Profiling Strain-Level Diversity pipeline

Bacterial strain transmission

(StrainPhlan)

A pipeline of the Vonaesch Lab for the curnagl cluster

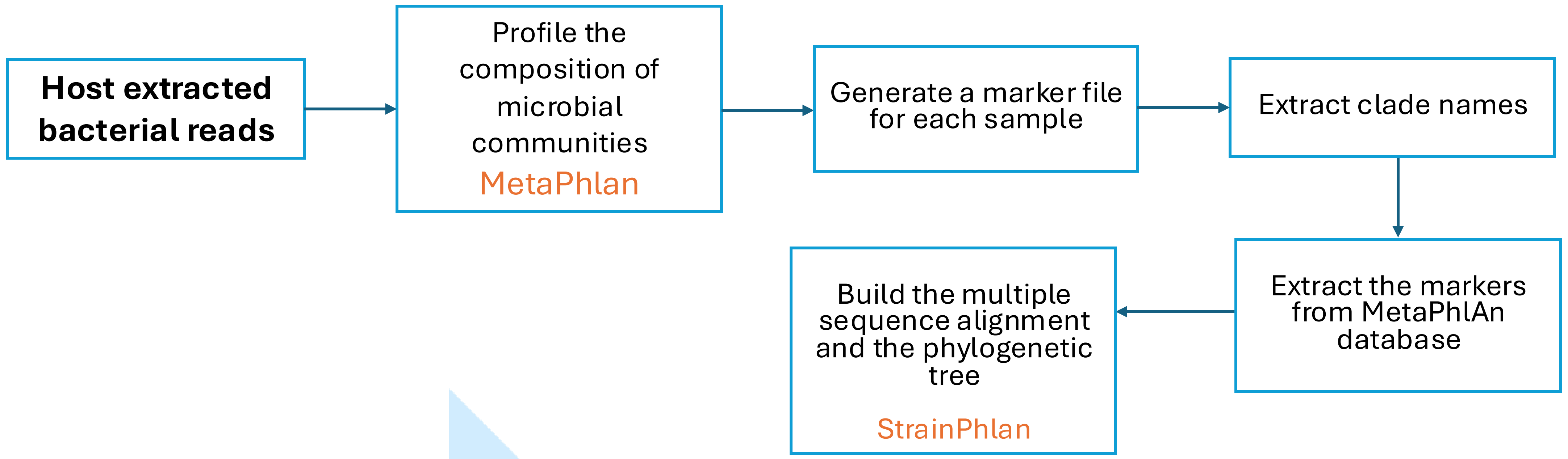
Margaux Creze & Simon Yersin

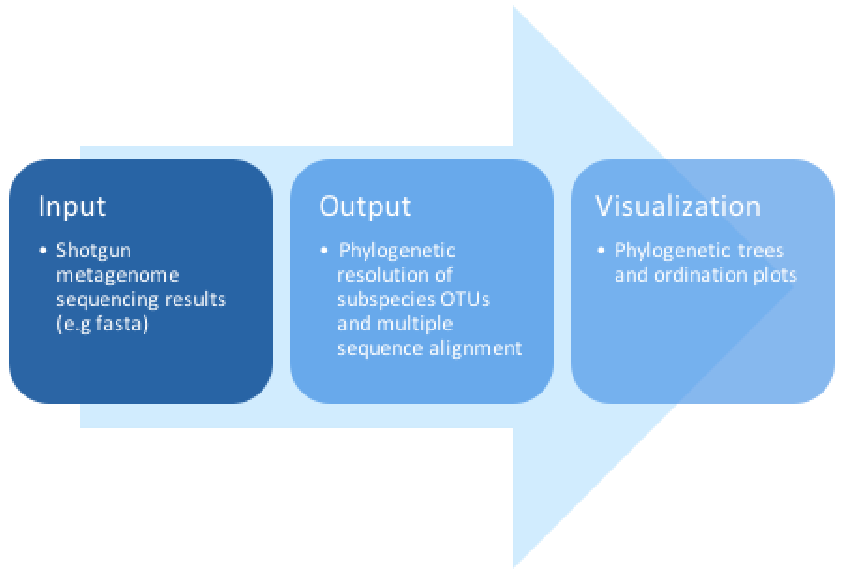
Profiling strain-level diversity

- StrainPhlAn is a computational tool for tracking individual strains across a large set of samples. The input of StrainPhlAn is a set of metagenomic samples and for a given species of interes, the output is a multiple sequence alignment (MSA) file of the strains of the species of interest reconstructed from the samples. From this MSA, StrainPhlAn calls PhyloPhlAn to build the phylogenetic tree showing the phylogenetic relationship between strains identified in each sample.
- For each sample, StrainPhlAn is able to identify strains of a species by merging and concatenating all reads mapped against species-specific MetaPhlAn markers.
- StrainPhlAn can distinguish between different strains of the same microbial species, providing a finer granularity in microbial community analysis
- It reconstructs phylogenetic trees of microbial strains based on their genomic markers, allowing researchers to study evolutionary relationships and strain-level variations
- Utilizes consensus markers derived from MetaPhlAn's profiling to perform the strain-level analysis
- Example of cohorts that used StrainPhlan for strain transmission
  - MicrobeMom cohort (Feehily et al, 2023)

How StrainPhlan works

1. Input Preparation:
    1. Consensus Markers: StrainPhlAn uses consensus markers generated from metagenomic samples. These markers are representative sequences of specific microbial strains found in the samples.
    2. Clade-specific Markers: For detailed analysis, clade-specific marker genes are used to focus on particular microbial clades.
    3. Reference Genomes: Reference genomes provide a basis for comparison and help in phylogenetic reconstruction.
2. Phylogenetic Analysis:
