September 2019

HIV-1 sample processing pipeline

Purpose

This pipeline allows the automated analysis of HIV-1 pol samples provided in fasta format. Sample data should be anonymised prior to input using the provided macro for Microsoft Excel and then saved in fasta format. The pipeline initially queries the HIV-DB Sierra GraphQL Webservice using the sierrapy Python package and returns HIV-1 subtype predictions for each sample with additional information related to drug resistance-associated mutations. Reference sequences are added to the aligned files using MAFFT and a phylogeny is generated by RAxML. The resulting phylogeny with bootstrap values is saved as a pdf.

Installation/set-up

- Requirements:
    - Microsoft Excel
    - Access to the Alpha server hosted at CVR
    - Python3.6
    - SierraPy
    - MAFFT
    - RAxML

- Python dependencies:
    - docx
    - SeqIO (BioPython)
    - ete3
    - pandas
    - json

Procedure

1. RenameSequences.xlsm

RenameSequences.xlsm contains a macro written for Microsoft Excel 2019 version 16.28. The macro (Multi_FindReplace) is designed to relabel a specific set of data generated at the WoSSVC. Entries in sheet2 can be re-labelled based on the columns in sheet1. This will replace the lab number with the sample ID for all entries in sheet2. Sequence entries in fasta format that have a traceable ID (lab ID) can be pasted into sheet2. Based on the identifier columns in Sheet1, the traceable lab ID will be replaced with the sample ID. The anonymised sequence entries can then be copied back out to a text file and saved in fasta format.

In Excel, Go to Tools → Macro → Macros… → Click on Multi_FindReplace → Click Run

2. Transfer the saved fasta file to Alpha.cvr.gla.ac.uk

Log in using ssh and -Y to enable phylogenetic trees to be visualised as pdf. (Or login using MobaXterm or X2Go – these have not been tested yet but should work as the X-window is built in)

e.g. ssh -Y user@Alpha.cvr.gla.ac.uk

pipeline location on Alpha= home2/db/HIV-cluster
(currently full read/write access restricted to few users)

3. preprocessing.sh

This bash script takes a single multi-sample fasta file as input and runs the preprocessing pipeline in two steps: 1. subtype and drug resistance query; 2. Add aligned HIV-1 reference sequences and generate phylogeny.

Example Usage

preprocessing.sh [-h -f] -- program to split fasta sequences by subtype and generate a phylogeny
where:

-h show this help text
-f input sequences in single multi-sample fasta format file

Command:

```
home2/db/HIV-cluster /preprocessing.sh -f <input_samples.fa>
```

Output:

reports_{date}/ - directory containing all results files
{date}.{input}.fasta - input fasta samples aligned to reference sequences
{date}.{input}.txt - text file containing sample to subtype information
{date}.{input}.json - HIVDB query response in json format, this is parsed to generate the individual .docx reports
{date}_DRM-overview.txt - overview of drug resistance associated mutations across all queried samples
RAxML_tree-rerooted.pdf - visualisation of phylogenetic tree of samples and reference sequences
RAxML/ - directory containing additional output files from generating and re-rooting the phylogeny using RAxML

Individual Scripts

- RenameSequences.xlsm
- preprocessing.sh

- bin/
    1) perform_query.py
    2) parse_json_write_docx.py
    3) parse_json_store_metadata.py
    4) visualise_phylogeny.py

1) **perform_query.py**: a Python module to query the HIVDB Sierra GraphQL Webservice using the SierraPy package (https://github.com/hivdb/sierra-client/tree/master/python). Requires HIV Pol samples in fasta format and returns HIV subtype information.

2) **parse_json_write_docx.py**: this script will generate a report for each sample in Microsoft word docx format detailing the subtype and information regarding drug-resistance associated mutations.

3) **parse_json_store_metadata.py**: generates an overview of the drug resistance associated mutations present in all samples from the current run and writes this to a tab delimited text file.

4) **visualise_phylogeny.py**: The phylogeny generated by RAxML is then visualised in pdf format. The tree will be rerooted by rooting it at the branch that best balances the subtree lengths.