THE UNIVERSITY *of* EDINBURGH
School of Biological Sciences

[ICA2]
[Bioinformatics Programming & System Management]

Student Exam Number: B221734

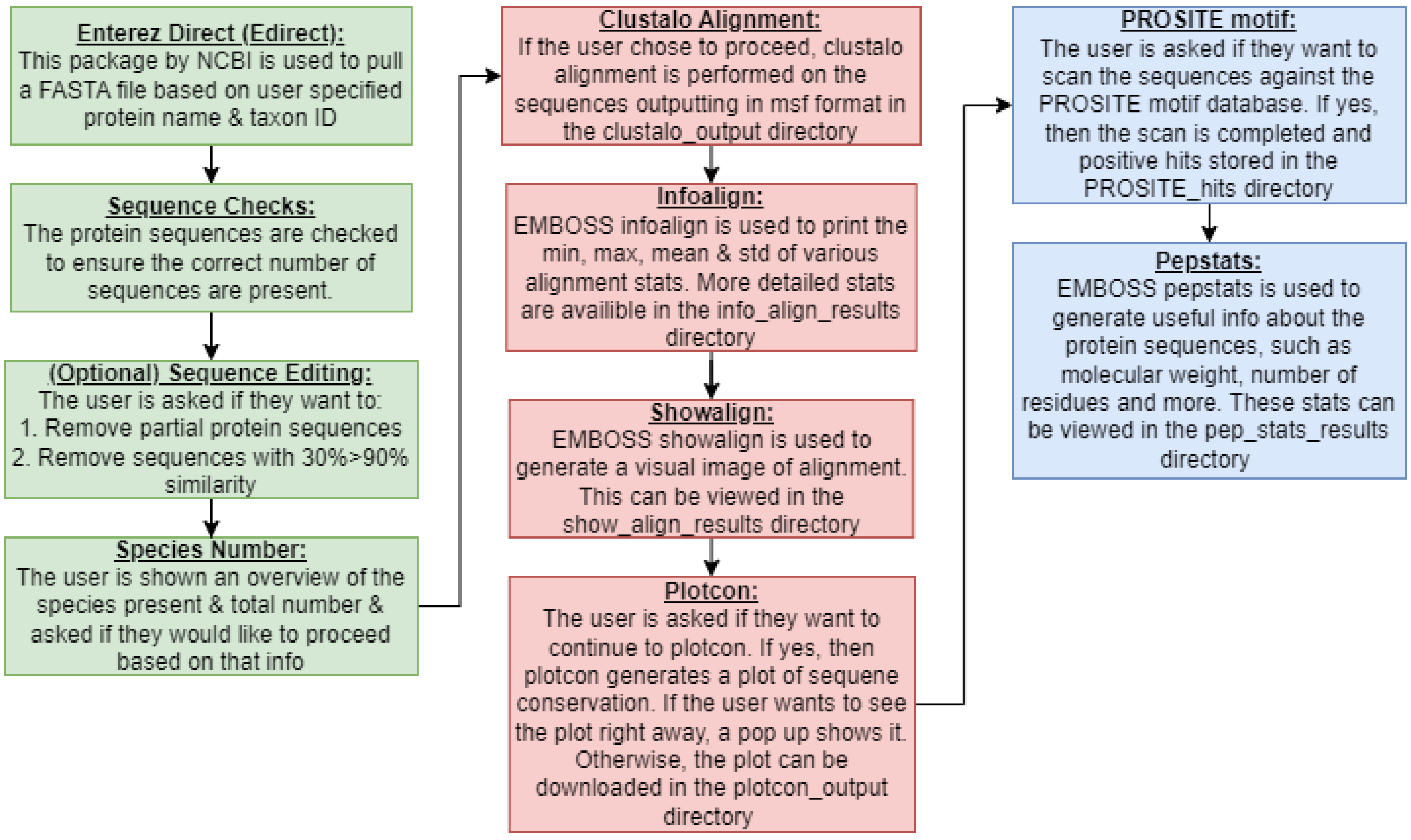
The file name of your uploaded document **must** include your exam number

Github repository link:
https://github.com/B221734-2022/ICA2

Decryption Key:
12345678

User Manual:

Programme Overview:


Enterez Direct (Edirect):
This package by NCBI is used to pull a FASTA file based on user specified protein name & taxon ID
Sequence Checks:
The protein sequences are checked to ensure the correct number of sequences are present.
(Optional) Sequence Editing:
The user is asked if they want to:
1. Remove partial protein sequences
2. Remove sequences with 30%>90% similarity
Species Number:
The user is shown an overview of the species present & total number & asked if they would like to proceed based on that info
Clustalo Alignment:
If the user chose to proceed, clustalo alignment is performed on the sequences outputting in msf format in the clustalo_output directory
Infoalign:
EMBOSS infoalign is used to print the min, max, mean & std of various alignment stats. More detailed stats are availible in the info_align_results directory
Showalign:
EMBOSS showalign is used to generate a visual image of alignment. This can be viewed in the show_align_results directory
Plotcon:
The user is asked if they want to continue to plotcon. If yes, then plotcon generates a plot of sequene conservation. If the user wants to see the plot right away, a pop up shows it. Otherwise, the plot can be downloaded in the plotcon_output directory
PROSITE motif:
The user is asked if they want to scan the sequences against the PROSITE motif database. If yes, then the scan is completed and positive hits stored in the PROSITE_hits directory
Pepstats:
EMBOSS pepstats is used to generate useful info about the protein sequences, such as molecular weight, number of residues and more. These stats can be viewed in the pep_stats_results directory

