Multiple Sequence Alignment

Lisa Pokorny & Marina Marcet-Houben

Multiple Sequence Alignment (MSA)

The true multiple alignment

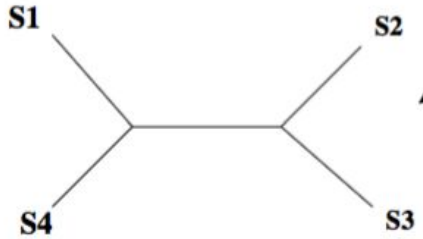
- **Reflects historical substitution, insertion, and deletion events**
- **Defined using transitive closure of pairwise alignments computed on edges of the true tree**

Fig. 9.2. Warnow. 2017. *Computational Phylogenetics. An Introduction to Designing Methods for Phylogeny Estimation.* CUP.

Multiple Seq Alignment (MSA)

Standard two-phase approach: 1st ALIGNMENT (positional homology)

```
S1 = AGGCTATCACCTGACCTCCA          S1 = -AGGCTATCACCTGACCTCCA
S2 = TAGCTATCACGACCGC              S2 = TAG-CTATCAC--GACCGC--
S3 = TAGCTGACCGC                   S3 = TAG-CT-------GACCGC--
S4 = TCACGACCGACA                  S4 = -------TCAC--GACCGACA
```

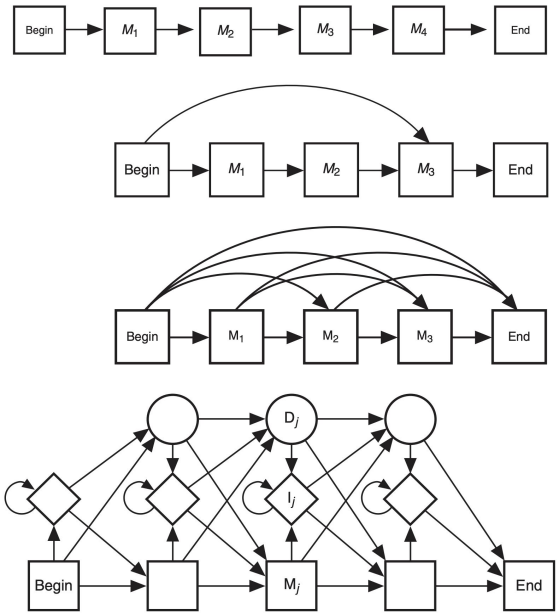


2nd TREE BUILDING

Fig. 9.1. Warnow. 2017. *Computational Phylogenetics. An Introduction to Designing Methods for Phylogeny Estimation.* CUP.

Optimization Problems & MSA Methods (MSAMs)

- Sum-of-Pairs Alignment (SOP)
- Tree Alignment (TL) and Generalized TL
- Sequence Profiles
- Profile Hidden Markov Models (HMM)
- Reference-based Alignments
- Template-based Methods
- Seed Alignment Methods
- Weighted-Homology Pair Methods
- Progressive Methods
- Divide-and-Conquer Methods
- Co-estimation of Alignments and Trees
- Structure Informed Methods, etc.



Figs. 9.4, 9.6–9.8. Warnow. 2017. *Computational Phylogenetics. An Introduction to Designing Methods for Phylogeny Estimation.* CUP.

Phylogenetic Tree Estimation w/o Alignment?

Syst. Biol. 66(2):218–231, 2017

DOI:10.1093/sysbio/syw074
Advance Access publication September 14, 2016

Phylogenetic Tree Estimation With and Without Alignment: New Distance Methods and Benchmarking

MARCIN BOGUSZ AND SIMON WHELAN*

Department of Evolutionary Biology, Evolutionary Biology Centre, Uppsala University, Norbyvägen 18D, 752 36 Uppsala, Sweden
**Correspondence to be sent to: Department of Evolutionary Biology, Evolutionary Biology Centre, Uppsala University, Norbyvägen 18D, 752 36 Uppsala, Sweden; E-mail: simon.whelan@ebc.uu.se.*

Received 3 February 2016; reviews returned 12 June 2016; accepted 23 August 2016
Associate Editor: David Bryant

Abstract.—Phylogenetic tree inference is a critical component of many systematic and evolutionary studies. The majority of these studies are based on the two-step process of multiple sequence alignment followed by tree inference, despite persistent evidence that the alignment step can lead to biased results. Here we present a two-part study that first presents PaHMM-Tree, a novel neighbor joining-based method that estimates pairwise distances without assuming a single alignment. We then use simulations to benchmark its performance against a wide-range of other phylogenetic tree inference methods, including the first comparison of alignment-free distance-based methods against more conventional tree estimation methods. Our new method for calculating pairwise distances based on statistical alignment provides distance estimates that are as accurate as those obtained using standard methods based on the true alignment. Pairwise distance estimates based on the two-step process tend to be substantially less accurate. This improved performance carries through to tree inference, where PaHMM-Tree provides more accurate tree estimates than all of the pairwise distance methods assessed. For close to moderately divergent sequence data we find that the two-step methods using statistical inference, where information from all sequences is included in the estimation procedure, tend to perform better than PaHMM-Tree, particularly full statistical alignment, which simultaneously estimates both the tree and the alignment. For deep divergences we find the alignment step becomes so prone to error that our distance-based PaHMM-Tree outperforms all other methods of tree inference. Finally, we find that the accuracy of alignment-free methods tends to decline faster than standard two-step methods in the presence of alignment uncertainty, and identify no conditions where alignment-free methods are equal to or more accurate than standard phylogenetic methods even in the presence of substantial alignment error. [Alignment-free; distance-based phylogenetics; pair Hidden Markov Models; phylogenetic inference; statistical alignment.]

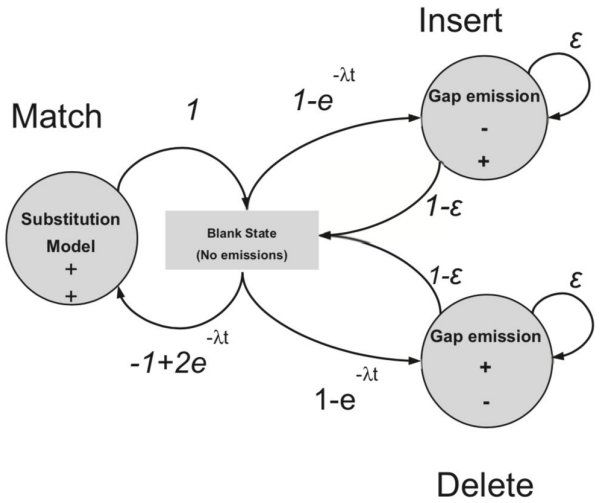
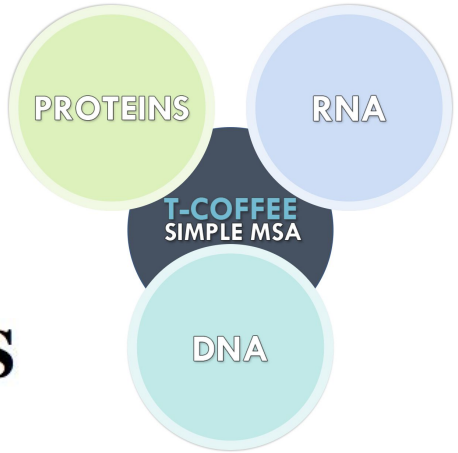


Fig. 1. Bogusz & Whelan. 2017. *Syst. Biol.* 66(2):218–231

Comparing MSAMs

Tool	Options	Algorithm	Alphabet
ClustalW	Defaults	Progressive	Amino Acid
Muscle	Defaults	Progressive (iterative)	Amino Acid
MAFFT	Defaults	Progressive (iterative)	Amino Acid
ProbCons	Defaults	Consistency	Amino Acid
ProbAlign	Defaults	Consistency	Amino Acid
Mummals	Defaults	Consistency/Structure	Amino Acid
Dialign-TX	Defaults	Greedy/Progressive	Amino Acid
Prank (AA)	+F (AA)	"Phylogenetically-aware"	Amino Acid
Prank	+F -codon	"Phylogenetically-aware"	Codon
BAli-Phy	Model M0	Statistical Alignment	Codon
BAli-Phy samples	Model M0	Statistical Alignment	Codon
BAli-Phy integrated	Model M0	Statistical Alignment	Codon



BAli---Phy

MUMMALS

MASS

Multiple Alignment by Secondary Structures

CLUSTAL

CLUSTAL

ESPript 3.0

PROBCONS

Probabilistic Consistency-based Multiple Alignment of Amino Acid Sequences

EMBL-EBI

MUSCLE

PRANK

MAFFT version 7

Multiple alignment program for amino acid or nucleotide sequences

Blackburne & Whelan. 2013. *Mol. Biol. Evol.* 30(3):642–653.

Mean dist btw MSAMs

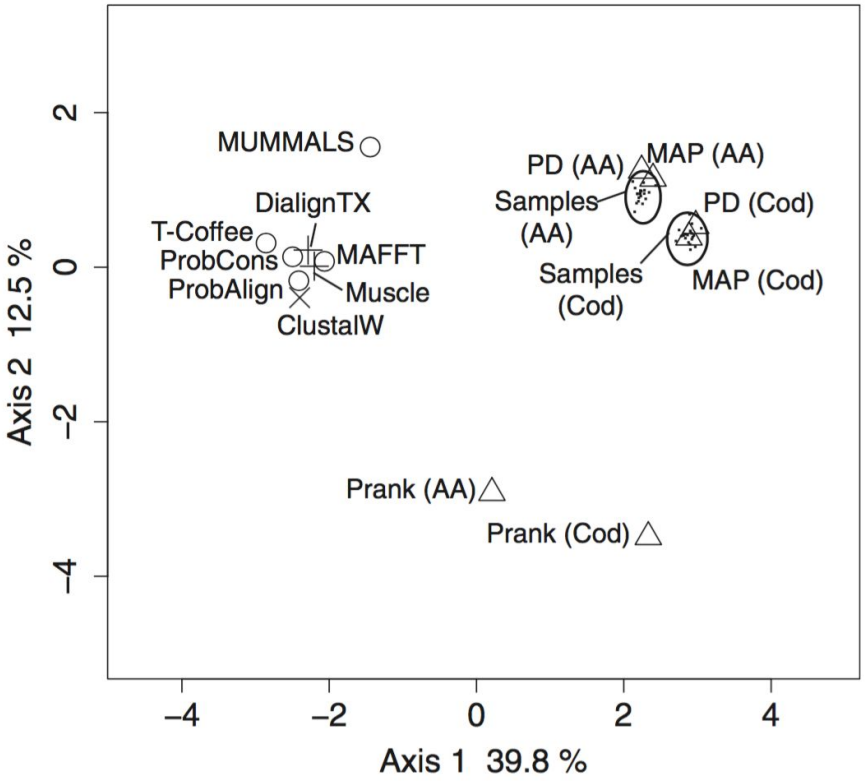


Fig. 1. Blackburne & Whelan. 2013. *Mol. Biol. Evol.* 30(3):642–653.

