4.1 (E) PSSM

During the exercise session we will take a detailed look at simple PSSMs and several possible improvements. Many of the slides contain examples and formulas needed for this exercise. Make sure to carefully read them before trying to solve the homework (as well as the comments in the code template).

4.2 (H) From MSA to PSSM (15)

Your task is to complete the MSA class in the pssm.py file. The __init__ method should accept only a single parameter: a list of sequences representing a multiple sequence alignment (MSA); its contents should be stored within a class variable for later use. Further, it is advised to pre-calculate and store most of the statistics needed for the other methods. Unless explicitly noted otherwise, the following methods do not take any parameters. Please read the comments in the code template carefully, as they provide additional and more detailed information about the input and output specifications of each method. Further, to keep the exercise sheet clean and short, we did not include any formulas in the task descriptions. Please reference the in-class slides.

(a) After initialization, the MSA class should immediately check if the list contained a valid MSA. If not, raise a TypeError. Consider the following requirements:

- The MSA contains at least one sequence
- All sequences have the same length (including gaps)
- All sequences contain only valid amino acids and gap characters

(b) Complete the get_size() method to return a tuple containing the number of sequences in the MSA and the MSA length (number of columns).

(c) Complete the get_primary_sequence() method to return the ungapped (i.e. without gaps) primary sequence of the MSA. In this exercise, the primary sequence is always the first sequence in the MSA.
(d) Not all sequences in a MSA carry the same amount of information: calculate the sequence weights. Complete the `get_sequence_weights()` method to return the weights for all sequences.

(e) To estimate the amount of information the whole MSA carries, we need to estimate the number of independent observations $N$. Complete the `get_number_of_observations()` method to return the estimated number of observations.

(f) Complete the `get_pssm()` method to return a PSSM for the MSA. The method takes several parameters determining which refinements are applied (real background frequencies, redistributed gaps, sequence weights, pseudocounts). Mind the order of those refinement steps! You will have to calculate the vector of background frequencies from the matrix of amino acid pair frequencies (cf. in-class slides). In the final PSSM, each row represents a position in the primary sequence (without gaps) and each column one of the 20 amino acids (use the provided indices in the code template). Substitute values that would be $-\infty$ in the final PSSM with -20.