




# CIMSA: Customizable Integrated Multiple Sequence Alignment Package

Mengmeng Kuang<sup>1</sup>

<sup>1</sup> University of Hong Kong, Hong Kong SAR

DOI: [10.21105/joss.02493](https://doi.org/10.21105/joss.02493)

## Software

- [Review](#) 
- [Repository](#) 
- [Archive](#) 

Editor: [Pending Editor](#) 

Submitted: 17 July 2020

Published: 19 July 2020

## License

Authors of papers retain copyright and release the work under a Creative Commons Attribution 4.0 International License ([CC BY 4.0](https://creativecommons.org/licenses/by/4.0/)).

## Summary

CIMSA, as a python package, is provided to simplify the writing of repeated code in the study of multiple sequence alignment (MSA), so that researchers can pay more attention to algorithm design. MSA, as a comparative basis in Bioinformatics, has received much attention in the past few decades. In the fruitful research on MSA methods, progressive methods are undoubtedly the most eye-catching kindest. This type of MSA methods usually includes five main steps. MSA algorithm researchers often need to write a lot of established steps for a certain part of innovation, which very affects efficiency. This research develops an MSA program package that can select different operating steps by simple parameters, which is an efficient assistance for future MSA algorithms research.

## Statement of need

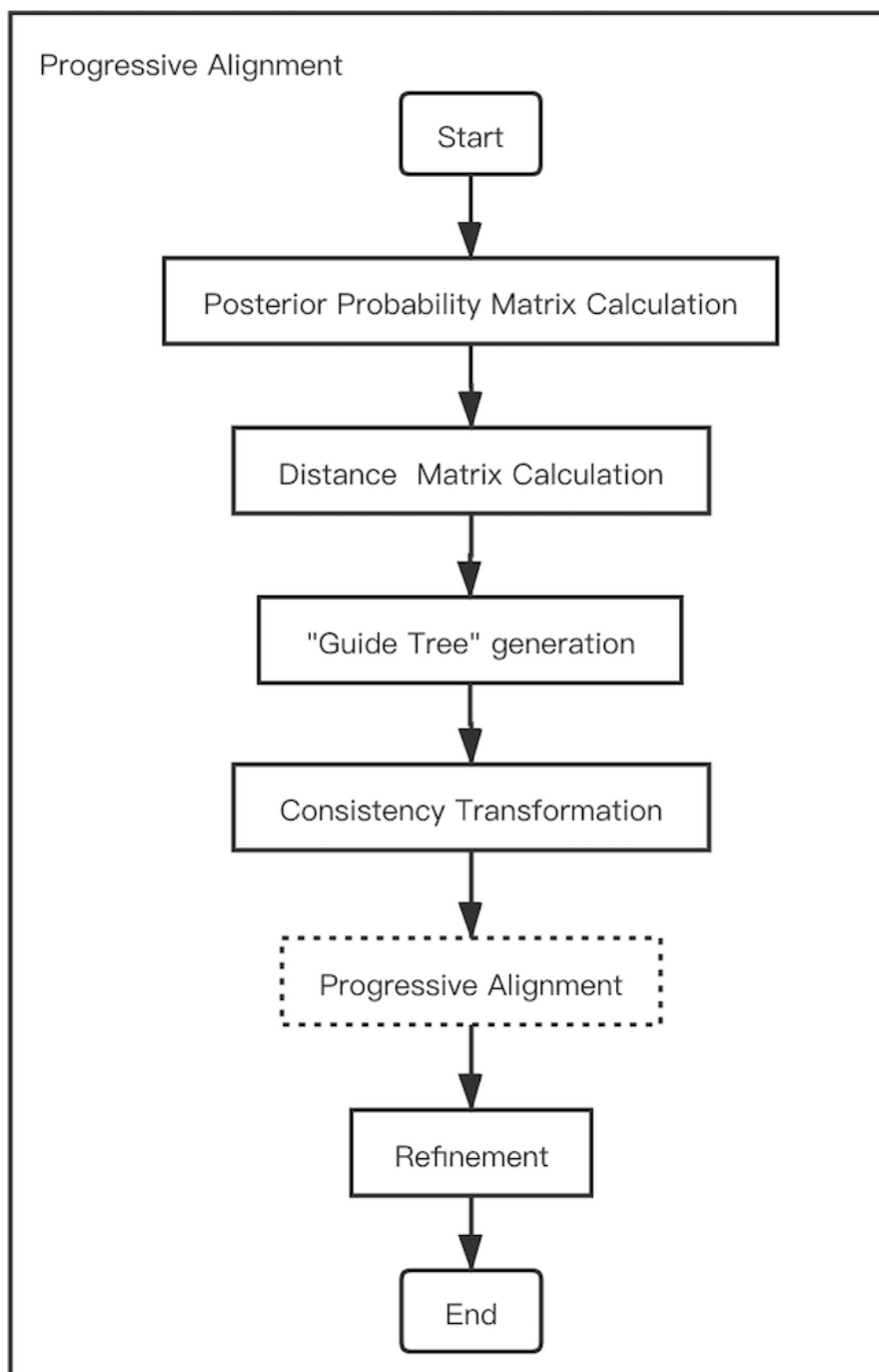
Till now, there is no efficient and accurate MSA tool of progressive strategy using Python programming language or providing a Python interface. In this study, in addition to adding a Python interface to the famous MSA programs, we have rewritten and integrated many existing progressive MSA programs to select different execution methods for each step through different parameter settings. These will definitely contribute to the improvement of MSA algorithm research by improving the development and verification efficiency of MSA strategies.