Dist btw trees w ≠ MSAMs

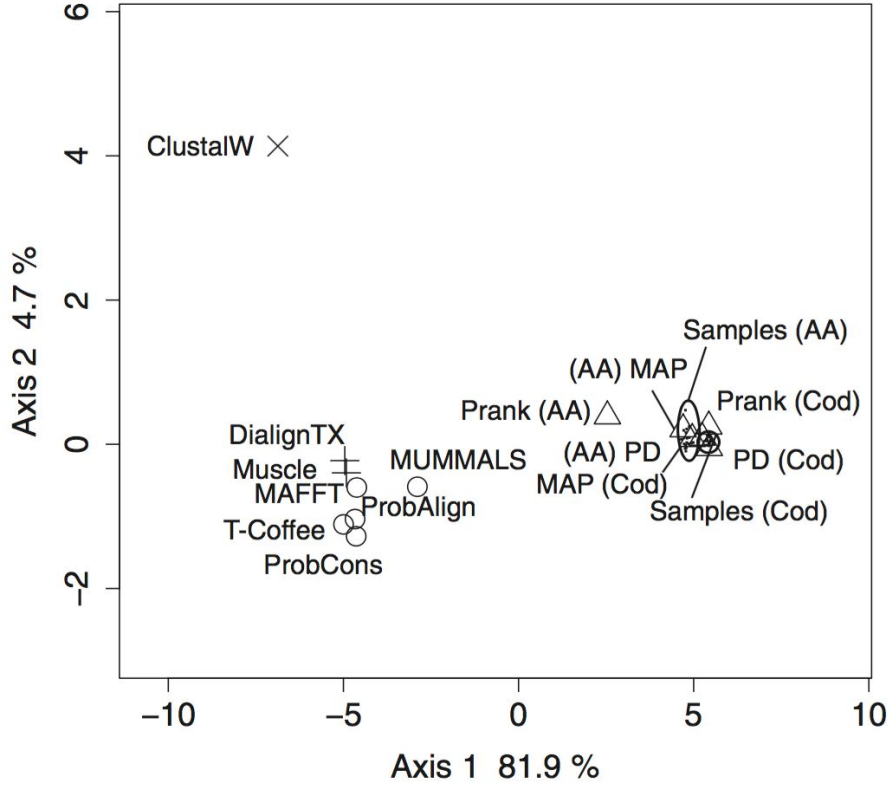


Fig. 2. Blackburne & Whelan. 2013. *Mol. Biol. Evol.* 30(3):642–653.

Even more MSAMs comparisons

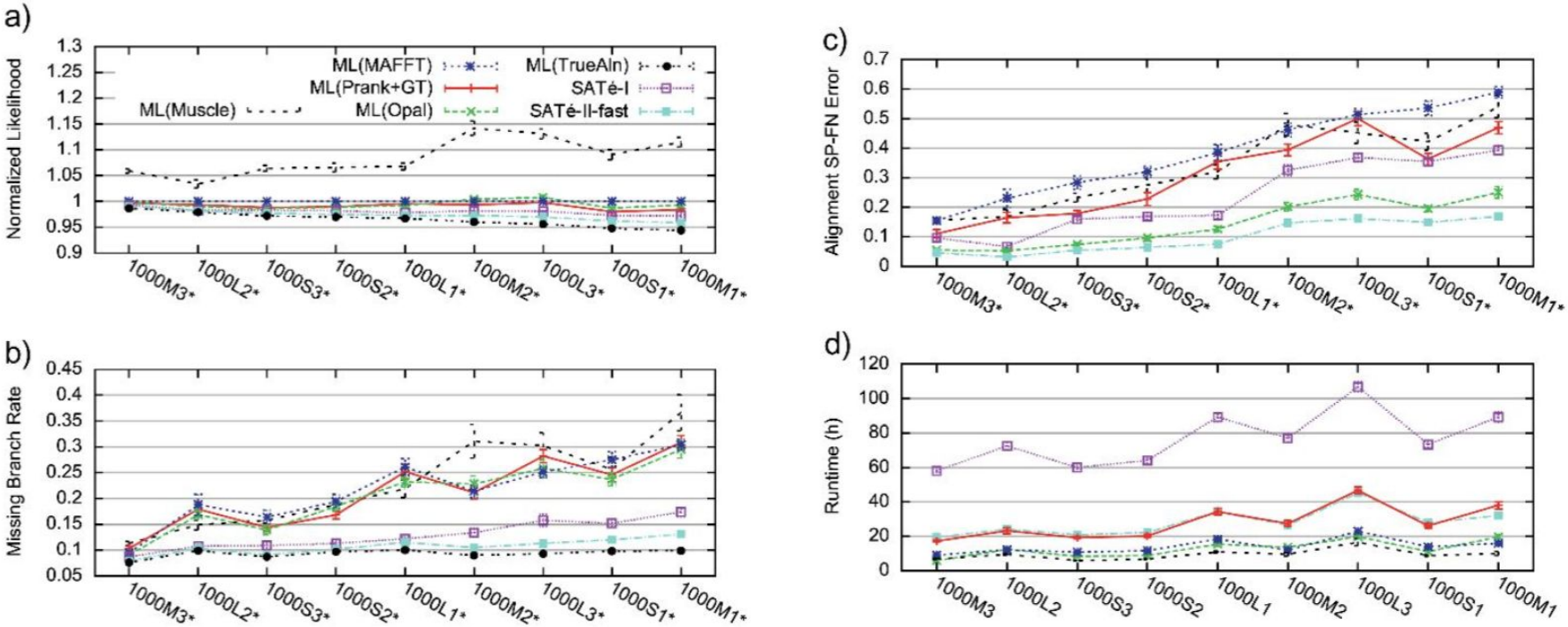


Fig. 4. Liu et al. 2013. *Syst. Biol.* 61(1):90–106.

Modeler Precision vs. Recall

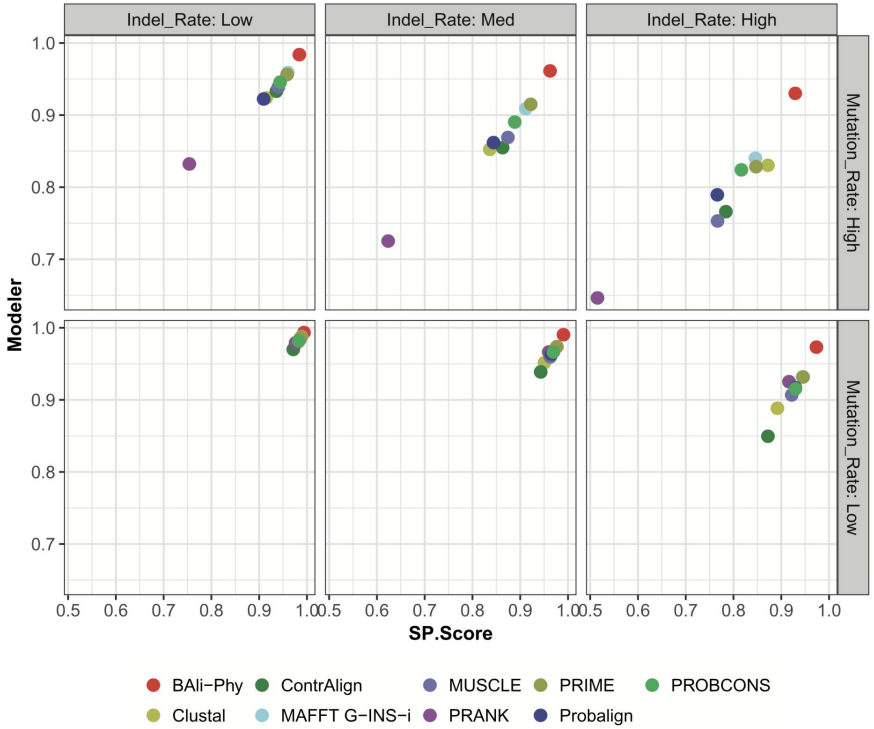


Fig. 6. Nute et al. 2018. *Syst. Biol.* https://doi.org/10.1093/sysbio/syy068

Expansion Ratios

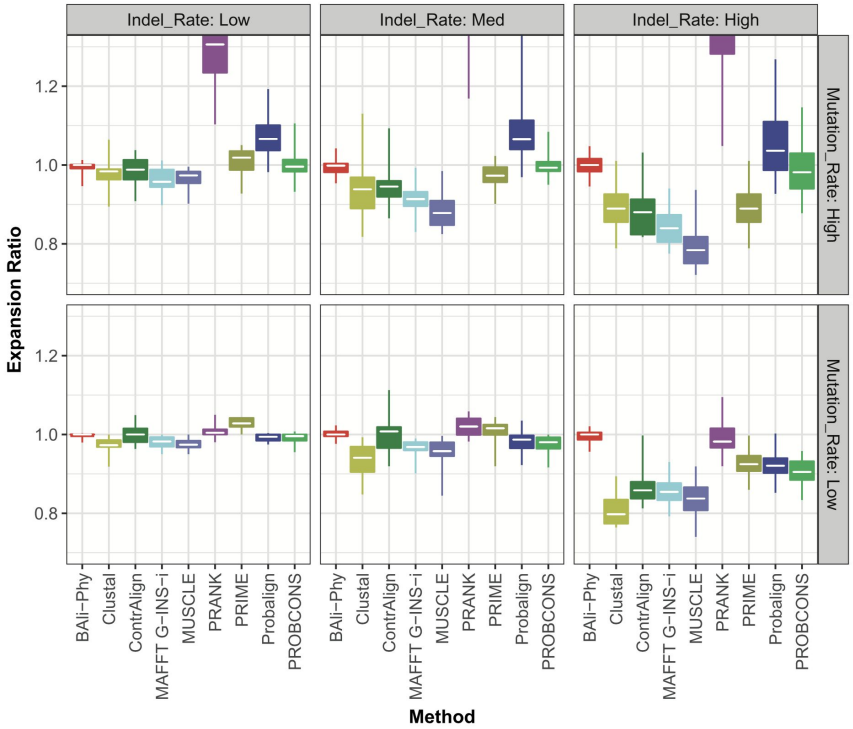


Fig. 7. Nute et al. 2018. *Syst. Biol.* https://doi.org/10.1093/sysbio/syy068

PROTEINS

RNA

T-COFFEE
SIMPLE MSA

DNA

BAli---Phy

MUMMALS

MASS

Multiple Alignment by Secondary Structures

CLUSTAL Ω

CLUSTAL W2

ESPript 3.0

PROBCONS

Probabilistic Consistency-based Multiple Alignment of Amino Acid Sequences

EMBL-EBI

MUSCLE

PRANK

MAFFT version 7

Multiple alignment program for amino acid or nucleotide sequences

Know Your Limits

Data type → DNA vs. RNA, coding vs. non-coding nucleotides (wobble bp), AAs, proteins, etc.

Data properties → substitution (≠ mutation) rate strength (↑ vs. ↓), indel size and rate (% gap & gap length), pairwise sequence identity (PID), etc.

Data matrix properties → # of tips, # of sequences, (alignment length ∝) data matrix weight, e.g., light (K, M) vs. heavy (G, T), etc.

CPU time and RAM memory → computing resources available

Divide and Conquer Method: PASTA

PASTA estimates **alignments and ML trees** from unaligned sequences using an **iterative approach**. In each iteration, it first estimates a multiple sequence alignment using the current tree as a guide and then estimates an ML tree on (a masked version of) the alignment. By default, PASTA performs 3 iterations, but a host of options enable changing that behavior. In **each iteration**, a **divide-and-conquer** strategy is used for estimating the alignment. The **set of sequences** is **divided into smaller subsets**, **each** of which is **aligned using** an external alignment tool (default is **MAFFT**). These **subset alignments** are then **pairwise merged** (by default **using Opal**) and finally the **pairwise merged alignments** are **merged** into a **final alignment using** a **transitivity merge** technique. The division of the dataset into smaller subsets and selecting which alignments should be pairwise merged is guided by the tree from the previous iteration. The first step therefore needs an initial tree.

