

Lab 4 Write-up

1. **Explain the time and space complexity of your algorithm by showing and summing up the complexity of each subsection of your code (this can be included in the comments of the code).**
- a. **[10 points] Your analysis should show that your unrestricted algorithm is at most $O(nm)$ time and space.**

align_all()

Runs for each sequence

Unrestricted Algorithm

For loop for each character in sequence 1 (n)

For loop for each character in sequence 2 (m)

Time: $O(nm)$

Space: Each iteration adds one value to the grid so $O(nm)$

- b. **[10 points] Your analysis should show that your banded algorithm is at most $O(n+m)$ time and $O(nm)$ space.**

align_all()

Runs for each sequence

Banded Algorithm

For each character in sequence 1 that is 3 away

For each character in sequence 2 that is 3 away

The bandwidth for each of these is 7

There are a fixed amount of computations for each for loop

That is not dependent on n or m

Each loop runs n and m times

Time: $O(n+m)$

Space: Each iteration may not make comparisons but each cell in the grid will need a value so the space requirement is $O(nm)$

2. **[10 points] Write a paragraph that explains how your alignment extraction algorithm works, including the backtrace**

First, I construct a grid. The grid is composed of a 2 dimensional array. Each cell is a dictionary that holds the cost at that point and the backtrace. Second, I fill the table with values according to the Needleman-Wunsch algorithm. After the lowest alignment cost is identified, I take my grid and I start at the bottom right corner. I look at the backtrace and if the backtrace is a diagonal then I write a character from sequence 1 and sequence 2 to each of their respective buffers. I increment both i and j . If the backtrace points to the top or the left then I write a hyphen

3. [20 points] Include a “results” section showing both a screen-shot of your 10x10 score matrix for the unrestricted algorithm with align length $k = 1000$ and a screen-shot of your 10x10 score matrix for the banded algorithm with align length $k = 3000$.

Gene Sequence Alignment

	seq1	seq2	seq3	seq4	seq5	seq6	seq7	seq8	seq9	seq10
seq1	-30	-1	4956	4956	4956	4956	4956	4956	4956	4956
seq2		-33	4948	4948	4948	4948	4948	4948	4948	4948
seq3			-3000	-2996	-2956	-2944	-1431	-1448	-1399	-1448
seq4				-3000	-2960	-2948	-1431	-1448	-1399	-1448
seq5					-3000	-2988	-1423	-1452	-1391	-1448
seq6						-3000	-1426	-1452	-1394	-1448
seq7							-3000	-2771	-2814	-2767
seq8								-3000	-2731	-2996
seq9									-3000	-2727
seq10										-3000

Label I:

Sequence I:

Sequence J:

Label J:

☐ Banded Align Length:

Done. Time taken: 1 mins and 10.261 seconds.

Gene Sequence Alignment

	seq1	seq2	seq3	seq4	seq5	seq6	seq7	seq8	seq9	seq10
seq1	-30	-1	None	None	None	None	None	None	None	None
seq2		-33	None	None	None	None	None	None	None	None
seq3			-9000	-8984	-8888	-8848	-2735	-2743	-1429	-2735
seq4				-9000	-8888	-8848	-2739	-2748	-1426	-2740
seq5					-9000	-8960	-2711	-2739	-1426	-2727
seq6						-9000	-2708	-2728	-1415	-2716
seq7							-9000	-8103	-1256	-8099
seq8								-9000	-1310	-8980
seq9									-9000	-1315
seq10										-9000

Label I:

Sequence I:

Sequence J:

Label J:

☒ Banded Align Length:

Done. Time taken: 2 mins and 13.738 seconds.

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 $k = 1000$. 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 $k = 3000$.

sequences #3 and #10 $k = 1000$ unrestricted

```
gattg-gagcgaatttgcggtgcgtgcatcccgcttc-actg--at-ctcttgtagatcttttc-taatctaaactttataaa--catc-actccctgta-  
-ataa-gagtgaattggcggtccgtacgtaccctttc-actc--aa-ctcttgtagtttaaatac-taatctaaactttataaa--cggc-acttcctgtg-
```

sequences #3 and #10 $k = 3000$ banded

```
gattg-gagcgaatttgcggtgcgtgcatcccgcttc-actg--at-ctcttgtagatcttttc-taatctaaactttataaa--catc-actccctgta-  
-ataa-gagtgaattggcggtccgtacgtaccctttc-actc--aa-ctcttgtagtttaaatac-taatctaaactttataaa--cggc-acttcctgtg-
```

5. [30 points] Included your documented source code for both your unrestricted and banded algorithms.

See github