Part 2.1 **Student:** kgxj22

Bioinformatics Report

I have defined a scoring function consisting of *three parts* for alignments over the alphabet $\Sigma = \{A, B, C\}$

PART 1 - Aligning a B with any letter, other than itself, is costly

Figure 1
***Single letter** substitution costs*

	A	**B**	**C**	**-**
A	1	-3	2	-1
B	-3	**5**	-3	-1
C	2	-3	2	**-1**
-	-1	-1	-1	0

Description
As shown in figure 1, this particular scoring function punishes matches of any letter with a B, except for B itself, which is greatly incentivised *(+5)*. Consequently, I expect an optimal alignment of two sequences to contain matches between B's if such a match is possible *(i.e. the aligned sequences each contain at least one B)*. Secondly, I expect there to be few *other* occurrences of B due to the fact that it is cheaper to insert or delete B than to substitute it with any letter other than itself. It is more rewarding to align C *(+2)* with itself, than it is to align A with itself *(+1)*.

PART 2 - The cost of single letter substitutions differs from that of multi-letter substitutions

Figure 2
***Multi-letter** substitution costs*

	$[AC]^n$
B^n	-3

Description
Figure 2 displays the cost of multi-letter substitutions. The rule described in figure 2 is a special case, whereby there is an upfront cost to aligning A or C with a B. Following that, any consecutive Bs can be freely aligned with an A or a C. This multi-letter scoring approach increases the likelihood that long sequences of Bs will appear in an alignment.

NB. One biological application of this is that we can reward the presence of meaningful subsequences (i.e. a particular protein), so that it is more likely to appear in the alignment and we can easily detect its presence.

PART 3 - Alignments containing palindromic subsequences are awarded a bonus

Figure 3

```
_ B C C B C C A _ B
  | | |       | |
A C C B _ _ C B C _
```

Each alignment is awarded **n additional points** - where n is equal to the length of the longest contiguous (i.e. without an indel) palindromic subsequence. This subsequence can appear in *either* sequence of the alignment. For example, figure 3 shows the alignment between *BCCBCCAB* and *ACCBCBC*. The longest palindromic subsequence in the alignment of the first sequence (_ BCCBCCA_B) is CCBCC (length 5). Similarly, in the alignment of the second sequence (ACB_ _CBC_), the longest palindromic subsequence is CBC (length 3). Hence, the longest palindromic subsequence is of length 5, and a bonus of 5 is added to the total alignment score

NB. Every alignment containing *at least one letter* is granted a *minimum additional score of 1*.

EXAMPLES

Alignment	**Individual Scores**	**Total Score**	**Explanation**
A B B _ C | | | A C B B _	A → A = 1 B → C = -3 B → B = 5 _ → B = -1 C → _ = -1	1+ 2 = **3** (bonus for the palindrome BB)	Perform *single-letter substitutions* for each letter as we have no instances where sufficiently long sequences of Bs are substituted for an equally long sequence of As and Cs.
C A A C | | | B B B _	CAA → BBB = -3 C → _ = -1	-4 + 4 = **0** (bonus for the palindrome CAAC)	Perform a *multi-letter substitution* of CAA for BBB, for a total cost of -3. This is cheaper than substituting each character independently (-3 each). The longest palindrome in either sequence of the alignment is CAAC (length 4), therefore n additional 4 points are added to the alignment score.