Acknowledgment: The current **PASTA** code is heavily based on the **SATé** code developed by Mark Holder's group at KU.

https://github.com/smirarab/pasta/blob/master/pasta-doc/pasta-tutorial.md

PASTA

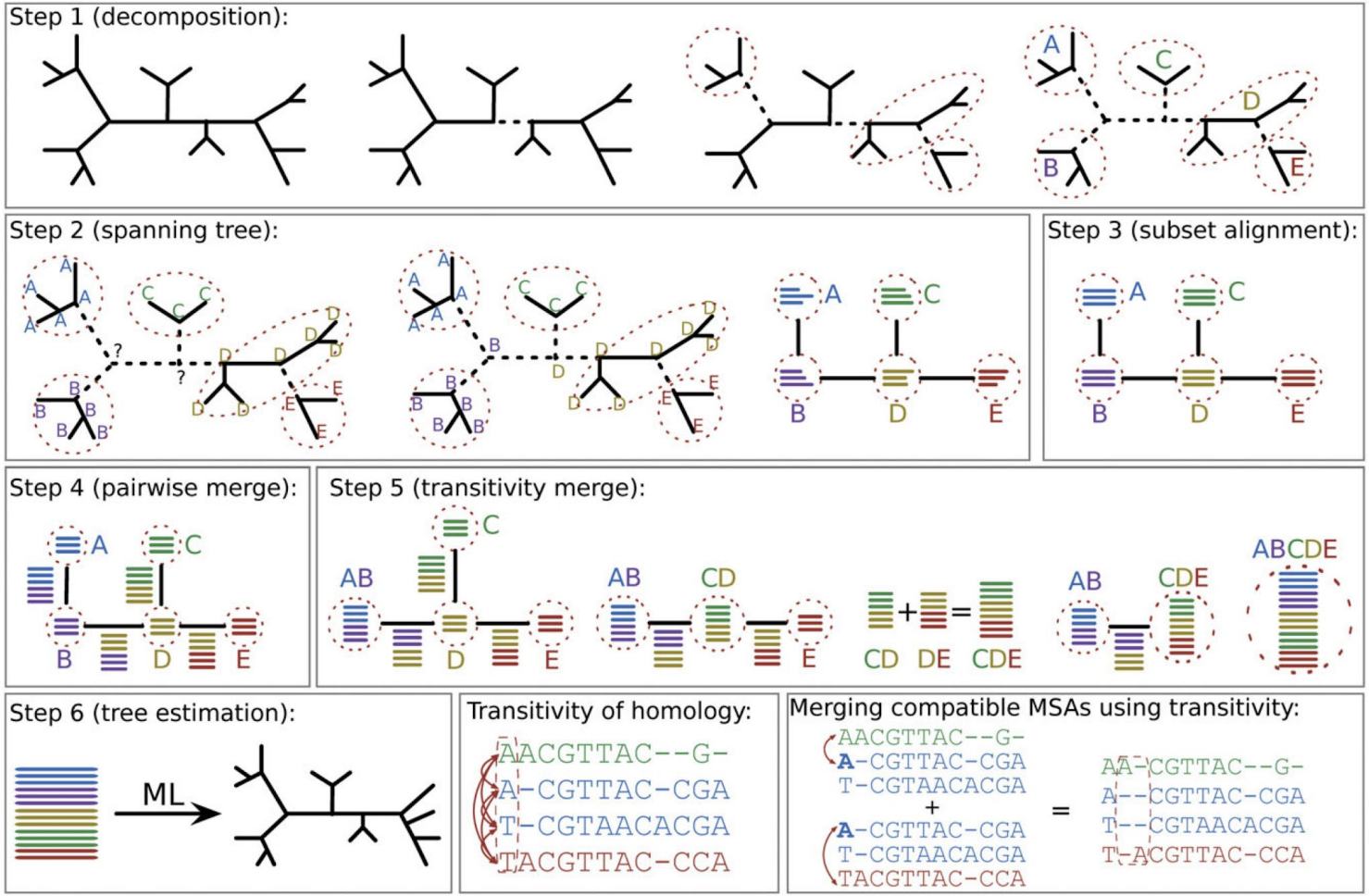


Fig. 1. Mirarab et al. 2015. *J. Comp. Biol.* 22(5):377–386.

PASTA for nucleotides and AA

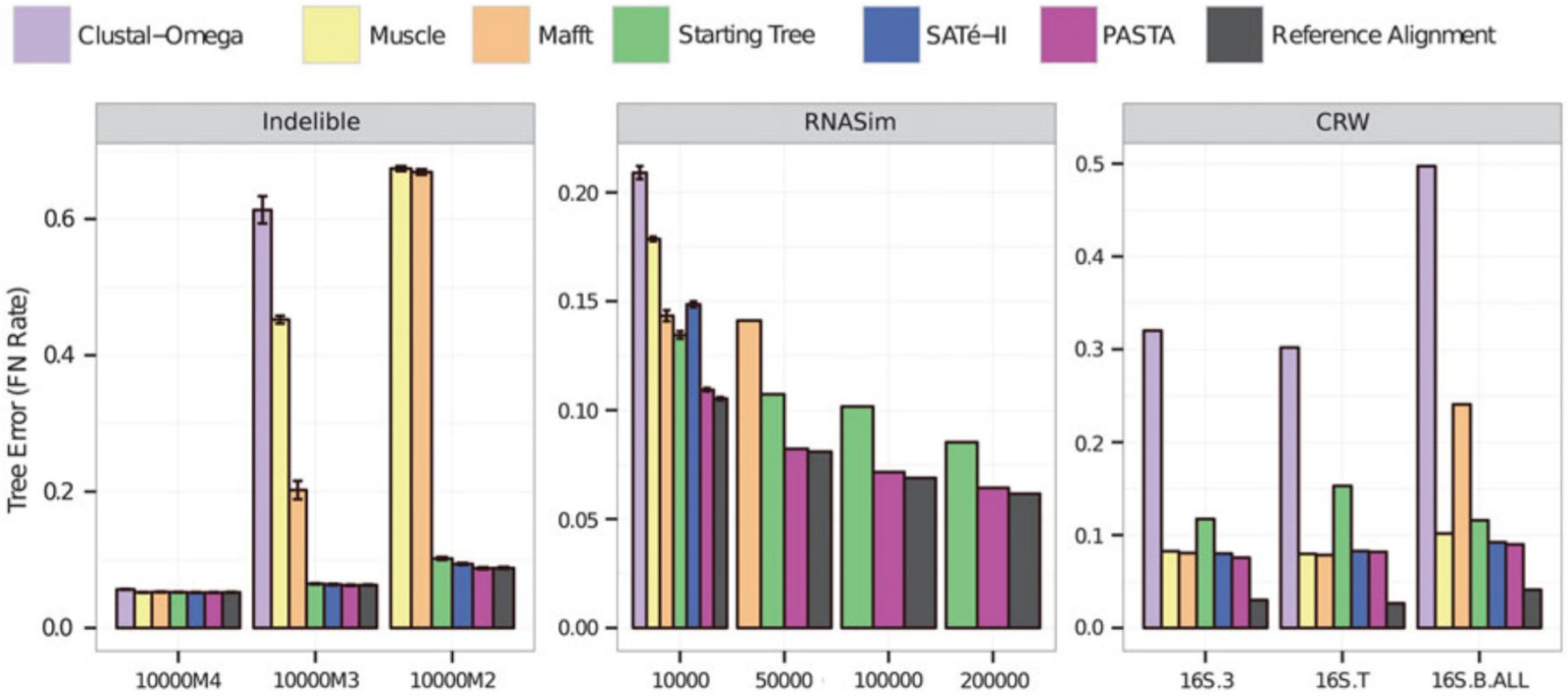
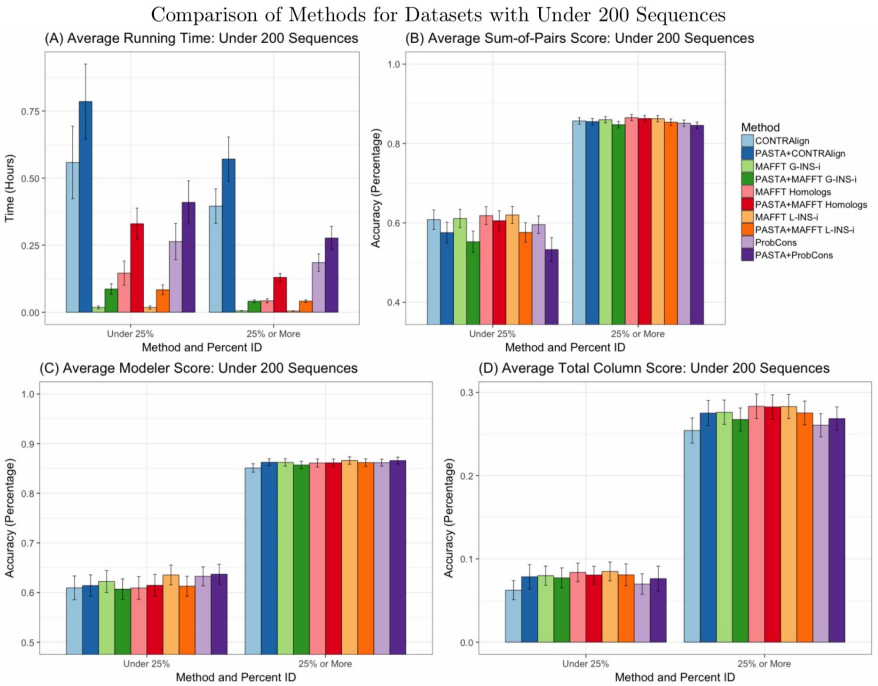
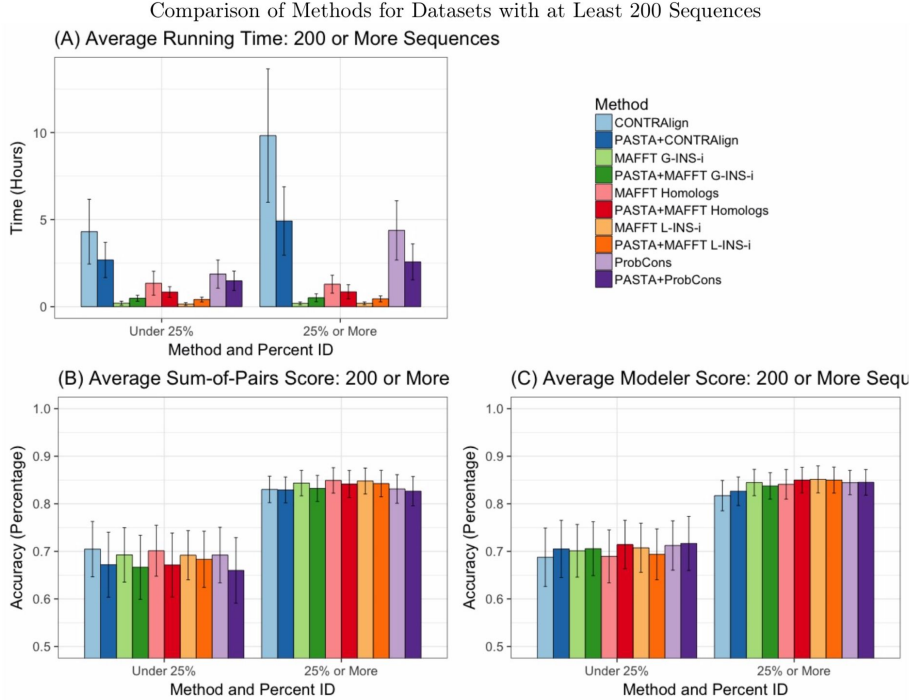


Fig. 2. Mirarab et al. 2015. *J. Comp. Biol.* 22(5):377–386.

PASTA for proteins





Sup. Figs. 1 & 2. Collins & Warnow. 2018. PASTA for proteins *Bioinformatics* 34(22):3939–3941.

Running PASTA (from Command-line)

If your installation is successful, you should be able to run **PASTA** by running the following command from any location. Open up a terminal window and type:

```
run_pasta.py --help
```

Running **PASTA** with the `--help` option produces the list of options available in **PASTA**. **PASTA** automatically picks its algorithmic settings based on your input, so you can ignore most of these options (but `-d` is essential if you have anything other than DNA sequences). The basic command-line usage you need to know is:

```
run_pasta.py -i input_fasta_file
```

https://github.com/smirarab/pasta/blob/master/pasta-doc/pasta-tutorial.md

Running PASTA (from Command-line)

The `-i` option is used to specify the input sequence file. The input file needs to be in the relaxed FASTA format. This command will start **PASTA** and will run it on your input file.

