**
AGAP, Univ Montpellier, CIRAD, INRA, Montpellier SupAgro, Montpellier, France.
vincent.ranwez@supagro.fr
https://orcid.org/0000-0002-9308-7541

**Nathalie Chantret**
AGAP, Univ Montpellier, CIRAD, INRA, Montpellier SupAgro, Montpellier, France.
nathalie.chantret@inra.fr
https://orcid.org/0000-0002-2512-7644

## Abstract

Multiple sequence alignment (MSA) is a prerequisite for most phylogenetic analyses. Aligning sequences to unravel residue homology is a challenging task that has been the focus of much attention in recent decades. Research in this field has been extremely active from both theoretical and practical standpoints. Numerous tools have been developed to align sequences and, more recently, to post-process those alignments and filter out their most dubious parts. Whether or not the inclusion of alignment filtering in a phylogenetic pipeline improves the quality of the inferred phylogenies is still debatable.

The goal of this chapter is not to provide an exhaustive list of all tools available to produce or filter an MSA, but rather to cover the limitations of current alignment methods and their causes, to highlight key differences among MSA filtering methods and provide some practical MSA filtering guidelines.

We consider that filtering methods can be subdivided into two main categories. The first one includes methods that filter MSA by entirely removing some sites or sequences from the MSA. The second category contains MSA filtering methods that mask residues and are able to extract some pieces of information from a site or sequence, while disregarding the remaining information—we called these *picky-filtering* methods. In our benchmark, the filtering methods that perform best are, as expected, in the picky category. When inferring phylogenies, MSA filtering impacts the inferred tree topology but it also seems to significantly improve branch length estimations, especially when a picky-filtering method is used.

**How to cite:** Vincent Ranwez and Nathalie Chantret (2020). Strengths and Limits of Multiple Sequence Alignment and Filtering Methods. In Scornavacca, C., Delsuc, F., and Galtier, N., editors, *Phylogenetics in the Genomic Era*, chapter No. 2.2, pp. 2.2:1–2.2:36. No commercial publisher | Authors open access book. The book is freely available at https://hal.inria.fr/PGE.

**Funding** This work was supported by the CIRAD UMR AGAP HPC Data Center of the South Green Bioinformatics platform (`http://www.southgreen.fr/`).

## 1 Introduction

Multiple sequence alignment (MSA) is used for several kinds of molecular analysis such as identifying "specific determining positions" (SDP) involved in interactions (e.g., protein complexes), post-translational modification sites (phosphorylation, glycosylation, etc.) and, of course, phylogeny inference (Thompson et al., 2011). MSA is a crucial step since phylogenetic inference methods assume that residue homology relationships are correctly reflected by the


input MSA (Chapter 2.1 [Simion et al. 2020]). Parsimony scores, tree likelihoods and tree posterior probabilities are meaningless without this assumption.

Not long ago, each newly sequenced DNA fragment was eye checked by expert biologists and MSA software generated multiple alignments of those highly reliable sequences were diligently curated manually. In the past two decades, sequencing technologies have rapidly progressed, while exomes, transcriptomes and even genomes are now routinely sequenced. This rapid increase in dataset sizes has surely improved the reliability of most of inferred phylogenies as they are now based on hundreds, or even thousands, of loci instead of only a handful. When it comes to working with such large datasets, MSA manual curation is no longer a realistic option and, even when tractable, is a questionable practice as it goes against reproducibility. Many tools have been developed to try to automatically curate alignments by removing part of them, not by correcting them. Overly conservative filtering could thus drastically reduce the available phylogenetic signal along with the sequence alignment errors. There is still an ongoing debate as to whether it is better or not to filter sequence alignments prior to phylogeny inferences since some filtering processes may tend to remove too much of the phylogenetic signal along with phylogenetic noise — so the cure could be worse than the disease. Alignment quality checks are hence sometimes simply ignored, while assuming that errors will somehow be averaged and have little impact on the final biological conclusions, with the vast amount of correct data overwhelming the few incorrect data. However, this argument would not apply to downstream analyses based on the mean deviation. For instance, when searching for branches or loci undergoing positive selection based on non-synonymous vs synonymous substitution rate (dN/dS) analyses (see Chapter 4.5 [Lowe and Rodrigue 2020]), curating the alignment is crucial as alignment errors induce false positives.

MSA methods have been developing since the early 1980s, and it is sill a very active field of research, as illustrated by the number of publications with "multiple sequence alignment" in their title (Figure 1). There was a marked increase in the number of publications at the beginning of the $21^{st}$ century when sequencing methods became more accessible in the labs, thus further increasing the need for alignment methods. The importance of MSA methods is also reflected by the number of citations of the most successful programs (e.g. $>$ 47.800 citations for ClustalW [Thompson et al. 1994] according to the Web of Science).

Details on aligning two sequences are extensively described in most bioinformatics textbooks, but the steps required to go from pairwise alignments to multiple sequence alignments are often only briefly covered, or not at all. In this chapter we propose to outline those steps, while focusing on the underlying assumptions and shortcuts. We believe that this could be useful for users who may otherwise not understand why even the current best MSA software sometimes generate alignments containing obviously erroneous parts. When using pipelines that automatically chain multiple bioinformatics software to simultaneously analyse thousands of loci, operators may easily lose sight of the imperfections in the underlying methods and the need for caution in interpreting the final results. The next section starts by detailing the strengths and limits of alignment methods. In this respect, Section 2 provides an in-depth explanation of how MSA works in a non-algorithmic way while several figures provide visual schematic representations of the key steps. Then, Section 3 presents the key principles behind MSA filtering methods and pinpoints the inherent limitations of some of them. Finally, based on a large biological dataset, Section 4 provides some insight regarding the impact of MSA filtering methods on the inferred phylogenies.

## 2 Alignment Methods, Strengths and Limits

Intuitively, an MSA method inserts gap characters '-' inside input sequences to produce a set of longer sequences that are all of the same length, such that residues at the same position in different sequences (aligned residues) share some common properties. More formally, an MSA for a set of $n$ sequences $s_1 \dots s_n$ defined with alphabet $\sum$ is a set of $n$ sequences $S_1 \dots S_n$ which are defined on an enriched alphabet $\sum \cup \{\text{'-'}\}$ such that all $S_i$ have the same length $L$ and, $\forall i$, removing '-' from $S_i$ leads to $s_i$. The main strong point of MSA methods is that they are the only feasible solution to handle the large datasets that are currently being dealt with. They may be imperfect, they may need to be filtered or manually corrected, but despite their limits they constitute the first step in the alignment process. Manually aligning dozens of sequences of several kb in length from scratch is almost unfeasible and much more tedious than manually curating an (imperfect) alignment generated by MSA software package, which generally return alignments that are very good overall. That said, we think it is worth briefly specifying some of the key steps of MSA methods to gain insight into their current limits and explain why we believe it is crucial to manually or automatically check them before going any further into phylogenetic analysis.

### 2.1 Multiple Sequence Alignment, Multiple Aims, Multiple Truths

The aims must be clarified before designing software to solve a problem. What is the expected output of the software for a given input? Next it is essential to determine whether the software could actually be designed to meet (in a reasonable amount of time) the specific needs. Regarding multiple sequence alignment, things are not as simple as they may seem at first glance. As pointed out by Morisson in 2006, sequence alignments may be done with different objectives in mind and the ideal alignment for one objective may differ from the ideal one for another application (Morrison, 2006). He listed four distinct objectives for

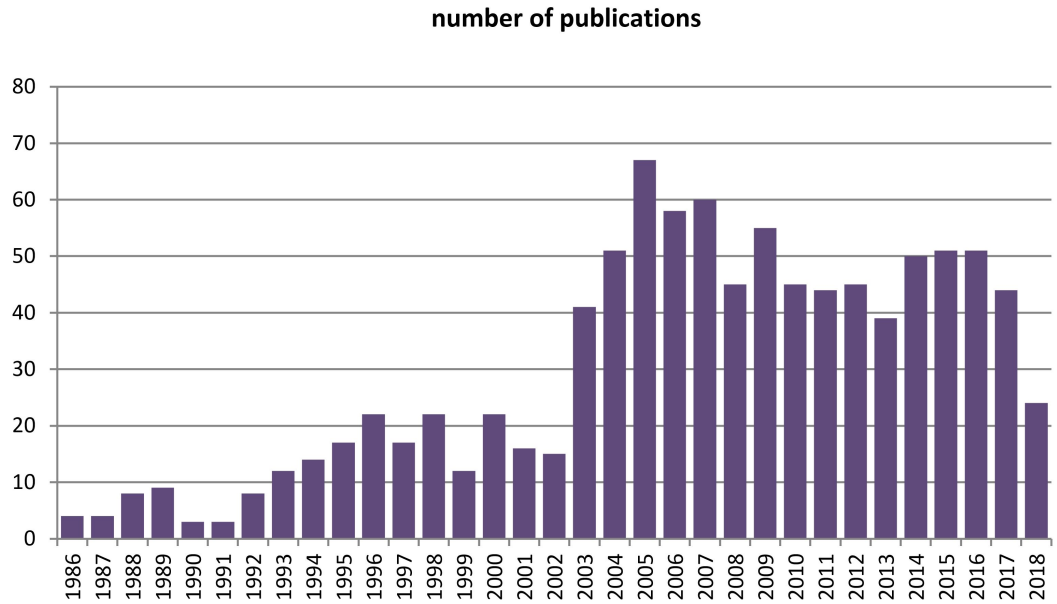


**Figure 1** Number of publications including "multiple sequence alignments" in their title. Source: Web of Science.

aligning sequences:

1. Structure prediction that aims to align residues that occupy the same 3D position in the protein
2. Sequence comparison that aims to align preserved functional motifs
3. Database searching that aims to maximize the difference between sequences that are (partially) homologous to the query and those that are not
4. Phylogeny inference that aims to align homologous residues

We have clearly positioned this chapter in the phylogenetic framework for the purposes of the present book. In this context, the aim of an MSA is to match homologous residues together. Thus, residues within the same site (i.e. column) of the MSA are assumed to be homologous, i.e. derived from a common ancestral residue. Given these homology relationships, the plausibility of an evolutionary history (phylogenetic tree) can then be estimated for every site, as well as for the whole alignment (see Chapter 1.1 [Pupko and Mayrose 2020]). For this last step, it is often necessary to assume that sites evolve independently of one another, but this is another story (see Chapter 4.6 [Zou and Zhang 2020]).

## 2.2 From verbal Aim to imperfect Objective Functions: the big gap

The aim of (phylogenetic related) sequence alignment is thus intimately related to evolution as it is striving to unravel the homology relationships of the residues. That said, the next step towards developing an MSA software is to turn this somewhat abstract objective into a practical measurement of the quality of a candidate alignment with respect to this aim. In computer science, this is called the objective function. The objective function generates a numerical score for each possible solution (here a possible alignment of the input sequences); then we just have to search among all possible solutions to find the one with the highest score. In some fields, the objective function is directly linked to the final objective. For instance, finding a way to place three new antennae in a town to maximize the area receiving a 4G signal; finding the fastest (or shortest) route to go from Paris to Brussels; or finding the longest reading frame in a DNA contig. This is seldom the case in bioinformatics, however, and a large part of the imperfection of bioinformatics method outputs is due to the inability to perfectly transform a biologist's expertise into an objective function with the highest values for the expected output.

An MSA is composed of predicted homologous residues and insertion/deletion events. Hereafter we stipulate how these events are scored (Sections 2.2.1 and 2.2.2) before discussing how pairwise alignments are evaluated based on these scoring schemes (Section 2.2.3), and we then explain how this scoring is extended to MSA (Sections 2.2.4 and 2.2.5).

### 2.2.1 Cost matrices: principles and limits

As far as amino acid sequences are concerned, the scoring schemes are derived from benchmark alignments that are supposed to be true since they are based on known 3D protein structures and/or have been manually curated. Given those golden standard alignments, the frequency $f_{ij}$ with which two residues $r_i$ and $r_j$ are observed facing each other within a same site (i.e. are homologous) may be compared with the expected frequency $f_i f_j$ of such events, given the two residue frequencies $f_i$ and $f_j$, if the residue placement is fully random. The log-odds ratio, $log(f_{ij}/f_i f_j)$, is used for this comparison and this is the key principle underlying the construction of a substitution matrix such as the PAM and BLOSUM matrix. The probability of observing two residues at the same site depends on the overall divergence of the considered sequences. There is thus not just one PAM (BLOSUM) matrix but rather a series

of such matrices. Different sequence divergence levels are considered and for each of them a dedicated matrix is built using the subset of the golden standard alignments corresponding to alignments with this degree of sequence divergence. The log-odds ratios are rounded to integer values for the purpose of speeding up the MSA software and reducing the memory space required.

This scoring approach seems mathematically well founded and in agreement with the MSA homology objective, but it has some limits. The learned scores depend on the initial golden benchmark, which could be biased or not representative enough of the sequences to be aligned. Some authors have proposed that substitution matrices could be learned for specific taxonomic groups or sequence types, e.g. for mitochondrial genes (Adachi and Hasegawa, 1996). The LG substitution matrix (Le and Gascuel, 2008) was constructed for phylogeny inference using a broad set of alignments from the (PFAM) protein families database, and the associated web server allows upload of user alignments to learn specific substitution scores for them. Apart from the impact of the initial golden benchmark, the most striking limitation of substitution matrices is that they are scored for a pair of residues and the residue homology is transitive, whereas matrix residue scoring is not. Consider for instance BLOSUM62 scores for the three following residues: phenylalanine (Phe;F), tyrosine (Tyr;Y) and histidine (His;H). Y and F are both aromatic and uncharged and their BLOSUM62 score is +2. Y and H are both polar and hydrophilic and their BLOSUM62 score is +3. However, H and F differ with regard to the four properties and have a BLOSUM62 score of -1. Phylogenetic inference methods using 20 x 20 transition matrices deal with the same problem. The CAT model implemented in phyloBayes (Lartillot and Philippe, 2004) accounts for the diversity of amino acid frequencies among sites, which is poorly captured by classical 20 x 20 matrices. This site heterogeneity is also handled by HMM profiles that are used to model sequence alignment and to search for sequences fitting this model in a database, but as far as we know this feature has yet to be implemented in MSA software.