    1. StrainPhlAn aligns the consensus markers against the reference genomes.
    2. Constructs a phylogenetic tree to visualize and analyze the relationships between different strains.
3. Output:
    1. The output includes phylogenetic trees, strain-specific profiles, and other data that can be used to interpret the strain-level diversity in the microbial community.


Host extracted bacterial reads
Profile the composition of microbial communities
MetaPhlan
Generate a marker file for each sample
Extract clade names
Extract the markers from MetaPhlAn database
Build the multiple sequence alignment and the phylogenetic tree
StrainPhlan


Input
• Shotgun metagenome sequencing results (e.g fasta)
Output
• Phylogenetic resolution of subspecies OTUs and multiple sequence alignment
Visualization
• Phylogenetic trees and ordination plots

Installation

StrainPhlan 4 must be installed together with MetaPhlan 4.

For our group, we already installed MetaPhlan in a designed environment that you can use. To do so, use:

```
# Module
module load gcc/11.4.0
module load miniconda3/22.11.1

# Activate conda environment
eval "$(conda shell.bash hook)"
conda activate /work/FAC/FBM/DMF/pvonaesc/vonasch_lab_general/syersin/MetaPhlan/metaphlan
```

Step 0: Preparation

- In the scratch, prepare the following directories architecture:
  - StrainPhlan_scratch
    - std_output
    - Host_extracted_reads
    - MetaPhlan_output (will be created while running the script)
      - sams
      - bowtie2
      - profiles
    - Consensus_markers (will be created while running the script)
    - CladeMarkers
    - StrainPhlan_output

Step 1: MetaPhlan

Taxonomic profiling with MetaPhlAn

Aim: Obtain the sam files from your samples by mapping them against MetaPhlAn database

This step will run MetaPhlAn: the metagenomic samples will be mapped against the MetaPhlAn marker database and then SAM files (*.sam.bz2) will produced. Each SAM file (in SAM format) contains the reads mapped against the marker database of MetaPhlAn.

