CIAlign Release 1.1.0

Improvements (**for additional information, please see notes on pages 2-4)**

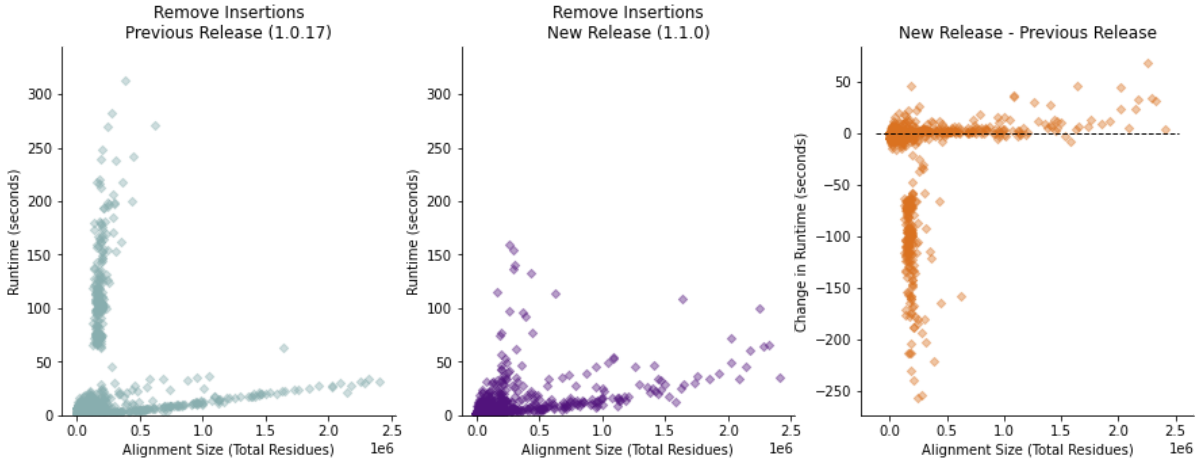
- The "**remove insertions**" function now runs much faster, particularly when the maximum insertion size is very large (see notes on **page 2** of this document).
- A new "**crop divergent**" function is now available, see documentation here. Examples are shown on **pages 3 and 4** of this document. This function allows the user to isolate the most conserved region of the alignment by specifying a minimum percentage of identical residues and percentage of non-gap residues, over a number of columns, which defines the edge of the alignment.
- CIAlign can now generate **position weight**, **position probability** and **position frequency** matrices, to represent an alignment numerically and to use as input for other software, see documentation here. Outputs can be formatted for the MEME and BLAMM bioinformatics software.
- Users can now specify, using the **retain** functions, sequences which they would like CIAlign not to remove, either with any cleaning function or with a specific cleaning function. The sequences to be retained can be specified as a command line parameter, a file with a list of sequence names or a regular expression
- The **plot_coverage** function is now the **plot_stats** function and will generate three plots, showing coverage, Shannon entropy and information content per column.
- Users can now extract part of an alignment based on their own coordinates, using the **get_section** function, documented here.
- An **additional colour palette** is now available for mini alignments and sequence logos,
- It is now possible to **convert T to U** in alignments, as well as U to T
- The CIAlign manual is now hosted at ReadTheDocs - https://cialign.readthedocs.io - and is more readable and comprehensive.
- Unit test coverage is now 100% across all module files (excluding import lines and logging)

Module	*statements*	*missing*	*excluded*	*coverage ↓*
CIAlign/__init__.py	0	0	0	100%
CIAlign/consensusSeq.py	267	0	15	100%
CIAlign/cropDiv.py	27	0	3	100%
CIAlign/cropSeq.py	43	0	0	100%
CIAlign/insertions.py	91	0	5	100%
CIAlign/matrices.py	61	0	5	100%
CIAlign/miniAlignments.py	127	0	8	100%
CIAlign/palettes.py	10	0	0	100%
CIAlign/parsingFunctions.py	133	0	22	100%
CIAlign/similarityMatrix.py	26	0	3	100%
CIAlign/utilityFunctions.py	202	0	19	100%
tests/__init__.py	0	0	0	100%
tests/consensusSeqTest.py	92	0	11	100%
tests/cropSeqTest.py	27	0	6	100%
tests/helperFunctions.py	11	0	3	100%
tests/miniAlignmentsTest.py	66	0	12	100%
tests/palettesTest.py	15	0	3	100%
tests/parsingFunctionsTest.py	155	0	11	100%
tests/pwmTest.py	63	0	11	100%
tests/similarityMatrixTest.py	19	0	8	100%
tests/utilityFunctionsTest.py	172	0	13	100%
Total	**1607**	**0**	**158**	**100%**

