Guidelines for using CIAlign

Here we want to share some tips that might help you using CIAlign. They concern how to choose the parameters and what to do if you input a very big alignment.

However, we want to stress that your analysis depends on your MSA (size and number of gaps), the purpose of your downstream analysis and the computer you are using. Please keep in mind that we cannot make universal statements here, as it depends on your individual case.

Tips for choosing parameters

- Which functions you want to use and the parameters you should choose depend on your downstream analysis and on your MSA
- Playing around with different parameters and seeing their effect on the MSA in the markup mini alignment can help to find the ones you need
- It is better to start with more conservative parameters to avoid over-cleaning
- It is recommended to run the make_similarity_matrix_input function where possible before running remove_divergent to avoid removing sequences which are not outliers compared to the general diversity of your alignment.
- remove_insertions will remove internal regions of the alignment which are gaps in the majority of sequences, this in particular is not appropriate for every downstream application of MSAs (for example if insertions or deletions are of interest).
- The default parameter for remove_short may be too large if your alignment is very short.

Tips for creating sequence logos for large MSAs

- If your MSA is has many sequences or is very long and you want to create a sequence logo, you might not be able to create the image due to the restrictions of the Python library we are using (matplotlib allows no more than 2ˆ16 pixels in each direction).
    - For “text” logos (--sequence_logo_type text), each additional column in the alignment adds a letter to the logo. By default there are 50 letters per row in the resulting image and a DPI of 300.
        - The total height of the image (while stored internally by matplotlib) is (2 * DPI * number of rows). With the default parameters there can be a maximum of 109 rows (65400 pixels), or 5450 columns in the alignment.
        - The total width of the image is (DPI * number of letters per row), so at 300 dpi there can be a maximum of 218 letters per row.
        - This gives a theoretical maximum alignment size for text sequence logos of 23,762 columns, at the default DPI and 213,858 columns at 100 DPI.
    - For “bar” logos (--sequence_logo_type bar), each additional column in the alignment adds a bar to the logo. By default there are 50 bars per row in the resulting image and a DPI of 300.
        - The total height is determined in the same way as for the text logos and is subject to the same limits.

        - The total width of the image is (DPI * number of letters per row * 0.2), so at 300 DPI there can be a maximum of 1092 letters per row.
        - This gives a theoretical maximum alignment size for bar sequence logos of 119,028 columns at 300 DPI and 1,071,252 columns at 100 DPI.

    - Please bear in mind that the memory available on your machine may be a limiting factor before the matplotlib maximum imagine size is reached

    - Therefore, generally, we recommend for very large MSAs to focus on our main functions: the cleaning and the mini alignments functions

    - The sizes of the actual image files generated using this function may differ somewhat from the size predicted based on these equations, as the matplotlib “tight_layout” and rendering functions perform certain optimisations.