### 2.2.2 Gap penalties: principles and arbitrary default choices

Having a protein score matrix is not sufficient to score even a simple pairwise alignment, as the scoring could also penalize gaps inserted to align sequences. The main idea behind gap scoring is that a gap interval (a maximal subsequence of consecutive gap symbols) observed in one sequence results from a deletion of multiple residues in this sequence or from the insertion of as many residues in the other sequences. The penalty (i.e. a negative score that is sometimes called *cost*) for such an event depends on the length of the gap interval, where the penalty increases with the gap interval length. From a biological viewpoint, it seems reasonable that the penalty difference between no gap and a gap of 1 residue would be higher than that between a gap interval of 1 residue and of two residues, which in turn should be higher than the penalty difference between a gap interval of 201 vs. 200 residues. The logarithmic gap cost is in line with this intuitive assumption. According to this gap penalty scheme, the cost for an interval of gaps $IG$ of length $IG[length]$ is $gap_O + log(IG[length])gap_{ext}$, where $gap_O$ is the penalty for the existence of a gap (gap opening penalty) and $gap_{ext}$ is the penalty related to the gap length (gap extension penalty). In practice most MSA methods use an affine gap penalty, where the cost for $IG$ is simply $gap_O + IG[length]gap_{ext}$, where the creation of a new gap interval is penalized more than the extension of an existing one ($|gap_O| > |gap_{ext}|$). This method penalizes the extension of an existing gap interval of one extra residue regardless of the gap interval length. The main reasons for this choice is that, in practice, the affine gap cost leads to faster computation of the alignment cost and that the gain, if any, of using a log affine gap cost is not as clearly established (Cartwright, 2006).

Note that the same gap penalty is used over the entire sequence length whereas it would be reasonable to assume that gaps would be more penalized in some parts of sequences and less penalized elsewhere, e.g. based on the 2D or 3D structure of the corresponding proteins for amino acid sequences (Madhusudhan et al., 2006). There is extensive literature on various gap penalizations and on algorithmic solutions to efficiently compute gaps, but the affine gap cost is by far the most widespread since it is simple, fast to compute and almost as accurate as other more complex penalizations.

Even the simple affine gap cost requires ($gap_O$ and $gap_{ext}$) parameter setting. Setting the relative cost of those parameters is challenging but adjusting them so that, together with the substitution matrix cost, they will generate a meaningful alignment score is even more challenging. Those two parameters are known to have a strong impact on the output alignments. Without recognised methods to set those parameters, users generally leave them at default values set by developers to perform well on specific MSA benchmarks. Having a method to automatically adjust those parameters to the input sequence dataset is a challenge that has yet to be met although it could drastically improve the MSA accuracy (Wheeler and Kececioglu, 2007). The amount of curated alignments currently available and the progress that has been achieved on machine learning methods enhance the possibilities of learning those parameters via deep learning approaches.

### 2.2.3 Finding the optimal alignment for two sequences

A two sequence alignment consists of sites where two residues are facing each other (match) and of gap intervals or *indels* (where a sequence of consecutive residues in one sequence is facing gap characters in the other). The overall pairwise alignment score is simply the sum of the match scores (given by the selected cost matrix) plus the gap penalty cost. Figure 2 provides a detailed example of the scoring procedure for an alignment of two sequences $S_i$ and $S_j$. The overall score of this alignment is +2. If we slightly modify this alignment by moving the amino acid I of $S_j$ in front of the amino acid Q of $S_i$, we get an alternative alignment that has an overall score of -3 (with the +2 of L facing I being replaced by the -3 of Q facing I). Using this scoring framework, an optimal alignment (one of those with the highest score) can be found efficiently in a time that is proportional to the product of the input sequence lengths. If $l_1$ and $l_2$ denote the input sequence lengths, the time needed to solve this problem can be viewed as a polynomial function that tends to be proportional to $l_1$ times $l_2$, and the algorithm is said to have a time complexity of $O(l_1 l_2)$. This means that the number of operations needed to solve this problem is of the order of magnitude of $l_1 l_2$. The algorithmic solution, involving dynamic programming, is detailed in many algorithmic/bioinformatics textbook. It is derived from the solution of the problem of identifying the longest common subsequence (LCS) of two sequences, i.e. the longest series of elements that appear in the same order in the two input sequences. This simpler problem, although easy to solve for two sequences, is known to be NP-complete (it cannot be guaranteed that an optimal solution could be found in a reasonable amount of time) for more sequences (Wang and Jiang, 1994). Indeed, an exact solution for $n$ sequences of length $l$ has a time complexity of $O(l^n)$, i.e., requires a number of operations that grows exponentially with respect to the number of sequences, which would be unfeasible for more than a few short sequences.

### 2.2.4 What is an optimal alignment for more than two sequences?

Defining an objective function that reflects the overall quality of an alignment of several sequences is not straightforward. Let us consider a simple example to understand where

the difficulties lie. Given an alignment of 10 sequences, suppose that a site consists of eight tyrosines (Y) and two histidines (H). If the two sequences containing histidine are sister taxa, then this 8Y2H pattern could be explained by a single Y to H mutation, otherwise two separate mutations are needed. The objective function of an MSA could then be expected to assign a site cost that depends not only on the amino acids present at this site but also on the underlying phylogeny of the corresponding species. The same holds true for gaps, except that things are much more complicated for gaps as they spread along several sites and can overlap so that, even if the underlying phylogeny is known, finding the most parsimonious scenario for gaps would not be that easy.

Given the above remarks, it would be tempting to simultaneously estimate both the MSA alignment and the underlying phylogeny. Many attempts have been made in this direction, i.e. in the parsimony framework (Wheeler, 1996, 2003), as well as in the Bayesian (Herman et al., 2014; Lunter et al., 2005) and likelihood frameworks (Fleissner et al., 2005; Thorne and Kishino, 1992). The POY method (Wheeler, 1996, 2003) was one of the first available methods for simultaneous sequence alignment and phylogeny inference. It was the focus of some attention, as this parsimonious based method seemed to be elegant and perform well. More in depth testing concluded the practical superiority of the conventional two-step approach that first uses MSA software to produce an alignment and then uses it as input for phylogenetic inference (Ogden and Rosenberg, 2007). Indeed, 99.95% of the alignments produced by ClustalW (Thompson et al., 1994) were better than that produced by POY.

In practice, very few of the widely used MSA methods actually account for the underlying evolutionary process, Prank (Loytynoja and Goldman, 2008) being a notable exception.

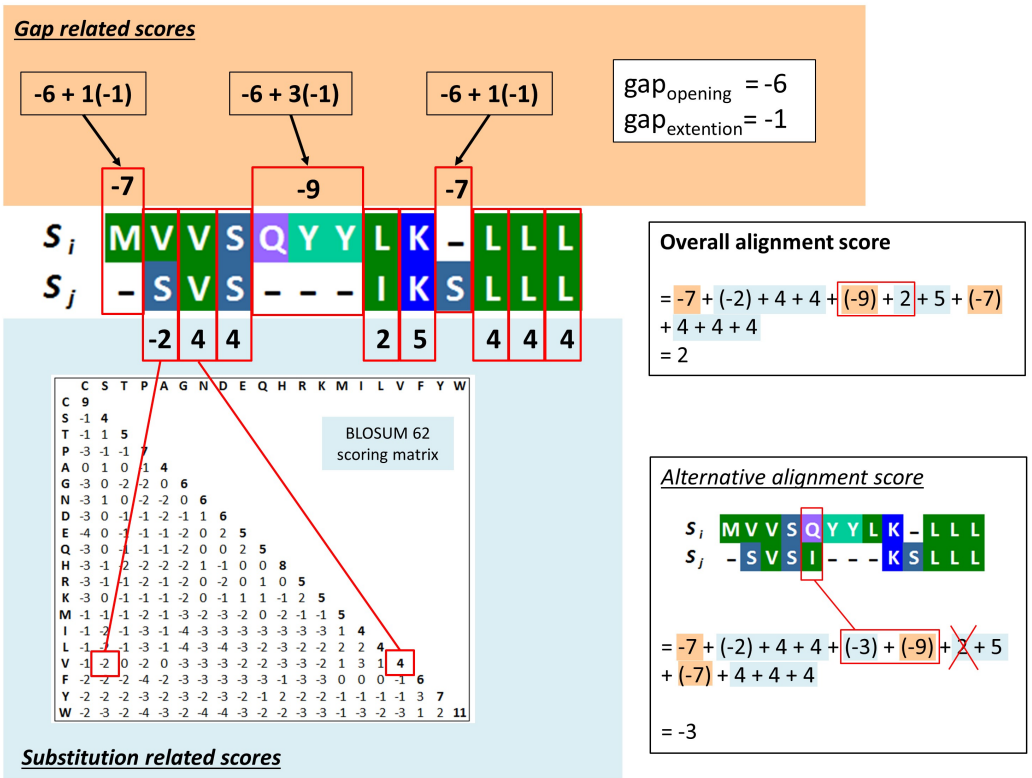


**Figure 2** Scoring a pairwise alignment, a detailed example.

There are at least three main reasons for this. Aligning sequences while accounting for the underlying process is harder and much more time consuming overall. Secondly, conditioning the alignment score to an evolutionary scenario can bias the alignment toward this scenario. This latter point is not a problem when the phylogeny is already known or when the main goal of the alignment is not to resolve the phylogeny but rather, for instance, to detect selection footprints. Prank software has proved to be very efficient for this task. In cases where the phylogeny has to be inferred, phylogeny inference routines included within the alignment software are usually much less powerful than dedicated software. For instance, Prank relies on the NJ algorithm, which is a reasonably good distance method but cannot at all compete with up to date probabilistic methods (see Chapters 1.2 and 1.4 [Stamatakis and Kozlov 2020; Lartillot 2020]). There is thus a risk that such methods could generate biased alignments that, by construction, would favour the erroneous phylogeny used to score the alignment (`https://code.google.com/archive/p/prank-msa/wikis/ExplanationDifferences.wiki`). Thirdly, alignments are made separately for each locus whereas phylogenetic inference may consider multiple loci simultaneously.

### 2.2.5 Objective functions for MSA: the SP-score and its numerous variants

As detailed in the previous section, most MSA software uses an objective function that does not rely on the evolutionary framework. Almost all MSA software uses a variant of the sum of pair scores, or SP-score in short. Given a multiple alignment $\mathcal{A}$, if we consider only its first two rows, then we get a pairwise alignment of the two first sequences aligned in $\mathcal{A}$ that can be scored using, for instance, the pairwise alignment scoring described in Section 2.2.3. The SP-score is basiacally the sum of pairwise scores across all sequence pairs, as illustrated in Figure 3.

More formally, the SP-score of an MSA is obtained by considering all possible pairwise alignments it induces: given two sequences $S_i$ and $S_j$ of the MSA $\mathcal{A}$, the corresponding induced pairwise alignment $(\mathcal{A}|S_i, S_j)$ consists of the two sequences $S_i^{'}$ and $S_j^{'}$ obtained by removing the '-' of $S_i$ (resp. $S_j$) whenever $S_j$ (resp. $S_i$) also has a gap at this position/site (see Figure 3 for an example). Conventional algorithms to compute the SP-score of an alignment $\mathcal{A}$, made of $L$ sites and $n$ sequences, proceed by summing up the pairwise scores of its $\binom{n}{2}$ induced pairwise alignments (of length proportional to $L$) and hence have an overall time complexity of $O(n^2L)$. Faster algorithms are now available for both the general gap penalty and the affine gap cost cases (Ranwez, 2016). Indeed, it turns out that this latter can be solved with a simple and efficient algorithm having a time complexity of $O(nL)$.

An alternative approach is to build an optimal pairwise alignment $\mathcal{A}_{ij}$ for each sequence pair $S_i$ and $S_j$ of the set $\mathbb{S}$ of sequences to be aligned. This is possible as the pairwise alignment problem can be solved in polynomial time. For a given MSA $\mathcal{A}$, the pairwise restriction $\mathcal{A}|S_i, S_j$ can then be compared to the computed optimal alignment $\mathcal{A}_{ij}$, and the closer they are the better it is because the multiple alignment is more consistent with the optimal pairwise alignments. An alternative way of scoring the MSA $\mathcal{A}$, which here we call SP-sym, is thus to sum up the similarity between (a sample of) its $\binom{n}{2}$ induced pairwise alignments and their optimal pairwise counterparts. This scoring scheme was first introduced in T-coffee software (Notredame et al., 2000). Based on this scoring, the searched MSA can be seen as the median of the considered optimal pairwise alignments.

Numerous MSA software packages including Clustal, MUSCLE and MAFFT use the SP-score framework, but they introduce variations in an effort to improve it. Some of them use different substitution matrices for the different pairwise comparisons, while others use a

weighted version of the SP-score, where all pairwise comparisons do not equally contribute to the overall SP-score, the gap scoring can also vary along the sequences based on the 2D structural information or on the gap frequencies in the pairwise alignments of the considered dataset. The most recent releases of MAFFT use a combination of SP-score and SP-sym to build their objective function. Our goal here is not to extensively review those MSA scoring variants but rather to outline the key underlying principles in order to highlight that those scoring schemes are both powerful – each newly launched software package brings new improvements – and imperfect – as they mostly overlook (for sound reasons) the evolutionary relationships of the compared sequences.

## 2.3 Heuristic search for optimal MSAs: the more you get, the worst the search

Finding an MSA with an optimal SP-score is known to be NP-complete (Wang and Jiang, 1994). An exact solution to this problem can thus only be found when dealing with a small number of short sequences. Hence, all widely used MSA software relies on a heuristic search to find an MSA that has a reasonably high score. This search involves two distinct phases: the first is to build an initial MSA of the input sequences, while the second is to improve this initial MSA through iterative refinement. Both steps extensively rely on the idea of aligning two alignments, i.e., merging two previously aligned, disjoint subsets of sequences into a new, larger alignment.

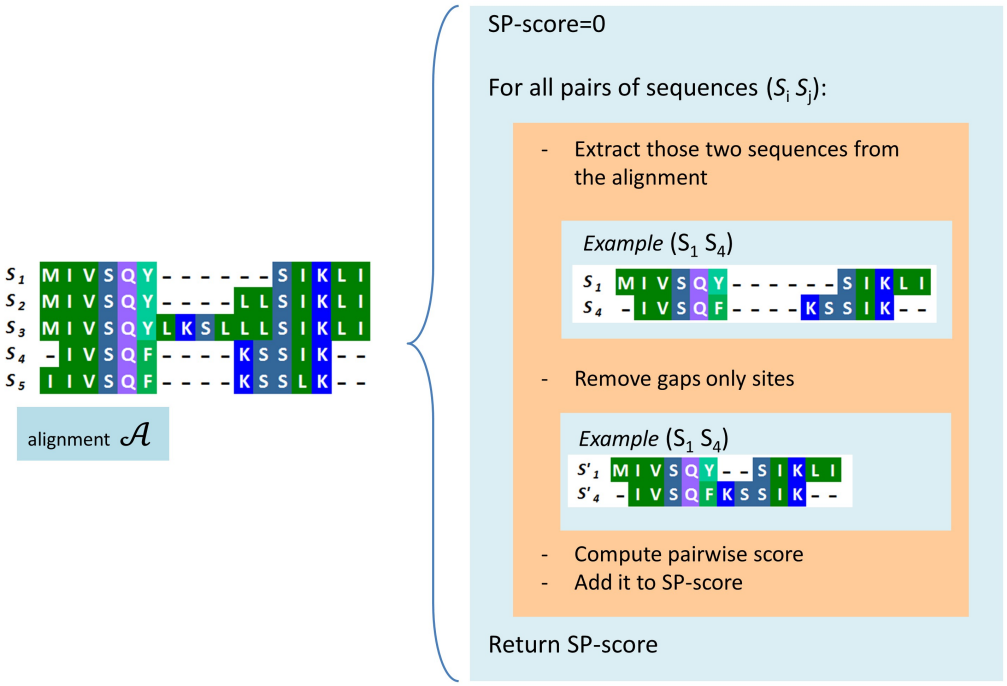


**Figure 3** SP-score computation principle for an MSA.

### 2.3.1 Aligning two alignments and gap penalty approximations

Given an alignment $\mathcal{A}_1$ of a set of sequences $\mathbb{S}_1$ and an alignment $\mathcal{A}_2$ of a disjoint set of sequences $\mathbb{S}_2$, an alignment $\mathcal{A}$ of $\mathbb{S}_1 \cup \mathbb{S}_2$ may be obtained by aligning $\mathcal{A}_1$ and $\mathcal{A}_2$. This alignment task aims to identify pairs of homologous sites of the two input alignments and position them in front of one another in a global alignment. This could be seen as a search for the best alignment of sequences of $\mathbb{S}_1 \cup \mathbb{S}_2$ that respect the homology relationships present in $\mathcal{A}_1$ and $\mathcal{A}_2$, i.e. such that $(\mathcal{A} \mid \mathbb{S}_1) = \mathcal{A}_1$ and $(\mathcal{A} \mid \mathbb{S}_2) = \mathcal{A}_2$.