## Introduction

Multiple sequence alignment (MSA) plays an important role in finding co-evolutionary ancestors of sequences and determining structural or functional roles of the sequences (Notredame (2007)), which is why it has attracted many researchers or developers in the past few years. The progressive strategy is the most concerned part of the MSA methods because of its fixed steps and high accuracy, which has always been favored by the majority of MSA algorithm researchers (Edgar & Batzoglou (2006)).

MSA programs that adopt progressive strategy with high efficiency and accuracy, such as ProbCons(Do, Mahabhashyam, Brudno, & Batzoglou (2005)), ProbAlign(Roshan & Livesay (2006)), MSAProbs(Liu, Schmidt, & Maskell (2010)), GLProbs(Ye et al. (2014)), and ProbPFP(Zhan et al. (2018)) all include five main modules: (1) Calculation of the posterior probability matrix, (2) Calculation of the distance matrix, (3) Guide tree generation, (4) Probabilistic consistency transformation and (5) Refinement as shown at [Figure 1](#). These popular MSA programs are all written in the C/C++ programming language which is difficult to read and hard to understand, which makes it very challenging for other researchers to analyze their contributions. Besides, because some of these programs perform the same or similar

steps, if managing their codes in a unified format, it will be easier for future researchers to make greater contributions on this basis.



**Figure 1:** The processes of a regular progressive alignment method.

Compared to C/C++, Python is a programming language that is easier to use and more readable. Many Bioinformatics researchers have started using Python as their preferred pro-

programming language for their research and development. As an indispensable tool for Bioinformatics, the MSA tool that uses Python as the entrance or the development language will be more conducive to the research on the upper levels in the Bioinformatics field.

## Design

The CIMSA package follows the traditional five-step progressive MSA method. In each step, in addition to implementing the methods that have been studied now, additional calculation methods that are not adopted by mainstream MSA tools are added to facilitate future researchers to compare or do more research.

**Posterior probability matrix**, as its name implies, needs to be calculated using a production model. Hidden Markov models (HMMs) have been widely used in previous decades of research, including double affine HMMs and local HMMs. What's more, the partition function can also get very good results in this task. In addition to implementing these three calculation methods separately in this study, we also added their each two combinations and all the three combinations by calculating their root mean square.

**Distance matrix** directly affects the guide tree generation, which means it has a great research value. The distance matrix calculations implemented in CIMSA all have the form as described in [Equation 1](#).

$$dist[a][b] = dist[b][a] = 1.0 - \frac{Score(a, b)}{Length(a, b)} \quad (1)$$

where  $a, b$  means two sequences,  $Score(a, b)$  could be the best probability or the number of matched pairs in optimal alignment of pair  $\langle a, b \rangle$  and  $Length(a, b)$  stands for the maximum, minimum or average length of  $\langle a, b \rangle$  or the length of the optimal pairwise alignment.

**Guide tree** is a binary tree data structure used for storing the similarity between sequences and determine the alignment order. The unweighted pair-group method with arithmetic means (UPGMA) is the most commonly used guide tree construction method. Besides, the weighted pair-group method with arithmetic means (WPGMA), Single-linkage(Sibson (1973)) and Complete-linkage(Defays (1977)) are all available clustering methods and accomplished in our package as well. The key variation between these four different clustering methods is the processing of the distances between the parent node and other nodes after merging two leaf nodes or two subtrees

**Probabilistic consistency transformation (PCT)** is to adjust the current transition probability between sequences by the posterior probability between other sequences. There are two main forms in the current research, one is the weighted version and the other is unweighted. The weighted version could be found at [Equation 2](#) and if all the weight  $w_s$  equal to 1, it was transferred to the unweighted one.

$$P'_{ab} = \frac{1}{wN} ((w_a + w_b)P_{ab} + \sum w_c P_{ac} P_{cb}) \quad (2)$$

where  $N$  means the number of sequences in protein sequence set  $S$  of that family and  $P$  is the posterior probability matrix.

