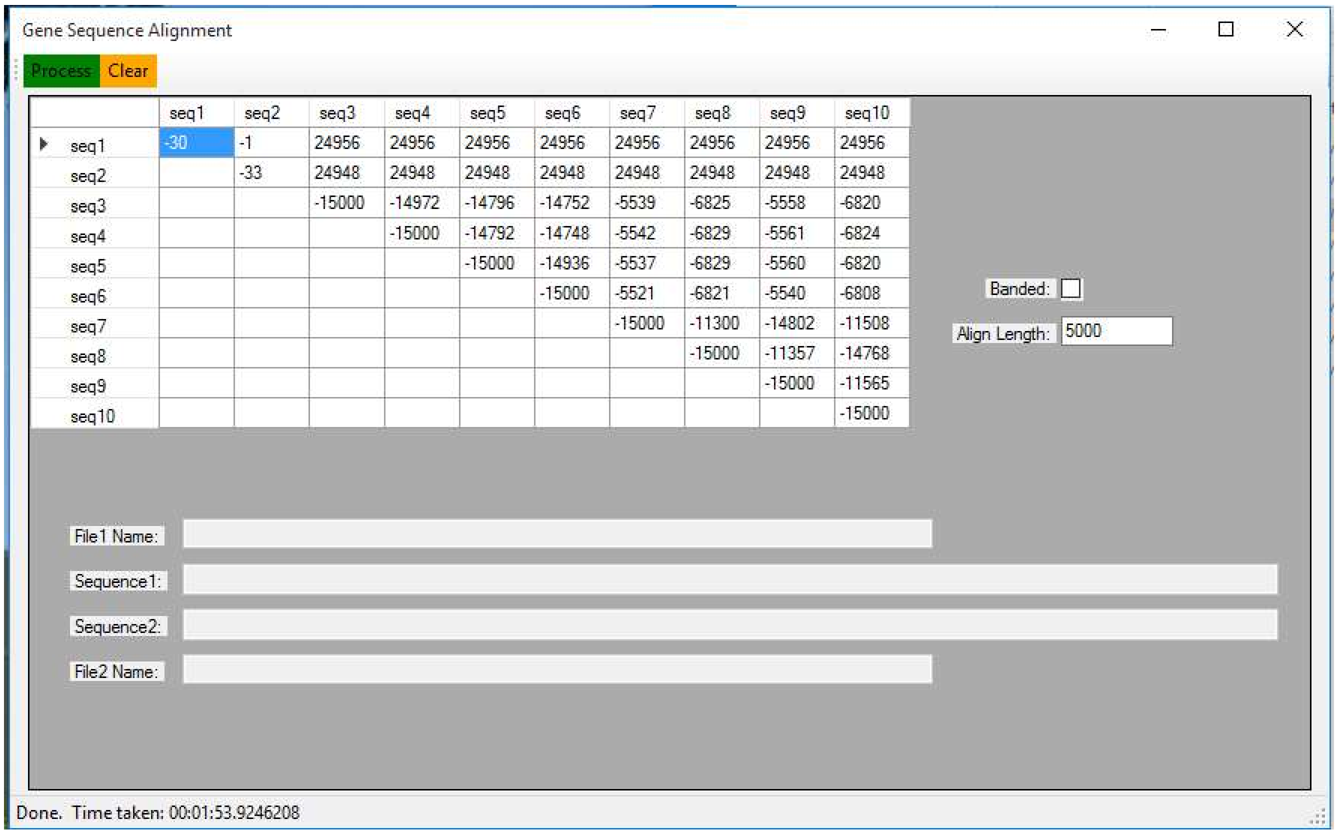
Taylor Cowley
CS 312 Lab 4: Gene Sequencing
November 03, 2016

1. Explain the time and space complexity of your algorithm by showing and summing up the complexity of each subsection of your code.
    a. [10 points] Your analysis should show that your unrestricted algorithm is at most *O*(*nm*) time and space.
    When doing the table calculation, we march through from 0 to A's length (n); each time going calculations from 0 to B's length (m). Because of this double for-loop action, the unrestricted algorithm is O(nm) in time. Because we store every result in a table of dimension n*m, the space complexity is also O(n*m).
    b. [10 points] Your analysis should show that your banded algorithm is at most *O*(*n*+*m*) time and *O*(*nm*) space.
    When doing the table calculation for the banded algorithm, we still go through 0 to A's length, but doing only 7 calculations with B each time. But because the difference in length from A and B can only be (at the maximum) 4, we get O(n+m) time complexity. We still, however, initialize a table that is n*m, hence taking up O(n*m) space complexity.
2. [10 points] Write a paragraph that explains how your alignment extraction algorithm works, including the backtrace.
    a. **We do the standard Needleman-Wunsch algorithm, with a cost of 5 for any “gap,” or insert/delete in the sequence, a cost of 1 for any substitutions, and a “cost” (really, a benefit) of -3 for a matching pattern. To compute the total alignment cost, dynamic programming is used to construct a table showing all the costs of aligning the two strings in every possible way. Any given character of the combined sequence is produced by either taking the character from both sequences- substituting one for the other or finding a match, or by taking a character from one sequence and forcing a gap in the other. This corresponds to coming from the upper-left diagonal or from the top or left side, respectively. Every node in the table stores the minimum cost of arriving at this alignment, as well as a pointer to its parent node- the one from which it achieves the current minimum cost. At the end, we take the last node in the bottom right corner, and follow its lineage- either up, left, or diagonally, until we reach the ultimate parent node at the top left corner. This corresponds with traversing our aligned strings from the end to the beginning.**
3. [20 points] Include a “results” section showing both a screen-shot of your 10x10 score matrix for the unrestricted algorithm with align length *k* = 5000 and a screen-shot of your 10x10 score matrix for the banded algorithm with align length *k* = 15000.
    Results:

a. [screenshot of unrestricted k=5000]



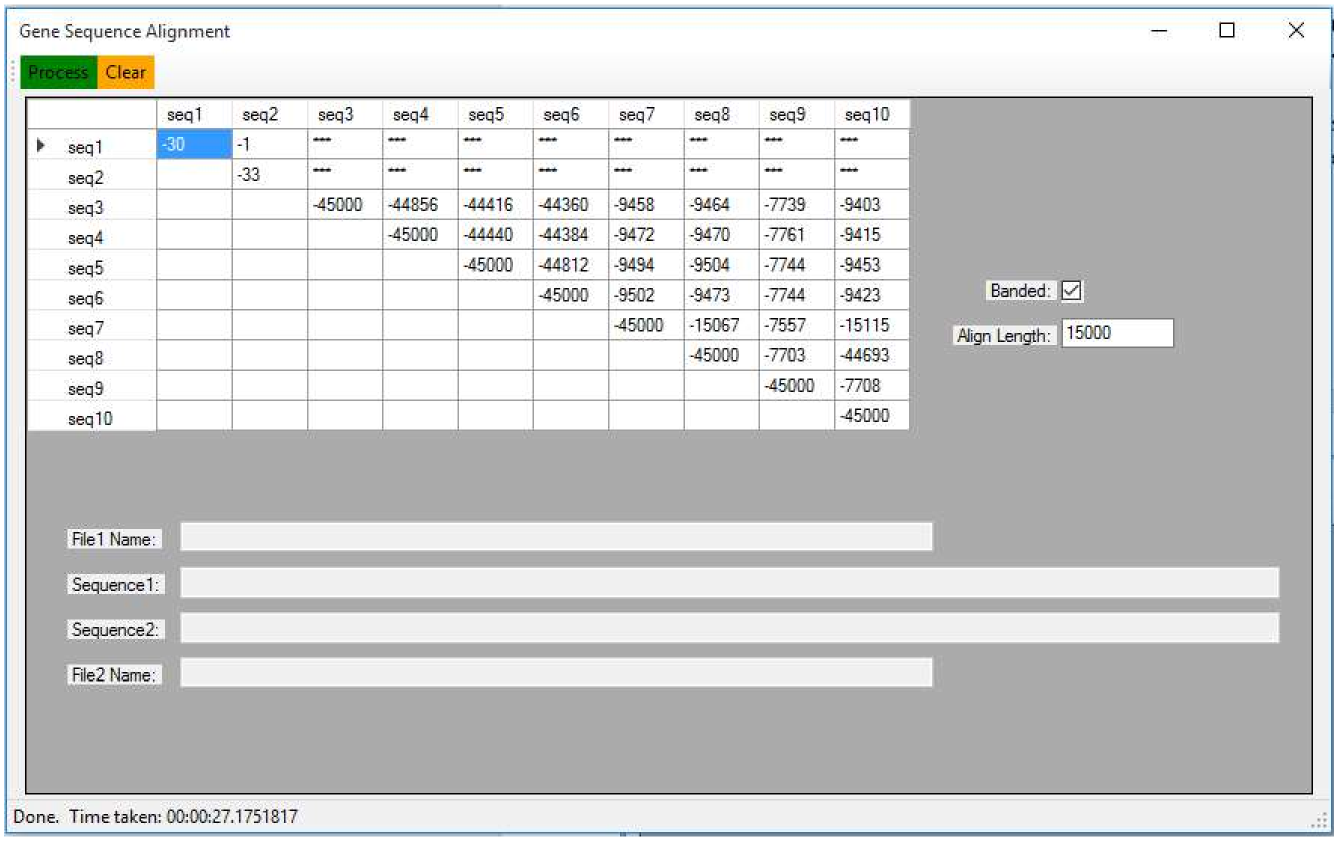
	seq1	seq2	seq3	seq4	seq5	seq6	seq7	seq8	seq9	seq10
seq1	-30	-1	24956	24956	24956	24956	24956	24956	24956	24956
seq2		-33	24948	24948	24948	24948	24948	24948	24948	24948
seq3			-15000	-14972	-14796	-14752	-5539	-6825	-5558	-6820
seq4				-15000	-14792	-14748	-5542	-6829	-5561	-6824
seq5					-15000	-14936	-5537	-6829	-5560	-6820
seq6						-15000	-5521	-6821	-5540	-6808
seq7							-15000	-11300	-14802	-11508
seq8								-15000	-11357	-14768
seq9									-15000	-11565
seq10										-15000