This is a script designed to automate sequence conservation analysis of protein sequences and scan those sequences against the PROSITE protein motif database.

To start the script, type “python master_script.py”

Ordinary User:

Enterez Direct (Edirect) & Sequence filtering:

- The user is asked to enter protein name and taxon ID.
- The script will search the NCBI protein database for matches with the user’s search query. If there are over 1000 sequences or less than 3 in the result, the user will be asked to adjust their search input accordingly. This is to both ensure speed, and to ensure the programme functions as intended, as clustalo requires at least 3 sequences to run.
- The user is asked if they would like to remove partial protein sequences.
- The user is asked if they would like to remove sequences with less than 30% similarity or more than 90% similarity.
- An overview of the species contained in the FASTA file is shown, along with the total number of species represented in the file. The user is asked if they would like to continue to the clustalo alignment based on this data.

Clustalo Alignment:

- Clustalo alignment is performed on the protein sequences. The resulting msf file is saved in the clustalo_output directory.
- EMBOSS infoalign is performed on the clustalo alignment, and an aggregate of stats (min, max, mean, standard deviation) for number of identical residues, number of different residues, number of similar residues and number and percentage change is displayed to the user. The full infoalign file with further information is saved in the info_align_results directory.
- EMBOSS showalign is performed on the clustalo_alignment. This creates a visual representation of the alignment. This can be viewed in the show_align_results directory.

Plotcon:

- The user is asked if they want to continue to plotcon analysis. This EMBOSS tool plots the amino acid conservation in the sequence alignment.
- The user is asked if they would like the view the plot now, if yes, a pop-up displays the plot.
- The plot is saved as a png in the plotcon_output directory where it can be downloaded and viewed later.
- To continue to PROSITE motif database scan, the graph pop-out window must be closed.

PROSITE motif database scan:

- The user is asked if they would like to continue to PROSITE protein motif database scan.
- EMBOSS patmatmotifs tool is used to scan the sequences against the PROSITE protein motif database.
- Positive hits are stored in the PROSITE_hits directory, where the .dbmotif files can be viewed by the user for the details of the hit. Sequences without any hits are stored in the PROSITE_dir directory.

Pepstats:

- EMBOSS pepstats is ran on the FASTA sequence file.
- This tool shows information such as molecular weight, number of residues, isoelectric point, and more
- The full information can be viewed in the pep_stats_results directory.

Python 3 Programmer:

Python Modules Used:

- os
- re
- shutil
- pathlib from Path
- pandas
- sys
- glob

Enterez Direct (Edirect) & Sequence filtering:

edirect is contained between lines 15-85 of the script, titled "## START OF ESEARCH ##"

- This script uses esearch, efilter and edirect.
- A function is used to take the user inputs the protein name and taxon ID, and to run the esearch, efilter and efetch commands. These values are stored as variables that are passed to the "eseach_cmd1". This variable does not contain efetch as it is used to check the sequence number before the FASTA file is downloaded. Errors are redirected to the error.txt file. This command is ran, then the <Count> line is stored in a temporary file. This file is read, and the sequence count is extracted with regex.
- An if loop is used to make sure the error.txt file is empty, ensuring the esearch query was valid. If the error.txt file contains something, then the esearch failed so the function is restarted. The user is told the search input was invalid and is asked to try again.
- Another if loop is used to check if the count number is greater than 1000 or less than 3. If the sequence number is outside these limits, a warning message is printed and the function is restarted to ask for user input again.
- If the count does not exceed the limits, then "esearch_cmd2" is ran. This variable contains the efetch command in order to download the FASTA file.
- If the command ran successfully, and there are less than 1000, but more than 3 sequences, then the function is ended, returning the protein_name_arg (protein name with space replaced with "_" for naming files) and taxonID variables.

Removal of partial sequences is contained between lines 86-112, titled "## REMOVE PARTIAL SEQEUNCES ##"

- The user is asked if they want to remove partial sequences, if they answer no then the script continues, if they answer yes then the sequences are removed.
- A combination of "while" and "if" loops are used to remove both the line containing "partial" and its corresponding sequence.

EMBOSS skipredundant is contained between lines 113-134 of the script, titled "## SKIP REDUNDANT ##"

- The user is asked if they want to remove sequences with less than 30% similarity and greater than 90% similarity. All parameters are set to the default parameters for skipredundant, however can be edited as necessary in the "skip_redundant_cmd" variable.
- If they answer yes to removing the sequences, skipredundant is ran, if they answer no then the command is not run.