For a test run, use the `cd` command to go to the `data` directory under your **PASTA** installation directory. From there, run

```
run_pasta.py -i small.fasta
```

This will start **PASTA** and will finish quickly (30 seconds to 5 minutes based on your machine). Read **PASTA** output and make sure it finishes without producing any errors. If **PASTA** runs successfully, it produces a multiple sequence alignment and a tree, which we will explore in the next step.

https://github.com/smirarab/pasta/blob/master/pasta-doc/pasta-tutorial.md

Inspecting the Output of PASTA

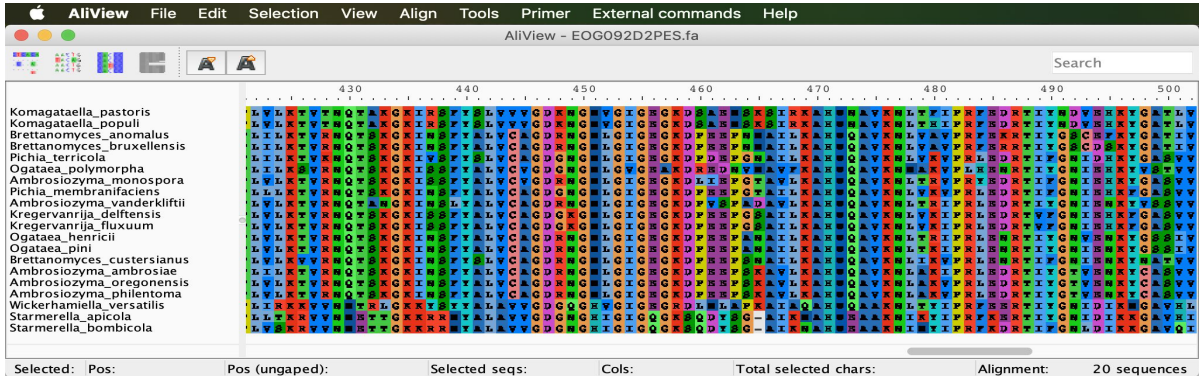
The two main outputs of **PASTA** are an alignment and a tree. The tree is saved in a file called `[jobname].tre` and the alignment file is named `[jobname].marker001.small.aln`. The `[jobname]` is a prefix which is by default set to `pastajob`, but can be changed by the user (see option `-j` below). When you start **PASTA**, if your output directory (which is by default where your input sequences are) already contains some files with the `pastajob` prefix, then the `pastajob1` prefix is used, and if that exists, `pastajob2` is used, and so forth. Thus the existing files are never overwritten. The name of your job and therefore the prefix used for output files can be controlled using the `-j` argument for command-line or the "Job Name" field on the GUI.

Tree Viewing Software (TVS) → https://en.wikipedia.org/wiki/List_of_phylogenetic_tree_visualization_software

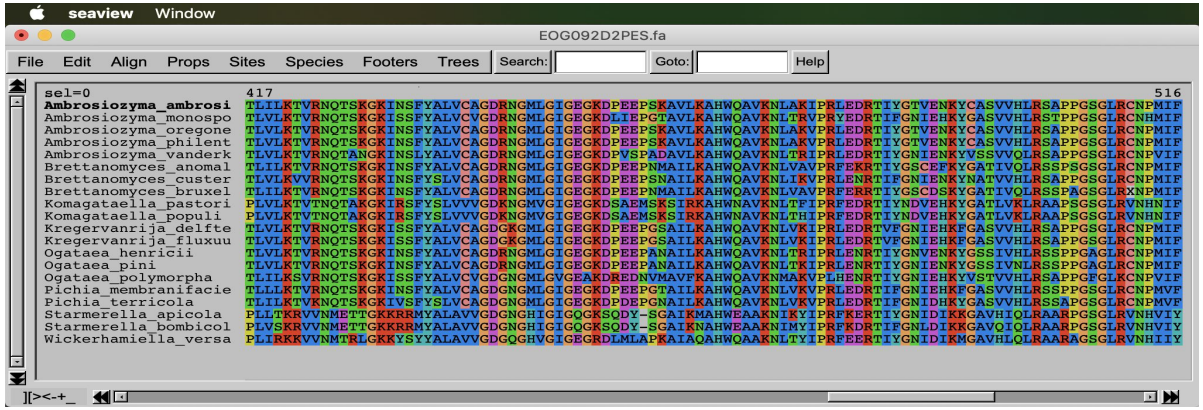
Alignment Viewing Software (AVS), e.g., http://doua.prabi.fr/software/seaview or http://www.ormbunkar.se/aliview/

Light-weight AVS — AliView & SeaView

AliView → Larsson, A. (2014). AliView: a fast and lightweight alignment viewer and editor for large data sets. *Bioinformatics* 30(22):3276–3278. http://dx.doi.org/10.1093/bioinformatics/btu531



SeaView → Gouy M., Guindon S. & Gascuel O. (2010) SeaView version 4 : a multiplatform graphical user interface for sequence alignment and phylogenetic tree building. *Molecular Biology and Evolution* 27(2):221–224. https://academic.oup.com/mbe/article/27/2/221/970247



Inspecting the Output of PASTA

What about bootstrapping?

PASTA does not perform bootstrapping. The tree outputted by **PASTA**, depending on the options used, might include support values on the branches. These are ***not* bootstrap** support values. Instead, they are **SH-like local** support values computed by **FastTree**, and are generally believed to be not as reliable as bootstrap support values. In our experience they tend to overestimate support. Thus, if you want to have support values that can be trusted, we suggest that you use the **PASTA** alignment and an external tool (e.g., **RAxML**) for bootstrapping. If your alignment is too big for bootstrapping using **RAxML**, you can always use **FastTree** or **IQ-tree** for bootstrapping.

Comparing alignments

When two alignments are generated on the same set of sequences, one can ask how similar they are. We have a tool called **FastSP** http://www.cs.utexas.edu/~phylo/software/fastsp/ that compares two alignments and tells you how similar or different they are. **FastSP** does not require installation. You can just download it and run it (Java is required). Assuming **FastSP** is located at `~/bin/`, you can, e.g., compare the reference alignment and your estimated alignment:

```
java -jar ~/bin/FastSP_1.6.0.jar -r 16S.E.ALL.referene.fasta -e pastajob.marker001.16S.E.ALL.unaligned.aln
```

Running PASTA (from Command-line)

You can script a while loop in bash to run **PASTA** on multiple fasta files. First open a text editor

```
nano pasta_loop.sh
```

Write your bash script

```
#!/bin/bash
while read targetname;
   do
      python ABSOLUTE_PATH_HERE/run_pasta.py -i "$targetname".fasta -j $targetname
   done < targetlist.txt
```

Close `CTRL+x` and save your script. This script assumes all target files are in the same folder in fasta format. It also assumes that folder contains a text file listing all targets. From there, run

```
bash pasta_loop.sh
```

Understanding and Using PASTA Options

The command line allows you to alter the behavior of the algorithm using a variety of configuration options. Running **PASTA** with the `-h` option lists all the options that can be provided to the command-line (see below for the most important ones). In addition to the command-line itself, **PASTA** can read the options from one or more configuration files. The configuration files have the following format:

```
[commandline]
option-name = value

[sate]
option-name = value
```

Note that as mentioned before, with every run, **PASTA** saves the configuration file for that run as a temporary file called `[jobname]_temp_pasta_config.txt` in your output directory. You can view one of these files in a Text editor for better understanding the format of the configuration file.

https://github.com/smirarab/pasta/blob/master/pasta-doc/pasta-tutorial.md

Understanding and Using PASTA Options

PASTA can read multiple configuration. Configuration files are read in the order they occur as arguments (with values in later files replacing previously read values). Options specified in the command line are read last. Thus these values "overwrite" any settings from the configuration files.

The following is a list of important options used by **PASTA**. Note that by default **PASTA** picks these parameters for you, and thus you might not need to ever change these (with the important exception of the `-d` option):

- **Initial tree**: As mentioned before, **PASTA** needs an initial tree for doing the first round of the alignment. Here is how the initial tree is picked.
    - If a starting tree is provided using the `-t` option, then that tree is used.

```
run_pasta.py -i small.fasta -t small.tree
```

https://github.com/smirarab/pasta/blob/master/pasta-doc/pasta-tutorial.md

Understanding and Using PASTA Options

PASTA can read multiple configuration. Configuration files are read in the order they occur as arguments (with values in later files replacing previously read values). Options specified in the command line are read last. Thus these values "overwrite" any settings from the configuration files.

The following is a list of important options used by **PASTA**. Note that by default **PASTA** picks these parameters for you, and thus you might not need to ever change these (with the important exception of the `-d` option):

- **Initial tree**: As mentioned before, **PASTA** needs an initial tree for doing the first round of the alignment. Here is how the initial tree is picked.
  - If a starting tree is provided using the `-t` option, then that tree is used.
  - If the input sequence file is already aligned and `--aligned` option is provided, then **PASTA** computes a ML tree on the input alignment and uses that as the starting tree.
    - If the input sequences are not aligned (or if they are aligned and `--aligned` is not given), **PASTA** uses the following procedure for estimating the starting alignment and tree. It 1) randomly selects a subset of 100 sequences, 2) estimates an alignment on the subset using the subset alignment tool (default **MAFFT-l-insi**), 3) builds a **HMMER** model on this "backbone" alignment, 4) uses **hmmalign** to align the remaining sequences into the backbone alignment, 5) runs **FastTree** on the alignment obtained in the previous step.

https://github.com/smirarab/pasta/blob/master/pasta-doc/pasta-tutorial.md

Understanding and Using PASTA Options

- **Data type**: **PASTA** does not automatically detect your data type. Unless your data is DNA, you need to set the data type using `-d` command. Your options are DNA, RNA, and PROTEIN.

```
run_pasta.py -i BBA0067-half.input.fasta -t BBA0067-half.startingtree.tre -d PROTEIN
```

https://github.com/smirarab/pasta/blob/master/pasta-doc/pasta-tutorial.md

Understanding and Using PASTA Options

- **Data type**: **PASTA** does not automatically detect your data type. Unless your data is DNA, you need to set the data type using `-d` command. Your options are DNA, RNA, and PROTEIN.
- **Tree estimation tool**: the default tool used for estimating the phylogenetic tree in **PASTA** is **FastTree**. The only other option currently available is **RAxML**. You can set the tree estimator to **RAxML** using the `--tree-estimator` option. However, Be aware that **RAxML** takes much longer than FastTree. If you really want to have a **RAxML** tree, we suggest obtaining one by running it on the final **PASTA** alignment. You can change the model used by FastTree (default: `-nt -gtr -gamma` for nt and `-wag -gamma` for aa) or **RAxML** (default `GTRGAMMA` for nt and `PROTWAGCAT` for AA) by updating the `[model]` parameter under `[FastTree]` or `[RAxML]` header in the input configuration file. The model cannot be currently updated in the command line directly as an option.
- **Subset alignment tool**: the default tool used for aligning subsets is **MAFFT**, but you can change it using the `--aligner` option. We strongly suggest alignment subset size should always be no more than 200 sequences, because for subsets that are larger than 200, the most accurate version of **MAFFT** (-linsi) is not used.
- **Pairwise merge tool**: the default merger too is **Opal**. You can change it using `--merger` option. If you have trouble with **Opal** (java version, memory, etc.) using **Muscle** should solve your problem and in our experience, it doesn't really affect the accuracy by a large margin.

