Fasta Tool - TA Eric

Create a python script that can be called on the command line that takes arguments to indicate the action to be performed on a given nucleotide FASTA file.

Example Functionality:

1. FASTA stats: provide a FASTA file and return some common FASTA stats

```
./fasta_multi_tool.py stats myfasta.fa
myfasta.fa
Seq Count: 10
Max len: 2000
Min len: 100
Avg len: 650
N50: 1000
L50: 5
GC%: 55%
```

2 Reverse complement: provide a FASTA file and return FASTA formated output containing the reverse complement of all sequences contained in the input FASTA.

```
./fasta_multi_tool.py revcomp myfasta.fa
```

3. Subseq: provide a FASTA file, a seq ID, and a coordinate range. Return a FASTA formated sequence containing the subseq requested

```
./fasta_multi_tool.py subseq myfasta.fa seqName 5:100
```

4.Translation: provide FASTA file of nucleotide sequences and return FASTA formated tranlslated protein sequence

```
./fasta_multi_tool.py translate myfasta.fa
```

5.Split: provide FASTA file. Return the number of files requested with the sequences in the input FASTA split across that file count

```
./fasta_multi_tool.py split myfasta.fa 10
```

You can come up with multiple other functions, analyses etc.

```
```

Ultimate fighter: May the best microbe win - TA Kirsten

Microbial interactions shape the diversity of life. Biosynthetic gene clusters (BGCs) are contiguous groups of genes within microbial genomes that work in concert to produce complex bioactive molecules. These molecules, often secondary metabolites, can mediate various interactions between microbes, including competition, communication, and cooperation. In this project, we will design a game that tests the capabilities of different microbes to compete against each other, based on various factors including their biosynthetic gene content.

Microbe Class:

1. Each microbe will have properties such as:

- Name
- Biosynthetic Gene Content
- Colony Size (input by the user)
- Strength (computed based on gene content and colony size)

1. Battle Function:

This will simulate a battle between two microbes and determine the winner based on their strengths.

1. Extensions

- Introduce more attributes for the microbes.
- Allow for user-created microbes.
- Use actual data from MiBIG (https://mibig.secondarymetabolites.org/) to inform the attributes and interactions.
    - Involves parsing a JSON formatted file.
    - Involves creating a scoring matrix to designate power levels assigned to BGC classes.

- Add more sophisticated battle mechanics, e.g., certain genes can counteract others.
- Introduce graphics and UI for a more interactive experience.

(Structural) Variant filtering and analysis - TA Riley

Starting with a set of variants, develop tools to score them based on evidence, removing false positives etc. The code should be able to handle several different forms of variants: SNPs, indels, SV etc. You may want to build different modules that are specific for the different variant types.

Features: Initial processing of sequencing data in unix to identify variants by graded category. In python, these variants can be binned by number of counts per sample, and an enrichment score will be developed (potentially using entropy calculations) and a threshold identified to select variants enriched and relatively specific to condition. As a bonus, machine learning paradigms from numpy/sklearn/other could be implemented for less supervised variant filtration.

-How this solves: This solution will allow us to rapidly process many samples in a script and select enriched variants based on statistics (and potential ML) to narrow the scope of wet lab querying.

-Inputs: In order to keep up with new technologies, I propose using long read sequencing input, which would be direct from patient data in the long run. For the sake of this project, it is possible to acquire long read data from recent publications off the GEO database, or there are datasets directly available from PacBio.

-Outputs: This analysis could potentially produce a mixed set of outputs for varied downstream use. Most simply, a filtered vcf could identify variants plainly, and could be accompanied by a table of metrics by condition set, and potentially a paired set of fasta files of reference vs alternate sequences to compare sequence motifs. (If time, motif disruption could also be calculated)

-Challenges: selecting an appropriate enrichment/filtration metric, and implementation of less familiar bonus analyses such as machine learning on condition vs non-condition, or identification of disrupted sequence motifs.

Bonus: add a module to detect the variants from fastq files

homo-log - an automated protein phylogeny builder - TA Jessen

Develop a program that takes as input:

    - 1 protein sequence query (either as a sequence or an accession)
    - 1 phylogenetic group (e.g., genus)

