

COMP 5970/6970 Project 1: 15 points 15% Credit

Submission due before 11:59 PM Monday February 18

Instructions:

1. This is a group project. You should do your own work while working collaboratively as a group. Any evidence of copying either from a public source or from the works of other groups without due credits will result in a zero grade and additional penalties/actions against all members of the involved groups.
2. **No show in project presentation or final submissions by email or late submissions (even by minutes) will receive a zero grade for the entire group.** No makeup will be offered unless prior permission has been granted, or there is a valid and verifiable excuse.

Submission:

For 5970, one member from each group will upload the following to canvas before 11:59 PM Monday February 18:

1. **Source Code (Member 1):** Python source files (upload .zip file in case of multiple files) containing your code only (no test data needed) and ReadMe.txt file (template provided) describing how to run your code. Note that we will NOT debug your code. If your code does not execute as described in ReadMe.txt, you will receive a zero grade.
2. **Presentation Slide (Member 2):** One slide only in PPT/PPTX/PDF format to be used during the oral presentations (see below). If you submitted file span more than a page, we will extract the first page for the oral presentation.
3. **Project Report (Member 3):** Completed report document in PDF format using template provided. Make sure to have all necessary sections of scientific writing: abstract, introduction, methods, results, discussion, references.

For 6970, one member from each group will upload the following to canvas before 11:59 PM Monday February 18:

1. **Source Code and Project Report (Member 1):** (i) Python source files (upload .zip file in case of multiple files) containing your code only (no test data needed) and ReadMe.txt file (template provided) describing how to run your code. Note that we will NOT debug your code. If your code does not execute after following your instructions laid out in ReadMe.txt, you will receive a zero grade. (ii) Completed report document in PDF format using template provided. Make sure to have all necessary sections of scientific writing: abstract, introduction, methods, results, discussion, references.
2. **Presentation Slide and Video Demo (Member 2):** (i) One slide only in PPT/PPTX/PDF format to be used during the oral presentations (see below). If you submitted file span more than a page, we will extract the first page for the oral presentation. (ii) A video demonstration not more than 5 minutes in duration containing a creative demonstration of the working dynamics of your program and the results achieved. Creative ways of visualization and use of graphic tools are encouraged. Please use widely recognized formats for videos.

Presentations:

Presentation will be during the class on **Wednesday February 20** and **Friday February 22**.

For 5970, the member submitting presentation slide will deliver 5 minutes flash presentation accompanied by the submitted slide:

1. At the least, your presentation should contain methods (i.e. implementation), results (e.g. output), and conclusion.
2. Practice your talk not to exceed the time limit or finish too early.
3. No need to bring your slides. We will set things up and decide the presentation sequence.

For 6970, the member submitting presentation slide and video demo will deliver 5 minutes flash presentation accompanied by the submitted slide followed by additional 5 minutes of demo accompanied by the submitted video:

1. At the least, your presentation should contain methods (i.e. implementation), results (e.g. output), and conclusion.
2. Practice your talk not to exceed the time limit or finish too early.
3. The video demo may be accompanied by oral presentation.
4. No need to bring your slides/demo. We will set things up and decide the presentation sequence.

Biological Sequence Alignment using Dynamic Programming

Implement Needleman-Wunsch and Smith-Waterman dynamic programming algorithms for global and local sequence alignment respectively for protein sequences.

Note: You must use standard Python programming language. You are NOT allowed to use non-standard packages or libraries (e.g. Biopython, scikit-learn, SciPy, NumPy, etc.).

A: Input format:

Protein sequences are provided in the FASTA format. In the FASTA format, the first row is the tag of the sequence with a leading character '>'. The following rows are the actual sequence. For example, it may look like:

```
>test protein X
EEEE
KKKK
AAAA
FFF
```

It represents a test protein sequence "EEEEKKKKAAAAFFF". Notice the length of each row is variable. Your program should accept two FASTA files for the alignment. Please use the BLOSUM62 (<http://www.ncbi.nlm.nih.gov/Class/FieldGuide/BLOSUM62.txt>) scoring matrix as the objective function.

B: Output format:

The output should be the resulting alignment score and an alignment between the two input sequences. For amino acids (characters) whose matches have positive scores in BLOSUM62, use '|' to indicate a positive match; otherwise use '*'. For alignment of amino acid and gap, no symbol should be placed. For example:

```
Score: 12345
EEEEKKKK
||||
EEEE-----

AAAAAFFF
*****|||
BBBBBFFF
```

Each row should contain 80 alignment columns such that the print out does not get messed up.