A naive way to do this, while also showcasing the key idea of the method, is presented below and illustrated in Figure 4. First, for the alignment $\mathcal{A}_1$, consisting of $n_1$ sequences and $l_1$ sites, a consensus sequence $S_{c1}$ of length $l_1$ such that the amino acid at position $k$ of this sequence is one of the most frequent at site $k$ of $\mathcal{A}_1$, is built. Then we proceed similarly to build $S_{c2}$ from the alignment $\mathcal{A}_2$, consisting of $n_2$ sequences and $l_2$ sites. In a second step, the consensus sequences $S_{c1}$ and $S_{c2}$ are aligned. In a third step, this resulting alignment is used to derive homologous sites of $\mathcal{A}_1$ and $\mathcal{A}_2$ by assuming that every time two amino acids of $S_{c1}$ and $S_{c2}$ are facing each other the corresponding $\mathcal{A}_1$ and $\mathcal{A}_2$ sites are homologous. To visualize the process, we could imagine that each amino acid of $S_{c1}$ and $S_{c2}$ has a skewer of amino acids attached to it that correspond to the site it summarizes. Furthermore, we imagine that each gap of $S_{c1}$ (resp. $S_{c2}$) has a skewer of $n_1$ (resp. $n_2$) gaps attached to it. Then, given the alignment of $S_{c1}$ and $S_{c2}$, we can build a global alignment of $\mathcal{A}_1$ and $\mathcal{A}_2$ by merging the two skewers facing each other.

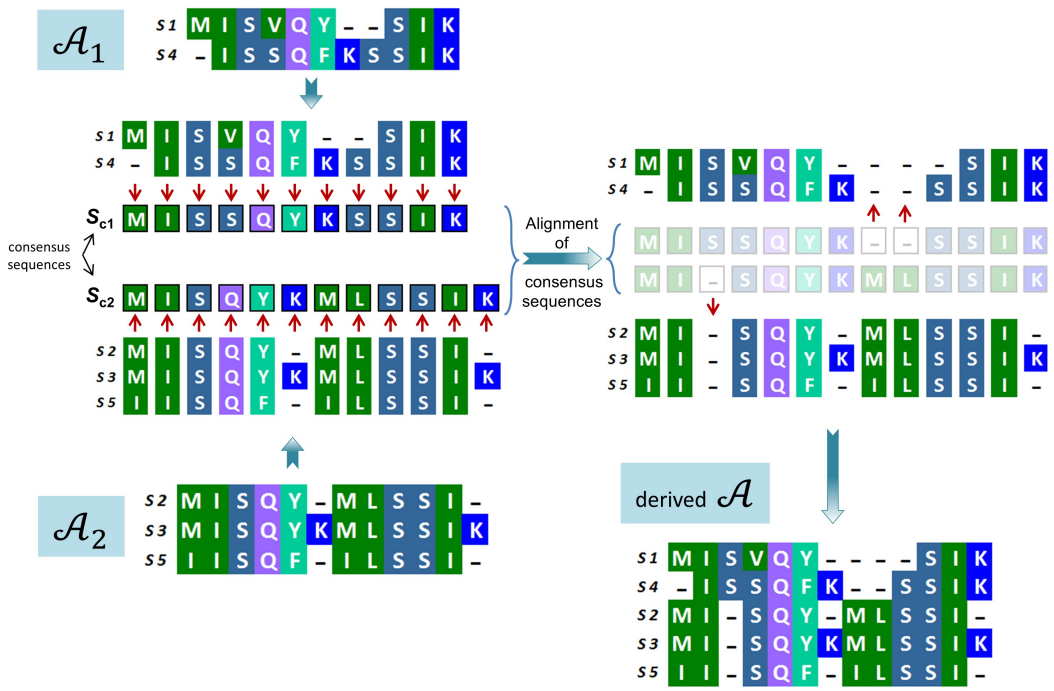


**Figure 4** Aligning two alignments, a naive approach.

The naive approach described above is not the one generally used, as summarizing all site information with a single character is a drastic compression that generates a major

information loss. In the example of Figure 4, the third site of $\mathcal{A}_2$ would be better aligned (at least w.r.t. the SP-score) if placed in front of the third one of $\mathcal{A}_1$, but this cannot be done when only the consensus sequences $S_{c1}$ and $S_{c2}$ are used. The dynamic programming approach to achieve optimal two-sequence alignments could be extended to better align two alignments, but it would no longer be certain that the resulting generalization would return an optimal solution. The SP-score of an alignment can be split into three parts: (1) $SP_{subst}$: the part induced by amino acid homology and computed using the substitution score matrix; (2) $SP_{gext}$: the part induced by gap extensions; and (3) $SP_{go}$: the part induced by gap openings. The first two are easy to estimate even when aligning alignments. If we consider putting site $i$ of $\mathcal{A}_1$ in front of site $j$ of $\mathcal{A}_2$, we can readily see how this would impact $SP_{subst}$, which could be accurately evaluated just by knowing the number of each amino acid present at sites $i$ and $j$. Similarly, the impact on $SP_{gext}$ may be assessed simply on the basis of the number of gap and non-gap characters at both sites. The situation is much more complicated regarding gap openings since the sites can no longer be considered independently. By considering two consecutive sites, it is possible to determine the upper bound (pessimistic gap count) and lower bound (optimistic gap count) regarding the number of gaps that would be opened by an alignment operation (e.g. placing site $i_{k1}$ in front of $j_{k2}$), but it is not sure that exact counts of the resulting gap openings would always be obtained (Altschul, 1989).

### 2.3.2 Building an initial MSA: not every tree is a phylogeny

Several heuristic methods are available to build an initial MSA. The most widespread one involves progressive alignment construction guided by hierarchical sequence clustering (Feng and Doolittle, 1987). This method follows a greedy strategy (the homology established at some point is never questioned afterwards) whereby a larger alignment is built by aligning two smaller ones until all input sequences are jointly aligned. Figure 5 illustrates the procedure, as detailed below.

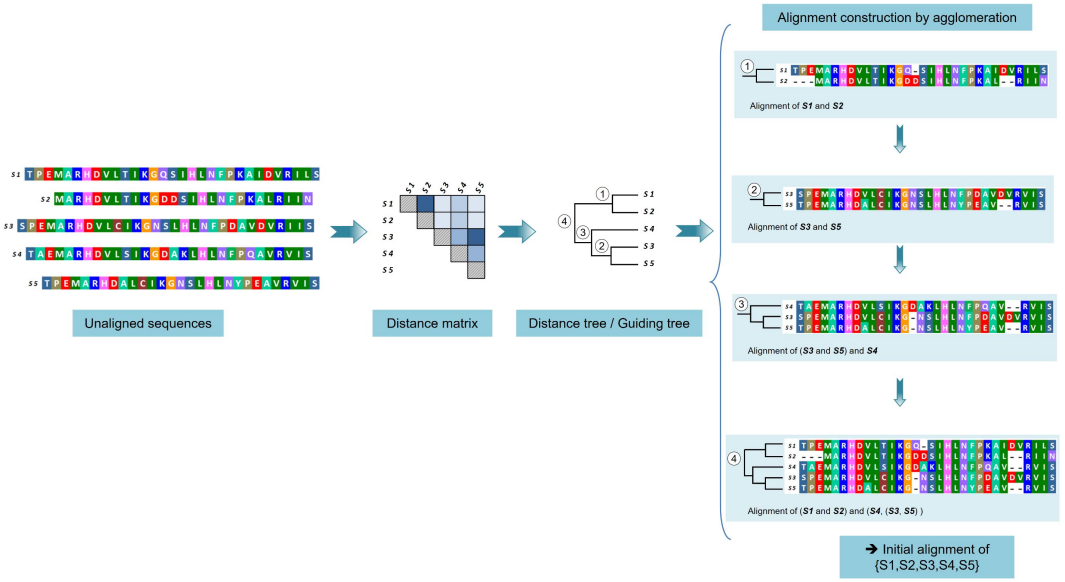


**Figure 5** Building an initial MSA using a guiding tree.

The advantage of using this greedy approach is that it is fast, since decisions taken at

one step are never questioned afterwards. The downside is that errors made in the early stages of the process, which condition subsequent choices, may have a devastating impact on the final result. To mitigate this problem, it is thus preferable to start by the easiest tasks that should be less error prone. If two sequences are fully identical, it is really easy to align them and the right solution is much better than any alternative possibility. If the two sequences differ by a single deletion of a few amino acids, the task remains straightforward but there may be some uncertainty concerning the alignment of amino acids at the frontier of the deletion. If one of the sequences is half the length of the other one and no motif longer than four consecutive amino acids is shared by the two sequences, it is quite likely that the alignment will contain (many) errors. The progressive alignment strategy thus first aims to jointly align the most similar sequences as it is an easier and less error prone task. To this end, hierarchical sequence clustering, a so-called guiding tree, is done so as to group the most similar sequences together. Each leaf of this tree is thus associated with an input sequence (i.e. a trivial alignment) and each internal node will then be associated with the alignment of the sequence below it. Those alignments are built by processing the internal tree nodes from tips to the root, which ensures that a node is always processed after its two children. An internal node is simply processed by producing the alignment associated with this node, i.e., by aligning the two alignments of its two children nodes. At the end of the process, the root node is associated with an alignment of the whole sequence set. Note that the guiding tree should not be confused with the input sequence phylogeny. There is surely a link between sequence similarity and species relationships, but most similar sequences are not necessarily derived from the most related species as evolutionary rates can vary during evolution. For a given gene, a human sequence may be more similar to a cow sequence than to a mouse sequence because mouse genes evolve faster. In such cases, it makes sense that the guiding tree pools human and cow sequences so as to postpone the harder task of aligning a divergent mouse sequence with the others, even though humans and mice are known to be closer relatives than humans and cows. In such a situation, the guide tree justifiably differs from the true species tree.

The guiding tree is built based on a distance matrix that provides a measurement of the similarity/divergence between any two sequences of the input set. This similarity can be derived using the pairwise alignment of the two concerned sequences, but performing $\binom{n}{2}$ pairwise alignments is extremely time consuming and most MSA software packages rely on a pairwise distance estimation based on $k$-mer contents which are much faster to obtain – a $k$-mer is a set of consecutive $k$ amino acids in the sequence. The sequence similarity is assumed to increase as the proportion of shared $k$-mer increases. Finally, note that once a first MSA alignment is obtained in this way, some software uses it to derive an improved distance matrix based on the pairwise sequence alignment induced by this first MSA. Then they build a new guiding tree and infer a new MSA using it. This idea of repeating the progressive alignment construction with an improved distance matrix was introduced, to our knowledge, in the first release of MUSCLE (Edgar, 2004). The initial MSA produced at this stage is highly important since not only is it the starting point for the subsequent alignment refinement stage but also its associated guiding tree will also guide the refinement search.

### 2.3.3 Optimizing the initial MSA and why this is better done with fewer sequences

The progressive alignment step is explained in great detail in many courses and textbooks whereas, despite its importance in practice, the refinement step is often rapidly described at best. Indeed, an alignment produced just using the progressive strategy can be of very poor

quality due to the greedy approach used to build it.

For most software, this refinement step relies on a "hill climbing strategy" to optimize the objective function: variants of the current alignments are produced and each time a variant better than the current solution is encountered it becomes the new current solution. The optimization process, as illustrated in Figure 6, stops when no better variant is found and the current solution thus becomes a local optimum.

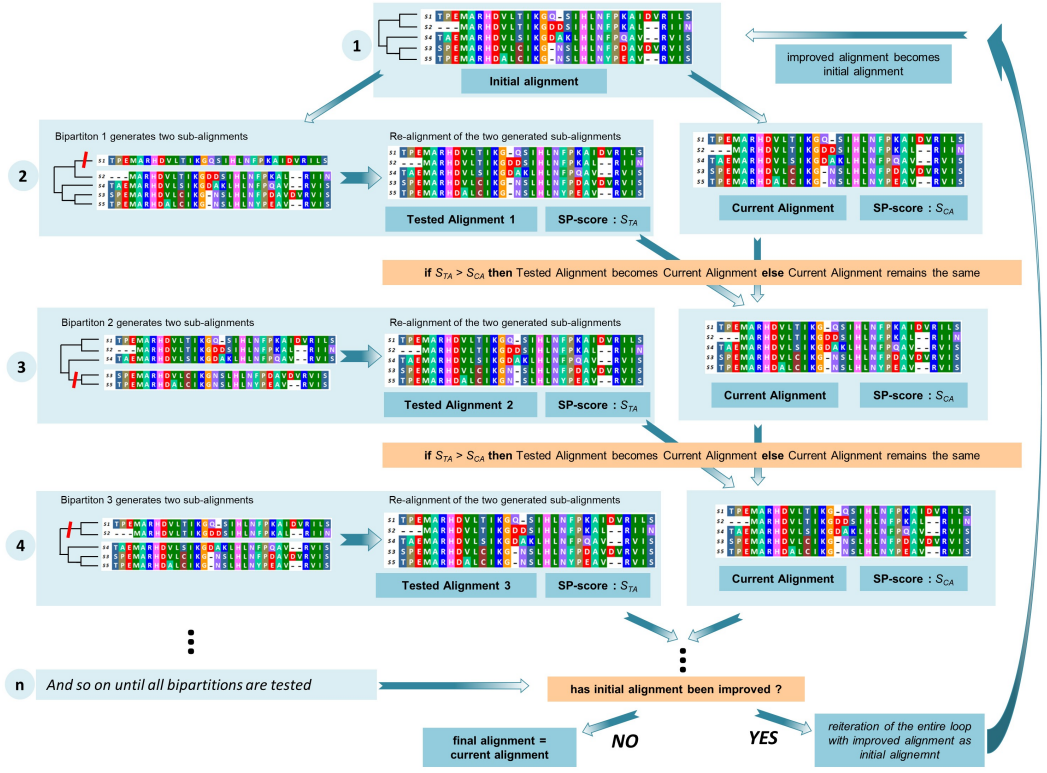


**Figure 6** Schematic representation of the 2-cut strategy used to refine an initial MSA.

A maximum number of iterations can also be set to reduce the computation time. The way the variants to test are produced can vary, but in many cases this is done by splitting the alignment in two and realigning the two resulting subalignments. This is sometimes called the 2-cut refinement strategy. Given an alignment $\mathcal{A}$ of a set of sequences $\mathbb{S}$ and a bipartition of $\mathbb{S}$ into $\mathbb{S}_1$ and $\mathbb{S}_2$ ($\mathbb{S}_1 \cup \mathbb{S}_2 = \mathbb{S}$ and $\mathbb{S}_1 \cap \mathbb{S}_2 = \emptyset$), the two induced alignments $\mathcal{A}\,|\mathbb{S}_1$ and $\mathcal{A}\,|\mathbb{S}_2$ are aligned to get a variant of $\mathcal{A}$ that will replace it if and only if it has a better score. As the number of possible bipartitions of $\mathbb{S}$ ($2^{n-1}$) grows exponentially with the number of sequences $n$, it is impossible to test all of them, except for datasets with less than a handful of sequences. The guiding tree is then used to define the bipartitions to be tested, and a refinement loop consists of testing all bipartitions corresponding to a branch of the current guiding tree. The current alignment is updated every time an alignment variant with a better score is found, and hence it could be changed several times during each loop. The process stops at the end of a loop if no better alignment has been found, or otherwise a new loop starts. The guiding tree may or not be updated after each loop based on the new best alignment found at that point.