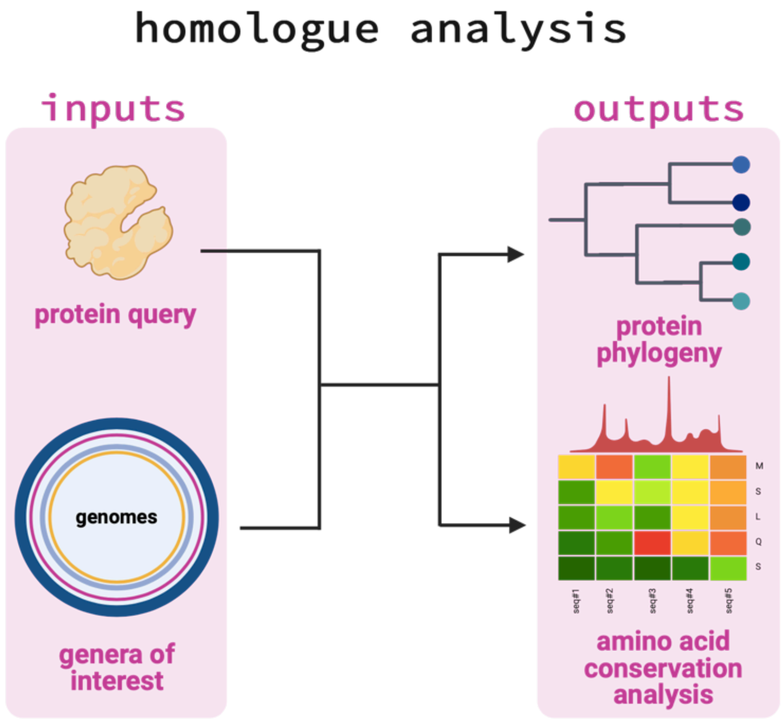
and produces as output:

- A list of all homologous proteins from the provided database in .tsv format with all scoring metrics, genome coordinates, protein length, and accessions
- A FASTA file with all homologous protein sequences

    - A graphical representation of the evolutionary distance between protein homologues (e.g., a phylogenetic tree)
    - A graphical representation of each amino acid in the query protein and their conservation among most closely-related sequences in the group (e.g., closely-related sequences on x-axis, query on y-axis, percent similarity as heat map for each position amongst all results in the group)
    - Optional: a graphical alignment combining phylogenetic information (a tree) paired with amino acid conservation information (an alignment)

using:

- Command line to run program
- Code to automatically extract data from online databases based on user-input
- A combination of Python and Unix to parse inputs and intermediate data
- Python and/or R to produce graphics & plots


homologue analysis
inputs
outputs
protein query
genomes
genera of interest
protein phylogeny
M
S
L
Q
S
seq#1
seq#2
seq#3
seq#4
seq#5
amino acid conservation analysis

Combined *-seq Analysis - TA Simon

We will work to integrate sequence-based (*-seq) and other data types.

We'll start with two combinations; others can be added as desired.

1) RNA-seq (to link differentially-expressed genes...) + ChIP-seq (... with TF binding)

This tool would first annotate called peak regions from a ChIP-seq dataset (output of Macs2 or a similar peak caller) to the nearest gene throughout the reference genome using one of many available peak annotation tools. The two datasets (RNA-seq differential expression and ChIP-seq annotated peaks) would then be combined to identify annotated peaks that are shared with significantly upregulated or downregulated genes in the RNA-seq dataset. These shared genes would be utilized to report a number of metrics in both written and graphical formats. This tool would allow the user to quickly and easily visualize relationships between transcription factor binding and differential gene expression.

Inputs: **1)** A tab separated file output by DEseq2 that contains gene ids, fold changes (log2 transformed), and p-values. **2)** A .bed or .narrowPeak formatted file **3)** Genome **4)** p-value cutoffs etc **5)** Colors for output bar graphs (default: something nice and colorblind friendly!)

Outputs: **1)** A list of genes annotated to peaks (this could be done with a tool from the Homer suite or a number of other available peak annotation tools). **2)** A file containing lists of annotated peaks overlapping with up or down-regulated genes, and percentage of overlap. **3)** A bar graph showing percent of up- or down-regulated genes overlapping with called peaks. **4)** Results from pathway analysis (over-representation analysis) run on the different lists of genes. **5)** Graphical representation (dot plots?) of pathway analysis results.

2) RNAseq dataset (H5 format)--that is used to store multidimensional arrays plus antibody based ADT (barcodes).

The goal of the project will be to develop modules that will enable parsing and manipulation of this data, and merging of the datasets to define cell types with ADT components and overlay the scRNAseq data, once the cell types are defined. There is also a Flo-data set. Data is published as part of a larger paper/dataset:Swanson E, Lord C, Reading J, Heubeck AT et al. Simultaneous trimodal single-cell measurement of transcripts, epitopes, and chromatin accessibility using TEA-seq. Elife 2021 Apr 9;10. PMID: 33835024 https://pubmed.ncbi.nlm.nih.gov/33835024/

GEO info: https://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSM5123955 CITE-seq, scATAC-seq, single-cell and single-nuclei 10x Multiome ATAC + Gene Expression, and TEA-seq libraries from the same PBMC sample in parallel.

Data: https://github.com/AllenInstitute/aifi-swanson-teaseq#cite (The .h5 file has the RNA-seq data) The .csv file has the matrix of surface marker ADT counts

Outputs:
Parsed data files that enable analysis
Alignments
Probably most important is the ability to couple the ADT data to improve the Cell Typing in the scRNA seq data.
Data can be analyzed as in the manuscript:
UMAPs; Heat Maps, Other annotation Other resources, cell types fractions of total cells