C: Test case:

Three sets of test protein sequences containing various deadly viruses are supplied:

Pair 1: Dengue virus and Zika virus

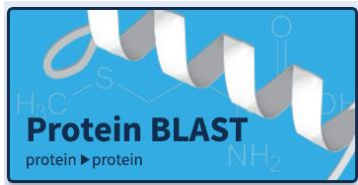
Pair 2: Human papillomavirus (HPV) and Human immunodeficiency virus (HIV)

Pair 3: Poliovirus and Rhinovirus

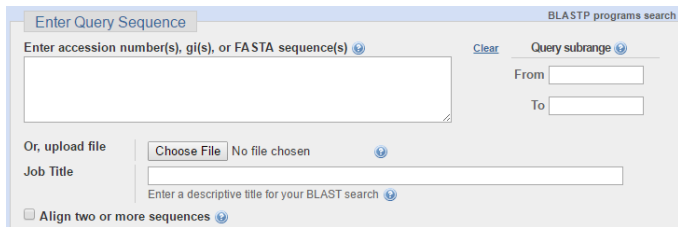
For each pair, perform global and local alignments using Needleman-Wunsch and Smith-Waterman dynamic programming respectively and compare the resulting alignments with the alignment generated by Basic Local Alignment Search Tool (BLAST) (see below).

To use BLAST, go to the online server at <http://blast.ncbi.nlm.nih.gov/Blast.cgi>.

Click on “Protein BLAST”



Check the box that says “Align two or more sequences” (at the bottom of the figure).



Enter Query Sequence

Enter accession number(s), gi(s), or FASTA sequence(s) [?](#) [Clear](#)

Query subrange [?](#)

From

To

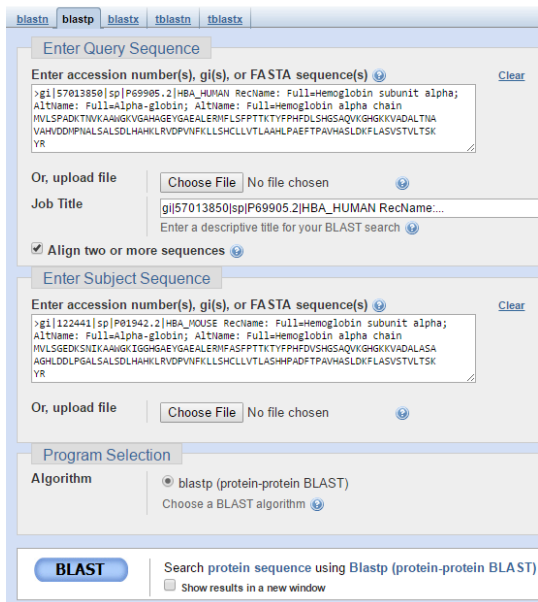
Or, upload file No file chosen [?](#)

Job Title

Enter a descriptive title for your BLAST search [?](#)

☐ Align two or more sequences [?](#)

Copy the sequences into the boxes.



blastn **blastp** blastx tblastn tblastx

Enter Query Sequence

Enter accession number(s), gi(s), or FASTA sequence(s) [?](#) [Clear](#)

Or, upload file No file chosen [?](#)

Job Title

Enter a descriptive title for your BLAST search [?](#)

☒ Align two or more sequences [?](#)

Enter Subject Sequence

Enter accession number(s), gi(s), or FASTA sequence(s) [?](#) [Clear](#)

Or, upload file No file chosen [?](#)

Program Selection

Algorithm ☒ blastp (protein-protein BLAST) [?](#)

☐ Choose a BLAST algorithm [?](#)

BLAST Search protein sequence using Blastp (protein-protein BLAST)

☐ Show results in a new window

Scroll down and view the alignment generated by BLAST.

gi|122441|sp|P01942.2|HBA_MOUSE RecName: Full=Hemoglobin subunit alpha; AltName: F
Sequence ID: Query_160837 Length: 142 Number of Matches: 1

Range 1: 1 to 142 [Graphics](#) [Next Match](#) [Previous Match](#)

Score	Expect	Method	Identities	Positives	Gaps
253 bits(645)	4e-93	Compositional matrix adjust.	122/142(86%)	131/142(92%)	0/142(0%)
Query 1	MVLSPADKTNVKAANGKVGAGHAGEYGAELRMFLSFPTTKTYFPHFDLSHGSAQVKGHG	60			
Sbjct 1	MVLSGDEKSNIKAAWGKIGGHGAEYGAELRMFLSFPTTKTYFPHFDVSHGSAQVKGHG	60			
Query 61	KKVADALTNAAVHVDDMPNALSALSDLHAHKLRLVDPVNFKLLSHCLLVTLAAHLPAEFTP	120			
Sbjct 61	KKVADAL +A H+DD+P ALSALSDLHAHKLRLVDPVNFKLLSHCLLVTLA+H PA+FTP	120			
Query 121	AVHASLQKFLASVSTVLTSKYR	142			
Sbjct 121	AVHASLQKFLASVSTVLTSKYR	142			

D: Analysis:

For each of the three sets of test protein sequences, compare alignments generated by your implementations of local and global alignments with that generated by BLAST to identify and discuss similarities and differences.