FASTA to dataframe section of the code is contained between lines 135-203 of the script, titled "## START OF FASTA -> DATAFRAME/CSV ##"

- Firstly, lists for the id (accession number), name (protein name), organism, and sequence are defined.
- A "for" loops is used to read through all the lines in the file.
- And "if" loop is used to identify the FASTA title line that always starts with '>'.
- The line is split into a list. The accession number is always the first thing in the fasta title line, so it will be index [0] in the list. This is appended to the id list created earlier.
- A re.search is used to find the organism name which I always contained between "[]". The regex expression assigns the organism name to group(1). This is then removed from the line using the replace() method, leaving only the ID and protein name in the list. Since the protein name will be the only thing after the ID, the index of [1:] will grab only the protein name, which is appended to the protein name list. The organism name that was assigned to group(1) in regex is appended to the organism name list.
- If the line does now start with '>' in FASTA format, it is a sequence. Therefore, in the "if" loop any line that does not start with '>' can be directly added to the seq list.
- The lists are converted to series, which are then used to create a panda dataframe.
- The dataframe is used to print out a summary of organisms present, and a sum of the total number of organisms.
- The dataframe is saved as a csv file in the "csv_file_dir".
- The user is then asked if based on the data shown, would they like the continue to clustalo alignment. If yes, the script continues, if no then the script exists.

<u>Clustalo, infoalign and showalign:</u>

Clustalo section of the code is contained between lines 204-229 titled "## START OF CLUSTALO ##"

- Clustalo is ran on the FASTA file with the parameters: --outfmt=msf (output is in msf format), --threads=32 (assigns 32 threads to speed up alignment), --wrap=80 (defines how many AA per line), --force (overwrites existing files).
- Clustalo alignment is output to the clustalo_output directory.

Info align section of the code is contained between lines 230-255, titled "## INFO ALIGN ##"

- This section is my "wild card". It takes the output from the infoalign EMBOSS tool, converts it to a dataframe, then summarises important features of the alignment before printing it to the screen for the user to see.

- Two info align commands are used. The first is used to generate the dataframe, the second is used to generate the output infoalign file.
- The parameters of infoalign are to limit the output to make it easier to read, however if all possible information is needed, deleting the parameters, leaving only -sequence and -outfile, will produce an output file with much more information.

The show align section of the code is contained between lines 256-279, titled "## SHOW ALIGN ##"

- Show align is used to create a visual representation of the alignment.
- The user is then asked if they would like to continue to plotcon, if yes then the script continues, if no then the script is exited.

Plotcon:

The plotcon section of the code is contained between lines 256-332, titled "## START OF PLOTCON ##"

- The plotcon uses two different plotcon commands. One is for outputting the plot directly as a pop-up, and one to save it to a png file in the plotcon_output directory.
- The user is asked if they want to view the plot right away, if they do then the plotcon_input2 is ran, which outputs the plot in a pop-up tab. If they say no, only plotcon_input1 is ran, saving the png of the plot.
- >/dev/null redirects the output so that it is muted, avoiding spamming the user's screen while the script is running.
- The user is asked if they want to continue to PROSITE motif database scan. If yes, then the script continues, if no then the script exists.

PROSITE motif database scan:

The PROSITE motif database scan section of the code is contained between lines 333-393, titled "## START OF PROSITE DATBSE SCAN ##"

- The PROSITE scan contains 2 EMBOSS commands, seqretsplit and patmatmotifs.
- Patmatmotifs will only accept 1 sequence from 1 file at a time, so seqretsplit was used to split the one FASTA file containing many sequences, into many fasta files containing 1 sequence each.
- Seqretsplit outputs the individual files into the "split_fasta_dir" directory, naming the files by the sequence accession number.
- A for loop was used to pass the files in the "pathlist" variable defined using the Path module with .glob to match all file paths matching the name. Patmatmotifs then outputs the files to the split_fasta_dir" directory.
- A for loop using the glob module was used to move all the files ending in ".dbmotif" to the "PROSITE_dir" directory.
- To select only the files which has a hit, the files were read with a "for" loop. An "if" loop was used to find the lines containing "HitCount".
- Lines with "HitCount" were stored in the matching list. This list is made into a string, then split again using ":" as a separator. Since the hit count number is after the ":", it would be at index [1] in the list. This was turned into an int, then if the value was > 0, the file was moved to the "PROSITE_hits" directory.

Pepstats:

The pepstats section of the code is contained between lines 394-403, titled "## PEP STATS ##"

- This uses the EMBOSS pepstats command to obtain useful information about the protein sequences. The default parameters were used for this command.

Additional Features that could be added:

- An output file containing a summary of information generated by the script.
- The user could be given the option to compare between species for the sequence conservation analysis.

These are features that would be useful additions in the future.