The number of bipartitions/branches within a tree of $n$ sequences is $2n-3$ whereas the

total number of possible bipartitions among $n$ sequences is $2^{n-1}$. The fraction of bipartitions that are considered to try to improve the alignment is thus $(2n-3)/2^{n-1}$, i.e. a fraction that exponentially tends toward zero as $n$ grows. For 10 sequences, we test $\sim 3\%$ of all possible bipartitions but only about $\sim 3 \times 10^{-26}$ % for 100 sequences. If some sequences $\mathbb{S}_1$ share a common gap that is misplaced with respect to the other sequences $\mathbb{S}_2$, the chances of precisely testing the alignment variant associated with bipartition $\mathbb{S}_1 \mid \mathbb{S}_2$, and hence correcting the misplaced gaps, are thus almost inexistant for a 100-sequence dataset. This difficulty is illustrated in Figure 7, using a small and simple example where an alignment cannot be improved with the current guiding tree despite the fact that a better alignment obviously exists.

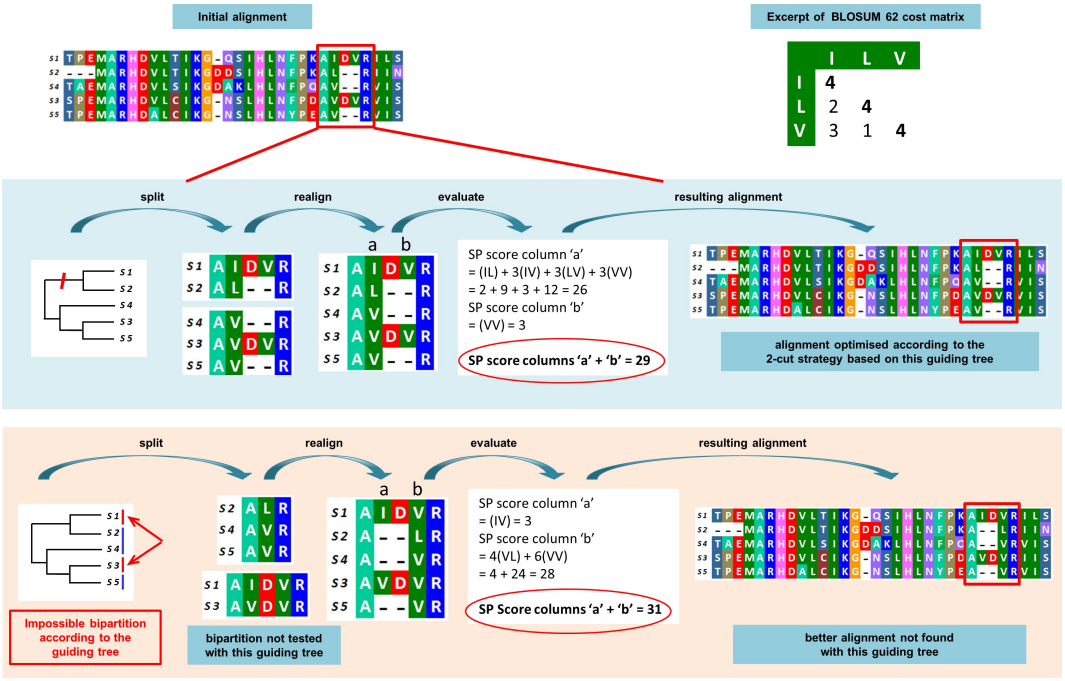


**Figure 7** Limits of MSA refinement when bipartitions are made according to the guiding tree. Starting with the initial alignment (top right), the better alignment (depicted on the bottom right) cannot be found with the current guiding tree (depicted on the left) since this tree does not contain the bipartition $\{S1, S3\}|\{S2, S4, S5\}$ that would help find a better alignment.

This somewhat pessimistic view has to be qualified by the fact that, rather than testing randomly picked bipartitions, we test those observed in the guiding tree, which are expected to be more promising bipartitions than random ones. Yet, this phenomenon cannot be overlooked, and is probably one of the reasons that led R. Edgar to state in the MUSCLE 3.8 user guide that "If you have thousands of sequences, then attempting to create a multiple alignment is dubious for many technical reasons. It may be better to cluster first, then align the reduced set of sequences".

## 3 Filtering alignments, less is more, well more or less

Alignments are the foundation upon which molecular phylogenetic analyses rely. In this part we will consider an alignment obtained by any MSA method as a starting point (alignment methods and optimization were discussed earlier). Once the alignment has been achieved, it intrinsically contains all relevant information that will be used by evolutionary models to reconstruct the history of the aligned sequences, and therefore any error in the alignment could seriously affect the inferred phylogeny.

As explained in the previous sections, MSA methods have numerous shortcomings (they rely on heuristic searches guided by imperfect objective functions). It is thus inevitable that in most cases their output, even if satisfactory overall, is tainted with errors. Many software programs have been designed with the aim of filtering MSAs in order to keep only their most reliable regions. This filtering is done by removing sites, sequences or masking residues (replacing them by the gap symbol '-' or by a symbol representing ambiguity '?', 'N' or 'X'). Filtering MSA would be a reasonable thing to do, but we have to avoid throwing out the baby with the bathwater. It is important to be sure that the filtering does not remove the signal along with the noise caused by the misaligned regions. For MSA filtering, as for any signal filtering process conducted to improve the image or sound quality for instance, the balance between noise reduction and signal loss is key. This balance partly depends on the planned downstream analysis. For example, misaligned regions impacting a single sequence at one time will have little impact on the phylogeny inference, apart from terminal branch length estimations, but they will induce many false positives when searching for loci under positive selection. The efficacy of cleaning methods have been a highly topical issue in recent years with the advent of high-throughput sequencing methods that have dramatically increased dataset sizes, up to the size of whole genomes. MSA manual curation is still applied in small or medium size datasets but is impossible for larger ones. Moreover, the crucial reproducibility issue is another reason for developing automatic MSA cleaning methods.

We consider that filtering methods can be subdivided into two main categories. The first one contains methods that filter MSA by entirely removing some sites or sequences from the MSA. These methods give you only two choices per site and sequence: you either "take it or leave it", which is why we have called them TILI-filtering methods. The second category contains MSA filtering methods that work by masking residues (replacing them by the gap character '-' or by a symbol representing ambiguity '?', 'N' or 'X', depending on the sequence type). Such methods take pieces of information from a site or sequence, while excluding the rest of it, so we have called these picky-filtering methods. Before delving deeper into filtering methods, it would be worthwhile to briefly outline the different cases of poor quality alignment and their causes, along with some possible upstream remedies.

### 3.1 What are the problematic regions of an alignment?

There are two problematic regions within an MSA, which we call poorly informative regions and wrongly aligned regions. Poorly informative regions are regions where most of the sites contain many gaps. It is hard to say whether the alignment is correct in these regions but, regardless, these regions carry little evolutionary signal. On the other hand, wrongly aligned regions can be defined as regions for which the hypothesis whereby aligned residues would have a common ancestor is unlikely to be correct. Anyone who has ever aligned sequences automatically and carefully looked into the generated alignments, has witnessed such regions. Many alignments include both poorly informative and wrongly aligned regions, or their combination. As specified below, problematic regions may have diverse impacts on the

downstream analysis depending on their characteristics.

### 3.1.1 Patchy regions

One group of poorly informative alignment regions could be described as 'patchy'. This is when the alignment algorithm has inserted so many gaps that there are long stretches of sites at which gaps predominate (Figure 8a). This situation is obviously caused by the presence of highly divergent regions in the sequence. These patchy regions do not necessarily induce errors in the tree topologies, as they basically do not contain any phylogenetic signals but they may disrupt the bootstrap procedures. Indeed, if too many sites are sampled in those regions for some bootstrap replicates, it may be impossible to compute a tree for those datasets and the inference program may crash without many clues to enable you to spot the problem.

### 3.1.2 Regions in the vicinity of patchy regions

In the vicinity of patchy regions, you may also find sites for which a greater number of sequences are present (short fragments, like "islands"). In these cases the homology between aligned fragments is often doubtful (Figure 8b). This is a common situation at the 5' and 3' ends of genes, i.e. regions that are known to diverge faster between lineages, in addition to often being subject to erroneous annotations. Such situations are more likely to happen if the number of analyzed sequences is large.

### 3.1.3 Misaligned regions

In some regions, suboptimal alignments, as shown in (Figure 8c), can be observed (see Section 2.3 for further details). In these cases, any phylogenetic signals will be blurred. These errors are also typically caused by a large sample size in low similarity regions. Indeed, the sequences in the example of Figure 8c were among the most distantly related ones in the dataset, and the algorithm failed to optimize the alignment of those small isolated regions.

### 3.1.4 Low complexity regions

Repeated characteristics, or small repeated motifs, can lead to another kind of wrongly aligned region, as presented in Figure 8d. The region depicted in this figure is a transmembrane domain, and as such requires a high prevalence of hydrophobic residues. Because of the relatively small number of hydrophobic amino-acids (mainly Ala, Ile, Leu, Met, Phe and Val), this region is particularly prone to reverse substitutions and convergent evolution (homoplasy), resulting in disordered repetitive stretches of hydrophobic amino acids. The resulting nonsense or wrong alignment may distort the phylogenetic signal.

## 3.2 What causes problematic MSA regions?

Aligning highly similar sequences is quite easy. Note that this idea underlies the greedy MSA strategy which, by using a guiding tree, aligns the most similar sequences first and postpones the harder task of aligning divergent ones. MSA software hence generally produces high quality MSAs when the input sequences are highly similar over their entire length. Difficulties arise when sequences are, locally or globally, highly divergent, possibly due to annotation errors. Short or long motif repeats also cause confusion since it may be hard to

identify homologous motifs, and long indels are particularly challenging for MSA software since an affine gap cost penalty does not fit well such situations.

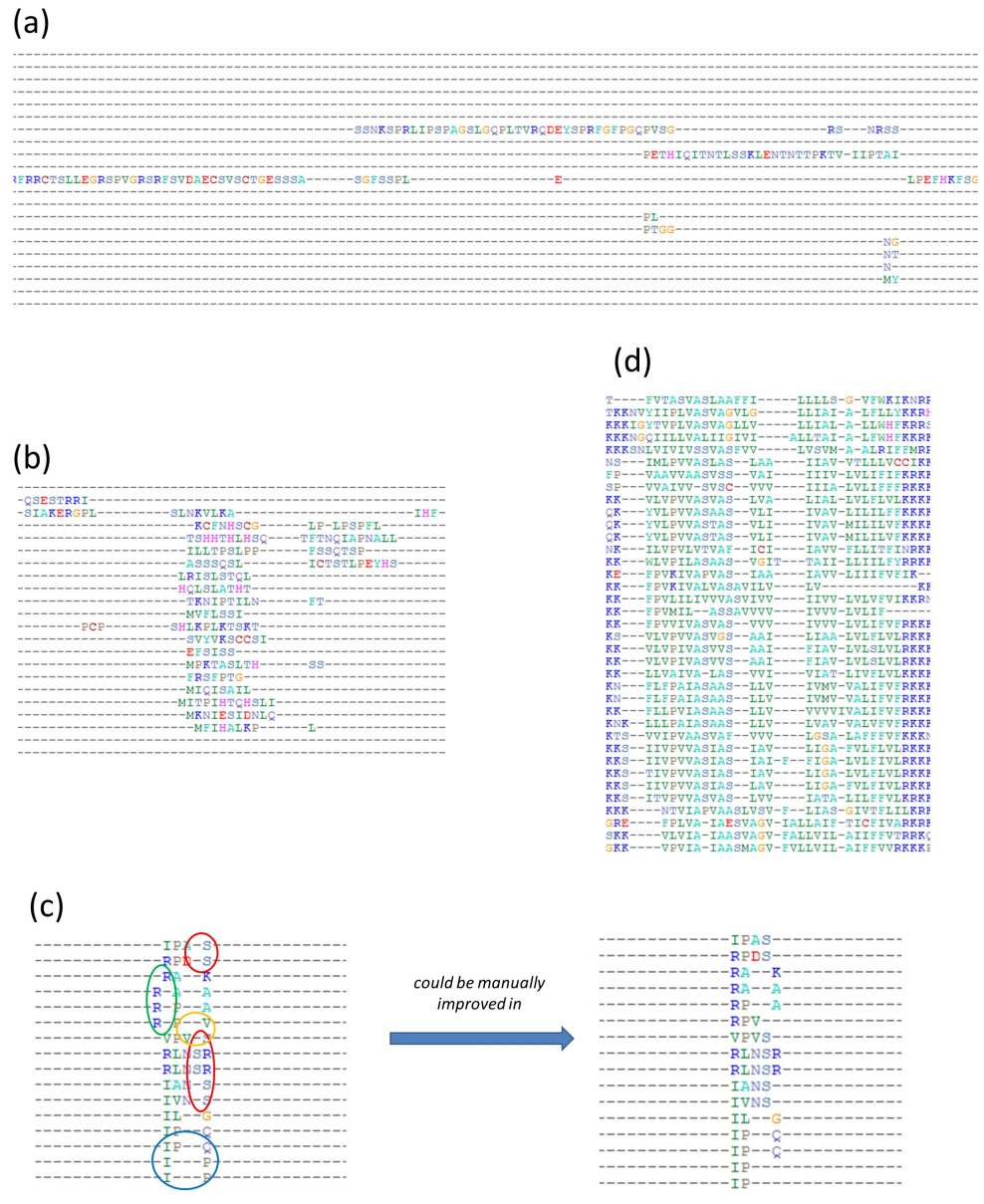


**Figure 8** Examples of problematic regions of an MSA.

### 3.2.1 Highly divergent or non-homologous sequence fragments

As alignment algorithms will always be able to propose an alignment of sequences even when they are only remotely related, the question of the reliability of the produced alignment arises. Note that even when sequences are too divergent, or not even homologous, MSA software will still produce an alignment. However, some phylogeny inference methods will refuse to take this alignment as an input and will produce an error message indicating that the input sequences are too divergent. This occurs, for instance, with most distance methods when the distance matrix, built from the input MSA, contains missing values. Indeed, the pairwise distance cannot be calculated between two sequences that do not share homologous residues according to the input MSA (i.e. no residues placed on the same site). This could indicate that the two sequences are only partially available (the 3' part is missing from one and the 5' end from the other). In this case it is fine to use alternative phylogeny methods (not based on distances). Alternatively, it could also indicate that the two sequences are indeed non-homologous and should not be simultaneously present in this MSA and that this alignment is hence unreliable.