Banded: ☐

Align Length: 5000

Done. Time taken: 00:01:53.9246208

b. [screenshot of banded k=15000]



	seq1	seq2	seq3	seq4	seq5	seq6	seq7	seq8	seq9	seq10
seq1	-30	-1	***	***	***	***	***	***	***	***
seq2		-33	***	***	***	***	***	***	***	***
seq3			-45000	-44856	-44416	-44360	-9458	-9464	-7739	-9403
seq4				-45000	-44440	-44384	-9472	-9470	-7761	-9415
seq5					-45000	-44812	-9494	-9504	-7744	-9453
seq6						-45000	-9502	-9473	-7744	-9423
seq7							-45000	-15067	-7557	-15115
seq8								-45000	-7703	-44693
seq9									-45000	-7708
seq10										-45000

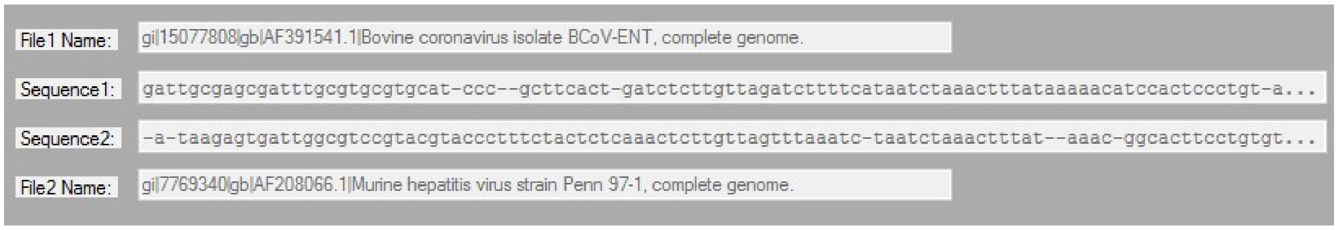
Banded: ☑

Align Length: 15000

Done. Time taken: 00:00:27.1751817

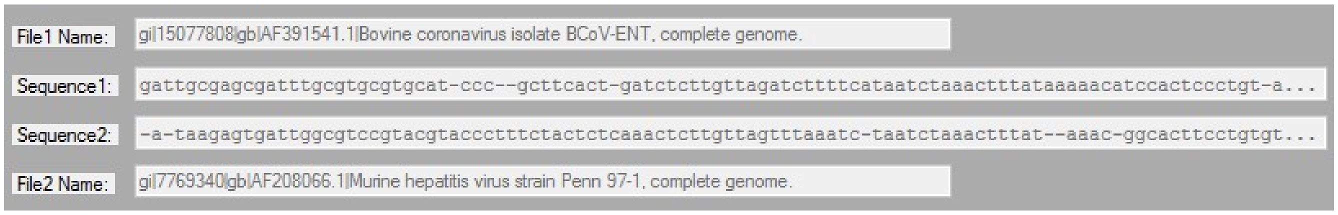
4. [10 points] Include in the “results” section the extracted alignment for the first 100 characters of sequences #3 and #10 (counting from 1), computed using the unrestricted algorithm with *k* = 5000. Display the sequences in a side-by-side fashion in such a way that matches, substitutions, and insertions/deletions are clearly discernible as shown above in the To Do section. Also include the extracted alignment for the same pair of sequences when computed using the banded algorithm and *k* = 15000.

Unrestricted algorithm #3 and #10, first 100 characters of aligned sequences



```
File1 Name:  gi|15077808|gb|AF391541.1|Bovine coronavirus isolate BCoV-ENT, complete genome.
Sequence1:   gattgcgagcgatttgcgtgcgtgcat-ccc--gcttcact-gatctcttgttagatcttttcataatctaaactttataaaaacatccactccctgt-a...
Sequence2:   -a-taagagtgattggcgtccgtacgtaccctttctactctcaaactcttgttagtttaaatc-taatctaaactttat--aaac-ggcacttcctgtgt...
File2 Name:  gi|7769340|gb|AF208066.1|Murine hepatitis virus strain Penn 97-1, complete genome.
```

Banded algorithm #3 and #10, first 100 characters of aligned sequences



```
File1 Name:  gi|15077808|gb|AF391541.1|Bovine coronavirus isolate BCoV-ENT, complete genome.
Sequence1:   gattgcgagcgatttgcgtgcgtgcat-ccc--gcttcact-gatctcttgttagatcttttcataatctaaactttataaaaacatccactccctgt-a...
Sequence2:   -a-taagagtgattggcgtccgtacgtaccctttctactctcaaactcttgttagtttaaatc-taatctaaactttat--aaac-ggcacttcctgtgt...
File2 Name:  gi|7769340|gb|AF208066.1|Murine hepatitis virus strain Penn 97-1, complete genome.
```

5. [30 points] Attach your commented source code for both your unrestricted and banded algorithms.

See next several pages