Assignment 2: Running (PSI-)BLAST

The goal for this practical is to generate **(PSI-)BLAST E-values** for a set of **209 proteins**.

All coding will be done in skeleton scripts. These are supportive scripts that will provide some code that you have to finish by completing the following blocks:

```py
########################
# START CODING HERE #
########################
```

To run this practical, you can use the jupyter hub environment and the accompanying notebook file. For instructions, see the Canvas page on jupyter hub. Make sure to download the files you need for this assignment from Canvas as well.
Please create the following folder structure, where we will reference to the folder names (**three separate directories!**):

```bash
$ mkdir queries data results
```

You can find all required files and skeleton scripts on the Canvas Assignment page. Save the SCOP_selection.txt file into "*./data/*" and save the skeleton files into the current working directory (`pwd`), the directory from which you can see '*queries*', '*data*' and '*results*' as available folders.

Question 1.1 [optional]

Both BLAST and PSI-BLAST need a substitution matrix. Why do we need a substitution matrix for (PSI-)BLAST?

Task 1.2: Fetch Data [points = 25]

Finish the code blocks inside of '*fetch_fasta_skeleton.py*'. Please read the supporting comments carefully.

For more information: http://www.UniProt.org/help/api

After you finished, run the script (either local, using Colab or on the VU Servers) using:

```bash
$ python3 fetch_sequences_skeleton.py \
        -i ./data/SCOP_selections.txt \
        -db ./data/yourDBname.fasta
```

Now that you have created your database file, the next step is to **format** the database file using the **makeblastdb** software:

```bash
$ makeblastdb \
        -in ./data/yourDBname.fasta \
        -parse_seqids \
        -dbtype prot
```

The next step is to perform (PSI-)BLAST using either *psiblast* or *blastp*. For more information on the application's options, you can use:

```bash
$ blastp -help
$ psiblast -help
```

In the following steps we are going to use protein **P00698** as a test case. Try to perform a default BLAST search for our P.O.I. Run (PSI-)BLAST with the following commands:

```bash
$ blastp -query ./queries/P00698.fasta \
        -db ./data/yourDBname.fasta

$ psiblast -query ./queries/P00698.fasta \
        -db ./data/yourDBname.fasta \
        -num_iterations ???
```

Note that in order to run PSI-BLAST, the `*-num_iterations*` parameter must be set to something greater than 1. The default of `*-num_iterations 1*` means that there are no iterations and that it's therefore the same as a normal BLAST search.

Question 2.1 [words = 50, points = 10]

Is it possible that PSI-BLAST stops before it reaches the number of iterations as determined by the `-num_iterations` parameter. If so, why?

Task 2.2: Run (PSI-)BLAST [points = 25]

Finish the code blocks inside of the '*run_local_skeleton.py*'. Please read the supporting comments carefully. You can run the script using:

```bash
$ python3 run_local_skeleton.py \
        -i data/SCOP_selections.txt \
        -db data/yourDBname.fasta \
        -o results/blastEvalues.tsv
```

You should run the script twice, one for BLAST and one for PSI-BLAST. The files will contain the UniProt identifiers of all protein pairs and the corresponding E-values separated by tabs.

An example of the file content ('*NA*' means Not Available):

```tsv
P000001    P000002    0.0001
P000001    P000004    0.0000089
P000002    P000004    0.004
P000001    P000003    0.002
P000002    P000003    NA
```

Question 3.1 [words = 100, points = 20]

Create histograms with the distribution of E-values from BLAST and PSI-BLAST using the run_local_skeleton script. Also give the number of hits with an E-value lower than 0.002 for both BLAST and PSI-BLAST (see the plotting function inside the skeleton script, add the numbers as an answer to this question).

example script call (plotting):

```bash
$ python3 run_local_skeleton.py \
        -r ./results/blastEvalues.tsv \
        -o ./results/BLAST_histogram.png
```

Do you observe any difference between the output of BLAST and PSI-BLAST and what could cause this?

Question 3.2 [words = 50, points = 10]

Take a pair of proteins; **P00701** and **P00698**. Blast protein **P00701** on the SwissProt database (you can download this from the assignment page).

Compare the results with your output from BLAST on your own database.

Can you find protein **P00698** in the hit list of both BLAST searches? If so, are the alignments and E-values exactly the same? Explain how these findings can be rationalized.

Task: Choose One Topic [words = be concise, points = 10]

This is a mini-project. You can choose either topic **A)** or **B)**. Only do this task if you have finished all other questions. This is a bonus question that will allow you to go from a 9 to a 10.

A) The aim is to create a more sensitive version of PSI-BLAST. You can do this for example by using larger (protein sequence) databases. Describe your approach. Implement this in your scripts and compare the number of significant hits from your previous results with the new results. What conclusions can you make?

B) The aim is to investigate if the BLAST and PSI-BLAST scores are symmetric: if your query protein is A and your hit is B, does that give exactly the same e-values and bit-scores, as when your query is B and your hit is A? First, explain what you would rationally expect for BLAST and PSI-BLAST. Then Investigate in your output if this is indeed the case. Lastly, you should try to give an explanation for your findings; you will probably need to look in the paper (Links to an external site.) on "*composition-based statistics*" from *Altschul et al* in 2001, for part of the BLAST method that causes this effect.

Submitting

Please upload all your files (histogram for PSI-BLAST and BLAST, .tsv files and your answers to the questions) as specified on Canvas.

You're done!