When sequences are highly divergent, they contain little (if any) information about homology relationship between residues. The MSA that could be inferred from such a dataset would hence be almost random, just as would be a phylogeny only inferred based on saturated sites. Even when sequences are generally similar, the question of alignment reliability remains relevant given that in most cases similarity levels are not homogenous all along the sequences being compared. Heterogeneous similarity levels among sequences are often observed when analyzing gene family. Gene families are often defined by the presence of a conserved active domain, while the rest of the protein, since it is a lot less constrained, may evolve at much faster rates and rapidly diverge.

Somewhere along the gradient from highly similar sequences to highly divergent sequences, there is a critical point beyond which to align sequences is not possible, or biologically meaningful - too many substitutions or indels have occurred. Beyond this point, alignment makes no sense since it is impossible to guarantee that the aligned positions are derived from an ancestral state. Just before this point, computational limitations of alignment methodologies induce errors and unreliable alignment regions may be frequent.

### 3.2.2 High-throughput sequencing, annotation errors and possible remedies

Dealing with high-throughput sequencing to conduct phylogenomic analysis introduces problems that are absent from smaller manually curated datasets.

Firstly, sequencing errors may occur, especially in the presence of homopolymers. For DNA sequences this would lead to small indel events when aligning sequences at the nucleotide level, but if they are coding sequences then their translation into amino acids will be erroneous as those events may induce artefactual frameshifts. Artefactual frameshifts can also be caused by erroneous exon boundary annotations. MACSE, an MSA software program that explicitly accounts for the underlying codon structure of protein-coding nucleotide sequences, may be used to correctly handle such coding sequences (Ranwez et al., 2011, 2018). Its unique features can help build reliable codon alignments even in the presence of (real or apparent) frameshifts.

Secondly, errors in homology annotations may lead to the inclusion of sequences erroneously considered as being homologous to others in the MSA. In this case, some MSA filtering methods may eventually be able to detect them and filter them out, but their mere

presence during the MSA process could significantly slow down the alignment process and alter the final result. If such unrelated sequences are detected, it may be worth re-aligning the remaining sequences once the rogue sequences have been removed. A similar problem arises (locally) when different splicing variants are mixed. The presence of (long fragments of) non-homologous sequences may seriously slow down and alter the alignment process. It may hence be worth trying to detect them before performing the actual alignment. This can be done using the *TrimNonHomologousFragments* subprogram of MACSE or the PREQUAL program (Whelan et al., 2018), which were specifically developed to remove long sequence fragments that are unrelated to other sequences.

A third case is related to orthology annotation errors. This more tricky case occurs when all considered sequences are homologous but some are erroneously considered as being orthologous (derived from ancestral copy by speciation) while actually being paralogous (derived from ancestral copy by duplication). When the objective is to reconstruct the species phylogeny, mixing orthologous and paralogous sequences can lead to erroneous conclusions (Chapter 2.4 [Fernández et al. 2020]). Indeed, such errors can strongly impact the phylogeny inferred since species will tend to group depending on the gene copy used to represent them rather than on their "relatedness". MSA filtering and the MACSE *TrimNonHomologousFragments* subprogram are both unhelpful in this case as the sequences are homologous and correctly aligned (residues present at the same site are homologous). However, in a phylogenomic context, and when no horizontal gene transfer is expected within the taxonomic group, such problems could be detected using the alignment of the hundreds of other genes at hand (see details in Section 3.3.6).

## 3.3 Principles underlying filtering methods

The underlying key ideas explained in this section are the basis of MSA filtering methods.

### 3.3.1 Gaps indicate hard to align and possibly saturated regions

Ultimately, sequence alignment simply consists of inserting gaps within sequences. The more gaps there are in a region, the more work the alignment method has to do, and the more likely it is that the method will generate errors. From a biological viewpoint, it is often assumed that in proteins insertions and deletions are less frequent than point substitutions. Hence, a region with multiple gaps indicates an unlikely evolutionary pattern that is most probably attributable to an MSA problem. In such regions, multiple mutations occurring at the same site are expected to be frequent and likely to obscure the phylogenetic signal.

### 3.3.2 Few/similar residues are expected per site

Residues within the same site are supposed to be homologous, so they are likely to share some characteristics - particularly as far as amino acid sequences are concerned. If all amino acids within a site are identical then we may be much more confident that they derive from the same ancestral amino acid than if 20 different amino acids are observed at this site. The latter case would imply not only that at least 19 substitutions have occurred to get this pattern (which could indicate a saturated site) but also that the protein has remained functional regardless of the physicochemical properties of the residue at this position. In such a situation, to filter out this part of the alignment would appear safe. Conversely, sites showing residues that share a common property – e.g., hydrophobic or positively charged – should probably be kept. Measuring residue conservation within a site can be done in

different ways, i.e. via basic measurements (number of different amino acids observed at this site, frequency of the most common amino acid) or more complex ones (measurement of site entropy, probability that two randomly picked residues are identical).

### 3.3.3 Models of sequence alignment

Extending the above idea, an MSA can be used to derive a model of the sequences it includes based on the observed spectrum of residues at each site. The Hidden Markov Model (HMM) provides a well-defined probabilistic model of a sequence alignment. This has many applications, such as improving homologous sequence search by BLAST-like algorithms. Of interest here is the fact that MSA HMM profile offers the opportunity to calculate a score that measures how much a given sequence fit the considered MSA. Examination of the variations of this score along the sequence can be useful to detect a potentially misaligned fragment.

### 3.3.4 Reliable regions are likely more robust to MSA method variations

As previously explained, MSA methods are heuristics that strive to find the MSA maximizing the chosen score (gap opening penalty, substitution matrix scores etc.). The fact that the MSA method output should be considered with caution was admirably highlighted by Landan and Graur (2007), that compare the alignment obtained with direct input DNA sequences with the reverse of the alignment obtained by aligning the reverse sequences. In a perfect world, those two alignments should be identical. In practice, they often differ, and the positions where they disagree pinpoint questionable alignment regions. These differences can be due to the presence of equally optimal scenarios at some stage of the heuristic, while the optimal scenario retained depends on the sequence orientation, and this choice impacts the next steps of the heuristic search. This heuristic search is also strongly impacted by the chosen guiding tree. Hence, tools such as Guidance (Sela et al., 2015) measure the alignment region reliability based on their stability with respect to a change in the guiding tree. Other methods go a step further and question whether the predicted residue homology relationships are stable when different penalty schemes are used (e.g. a slightly higher cost for gap opening or a different substitution matrix) or when different MSA programs are used.

### 3.3.5 Homologous (fragment of) sequences are expected to be similar (pre-filtering)

For most pipelines, sequence similarity is an initial criterion used to identify homologous sequences. This guarantees a minimal level of overall similarity among the sequences to be aligned. Despite this, it sometimes happen that, in a limited region of the alignment, a fragment of one (or a few) sequence does not resemble at all the rest of the alignment in this region. This sequence is thus likely not homologous to the others ones in this region, either because it was misaligned and the fragment is homologous to a distinct part of the alignment, or because it shares no homology at all with the rest of the sequences (e.g. due to alternative splicing or annotation errors). This latter situation could potentially be detected even before trying to align the sequences and a few tools have recently been developed to this effect. It is worth doing this filtering before aligning sequences since the MSA can be drastically slowed down and degraded by the presence of, particulaly, long insertions present in only one or a few sequences. The *TrimNonHomologousFragments* MACSE V2 subprogram (Ranwez et al.,

2018) and the PREQUAL program (Whelan et al., 2018) were both developed to remove fragments unrelated to other sequences, even before aligning them.

### 3.3.6 Orthologous sequences are supposed to be congruent over loci (post-filtering)

If an alignment contains a sequence that is not orthologous, while still being homologous, to others it is almost impossible for alignment filtering methods to detect and remove this sequence based solely on this alignment – since the problematic sequence is still highly similar to others. But in a phylogenomic context, it is possible to simultaneously consider the MSA of all considered loci to try to detect such problems. Intuitively, the distance between two sequences within a given MSA depends on the taxa from which those sequences derive (some species evolve faster than others) and on the locus represented by this MSA (some loci evolve faster than others). Having a large number of genes and taxa facilitates learning of loci and taxa evolutionary rates and hence detection of MSA having sequences significantly more distant from others than expected, considering the parameters learned using the whole set of available alignments. A simple solution to detect such non-orthologous sequences is included in the OrthoMaM v10 pipeline, while a more elaborate solution is provided by the Phylo-MCOA software package (de Vienne et al., 2012). Of course, this does not apply when the evolution of a whole gene family (orthologous + paralogous sequences) is the focus of the study.

## 3.4 TILI filtering methods and why they are fated to remove signals along with noise

Although some preliminary work was carried out in the early 1990s (e.g. Fernandes et al. 1993; Gatesy et al. 1993), the paper of Castresana introducing his famous Gblock software was clearly a turning point (Castresana, 2000) in the MSA literature. With more than 4,000 citations, this remains one of the most noteworthy studies in the field. Gblock defines a measurement of site conservation and basically removes any site that contains a gap and adjacent non-conserved sites, as well as stretches of non-conserved sites. Gblock is clearly a TILI-filtering method, for each site Gblock uses a series of criteria to decide whether to "take it or leave it".

Note that the number of sequences in a typical MSA has substantially increased since then, so in many applications removing any site with at least one gap is no longer a reasonable option. The latest releases of Gblock include a threshold that helps set the percentage of gaps allowed for a site. However, even in these latest versions, using default parameters will lead to the removal of any site with a gap. To really understand why removing any site with a gap is not reasonable, let us introduce an example that we will use to illustrate the inherent limitations of TILI filtering methods. Suppose you are trying to infer the phylogeny of 100 species for which you have an alignment of 5000 sites, where the sole gaps are as follows: a deletion going from sites 1 to 50 within the $1^{st}$ species, a deletion going from sites 51 to 100 for the $2^{nd}$ species, and so on up to the $100^{th}$ species, which has a deletion from sites 4951 to 5000. This is a great MSA matrix with only 1% of gaps, potentially conveying a strong phylogenetic signal. The Gblock default parameters, however, would remove all the sites of the alignment since they all contain at least one gap. Note that this example is intended as a criticism of the widespread blind usage of default software parameters, rather than a criticism of Gblock itself. Slightly modifying this example illustrates the problem inherent to any TILI filtering method. Suppose now that instead of gaps you have non-homologous fragments of

50 amino acids while the rest of the alignment is perfectly clean. Hence we now have, for the first 50 sites, 99 fragments of 50 amino acids that are orthologous to each other and perfectly aligned, plus a stretch of 50 unrelated amino acids (in sequence 1). Similarly, for the sites 51 to 100, we have 99 fragments of 50 amino acids that are orthologous to each other and perfectly aligned, plus a stretch of 50 unrelated amino acids in sequence 2 (see Figure 9 for an illustration of this kind of configuration). If we are using a TILI method that can only remove or keep entire sites, then there is no choice, either we keep those non-homologous fragments in the final alignment or we lose all of the sites. If a TILI-filtering approach is used for entire sequences, obviously we will end up with the same dilemma caused by the same problem. Theoretically, these non homologous fragments should generate insertions in the alignment, in practice they often don't. This problem of over alignment is well documented and hard to tackle (Katoh and Standley, 2016). It has been, for instance, emphasized on the Hedgehog-interacted protein (HHIP) to introduce the PREQUAL filtering method (Whelan et al., 2018), see https://natureecoevocommunity.nature.com/users/54859-iker-irisarri/posts/37479-automated-removal-of-non-homologous-sequence-stretches-in-phylogenomic-datasets.

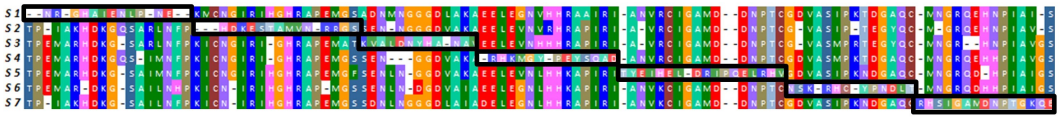

**Figure 9** Schematic representation of an example of MSA where TILI-filtering methods are useless. When assuming that the alignment is perfect except in regions surrounded by the black rectangles, using a TILI approach cannot get rid of all imperfect regions without removing all the sites/sequences of the MSA. Conversely, picky methods can simply mask the residues inside those black boxes while preserving the rest of the alignment.

Note that for a TILI-filtering approach on sites, we will also end up losing the entire signal or be obliged to keep all of the noise when we have a perfect alignment of 5000 sites for 99 sequences and a last sequence completely unrelated to the others. The problem when measuring the overall conservation of a site to decide whether to keep it or not is that, as the number of sequences increases, the impact of a misaligned fragment within a sequence decreases and is masked by the conservation of the rest of the site.

Despite these limitations, TILI-filtering methods could still do a great job regarding phylogeny inference if they are able to correctly identify and remove sequences and sites containing more noise than signal. Small misaligned fragments have little impact in such cases. These limitations are much more problematic when the planned downstream analysis includes tasks such as searching for selection footprints using dN/dS or branch length analyses. In this case, every misaligned fragment has a high probability of becoming a false positive in the analysis.

## 4 Evaluation of MSA filtering methods

Although it would seem reasonable to rely on filtering alignment methods to obtain trustworthy alignments upon which phylogenetic analyses could rely, how to implement such automatic filtering tools is still an active yet not completely mature research field. Consequently, whether currently available automatic alignment filtering methods offer satisfactory performance or worsen the situation is still debated. Whereas each new filtering method claims to improve things, the paper of Tan, 2015, casts serious doubts on their relevance

in phylogenetic pipelines (Tan et al., 2015). However, their analysis only considered TILI filtering methods and was focused on tree topology inference. We thus decided to conduct additional tests to further evaluate the performances of TILI and picky filtering methods in a phylogenomic framework.