https://github.com/smirarab/pasta/blob/master/pasta-doc/pasta-tutorial.md

Understanding and Using PASTA Options

- **CPUs**: **PASTA** tries to use all the available cpus by default. You can use `--num_cpus` to adjust the number of threads used.

```
run_pasta.py -i small.fasta --num_cpus 1
```

https://github.com/smirarab/pasta/blob/master/pasta-doc/pasta-tutorial.md

Understanding and Using PASTA Options

- **CPUs**: **PASTA** tries to use all the available cpus by default. You can use `--num_cpus` to adjust the number of threads used.
- **Number of iterations**: the simplest option that can be used to set the number of iterations is `--iter-limit`, which sets the number of iterations **PASTA** should run for. You can also set a time limit using `--time-limit`, in which case, **PASTA** runs until the time limit is reached, and then continues to run until the current iteration is finished, and then stops. If both options are set, **PASTA** stops after the first limit is reached. The remaining options for setting iteration limits are legacies of **SATé** and are not recommended.
- **Masking**: Since **PASTA** can produce very gappy alignments, it is a good idea to remove sites that are almost exclusively gaps before running the ML tree estimation. By default, **PASTA** removes sites that are more than 99.9% gaps. You can change that using the `--mask-gappy-sites` option. For example, using `--mask-gappy-sites 10` would remove sites that are gaps for all sequences except for (at most) 10 sequences. Increasing the masking can make **PASTA** a bit faster and can potentially reduce the memory usage. But it could also have a small effect on the final tree. If unsure, leave the option unchanged. Note that the final alignment outputted by **PASTA** is NOT masked, but masked versions of the output are also saved as temporary files (see below).

https://github.com/smirarab/pasta/blob/master/pasta-doc/pasta-tutorial.md

Understanding and Using PASTA Options

- **Maximum subset size**: two options are provided to set the maximum subset size: `--max-subproblem-frac` and `--max-subproblem-size`. The `--max-subproblem-frac` option is a number between 0 and 1 and sets the maximum subset size as a fraction of the entire dataset. The `--max-subproblem-size` option sets the maximum size as an absolute number. When both numbers are provided (in either a configuration file or the command line), the *LARGER* number is used. This is an unfortunate design (legacy of **SATé**) and can be quite confusing. Please always double check the actual subset size reported by **PASTA** and make sure it is the value intended. The default subset sizes should work just fine. In our limited experiments, we have noticed that reducing the maximum subset size from 200 to 100 for very large datasets increases speed with little or no effect on the final alignments.
- **Temporary files**: **PASTA** creates many temporary files, and deletes most at the end. You can control the behavior of temporary files using few options: `--temporaries` sets the directory where temp files are created, `-k` instructs **PASTA** to keep temporary files, and `--keepalignmenttemps` will keep even more temporary files. Note that these are different from the temporary files created in the output directory (which are always kept).
- **Dry run**: The `--exportconfig` option can be used to just create a config file and exit without actually running **PASTA**. This is useful for making sure the configurations are correct before actually running the job.

https://github.com/smirarab/pasta/blob/master/pasta-doc/pasta-tutorial.md

Running PASTA Using Configuration Files

The configurations used for running **PASTA** are all saved to a configuration file, and also, **PASTA** can be run using a configuration file. These configuration files are useful for multiple purposes. For example, if you want to reproduce a **PASTA** run, or if you want to report the exact configurations used. Always make sure to keep the produced configuration files for future reference. Note however, that configuration files can be used as input only using command-line.

Let's open `myjob_temp_pasta_config.txt` under the data directory and take a look at it. Notice that the options we referred to are all mentioned here.

Now imagine that we wanted to instruct **PASTA** to use the JTT model instead of WAG for a protein run. Here is how we can accomplish that. Copy the `myjob_temp_pasta_config.txt` file as a new file (e.g. `cp myjob_temp_pasta_config.txt jtt_config.txt`). Then open `jtt_config.txt` using a text editor of your choice. Find `model = -wag -gamma -fastest` under the `[FastTree]` header. Remove the `-wag` option and save the config file. Note that the default model in FastTree is JTT, and therefore, when the `-wag` is removed, it automatically switches to using JTT. To run **PASTA** using this new configuration file, run:

```
run_pasta.py  jtt_config.txt
```

https://github.com/smirarab/pasta/blob/master/pasta-doc/pasta-tutorial.md

Running PASTA Using Configuration Files

Adding custom parameters to aligners: It is also possible to add custom parameters to alignment and merge tools. To do so, you need to use the config file. Under each alignment tool in the config file, you can add an `args` attribute and list all the attributes you want to pass to that tool. For example, to run **MAFFT** with your choice of gap penalty value, edit the config file under the `[mafft]` heading to something like:

```
[mafft]
path = [there will be a path here to your pasta directory]/bin/mafft
args = --op 0.2 --ep 0.2
```

and use this config file to run **PASTA**.

Note that **PASTA** does not try to understand these extra parameters you pass to external tools. It simply appends these parameters to the end of the command it executes.

At this stage, if you have input files that you like to have analyzed, you know enough to start doing that.

Email: `pasta-users@googlegroups.com` for all issues.

https://github.com/smirarab/pasta/blob/master/pasta-doc/pasta-tutorial.md

MAFFT

MAFFT is a multiple sequence alignment program for unix-like operating systems. It offers a range of multiple alignment methods, **L-INS-i** (accurate; recommended for <200 sequences), **FFT-NS-2** (fast; recommended for >2,000 sequences), etc. Accuracy-oriented methods:

*__L-INS-i__ (probably most accurate; recommended for <200 sequences; iterative refinement method incorporating local pairwise alignment information):

```
mafft --localpair --maxiterate 1000 input [> output]
linsi input [> output]
```

*__G-INS-i__ (suitable for sequences of similar lengths; recommended for <200 sequences; iterative refinement method incorporating global pairwise alignment information):

```
mafft --globalpair --maxiterate 1000 input [> output]
ginsi input [> output]
```

*__E-INS-i__ (suitable for sequences containing large unalignable regions; recommended for <200 sequences):

```
mafft --ep 0 --genafpair --maxiterate 1000 input [> output]
einsi input [> output]
```

https://mafft.cbrc.jp/alignment/software/manual/manual.html

Bali-Phy

BAli---Phy

[Welcome!]
- Introduction
- References

Download

Examples

Documentation
- Users Guide
- Tutorial
- Manual pages
- Developers Guide

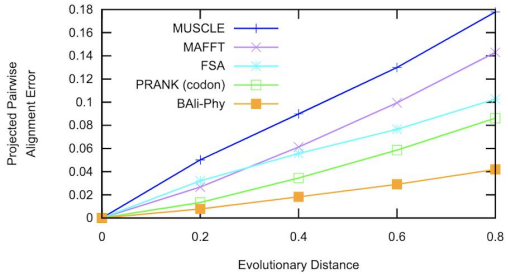
Contact
- Bug reports
- Mailing lists

Introduction

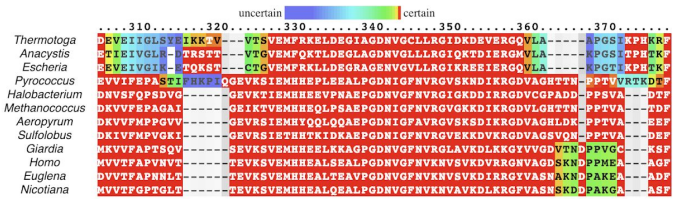
01/18/19: **BAli-Phy 3.4.1 released** - Download
16-state RNA stem models + character sets + bug fixes (release notes)

BAli-Phy is software by Ben Redelings that estimates multiple sequence alignments and evolutionary trees from DNA, amino acid, or codon sequences. It uses likelihood-based evolutionary models of substitutions and insertions and deletions to place gaps. It has been used in published analyses on data sets up to 117 taxa.

High alignment accuracy: Redelings (2014) showed that *BAli-Phy* had 3.5 times fewer alignment errors than MUSCLE and MAFFT on simulated data:



Eliminate bias: Fletcher and Yang (2010) showed that relying on a ClustalW alignment estimate could lead to a 99% false-positive rate in detecting positive selection. In general, inferring evolutionary trees, branch lengths, or positive selection from a single alignment can lead to bias if the alignment is ambiguous. *BAli-Phy* solves this problem by using MCMC and Bayesian methods to estimate evolutionary trees, positive selection, and branch lengths while averaging over alternative alignments.



http://www.bali-phy.org/ (find **PASTA** implementation here → https://github.com/MGNute/pasta)

Profile HMM Methods: SEPP, TIPP, UPP, & HIPPI

SEPP stands for "SATé-enabled Phylogenetic Placement", and addresses the problem of phylogenetic placement of short reads into reference alignments and trees.

TIPP stands for "Taxonomic Identification and Phylogenetic Profiling", and addresses the problem of taxonomic identification and abundance profiling of metagenomic data.

UPP stands for "Ultra-large alignments using Phylogeny-aware Profiles", and addresses the problem of **alignment of very large datasets, potentially containing fragmentary data**. UPP can align datasets with up to 1,000,000 sequences.

HIPPI stands for "Highly Accurate Protein Family Classification with Ensembles of HMMs", and addresses the problem of classifying query sequences to protein families.

https://github.com/smirarab/sepp

UPP

UPP is a modification of **SEPP** for performing **alignments of ultra-large** and **fragmentary datasets**. **UPP** operates in four steps:

- In the first step, **UPP partitions** set S into a **backbone set** and a **query set** and computes an alignment and tree on the backbone set using **PASTA**, which is a direct improvement to **SATé**.
- In the next step, **UPP decomposes** the **backbone alignment** into an ensemble of profile Hidden Markov Models (**HMMs**).
- The third step in **UPP searches** for the **best alignment** of the **query sequence** to each **HMM**.
- The final step **inserts** the **query sequence into** the **backbone alignment** using the best scoring **HMM**.

Our study shows that **UPP** results in accurate alignments, and that ML trees estimated on the alignments are also highly accurate. **UPP** has **good accuracy** on datasets that contain **fragmentary sequences**.

https://github.com/smirarab/sepp/blob/master/README.UPP.md

UPP

To run **UPP**, invoke the `run_upp.py` script from the `bin` sub-directory of the location in which you installed the Python packages.

To see options for running the script, use the command: `python <bin>/run_upp.py -h`

The general command for running **UPP** is: `python <bin>/run_upp.py -s <unaligned_sequences>`

This will run **UPP(Default)**. This will automatically select up to 1,000 sequences to be in the backbone set, generate a **PASTA alignment** and **tree**, and then align the remaining sequences to the backbone alignment.

https://github.com/smirarab/sepp/blob/master/README.UPP.md

UPP

To run **UPP**, invoke the `run_upp.py` script from the `bin` sub-directory of the location in which you installed the Python packages.

To see options for running the script, use the command: `python <bin>/run_upp.py -h`

The general command for running **UPP** is: `python <bin>/run_upp.py -s <unaligned_sequences>`

This will run **UPP(Default)**. This will automatically select up to 1,000 sequences to be in the backbone set, generate a **PASTA alignment** and **tree**, and then align the remaining sequences to the backbone alignment.