Remove Insertions Improvements

For the previous version, the time taken to run was linear with respect to the maximum insertion size specified. Now, the time will only increase if insertions of this size are actually present in your alignment.
The new function is faster in the majority of cases, but not in every case, for some complex alignments it is fractionally slower, as shown below. However, the longest runtimes for the previous version are now substantially faster

Runtime vs alignment size for 2494 alignments generated using various data simulation and alignment tools (alignments available at https://github.com/KatyBrown/benchmarking_data_cialign and described in the original CIAlign publication doi.org/10.7717/peerj.12983) with a minimum insertion size of 1 and a maximum insertion size of 500.

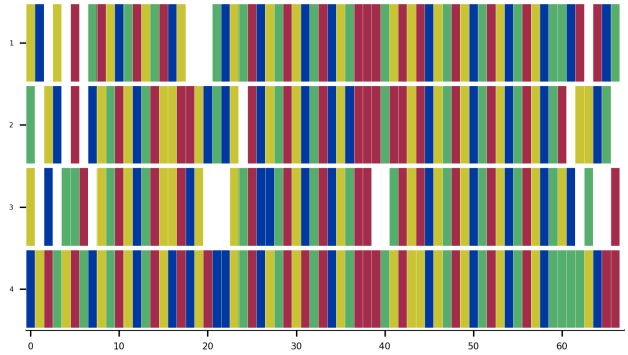


Please note: although the output of the new and original versions of remove insertions are very similar, there are some minor differences in the case of complex insertions, especially in cases where different sections of the insertion have different levels of coverage. In our simulations, this was the case for 907 of 779192 removed columns, or just over 0.1% of columns. In all 907 cases, the original remove insertions function removed columns which are not removed by the new function.

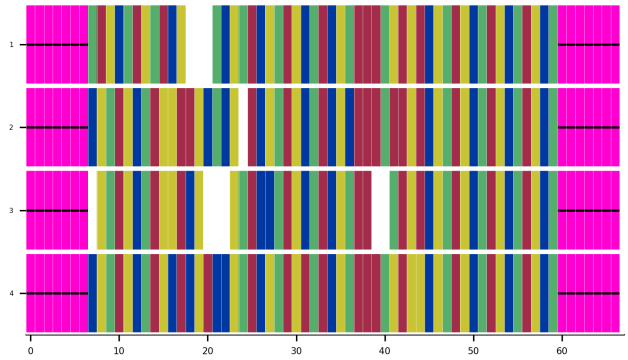
Crop Divergent Examples
Using a toy example

```
CIAlign--plot_input --infile alignment.fasta --crop_divergent
--crop_divergent_min_prop_ident 0.6 --crop_divergent_min_prop_nongap 0.5
--crop_divergent_buffer 5 --plot_output --plot_markup
```