## 4.1 A benchmark of 275 genes for 116 mammal species

We performed a comparison test to evaluate the efficiency of different alignment filtering methods. We opted to use a dataset generated from the tenth release of the OrthoMaM database. OrthoMaM is a database of orthologous exon and coding sequence (CDS) alignments and phylogenetic trees. We here focus on CDS markers. The latest OrthoMaM release (v10) gathers orthologous CDS sequences from 116 fully sequenced genomes present in Ensembl and NCBI database. For each human gene, the Ensembl annotation is used to gather 1-1 orthologous sequences present in Ensembl. This core set of sequences is then enriched by searching additional 1-1 orthologous sequences within mammalian genomes only present in the NCBI database. This leads to 14,509 sets of presumably 1-1 orthologous CDS sequences containing up to 116 sequences of diverse quality. The resulting set of sequences is then processed using a dedicated pipeline to generate high quality alignments and trees.

Using OrthoMaM to build an alignment filtering benchmark has several advantages. First, the database provides not only the filtered alignment used to build each gene tree but also the raw sequences collected for each species, which is exactly what is needed to test filtering in realistic conditions. Secondly, the evolutionary history of mammals is presumably devoid of events such as genome duplication, hybridization and gene transfers between distant taxa. The tree-like evolutionary history of the 116 mammals of our data set is therefore well established (except for a few irresolutions) and most gene trees are expected to share the same topology – branch lengths, however, are expected to vary since some genes may have evolved faster than others at different periods. Phylogenomic studies have resolved the phylogeny of these 116 mammals whose topology is well known (Figure 10). This species tree, obtained with the whole OrthoMaM dataset, provides us with a "reference tree" that can be used to assess the quality of each reconstructed gene tree, after the MSA was filtered or not. To facilitate the evaluation, we focused on the 275 CDS markers of OrthoMaM where all of the 116 species are represented. In practice, for each of these 275 markers, we aligned the 116 raw (unfiltered) sequences and compared the gene tree obtained using this alignment and those obtained using different filtered versions of it, with the reference tree.

## 4.2 Two criteria to assess the alignment quality

To evaluate the relevance of alignment filtering methods, it is crucial to come up with a criterion according to which it is possible to measure whether a given cleaned alignment is "better" or not than the initial one.

We used two criteria to measure the quality of a filtering method based on the 275 CDS trees that are inferred using the 275 filtered alignments. The underlying assumption is that the better the tree the better the alignment used to build it. See Chapter 2.5 (Tannier et al. 2020) for another, original criterion of gene tree quality assessment.

### 4.2.1 Consistency of gene trees and species tree topologies

Our first proxy to estimate the quality of the gene trees obtained is to measure how congruent they are with the species tree, i.e. the reference tree. As some genes may evolve faster

than others, we focus on the tree topology, i.e. ignoring the branch lengths. More precisely, we compute the quartet distance between the reference tree and each inferred CDS tree, i.e. the smaller this distance the better. Recall that, for this taxonomic group, most genes are expected to have undergone the evolutionary history depicted by the species tree (as expected in the absence of hybridizations and lateral gene transfer events).

Briefly, to calculate the distance between two trees, all possible quartets of leaves (i.e. species) are extracted from the trees. The topology of the quartet extracted from the first tree is compared to the topology of the same quartet extracted from the reference tree. For a given quartet, their topology is either the same or different. The distance is the number, or proportion, of quartets with a different topology. We compute this distance using the tqdist program (Sand et al., 2014).

Note that, using superTriplet (Ranwez et al., 2010) to combine, into a supertree, the 275 CDS trees with 116 species provided in the OrthoMaM database, leads to the exact same tree as the species tree obtained with the 14,509 CDS of the OrthoMaM database, which also is congruent with the literature and used as a reference on the OrthoMaM website (Figure 10).

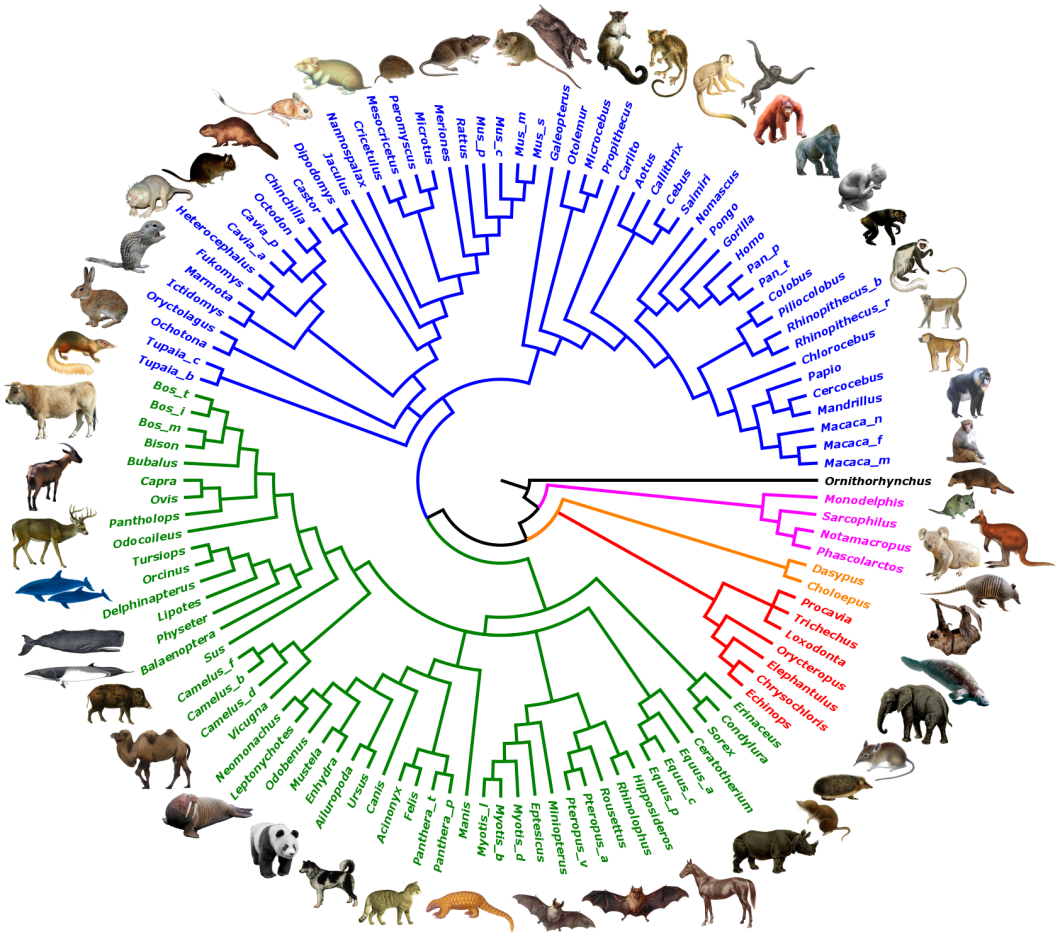


**Figure 10** The OrthoMaM v10 species tree (from the OrthoMaM website). This species tree is well resolved and only a handful of irresolution remains, e.g. the clade (Procavia, Trichechus, Loxodonta) in red.

### 4.2.2 Consistency of terminal branch lengths among gene trees

Although useful, the first quality measurement has a major limitation, i.e. it only takes the tree topology into account, which is not the only parameter upon which misalignment may have consequences. Our second proxy to estimate the quality of gene trees aims to capture errors in branch length estimations. This is a much harder task than comparing topologies for two reasons. First, as some genes are much more constrained than others there is no reason for branch lengths to be equal across gene trees, or to those of the species tree. Secondly, as gene trees may have different topologies, there is no direct link between the branches (lengths) observed in one tree and those observed in another one (even if both have the same leaves). This latter problem can be partly overcome by focusing only on terminal branches (i.e. branches connecting species to their first parental node) as they are present in all the trees to be compared. To tackle the first problem, as we lacked a reference branch length, we used the approach described below to detect abnormally long branches.

An abnormally long terminal branch in a gene tree reflects the accumulation of private residues at one particular gene in one particular species. This can be a signature of positive selection, pseudogenization, an indication that the sequence is not orthologous to others or has been misaligned. As positive selection is assumed to be a rare phenomenon, positive selection should affect only a few sites in a sequence and not lead to very long branches. All other cases are supposed to be detected by filtering methods and could lead to very long branches (if the problem affects the whole sequence) or just a small increase in its length (if the problem affects only part of the sequence).

That said, detecting abnormally long branches cannot be done with a simple threshold regarding branch lengths. For instance, a length of 0.4 is normal for the terminal branch associated with the early-branching Ornithorhynchus, but abnormal for the terminal branch associated to Pan troglodytes, which has only recently diverged from its common ancestor to Homo sapiens. In addition, the overall evolutionary rate of the considered gene also influences the expectations, longer branches being expected in fast-evolving genes. We used a linear regression to explain the terminal branch lengths observed on inferred ML trees. The terminal branch $b_{ij}$ leading to the species $i$ in the gene $j$ is viewed as $b_{ij} = \overline{b} + S_i + G_j + \epsilon_{ij}$, where $\overline{b}$ denotes the average length of all terminal branches of all gene trees; $S_i$ is the species effect; $G_j$ the gene effect and $\epsilon_{ij}$ the residual (the part of the branch length not captured by this model). We then considered $\epsilon^{\star}_{ij}$,the standardized residuals (a normalized version of $\epsilon_{ij}$), as a measurement of the terminal branch length consistency. This analysis was applied independently for each filtering method.

A similar approach is used in the OrthoMaM pipeline to remove non-orthologous sequences (having a standardized residual greater than 3). The same idea has also been applied to detect positive selection by identifying (reasonably) high residuals (Wu et al., 2017). Note that the alignment filtering methods we tested here consider only one locus/gene at a time, as does the tree topology inference. Intuitively, the standardized residuals of terminal branch lengths measure the deviation of the estimation of the branch length calculated on one dataset with the estimation done on the others, while accounting for the overall gene and species evlutionary rates). The lower (the absolute value of) these standardized residuals, the more congruent the terminal branch length estimations are.

## 4.3 The benchmark pipeline: from orthologous sequences to a gene tree

The whole benchmark pipeline, summarized in Figure 11, is detailed in the following sections.

### 4.3.1 Obtaining raw and filtered alignments

For each CDS marker, the nucleotide sequences were translated into protein sequences. The protein sequences were aligned with MAFFT and the nucleotide alignment was derived from the protein alignment using the back-aligned procedure provided by the 'egglib' python library (De Mita and Siol, 2012), see Figure11 A. Then, seven MSA filtering methods were applied on the 275 nucleotide raw alignments (Figure11 B).


**Figure 11** Schematic representation of the different filtering processes. Part A) depicts the alignment process: the nucleotide coding sequences are translated into amino acid sequences that are then aligned with MAFFT. The resulting amino acid alignment is then used to derive the nucleotide alignment. Part B) depicts the seven filtering processes that we compare to the "no filtering" approach.

Note that when CDS are the raw material for evolutionary study, the best strategy is to align them based on their translated sequences since amino acid sequences are more similar than their nucleotide counterparts (due to the genetic code redundancy and selective pressure acting at the amino acid level) and are hence easier to align. Once the protein alignments are obtained and cleaned, they may be used to derive the corresponding nucleotide alignments. Alignments are facilitated by using translated amino acid sequences. Filtering at the amino acid level is also easier thanks to the richer amino acid alphabet while ensuring that the codon structure is preserved. Whether inferring the phylogeny based on the (filtered) amino acid alignments or based on the (filtered) nucleotide alignments derived from them depends on the studied taxonomic level. Protein alignments will be better for deep phylogeny inferences (avoiding nucleotide saturation) while nucleotide alignments will be better for recent phylogenies (allowing observation of sequence differences that are masked at the amino acid level). According to what is done in the OrthoMaM pipeline, here we opted to infer gene trees based on the nucleotide alignments.

The filtering methods tested are presented in Table 1 (columns 1-3) and in Figure 11. The options used are detailed below.

- "no filtering": the nucleotide coding sequences are translated into amino acid sequences that are then aligned with MAFFT. The resulting amino acid alignment is then used to derive the nucleotide alignment, which serves as a reference point.
- Gblock is used to filter reference alignments with the codon option activated (option `-t=c`) and the possibility of having gaps (option `-b5=h`, i.e. only sites where gaps are present in less than half of the sequences could be kept).
- trimAl does not provide a codon filtering option, but its amino acid filtering provides, as

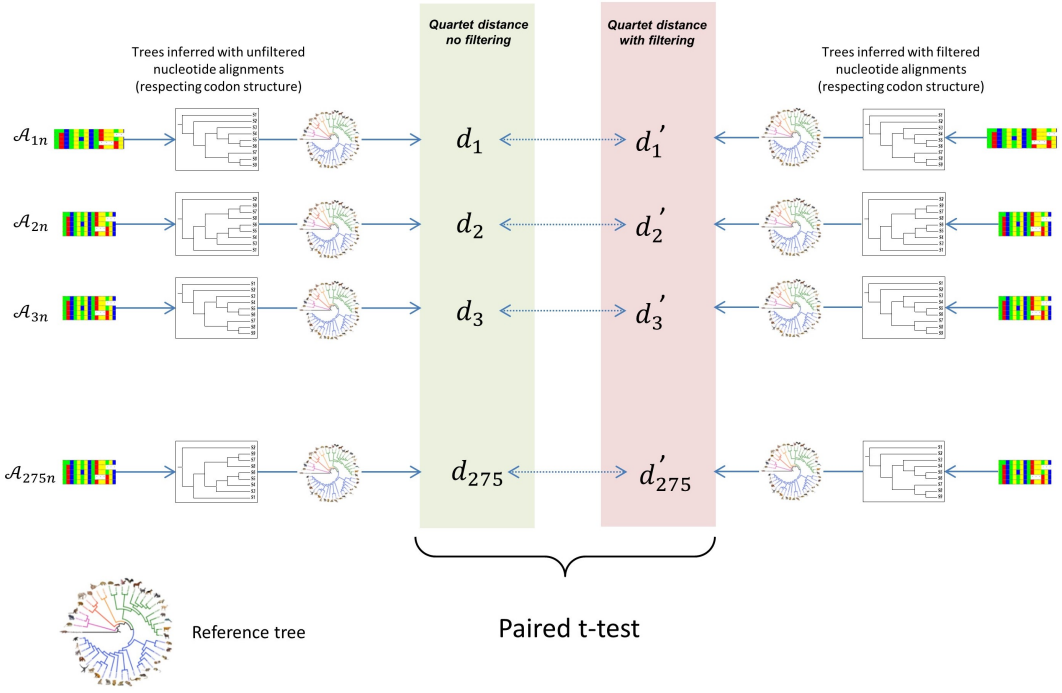


**Figure 12** Schematic representation of the test used to assess the impact of a given alignment filtering method on gene tree topologies.

| tested methods | | | % of removed nucleotides | | normalized q-distance | |
|---|---|---|---|---|---|---|
| name | ref | category | avg | median | avg | p-value |
| no filtering | - | - | 0 | 0 | 0.129 | - |
| Gblock | (Castresana, 2000) | TILI | 6.07 | 3.92 | 0.129 | 0.88 |
| trimAl | (Capella-Gutierrez et al., 2009) | TILI | 3.76 | 1.36 | 0.128 | 0.75 |
| BMGE | (Criscuolo and Gribaldo, 2010) | TILI | 4.77 | 3.11 | 0.127 | 0.23 |
| Guidance_TILI | (Penn et al., 2010; Sela et al., 2015) | TILI | 4.95 | 3.13 | 0.129 | 0.97 |
| Guidance_picky | (Penn et al., 2010; Sela et al., 2015) | picky | 2.90 | 1.99 | 0.128 | 0.59 |
| OMM_MACSE | (Scornavacca et al., 2019) | picky | 2.20 | 1.58 | 0.121 | 3.2E-3 |
| HmmCleaner | (Di Franco et al., 2019) | picky | 2.22 | 1.55 | 0.120 | 1.4E-5 |

**Table 1** Impact of filtering methods on the gene tree topologies. The first three columns provide information regarding each tested filtering method, its name, the corresponding paper(s) and the type of filtering applied. We computed the percentage of nucleotides removed from each filtered alignment; this led to 275 values per method whose average and median are provided in columns 4 and 5, respectively. We computed the normalized quartet distances between the reference tree and each inferred tree; this led to 275 values per method whose average is provided in column 6. For a given filtering method, the 275 normalized quartet distances are compared (using a paired t-test, see Figure 12) with the 275 normalized quartet distances obtained with "no filtering"; the p-value of this test is provided in the last column.

output, indices of the removed amino acid sites; we used this information to derive the corresponding filtered nucleotide alignment.