In addition to the two calculation methods mentioned above, we have also implemented a weighted and an unweighted "Subtree PCT" methods to improve efficiency, that is, instead of all sequences in the protein family used for relaxation, the set  $S$  is composed of the sequences in the smallest subtree contains the two sequences that need to be relaxed each time.

After these four main steps, in fact, we can already generate an MSA by performing pair alignments and profile-profile alignments. The greatest impact on the whole MSA process

is the starting pair alignments, so after having obtained a rough MSA, we consider using a refinement strategy to reduce the impact of starting pair alignment on correctness.

**Refinement** is to split the existing alignment into two different groups through a certain segmentation strategy and then do some fine-tuning. Iterative refinement and tree-based refinement have proven to be two efficient strategies, which randomly divide sequences into two groups or split the guide tree to obtain two different sequences.

## Results

CIMSA was developed using C/C++ and Python programming languages, which is a console program that can select different modules of the progressive alignment by entering different parameters.

CIMSA implements and integrates 7 posterior probability matrix calculation methods, 5 distance matrix measurement methods, 4 guide tree generation methods, 4 probabilistic consistency transformation methods, and 2 refinement methods. The total permutation can reach more than one thousand different MSA program streams. Some of these combinations have been researched and verified to be of high accuracy. At the meantime, each component retains the interface, so that researchers can write some extra modules to continue their research.

## Conclusion

CIMSA is a high efficient and parameter rich package for multiple sequence alignment in the extremely popular C/C++ and Python programming languages, and the only one of its kind. We, therefore, expect it to be a boon to current and future MSA algorithms researchers.

## Acknowledgments

We acknowledge contributions from the developers and researchers of the open source softwares mentioned in this paper.

## References

- Defays, D. (1977). An efficient algorithm for a complete link method. *The Computer Journal*, 20(4), 364–366. doi:[10.1093/comjnl/20.4.364](https://doi.org/10.1093/comjnl/20.4.364)
- Do, C. B., Mahabhashyam, M. S., Brudno, M., & Batzoglou, S. (2005). ProbCons: Probabilistic consistency-based multiple sequence alignment. *Genome research*, 15(2), 330–340. doi:[10.1101/gr.2821705](https://doi.org/10.1101/gr.2821705)
- Edgar, R. C., & Batzoglou, S. (2006). Multiple sequence alignment. *Current opinion in structural biology*, 16(3), 368–373. doi:[10.1016/j.sbi.2006.04.004](https://doi.org/10.1016/j.sbi.2006.04.004)
- Liu, Y., Schmidt, B., & Maskell, D. L. (2010). MSAProbs: Multiple sequence alignment based on pair hidden markov models and partition function posterior probabilities. *Bioinformatics*, 26(16), 1958–1964. doi:[10.1093/bioinformatics/btq338](https://doi.org/10.1093/bioinformatics/btq338)
- Notredame, C. (2007). Recent evolutions of multiple sequence alignment algorithms. *PLoS Comput Biol*, 3(8), e123. doi:[10.1371/journal.pcbi.0030123](https://doi.org/10.1371/journal.pcbi.0030123)

- Roshan, U., & Livesay, D. R. (2006). Probalalign: Multiple sequence alignment using partition function posterior probabilities. *Bioinformatics*, 22(22), 2715–2721. doi:[10.1093/bioinformatics/btl472](https://doi.org/10.1093/bioinformatics/btl472)
- Sibson, R. (1973). SLINK: An optimally efficient algorithm for the single-link cluster method. *The computer journal*, 16(1), 30–34. doi:[10.1093/comjnl/16.1.30](https://doi.org/10.1093/comjnl/16.1.30)
- Ye, Y., Cheung, D. W.-I., Wang, Y., Yiu, S.-M., Zhang, Q., Lam, T.-W., & Ting, H.-F. (2014). GLProbs: Aligning multiple sequences adaptively. *IEEE/ACM transactions on computational biology and bioinformatics*, 12(1), 67–78. doi:[10.1109/TCBB.2014.2316820](https://doi.org/10.1109/TCBB.2014.2316820)
- Zhan, Q., Wang, N., Jin, S., Tan, R., Jiang, Q., & Wang, Y. (2018). ProbPFP: A multiple sequence alignment algorithm combining partition function and hidden markov model with particle swarm optimization. In *2018 IEEE International Conference on Bioinformatics and Biomedicine (bIBM)* (pp. 1290–1295). IEEE. doi:[10.1109/bibm.2018.8621220](https://doi.org/10.1109/bibm.2018.8621220)