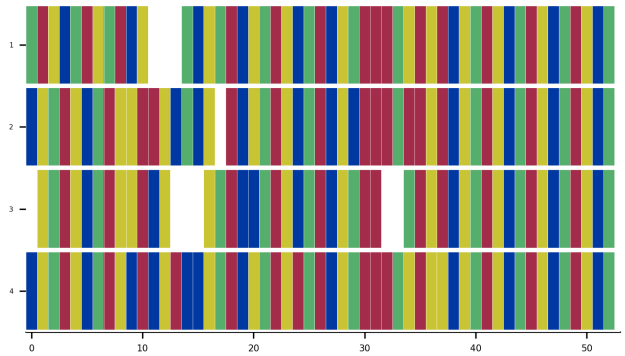
Input



Markup



Output



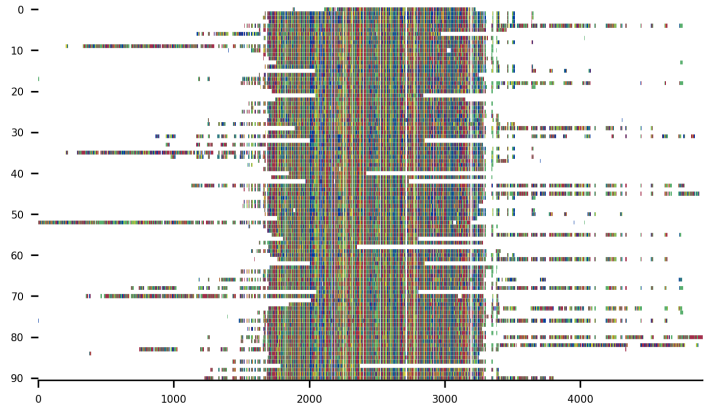
In this case, the cropped alignment starts when, for 5 consecutive columns (--crop_divergent_buffer 5), the percentage of identical (non gap) residues is greater than 0.6 (--crop_divergent_min_prop_ident 0.6) and the percentage of non-gap residues is greater than 0.5 (--crop_divergent_min_prop_nongap 0.5).

Using the CIAlign/example_files/example4.fasta

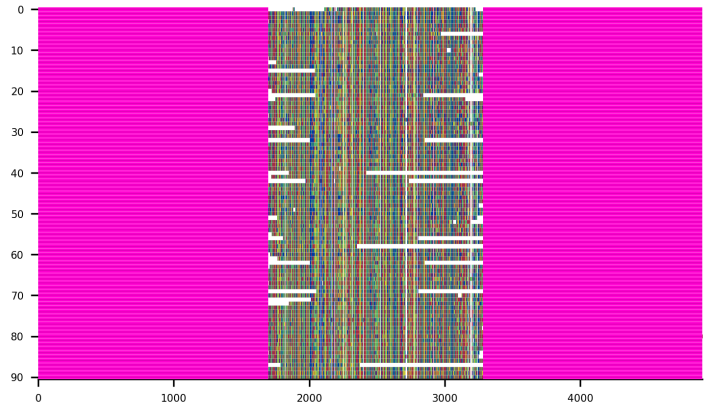
This file contains 91 putative COI gene sequences generated from raw transcriptomic datasets.

```
CIAlign --plot_input --infile ~/CIAlign/example_files/example4.fasta
--crop_divergent --crop_divergent_min_prop_ident 0.25
--crop_divergent_min_prop_nongap 0.75 --crop_divergent_buffer 15
--plot_output --plot_markup
```

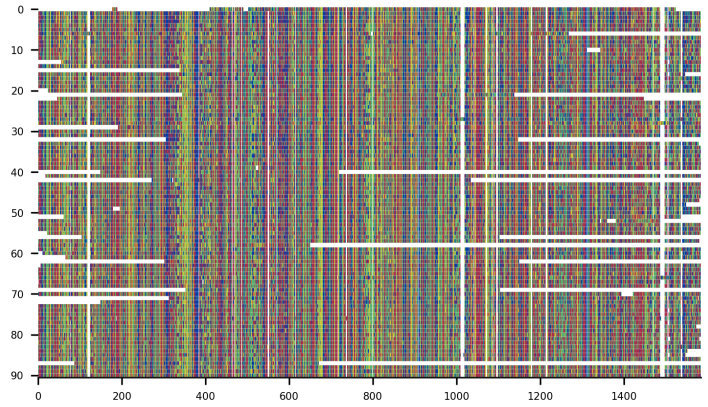
Input



Markup



Output



In this case, the cropped alignment starts when, for 15 consecutive columns (--crop_divergent_buffer 15), the percentage of identical residues is greater than 0.25 (--crop_divergent_min_prop_ident 0.25) and the percentage of non-gap residues is greater than 0.75 (--crop_divergent_min_prop_nongap 0.75).