- BMGE can handle coding nucleotide alignments with its codon option activated (option `-t CODON` for BMGE).
- Guidance works directly on the unaligned coding nucleotide sequences as it includes an alignment step able to handle amino acid translation and back translation ( options `--msaProgram MAFFT --seqType codon`). By default Guidance uses a TILI strategy.
- maskLowScoreResidues.pl is a handy perl script that comes with Guidance. It takes advantage of Guidance output files to filter the initial alignment at the residue level (picky approach). As the script provides no default threshold, we used a threshold of 0.9 (option `0.9 nuc`), as also carried out in the paper mentioned on the Guidance documentation of this feature (Privman et al., 2012).
- OMM_MACSE is used with default options. Starting with unaligned nucleotide coding sequences, this pipeline chains sequence pre-filtering (removing long non-homologous fragments) and detection of frameshifts (using MACSE), alignment of amino acid sequences (using MAFFT), filtering of the resulting alignment (using HmmCleaner) and post-filtering (using MACSE) to mask isolated residues (those surrounded by gaps after HmmCleaner filtering) and to completely remove sequences for which more than half of the residues were masked during the pipeline.
- HmmCleaner works on amino acid alignments and the release we used generated a filtered alignment where masked residues were replaced by a specific user defined character. We used the '$' symbol to mask residues and a threshold of 10 (options `--del-char '\$' 10`). The *reportMaskAA2NT* subprogram of MACSE was then used to derive, thanks to the positions of the '$' symbols, the filtered alignment of the corresponding nucleotide coding sequences.

### 4.3.2 Obtaining phylogenetic trees and quartet distances

We have eight alignments for each of the 275 CDS markers: one raw alignment and seven filtered ones. For each of these $8 \times 275$ alignments, a maximum likelihood tree was inferred with phyML under a GTR+gamma model (Guindon et al., 2010) and the (normalized) quartet distance between the inferred tree and the reference tree was computed using tqdist (Sand et al., 2014). Note that since tqdist can only compute the quartet distance between two trees having exactly the same set of leaves, this criterion is not applicable to the few cases in which OMM_MACSE removes an entire sequence from the alignment. For this filtering approach, the total number of quartets with different topologies is hence not directly comparable to that of other filtering methods and only the average standardized quartet distance is comparable.

## 4.4 Results

### 4.4.1 Some filtering methods improve gene tree topologies much more than others

The quartet distances observed between the gene trees obtained and the reference tree are summarized in Table 2. All filtering methods lead to gene trees whose topology is more similar to that of the species tree than those obtained without filtering. Indeed, the normalized quartet distances to the species tree are about 0.120 for HmmCleaner, 0.121 for OMM_MACSE, 0.127 for BMGE, 0.128 for trimAl and the picky version of Guidance, 0.129 without filtering or with Gblock and the TILI version of Guidance. This gene tree topology improvement is not significant, however (based on a paired t-test 5%), with the exception of two filtering methods: HmmCleaner and the OMM_MACSE pipeline (that also relies on HmmCleaner). These findings are in line with the idea that picky filtering methods, which are able to mask individual residues, are more powerful than TILI methods, which can only remove entire sites or sequences, as explained above (Section 3.4). Note that the picky version of Guidance (normalized quartet distance 0.128) is only slightly better than its TILI counterpart (0.129) and that it does not improve the gene tree topologies as much as HmmCleaner(0.120).

### 4.4.2 Filtering methods have a notable impact on terminal branch lengths

The fourth column of Table 2 provides the count of gene tree terminal branch lengths longer than 0.5 (on average one mutation every two sites along these branches), excluding long branches leading to the Ornithorhynchus outgroup. Such long branches are highly unlikely and generally result from the presence of non-homologous or misaligned sequences. With just 4 such branches, OMM_MACSE is the most efficient method for this criterion, closely followed by HmmCleaner which has 7 such branches. All other methods lead to more than 20 of these long branches. Guidance_picky seems especially inefficient with 60 of these long branches while there are only 30 without any filtering. This seems to at least partially be related to the fact that this method sometimes masks all residues of a sequence without removing it from the alignment, hence the position of this sequence in the gene tree, as well as its branch length, are completely meaningless. Such extreme cases are correctly handled by OMM_MACSE which removes the sequence from the alignment when too many residues are masked, but they do not seem to be handled by other methods, which may mask all, or almost all, of the residues in a sequence without warning the end user.

| tested methods | | | number of abnormally long branches | |
|---|---|---|---|---|
| name | ref | category | length > 0.5 excluding Ornithorhynchus | $\epsilon^{\star}_{ij} > 3$ and length > 0.01 |
| no filtering | - | - | 30 | 152 |
| Gblock | (Castresana, 2000) | TILI | 25 | 153 |
| trimAl | (Capella-Gutierrez et al., 2009) | TILI | 22 | 145 |
| BMGE | (Criscuolo and Gribaldo, 2010) | TILI | 23 | 157 |
| Guidance_TILI | (Penn et al., 2010; Sela et al., 2015) | TILI | 22 | 150 |
| Guidance_picky | (Penn et al., 2010; Sela et al., 2015) | picky | 60 | 85 |
| OMM_MACSE | (Scornavacca et al., 2019) | picky | 4 | 27 |
| HmmCleaner | (Di Franco et al., 2019) | picky | 7 | 31 |

**Table 2** Impact of filtering methods on terminal branch lengths of gene trees. The first three columns provide information regarding each tested filtering method, its name, the corresponding paper(s) and the type of filtering applied. Column four indicates the number of terminal branches longer than 0.5, which usually reveals an alignment problem except, occasionally for the outgroup taxon. Column five indicates the number of abnormally long branches according to a linear model using species and genes as fixed parameters. See main text for details.

The count of abnormally long branches detected using a more elaborate test (standardized residuals > 3 and branch length > 0.01) that takes the fact that the evolutionary rates depend on the considered species and gene, leads to the same overall results. As detailed in the fifth column of Table 2, trees produced using the OMM_MACSE pipeline and HmmCleaner have many fewer abnormally long branches. These two methods have about 30 problematic branches detected, while 145 to 157 such problematic branches are detected with Gblock, Guidance_TILI, BMGE, Gblock, and trimAl or in the absence of filtering (152 problematic branches). The only discrepancy between these two abnormally long branch measurements concerns Guidance. While Guidance_picky has many fewer abnormally long branches according to the residual errors than Guidance_TILI (85 vs 150), it has many more branches longer than 0.5 (150 vs 22).

Figure 13 below illustrates one such case of abnormally long branches, where the part of the sequence removed by OMM_MACSE and HmmCleaner is obviously not orthologous to the others.

The abnormal branches detected using the above tests are extreme cases where a large portion of a sequence should be masked by filtering. When only a relatively short fragment of a sequence is problematic, and should be masked, its presence may not have sufficient impact on the corresponding branch length or its standardized residual to make any difference when using these tests. Figure 14 highlights the overall impact of HmmCleaner filtering on terminal branch lengths. In this figure, we plotted the length of each terminal branch estimated with HmmCleaner filtering (Y-axis) and without filtering (X-axis). Obviously, the main trend is that most of the points are below the y=x line, clearly indicating that using HmmCleaner tends to reduce terminal branch lengths. The same trend is also observed with OMM_MACSE.

## 5 Conclusion

This chapter details the key steps of MSA with an emphasis on the underlying arbitrary choices (e.g. gap opening costs) and inherent limitations (e.g. heuristic searches to optimize the alignment score). Given these limitations, it would seem natural to try to post-process alignments and filter out the erroneous parts. Though some obvious errors are easily spotted by the expert human eye, defining criteria to automatically determine which parts of the alignment are correct and incorrect is a difficult task. In recent years, several alignment filtering methods have been proposed based on criteria that we informally explain here. We believe that, independently of the filtering criteria used, the main distinction between these filtering methods concerns whether or not they can remove part of a site (or sequence) by simply masking some residues of a site (or sequence). We have called those that can "picky" and those that cannot "TILI". Using a simple example, we explained why TILI methods are inherently less able to spot errors in large alignments.

We assessed the impact of alignment filtering methods by comparing their impact on the gene tree inference of 275 CDS datasets, each consisting of 116 putative orthologous mammal sequences. As expected, the picky HmmCleaner method, either used alone or included as part of the OMM_MACSE pipeline, performed best. HmmCleaner and OMM_MACSE had a positive impact on the topology of individual gene trees. Using one of those two filtering approaches led to gene tree topologies significantly closer to that of the species tree, compared to topologies obtained in the absence of alignment filtering. The other tested

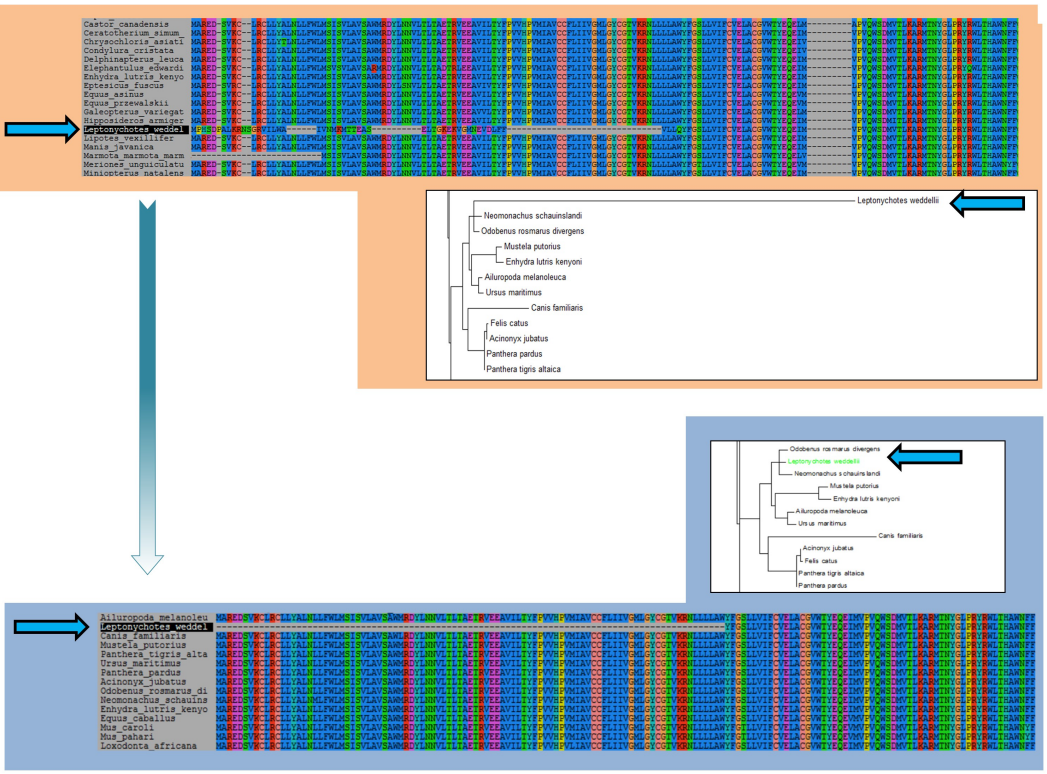


**Figure 13** A case of an abnormally long branch which has been corrected by OMM_MACSE and HmmCleaner but cannot be fixed by a PICKY filtering method.

filtering methods had no significant impact on gene tree topologies. Alignment filtering also had a major impact on branch lengths, which is an often overlooked aspect. Here again, HmmCleaner and OMM_MACSE yielded the best results in our tests, with a significant reduction in the number of abnormally long terminal branches and an overall increase in the consistency of the terminal branch lengths among the independently inferred gene trees. Note that those two methods are closely linked. Indeed OMM_MACSE is a coding sequence alignment dedicated pipeline that includes an HmmCleaner filtering step together with some specific pre- (before alignment) and post- (after HmmCleaner) cleaning steps.