UPP can also be run using a configuration file. To run using a configuration file, run: `python <bin>/run_upp.py -c sample.config`

To run **UPP(Fast)**, run: `python <bin>/run_upp.py -s input.fas -B 100`

UPP currently assumes that the input sequences are nucleotide sequences. To select the input data type, run: `python <bin>/run_upp.py -s input.fas -m [dna|rna|amino]`

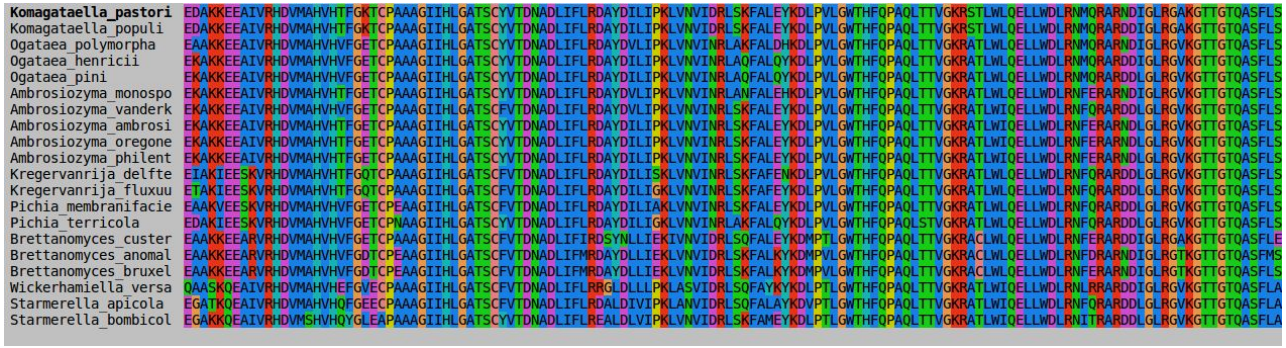
https://github.com/smirarab/sepp/blob/master/README.UPP.md

Trimming

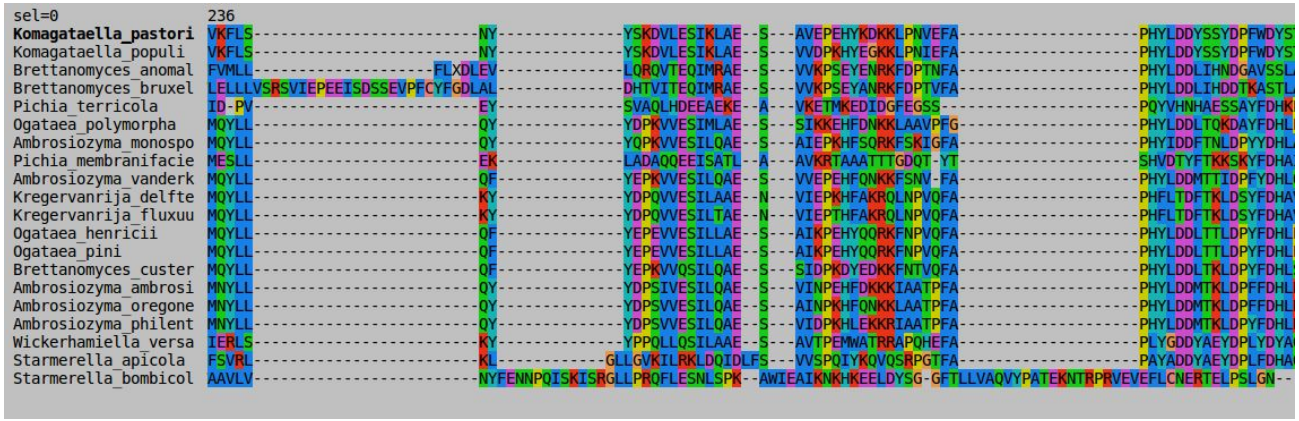
Lisa Pokorny & Marina Marcet-Houben

Multiple sequence alignments can have many different forms

Conserved



Noisy



Why can alignments be noisy?

Biological reasons: If we compare sequences from proteins from distantly related species, there is high chance that only the functional part of the protein is well conserved in terms of sequences. Other parts, such as loops, are more likely to have altered their amino acid sequence, bot in terms of amino acid content and with the presence of indels.
Even when the prediction of the multiple sequence alignment is correct, it may negatively affect the inference of the phylogenetic tree.

Errors:

- Errors derived from genome assembly problems
- Errors derived from gene prediction
- Errors derived from alignment of multiple sequences

Different trimming programs aim to solve some of the problems caused by the presence of gaps or badly aligned regions.

Biological reasons

Errors in multiple sequence alignments

Traditional block-based methods
(Gblocks, trimAl, BMGE, Zorro, …)

Errors in sequencing

Errors in gene predictions

Segment based methods
(Prequal, hmmcleaner)

To trim or not to trim

The trimming of multiple sequence alignments is still a controversial topic. Some authors claim that it's necessary to improve phylogenetic reconstruction and the detection of evolutionary events.

Improvement of phylogenies after removing divergent and ambiguously aligned blocks from protein sequence alignments.

Talavera G[1], Castresana J.

Whereas some others claim it either does not have any effect or it make the problem worse.

Current Methods for Automated Filtering of Multiple Sequence Alignments Frequently Worsen Single-Gene Phylogenetic Inference

Ge Tan,[1,2,3] Matthieu Muffato,[4] Christian Ledergerber,[1] Javier Herrero,[4,5] Nick Goldman,[4] Manuel Gil,[6,7] and Christophe Dessimoz[5,4,*]

Table 4
Topological accuracy of single-gene phylogenies

VERTEBRATA		
version	mean	loss (%)
RAW	65.64%	NA
RAW (long)	68.85%	NA
HMM	65.23%	5.61
HMM-L	65.41%	4.22
HMM-LS	65.58%	2.68
PREQUAL	65.73%	3.06
BMGE	64.83%	4.83
TrimAl	65.28%	1.8
HMM Random	64.56%	5.61
HMM + BMGE	63.94%	13.38
HMM + TrimAl	62.90%	13.76
MIN	68.71%	0.71
MIN + HMM	68.67%	6.43

There's little difference.

BMC Evol Biol. 2019 Jan 11;19(1):21. doi: 10.1186/s12862-019-1350-2.

Evaluating the usefulness of alignment filtering methods to reduce the impact of errors on evolutionary inferences.

Di Franco A[1], Poujol R[2], Baurain D[3], Philippe H[4,5].

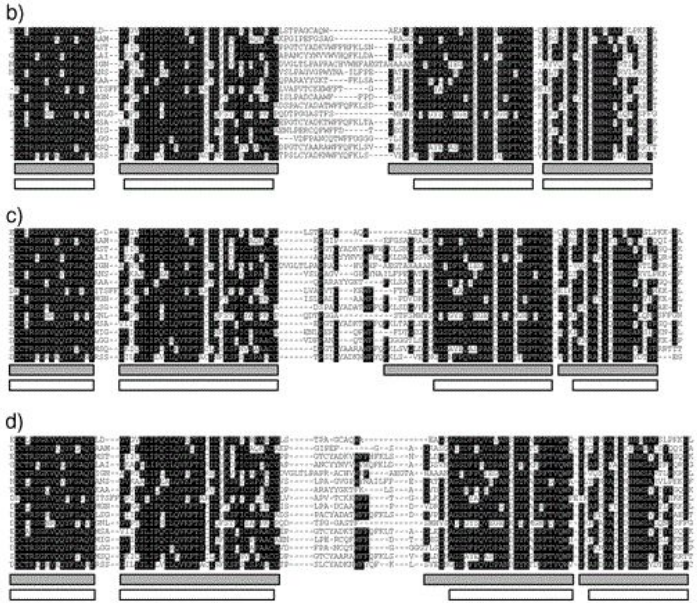
The probable truth: it depends on the dataset and the methodology used.

Program	Number of citations (Google scholar)
BMGE	381
trimAl	1737
Gblocks	5736

Trimming alignments tends to be part of a normal phylogenetic reconstruction pipeline.

The first one: GBLOCKS (http://molevol.cmima.csic.es/castresana/Gblocks.html)

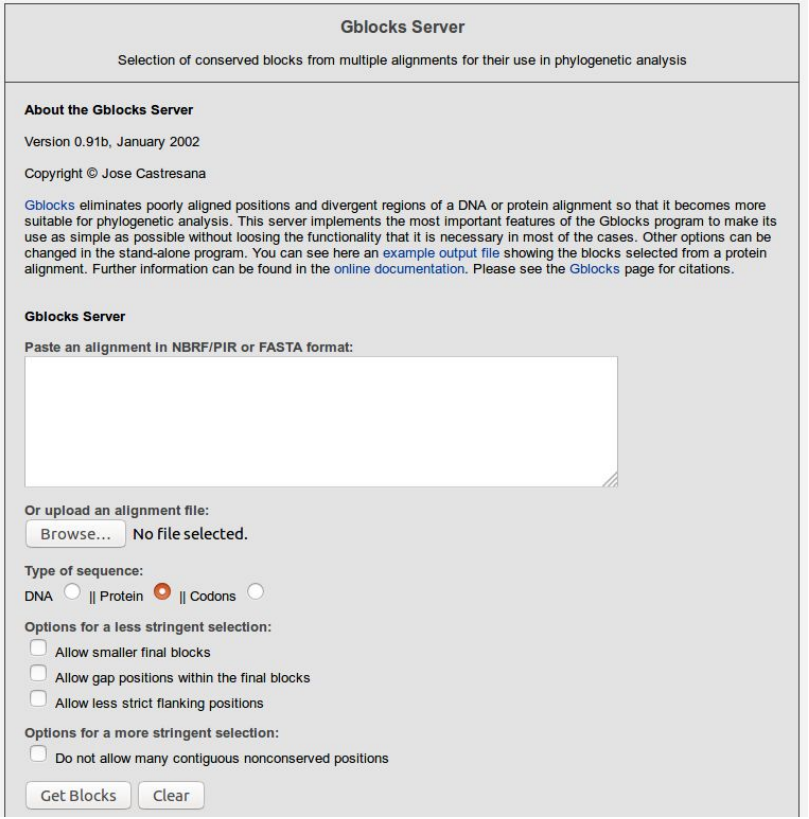
Gblocks selects blocks in a similar way as it is usually done by hand but following a reproducible set of conditions. The selected blocks must fulfill certain requirements with respect to the lack of large segments of contiguous nonconserved positions, lack of gap positions and high conservation of flanking positions, making the final alignment more suitable for phylogenetic analysis.



The white and grey blocks under the alignments represent the parts of the alignment that Gblocks would keep using a more relaxed and a more stringent approach.

How to run Gblocks: website

The is an on-line server that you can use if you only want to trim one alignment.



At the end of the alignment representation there's a link to obtain the trimmed alignment.

How to run Gblocks: command line

```
mmarcet@saturn:~/Desktop/evomics$ Gblocks
*************************************************
                  GBLOCKS 0.91b
SELECTION OF CONSERVED BLOCKS FROM MULTIPLE ALIGNMENTS
        FOR THEIR USE IN PHYLOGENETIC ANALYSIS
*************************************************

o. Open File

b. Block Parameters

s. Saving Options

g. (Get Blocks)

q. Quit


Your Choice:
```

Used to upload your alignment file (Fasta or NBRF/PIR format)

```
Your Choice: o


Please enter a DNA or protein alignment in NBRF/PIR or FASTA format or a paths file.
File Name: cytochrome.alg.forward.fasta

96 sequences and 1543 positions in the first alignment file:
cytochrome.alg.forward.fasta
```

Once the alignment has been introduced it will go back to the main menu and you can choose option g which will execute the program with the default parameters.

```
Your Choice: g


cytochrome.alg.forward.fasta
Original alignment: 1543 positions
Gblocks alignment:  148 positions (9 %) in 8 selected block(s)
```

Example Gblocks -

A- Start Gblocks and open the file called EOG092D2PES.alg (use the o option). Then use g to obtain the blocks with the default parameters. Pay attention to how long your resulting alignment is and which percentage of the alignment has been kept with the default parameters.

If you go to the folder where the sequences are, you will see that there are now two new files: EOG092D2PES.alg-gb contains the trimmed alignment while EOG092D2PES.alg-gb.htm has a html representation of the alignment and the blocks that have been kept. Open it and have a look. Afterwards, rename the file to EOG092D2PES.default.htm so that it is not lost.

Option b in the Gblocks menu allows you to change the parameters by which the blocks are defined.

B.- Use this option to adjust the minimum number of sequences needed to define a conserved position and the minimum number of sequences to define a flanking region. Assign a random number within the accepted scope. How does this affect your alignment?

Again have a look at the results. Do you see a change between the blocks that have been selected? Again rename the file so that it is not lost

C.- Now decrease the length of your conserved blocks. Did that have an effect on your alignment? What happens if you increase it?

Check out the results again and compare them with the previous runs of Gblocks.

D.- These alignments are in general pretty conserved. What do you think would happen if we were comparing more distantly related species?

trimAl was initially born because Gblocks could be too restrictive when automatically building thousands of alignments. Unlike Gblocks, trimAl implements different trimming strategies based on gap content, similarity or consistency across different alignment methods. It also implements a conservation score which always ensures that a percentage of the alignment is conserved.

