**Biological Sequence Alignment using Dynamic Programming**

**Kareith Dyce**+1**, James Ann**+1**, and Xavier Verdad** +1

1Auburn University, COMP, Auburn, 36849, USA \* corresponding.author@email.example

+these authors contributed equally to this work

**ABSTRACT**

In Biology the similarity between two sequences tells quite a bit about the two. The more similar their sequences are the closer they are related. Knowing this it can be easier to fight one virus if it is closely related to another one with a known cure. Using normal comparisons however to find the similarities would take forever as most of these sequences are over 100 characters in length. This is where Dynamic Programming comes in an vastly reduces the time spent calculating the alignment and quickly allows for finding the similarities between the sequences.

**1. Introduction**

Just like many things protein sequences can also undergo evolution. This means that there are many different protein sequences that came from a common original sequence. This similar means that if we know how to combat / treat one virus if we find that another one is closely related, we could combat it through similar means.

However, finding is not especially easy as these sequences evolve in three different ways, Substitution, Deletion and Insertion. As the names would suggest Insertion is adding to the sequence, Deletion is removing form the sequence and Substitution is changing letters in the sequence. Each of these evolutions can change more than just one letter which can make the alignment process harder. Also both Insertion and Deletion would change a sequences length so, just because two sequences have the same length does not mean that there are closely related. Also, the opposite is true so that just because the sequences vary in length does not make it impossible for them to be closely related. These small evolutions over time has lead to huge diversions from the common ancestral sequence.

Now being able to check every alignment possible between the two sequences to find the best global alignment which how the two full sequences compare to each other would take forever with normal programming practices to just to find a alignment with a length of 100. These sequences could get very long so this is where we would utilize dynamic programming.

**2. Methods**

Up to two levels of **subheading**s are permitted. Subheadings should not be numbered.

**2.1. Subsection**

Example text under a subsection. Bulleted lists may be used where appropriate, e.g.

• First item

• Second item

***2.1.1. Second-level section***

Here goes the text.

**3. Results**

Topical subheadings are allowed. Authors must ensure that their Methods section includes adequate experimental and characterization data necessary for others to reproduce their work.

**Table 1.** Legend (350 words max). Example legend text.

|  |  |  |
| --- | --- | --- |
| Condition | n | p |
| A | 5 | 0.1 |
| B | 10 | 0.01 |



**Figure 1.** Legend (350 words max). Example legend text.

**4. Discussion**

The Discussion should be succinct and must not contain subheadings.

**5. References**

1. Figueredo, A. J. & Wolf, P. S. A. Assortative pairing and life history strategy - a cross-cultural study. *Hum. Nat.* 20, 317–330 (2009).

Note: Use software such as EndNote for managing references.