Our results do not contradict those of (Tan et al., 2015). Indeed, these authors did not observe any benefit of using TILI filtering methods regarding the gene tree topologies, nor did we. We do not believe that this indicates that the filtering alignment is counter-productive for phylogeny inference, but it simply emphasizes the limitations inherent to TILI filtering methods. Our study also included some picky filtering methods and we were able to illustrate

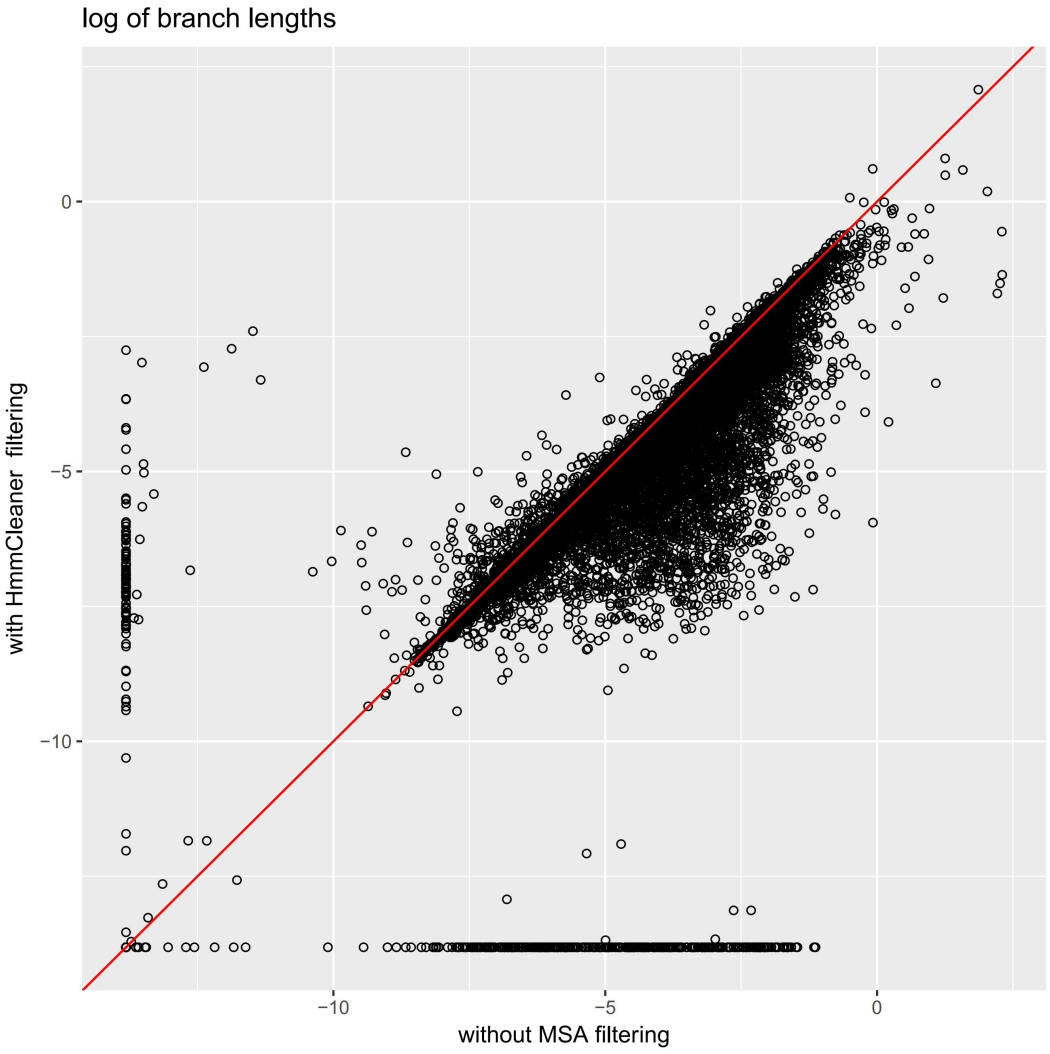


**Figure 14** Dot plots comparing the log of terminal branch lengths obtained without MSA filtering (x-axis) and with HmmCleaner filtering (y-axis). The $y = x$ line is drawn in red to facilitate interpretation.

the benefits of these. Moreover, it is an over simplification to consider phylogenetic inference only in terms of gene tree topologies. Here we have shown that alignment filtering may also have a marked impact on branch length estimations. Similarly, it would be worth investigating their impact on branch supports (bootstrap or posteriors), on the estimates of other evolutionary model parameters (e.g. substitution rates), and on downstream analyses – e.g. positive selection detection.

It should be noted that the potential effects (positive or negative) of MSA filtering methods also depend on the initial alignment quality. In other words, if great care is taken to ensure that the sequences to be aligned are homologous and do not contain long stretches of non-homologous residues, filtering methods will probably fail to substantially improve the MSA, and might even worsen it. However, when automatic homology search methods are applied in a phylogenomic project, applying picky-filtering methods is worthwhile.

## References

Adachi, J. and Hasegawa, M. (1996). Model of amino acid substitution in proteins encoded by mitochondrial dna. *J Mol Evol*, 42(4):459–68.

Altschul, S. F. (1989). Gap costs for multiple sequence alignment. *J Theor Biol*, 138(3):297–309.

Capella-Gutierrez, S., Silla-Martinez, J. M., and Gabaldon, T. (2009). trimal: a tool for automated alignment trimming in large-scale phylogenetic analyses. *Bioinformatics*, 25(15):1972–3.

Cartwright, R. A. (2006). Logarithmic gap costs decrease alignment accuracy. *BMC Bioinformatics*, 7:527.

Castresana, J. (2000). Selection of conserved blocks from multiple alignments for their use in phylogenetic analysis. *Mol Biol Evol*, 17(4):540–52.

Criscuolo, A. and Gribaldo, S. (2010). Bmge (block mapping and gathering with entropy): a new software for selection of phylogenetic informative regions from multiple sequence alignments. *BMC Evol Biol*, 10:210.

De Mita, S. and Siol, M. (2012). Egglib: processing, analysis and simulation tools for population genetics and genomics. *BMC Genet*, 13:27.

de Vienne, D. M., Ollier, S., and Aguileta, G. (2012). Phylo-mcoa: a fast and efficient method to detect outlier genes and species in phylogenomics using multiple co-inertia analysis. *Mol Biol Evol*, 29(6):1587–98.

Di Franco, A., Poujol, R., Baurain, D., and Philippe, H. (2019). Evaluating the usefulness of alignment filtering methods to reduce the impact of errors on evolutionary inferences. *Bmc Evolutionary Biology*, 19.

Edgar, R. C. (2004). Muscle: multiple sequence alignment with high accuracy and high throughput. *Nucleic Acids Res*, 32(5):1792–7.

Feng, D. F. and Doolittle, R. F. (1987). Progressive sequence alignment as a prerequisite to correct phylogenetic trees. *J Mol Evol*, 25(4):351–60.

Fernandes, A. P., Nelson, K., and Beverley, S. M. (1993). Evolution of nuclear ribosomal rnas in kinetoplastid protozoa: perspectives on the age and origins of parasitism. *Proc Natl Acad Sci U S A*, 90(24):11608–12.

Fernández, R., Gabaldón, T., and Dessimoz, C. (2020). Orthology: Definitions, prediction, and impact on species phylogeny inference. In Scornavacca, C., Delsuc, F., and Galtier, N., editors, *Phylogenetics in the Genomic Era*, chapter 2.4, pages 2.4:1–2.4:14. No commercial publisher | Authors open access book.

Fleissner, R., Metzler, D., and von Haeseler, A. (2005). Simultaneous statistical multiple alignment and phylogeny reconstruction. *Syst Biol*, 54(4):548–61.

Gatesy, J., DeSalle, R., and Wheeler, W. (1993). Alignment-ambiguous nucleotide sites and the exclusion of systematic data. *Mol Phylogenet Evol*, 2(2):152–7.

Guindon, S., Dufayard, J. F., Lefort, V., Anisimova, M., Hordijk, W., and Gascuel, O. (2010). New algorithms and methods to estimate maximum-likelihood phylogenies: assessing the performance of phyml 3.0. *Syst Biol*, 59(3):307–21.

Herman, J. L., Challis, C. J., Novak, A., Hein, J., and Schmidler, S. C. (2014). Simultaneous bayesian estimation of alignment and phylogeny under a joint model of protein sequence and structure. *Mol Biol Evol*, 31(9):2251–66.

Katoh, K. and Standley, D. M. (2016). A simple method to control over-alignment in the mafft multiple sequence alignment program. *Bioinformatics (Oxford, England)*, 32(13):1933–1942.

Landan, G. and Graur, D. (2007). Heads or tails: a simple reliability check for multiple sequence alignments. *Mol Biol Evol*, 24(6):1380–3.

Lartillot, N. (2020). The bayesian approach to molecular phylogeny. In Scornavacca, C., Delsuc, F., and Galtier, N., editors, *Phylogenetics in the Genomic Era*, chapter 1.4, pages 1.4:1–1.4:17. No commercial publisher | Authors open access book.

Lartillot, N. and Philippe, H. (2004). A bayesian mixture model for across-site heterogeneities in the amino-acid replacement process. *Mol Biol Evol*, 21(6):1095–109.

Le, S. Q. and Gascuel, O. (2008). An improved general amino acid replacement matrix. *Mol Biol Evol*, 25(7):1307–20.

Lowe, C. and Rodrigue, N. (2020). Detecting adaptation from multi-species protein-coding dna sequence alignments alignments. In Scornavacca, C., Delsuc, F., and Galtier, N., editors, *Phylogenetics in the Genomic Era*, chapter 4.5, pages 4.5:1–4.5:18. No commercial publisher | Authors open access book.

Loytynoja, A. and Goldman, N. (2008). Phylogeny-aware gap placement prevents errors in sequence alignment and evolutionary analysis. *Science*, 320(5883):1632–5.

Lunter, G., Miklos, I., Drummond, A., Jensen, J. L., and Hein, J. (2005). Bayesian coestimation of phylogeny and sequence alignment. *BMC Bioinformatics*, 6:83.

Madhusudhan, M. S., Marti-Renom, M. A., Sanchez, R., and Sali, A. (2006). Variable gap penalty for protein sequence-structure alignment. *Protein Eng Des Sel*, 19(3):129–33.

Morrison, D. A. (2006). Multiple sequence alignment for phylogenetic purposes. *Australian Systematic Botany*, 19(6):479–539.

Notredame, C., Higgins, D. G., and Heringa, J. (2000). T-coffee: A novel method for fast and accurate multiple sequence alignment. *J Mol Biol*, 302(1):205–17.

Ogden, T. H. and Rosenberg, M. S. (2007). Alignment and topological accuracy of the direct optimization approach via poy and traditional phylogenetics via clustalw + paup*. *Syst Biol*, 56(2):182–93.

Penn, O., Privman, E., Landan, G., Graur, D., and Pupko, T. (2010). An alignment confidence score capturing robustness to guide tree uncertainty. *Mol Biol Evol*, 27(8):1759–67.

Privman, E., Penn, O., and Pupko, T. (2012). Improving the performance of positive selection inference by filtering unreliable alignment regions. *Mol Biol Evol*, 29(1):1–5.

Pupko, T. and Mayrose, I. (2020). A gentle introduction to probabilistic evolutionary models. In Scornavacca, C., Delsuc, F., and Galtier, N., editors, *Phylogenetics in the Genomic Era*, chapter 1.1, pages 1.1:1–1.1:21. No commercial publisher | Authors open access book.

Ranwez, V. (2016). Two simple and efficient algorithms to compute the sp-score objective function of a multiple sequence alignment. *Plos One*, 11(8).

Ranwez, V., Criscuolo, A., and Douzery, E. J. P. (2010). Supertriplets: a triplet-based supertree approach to phylogenomics. *Bioinformatics*, 26(12):i115–i123.

Ranwez, V., Douzery, E. J. P., Cambon, C., Chantret, N., and Delsuc, F. (2018). Macse v2: Toolkit for the alignment of coding sequences accounting for frameshifts and stop codons. *Mol Biol Evol*, 35(10):2582–2584.

Ranwez, V., Harispe, S., Delsuc, F., and Douzery, E. J. (2011). Macse: Multiple alignment of coding sequences accounting for frameshifts and stop codons. *PLoS One*, 6(9):e22594.

Sand, A., Holt, M. K., Johansen, J., Brodal, G. S., Mailund, T., and Pedersen, C. N. (2014). tqdist: a library for computing the quartet and triplet distances between binary or general trees. *Bioinformatics*, 30(14):2079–80.

Scornavacca, C., Belkhir, K., Lopez, J., Dernat, R., Delsuc, F., Douzery, E. J. P., and Ranwez, V. (2019). Orthomam v10: Scaling-up orthologous coding sequence and exon alignments with more than one hundred mammalian genomes. *Mol Biol Evol*, 36(4):861–862.

Sela, I., Ashkenazy, H., Katoh, K., and Pupko, T. (2015). Guidance2: accurate detection of unreliable alignment regions accounting for the uncertainty of multiple parameters. *Nucleic Acids Res*, 43(W1):W7–14.

Simion, P., Delsuc, F., and Philippe, H. (2020). To what extent current limits of phylogenomics can be overcome? In Scornavacca, C., Delsuc, F., and Galtier, N., editors, *Phylogenetics in the Genomic Era*, chapter 2.1, pages 2.1:1–2.1:34. No commercial publisher | Authors open access book.

Stamatakis, A. and Kozlov, A. M. (2020). Efficient maximum likelihood tree building methods. In Scornavacca, C., Delsuc, F., and Galtier, N., editors, *Phylogenetics in the Genomic Era*, chapter 1.2, pages 1.2:1–1.2:18. No commercial publisher | Authors open access book.

Tan, G., Muffato, M., Ledergerber, C., Herrero, J., Goldman, N., Gil, M., and Dessimoz, C. (2015). Current methods for automated filtering of multiple sequence alignments frequently worsen single-gene phylogenetic inference. *Syst Biol*, 64(5):778–91.

Tannier, E., Bazin, A., Davín, A. A., Guéguen, L., Bérard, S., and Chauve, C. (2020). Ancestral genome organization as a diagnosis tool for phylogenomics. In Scornavacca, C., Delsuc, F., and Galtier, N., editors, *Phylogenetics in the Genomic Era*, chapter 2.5, pages 2.5:1–2.5:19. No commercial publisher | Authors open access book.

Thompson, J. D., Higgins, D. G., and Gibson, T. J. (1994). Clustal w: improving the sensitivity of progressive multiple sequence alignment through sequence weighting, position-specific gap penalties and weight matrix choice. *Nucleic Acids Res*, 22(22):4673–80.

Thompson, J. D., Linard, B., Lecompte, O., and Poch, O. (2011). A comprehensive benchmark study of multiple sequence alignment methods: current challenges and future perspectives. *PLoS One*, 6(3):e18093.

Thorne, J. L. and Kishino, H. (1992). Freeing phylogenies from artifacts of alignment. *Mol Biol Evol*, 9(6):1148–62.

Wang, L. and Jiang, T. (1994). On the complexity of multiple sequence alignment. *J Comput Biol*, 1(4):337–48.

Wheeler, T. J. and Kececioglu, J. D. (2007). Multiple alignment by aligning alignments. *Bioinformatics*, 23(13):i559–68.

Wheeler, W. (1996). Optimization alignment: The end of multiple sequence alignment in phylogenetics? *Cladistics*, 12(1):1–9.

Wheeler, W. C. (2003). Implied alignment: a synapomorphy-based multiple-sequence alignment method and its use in cladogram search. *Cladistics*, 19(3):261–8.

Whelan, S., Irisarri, I., and Burki, F. (2018). Prequal: detecting non-homologous characters in sets of unaligned homologous sequences. *Bioinformatics*, 34(22):3929–3930.

Wu, J., Yonezawa, T., and Kishino, H. (2017). Rates of molecular evolution suggest natural history of life history traits and a post-k-pg nocturnal bottleneck of placentals. *Current Biology*, 27(19):3025 – 3033.e5.

Zou, Z. and Zhang, J. (2020). The nature and phylogenomic impact of sequence convergence. In Scornavacca, C., Delsuc, F., and Galtier, N., editors, *Phylogenetics in the Genomic Era*, chapter 4.6, pages 4.6:1–4.6:17. No commercial publisher | Authors open access book.