(https://github.com/scapella/trimal)

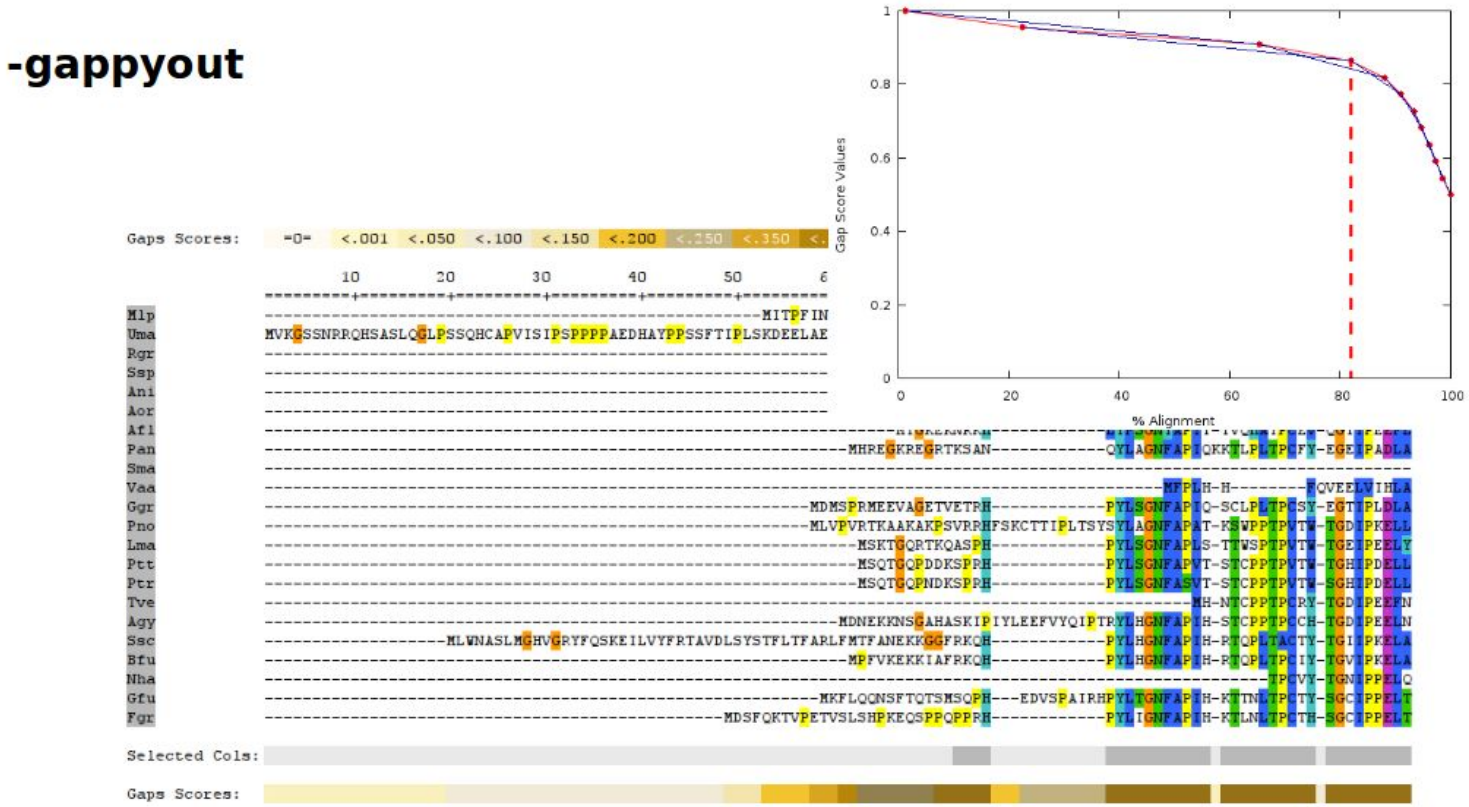
trimAl has a wide array of options, including user defined trimming parameters:

```
-gt -gapthreshold <n>    1 - (fraction of sequences with a gap allowed). Range: [0 - 1]
-st -simthreshold <n>    Minimum average similarity allowed. Range: [0 - 1]
-ct -conthreshold <n>    Minimum consistency value allowed.Range: [0 - 1]
-cons <n>                Minimum percentage of the positions in the original alignment to conserve. Range: [0 - 100]
```

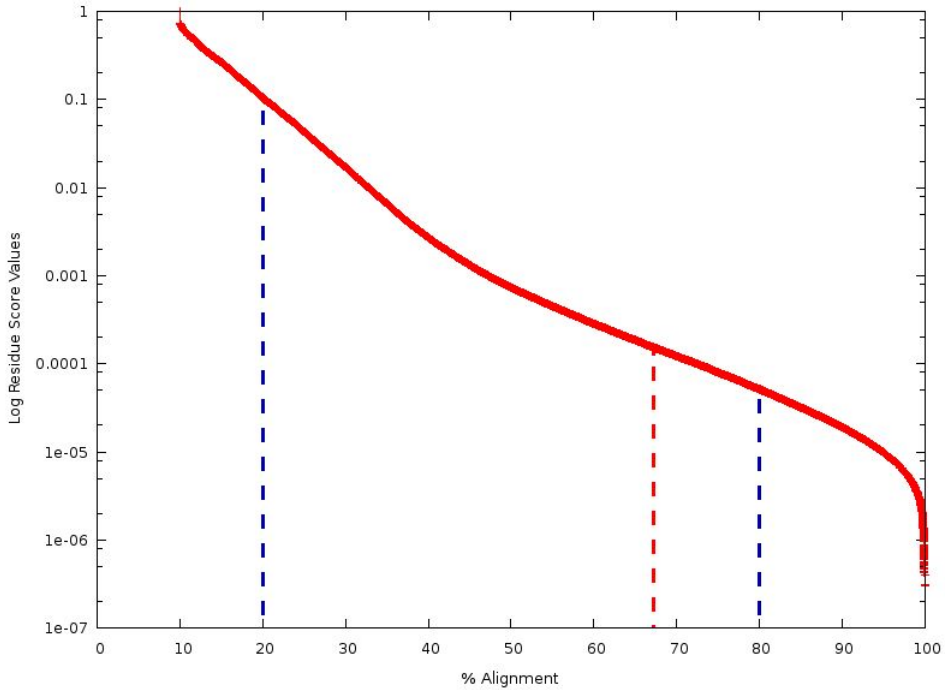
And automated methods that predict the best parameters for a given alignment:

```
-gappyout                Use automated selection on "gappyout" mode. This method only uses information based on gaps' distribution. (see User Guide).
-strict                  Use automated selection on "strict" mode. (see User Guide).
-strictplus              Use automated selection on "strictplus" mode. (see User Guide).
                         (Optimized for Neighbour Joining phylogenetic tree reconstruction).
```

Automated methods:



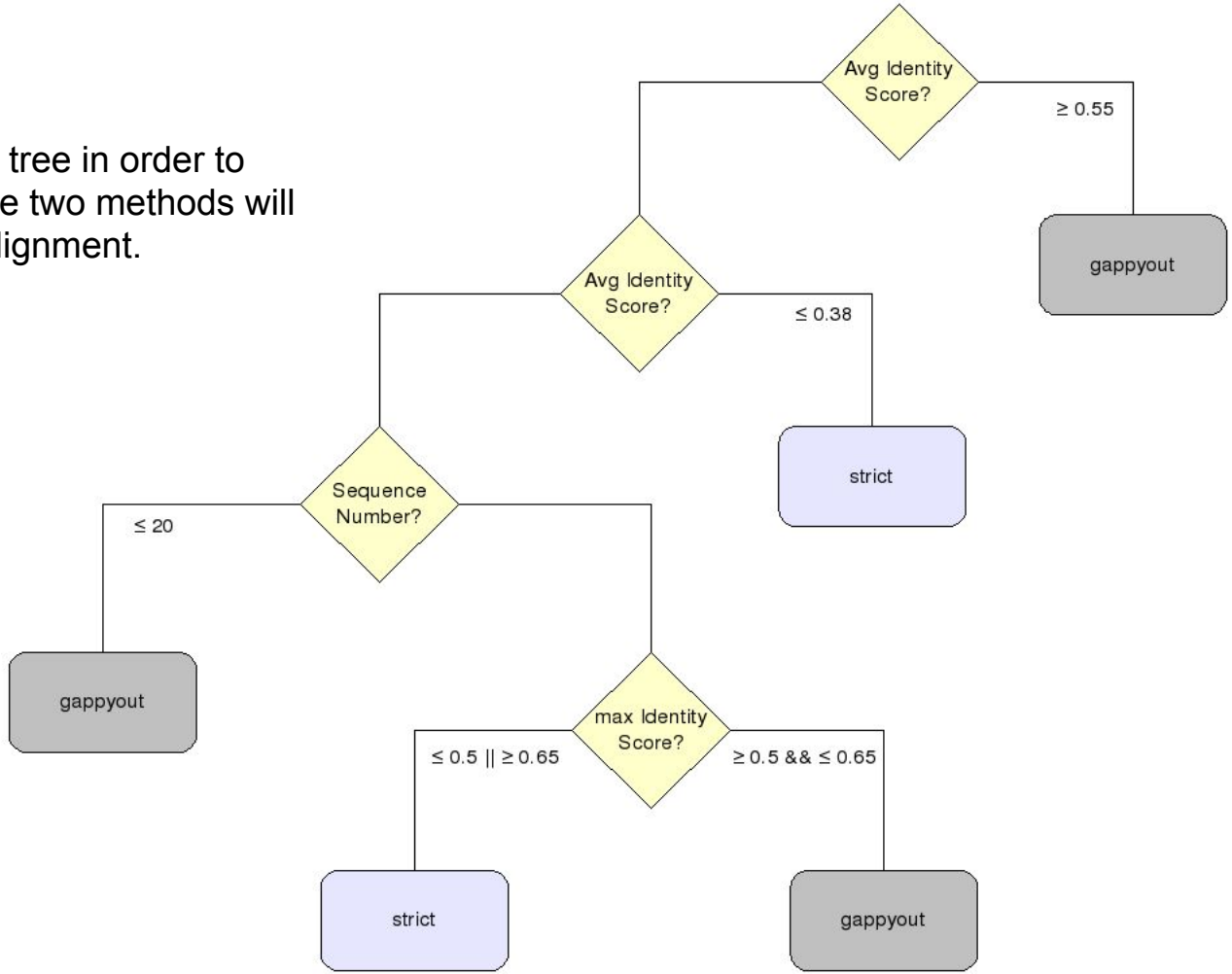
-strict -strictplus



Gappy-out + trimming by similarity scores → they will only delete blocks of data so if one column has been marked to be deleted but it is surrounded by non-marked columns it will be kept in the alignment. The two methods differ on how they define the block size.

-automated1

Will use a decision tree in order to choose which of the two methods will work best on the alignment.



readAl: Reformatting MSAs

One of the main problems of alignments is the fact that different formats exist, and there may not be a match between the output format of an alignment program and the input format the next program needs.

```
#NEXUS

[!Imported PHYLIP file "03O1O3phylip.phy" (Fri Jan 03 12:43:38 2003)]

Begin data;
    Dimensions ntax=29 nchar=949;
    Format datatype=nucleotide gap=- missing=? matchchar=. interleave;
options gapmode=missing;
    Matrix
487GJS        AACGTTACCAAACTGTTGCCTCGGCGGGAAAA--TTCCATCGCCCCGGG
484GJS        AACGTTACCAAACTGTTGCCTCGGCGGGAAAA--TTC-ATCGCCCCGGG
476GJS        AACGTTACCAAACTGTTGCCTCGGCGGGGAAA--TTCCATCGCCCCGGG
481GJS        AACGTTACCAAACTGTTGCCTCGGCGGGAAAA--TTCCATCGCCCCGGG
497GJS        AACGTTACCAAACTGTTGCCTCGGCGGGGAAA--TCTCATCGCCCCGGG
501GJS        AACGTTACCAAACTGTTGCCTCGGCGGGAAAA--TTTCAT-GCCCCGGG
477GJS        AACGTTACCAAACTGTTGCCTCGGCGGGAAAA--TTCCATCGCCCCGGG
493GJS        AACGTTACCAAACTGTTGCCTCGGCGGGAAAA--TTTCAT-GCCCCGGG
486GJS        AACGTTACCAAACTGTTGCCTCGGCGGGAA-----TTCAT-GCCCCGGG
TminuEX       AACGTTACCAAACTGTTGCCTCGGCGGGAAAA--TTTCAT-GCCCCGGG
Tharzia       GCCCATCTACGGAAGATCATTCCAGAACACCGCTGGTATTGGCCAGACT
Tvirens       GCCCATTTATGGACGATCATTCCAGAACACCGCTGGTATTGGCCAGACT
Thamatum      GCCCATCTATGGACGATCTTTCGAGAGCACCGGTGGAATTGGCCAGTCT
473gjs        GCCTATTTACGGACGATCCTTCGAGAGCACCACTGATATTGGCAAGTYT
Hpiluli       GCCTATTTACGGACGATCTTTTGAGGCCACNACNGATATTGGCCAGTTT
153DGJS       GCCTATTTACGGACGATCTTTTTAGKGAACCACTGATATTGGTCAGTCT
130DGJS       GCCTATTTACGGACGATCTTTTGAGGGCACCACTGATATTGGCAAGGCT
135DGJS       GCCTATTTACGGACGATCTTTTGAGGGCACCACTGATATTGGCAAGGCT
139DGJS       GCCTATTTACGGACGATCTTTTGAGGGCACCACTGATATTGGCAAGTCT
147GJS        GCCTATTTACGGACGATCTTTTGAGGGCACCACTGATATTGGCAAGTCT
138DGJS       GCCTATTTACGGGCGATCTTTCGAGGGCACCACTGATATTGGCAAGGCC
491GJS        GCCTATTTACGGACGATCTTTTGAGGGCACTACTGATATTGGCAAGGCT
460GJS        GCCTATTTACGGACGATCTTTTGAGGGCACTACTGATATTGGCAAGGCT
467GJS        GCCTATTTACGGACGATCTTTTGAGGGCACCACTGATATTGGCAAGGCT
124DGJS       GCCTATTTACGGACGATCTTTTGAGGGCACCACTGATATTGGCAAGGCT
150DGJS       GCCTATTTACGGACGATCTTTTGAGGGCACCACTGATATTGGCAAGGCT
croceum       GCCTATTTACGGACGATCTTTTGAGGGCACTACTGATATTGGCAAGGCT
polysp        GCCTATTTACGGACGATCTTTTGAGGGCACTACTGATATTGGCAAGGCT
toment        gcccatctacggacgatcattccagawcaccgmcggyattggccakact
    ;
End;
begin sets;
charpartition genes = ITS:1-383, EF:384-495, ECH:496-.;
;
end;
```

NEXUS format

```
>TRY2_RAT/24-239
----------------------IVGGYTCQENSVPYQVSLNSGY------------HFC
GGSLI------NDQ-WV-VSAAHCYKS----------RIQVRLGE-HNINVLEGN-----
-----EQFVNAAKIIKHPNFDRKT-L-------------------------NNDIMLIKLS
SP--VKLNARVATVALPS---SCA---PAGTQCLISGWGN----------TLSSGV-----
-------NEPDLLQ-CLDAP-LLPQADCEAS---YPGK--------ITDNMVCVGFL---
-EGG-KDSCQGDSGGPVVCNGE--------LQGIVSWG-YGCALPDN---PGVYTKVCNY
VDWI---------------

>Q16LB2_AEDAE/136-374
----------------------ILNGIEADLEDFPYLGALALLDNYT-----STVSYRC
GANLI------SDR-FM-LTAAHCLFG--------KQAIHVRMGTLSLTDNPDED------
----APVIIGVERVFFHRNYTRRPIT-----------------------RNDIALIKLN
RT---VVEDFLIPVCLYT---EQNDP-LPTVPLTIAGWGG----------NDSAS-----
-------LMSSSLM-KASVT-TYERDECNSL---LAKKI-----VRLSNDQLCALGRSEF
NDGLRNDTCVGDSGGPLELSIGR----RKYIVGLTSTG-IVCGNE-F---PSIYTRISQF
IDWI---------------
```

FASTA format

```
| 150 1075
Phy0007P00_GIBZE    ------------------------------------M---E-TQSSNG--N-APAT-K
Phy0007T05_GIBZE    ---------------MRG-A-FIP-WIDKYGELLRVCG---N---------VGTSEY-AR
Phy000D2RD_SCHPO    ------------------------------------------------------MG-K
Phy000FQYT_ASPFU    MFHT-----------AL-N-PRA-A--------------------------TGEERR-G
Phy000FR4G_ASPFU    ----LR--RRL--HHSKH-R-ICL-QFALSLRKTSIGSVDSM--------GESCKAP-Q
Phy000FRUS_ASPFU    ----------MDP----NSSSST-QSLPKVSLPRITTAK---P-R----------T---
Phy000FSLI_ASPFU    -----P--------S---PTAER-SVNSKVAIPRSANPS---S-W----------T--S
Phy000FU66_ASPFU    MASH-------RTS-LQ-PG-SSK-YINL-NERYKYAWM---Q-HQLQPS--S-PQRP-K
Phy000FUZH_ASPFU    MFHT-----------EG-P-GAS-APAK-----GRE--------------RQQSVG-R
Phy000FV0G_ASPFU    ---------------------------MSGHPSEELSAD----YQLQSGGYFHYARSP-L
Phy000FW1S_ASPFU    ------------------------------------MA-------DY-QN--VRPLRR-E
Phy000FWHP_ASPFU    ------------------------------------------------------MAS-T
Phy000FX8T_EMENI    ----G---SMSEQ-R---PSEPS-TPGSKIPIPRVSQLR---A-Y----------G
Phy000FXVQ_EMENI    -------------------------MDASVQPGHHLVRA---------PAPGAEP-S
Phy000FYYQ_EMENI    MFIA-----------EA-A-NAS-KDLESGPRNPKSV---T----------GRTSTG-R
Phy000FZYA_EMENI    ------------PNDIR-A-LIT-GASGGIGAACAHQLHLA---------LTNS-AV--
Phy000G015_EMENI    ------------------------------------------------M---SSRGPT-R
Phy000G06G_EMENI    GQYEALQVSPILRQRSTL-A-VIG-GDLLQNGHSA---SEAY---------GSVFRSR-H
Phy000G0IY_EMENI    ------------------------------------------------M---SKNMGS-T
Phy000G1Z6_EMENI    MFHT-----------EG-P-AAS-APARA----GRE--------------RQSSIG-R
Phy000G28K_EMENI    ---------------------------TDQPH-------GAQPGILHASRLQGR-K
Phy000G57Q_ASPNG    ------------------------------------------------------DISQSR-R
Phy000G8Y0_ASPNG    MYRS-------------------------------S--------------RHVE---K
Phy000G9MD_ASPNG    MASA--AD--EDDSDSFF-E-DHD-ASPGHDLMMKDDAL------GDGKP--DPLPMQ-K
Phy000GA9V_ASPNG    ------------------------------------------MNQTHREGEPLA-K
Phy000GBJT_ASPNG    ------------------------------------------------------MAS-T
Phy000GBKP_ASPNG    -----SVPDGFPTRFNSG-I-I-----PSTPTKLCNNIW----LEMDS--SGRSRGKS-R
Phy000GEPP_ASPNG    -------------K---R-----------------------------------SEV--
```

PHYLIP format

readAl: Reformatting MSAs

readAl is a sister program to trimAl that allows us to convert alignment between each other.

readal -in [input file] -format -out [output file]

Input file → Alignment file

Output file → Resulting file

Format → Can be any of the formats that readAl has and that you wish to use as output: Fasta, phylip, mega, nexus, clustal,...

Use one of the previous alignments to try out the following things
1.- Open the alignment file (EOG092D2PES.alg) and check in which format it has been generated.
Now use readal to (make sure each result is in a different file):

- Change the format of the current alignment to fasta format
- Change the format of the current alignment to nexus format
- Change the format of the current alignment to clustal format
- Use the -onlyseqs option

Open the different files and notice the differences between the alignment formats.

Tip: readal is run like this:

```
readal -in alignment_file -out trimmed_alignment_file -format
FORMAT_NAME
```

You can check out all the formats supported by readal by typing: readal -h

2.- Use trimAl to trim the alignment (EOG092D2PES.alg) according to a gap threshold using the following parameters:

- A gap threshold of 0.1 (-gt 0.1)
- A gap threshold of 0.5 (-gt 0.5)
- A gap threshold of 0.9 (-gt 0.9)

Make sure that the output of your alignment is in phylip format. Now you can visualize each alignment either using a text editor or using seaview. Which of the previous commands deletes the largest amount of columns?

3.- Now use the -gt 0.5 command but add a conservation score of different values: 30, 50 and 80 (-cons option). Again make sure that your output alignment is in phylip format. Which effect does it have on the trimmed alignment?

4.- Now instead of using the gap threshold, we'll be using the similarity threshold (-st). Repeat the trimming of the original alignment using different similarity thresholds (0.1, 0.5 and 0.9). Again, how does the alignment trimming vary? Which approach is more aggressive? How can you make sure you don't loose all the alignment?

6.- Now we are going to use the automated trimming methods. Trim your alignment using:

- Use the different automated trimming methods: -gappyout, -strict, -strictplus, -automated1
- Use the more radical methods to delete all the columns with gaps in your alignment: -nogaps

Of all the trimming strategies you've tried, which is the best one? Can you know?

BMGE (Block mapping and gathering through entropy)

BMGE is a trimming method that bases its trimming on the calculation of the entropy generated from moving between the different stated found in each column in an alignment. It compares these entropy value to standard substitution matrices to see whether the entropy values have biological meaning. For each column of the alignment a score is calculated. BMGE then removes the blocks with high entropy values (poorly conserved regions).

BMGE is a command line program that runs on Java, so you'll have to have Java installed for it to run.

The easy way to run BMGE is simply calling:

```
BMGE -i EOG092D2PES.alg -t AA -o EOG092D2PES.BMGE.alg
```

This will result in a trimmed alignment

You can obtain a visual output by using the option -oh

```
BMGE -i EOG092D2PES.alg -t AA -oh EOG092D2PES.BMGE.html
```

BMGE also implements other trimming methods such as gap based:

```
BMGE -i EOG092D2PES.alg -t AA -h 1 -w 1 -g 0.1 -o
EOG092D2PES.gapBased.phy
```

Where -h makes sure there is no entropy trimming and -w says that the sliding window should be of one so that all columns are considered for trimming.

Prequal

Another kind of trimming tools are those that try to address errors caused by miss-assemblies or gene prediction errors.

Different to the previous programs they work on the multi-sequence fasta and they try to identify non-homologous regions within the sequences included in the multi-fasta. These non-homologous regions are then masked or altogether removed from the sequence prior to the alignment.

How to run Prequal:

A.- Note that this time the input is an un-aligned multi fasta file.

```
prequal inputFile
```

The result will be a filtered fasta file where some parts of the alignments have been deleted and some others are masked (see stretches of X)

Alignment and trimming challenge

Lisa Pokorny & Marina Marcet-Houben

In phylogenomics we will not work with one single set of sequences that have to be aligned and trimmed. But rather with a large set. So, how can we work with them? Which programs can we use? How can we adjust the parameters to create good alignments?

Untangling the early diversification of eukaryotes: a phylogenomic study of the evolutionary origins of Centrohelida, Haptophyta and Cryptista

Article (PDF Available) *in* Proceedings of the Royal Society B: Biological Sciences 283(1823):20152802 · January 2016 *with* 730 Reads
DOI: 10.1098/rspb.2015.2802

https://datadryad.org/resource/doi:10.5061/dryad.rj87v