01_MetaPhlan.sh

After this step, you will have a folder "sams" containing the SAM files (*.sam.bz2) and other MetaPhlAn output files in the "bowtie2" and "profiles" folders.

Step 2: Strain-level profiling with StrainPhlAn

Extracting the samples markers

Aim: Produce the consensus-marker files which are the input for StrainPhlAn

For each sample, this step will reconstruct all species strains found in it and store them in a json file (*.json). Those strains are referred as *sample-reconstructed strains*.

02_Sample_to_markers.sh

The result is the same if you want to run several sample2markers.py scripts in parallel with each run for a sample (this may be useful for some cluster-system settings). After this step, you will have a folder consensus_markers containing all sample-marker files (*.json).

Step 3: Strain-level profiling with StrainPhlAn

Aim: Extract clade names from the marker files

From the marker files that have just been generated, extract the clade names and store them in a .txt file
📜 03_Clade_name.sh

Alternatively, and depending on the number of samples, you can just run the command on your computer without launching a bash script

Step 4: Strain-level profiling with StrainPhlAn

Extracting the species marker genes

Aim: Extract the markers of some strains of interest (or all) from MetaPhlAn database (to add its reference genome later)

This step will extract the markers of your strains of interest in the database and then StrainPhlAn will identify the sequences in the reference genomes that are closest to them (in the next step by using blast). Those will be concatenated and referred to as *reference-genome-reconstructed strains*.

04_Extract_markers.sh

After this step, you should have as many files (.fna) as your strain of interest in folder "db_markers”. Each .fna file contains the markers of the strain of interest.

For our group, the MetaPhlan database can be found here:
work/FAC/FBM/DMF/pvonaesc/vonasch_lab_general/syersin/MetaPhlan/metaphlan/lib/python3.7/site-packages/metaphlan/metaphlan_databases

Step 5: Strain-level profiling with StrainPhlAn

Strain-level profiling

Aim: Build the multiple sequence alignment and the phylogenetic tree

This step will filter the selected clade markers based on their presence in the *sample-reconstructed strains* (stored in the marker files produced in **step 2**) and *reference-genomes* (if specified). Also, the *sample-reconstructed strains* and *reference-genomes* will be filtered based on the presence of the selected clade markers. From this filtered markers and samples, StrainPhlAn will call PhyloPhlAn to produce a multiple sequence alignment (MSA) to then build the phylogenetic tree.

05_StrainPhlan.sh

After this step, you will find the different trees in output folder. You'll have one tree per SGB marker.

Step 6: Strain-level profiling with StrainPhlAn

Extraction of pairwise phylogenetic distances

Aim: Extract the sample pairwise distances from the phylogenetic tree into a tsv file.

We will next extract the sample pairwise distances from the phylogenetic tree into a tsv file. We will use the tree_pairwisedists.py script in PyPhlAn

📜 06_Pairwise_phylo_dist.sh

The -n parameter normalizes the distances by the total branch length, creating what we call nGD, i.e. normalized phylogenetic distance.

Step 6: Add Metadata

In order to add the metadata, they provide a script called add_metadata_tree.py

```
add_metadata_tree.py  -t  output/RAxML_bestTree.t__SGB1877.StrainPhlAn4.tre  -f  fastq/metadata.txt  -m
subjectID --string_to_remove .fastq.bz2
```

The script "add_metadata_tree.py" can accept multiple metadata files (space-separated, wild card can also be used) and multiple trees. A metadata file is a tab-separated file where the first row is the meta-headers, and the following rows contain the metadata for each sample. Multiple metadata files are used in the case where your samples come from more than one dataset and you do not want to merge the metadata files.
For more details of using "add_metadata_tree.py", please see its help (with option "-h")

Note that "sampleID" is a compulsory field.

After adding the metadata, you will obtain the tree files "*.tre.metadata" with metadata and view them by Archaeopteryx as in the previous step.
If you have installed graphlan, you can plot the tree with the plot_tree_graphlan.py script:

```
plot_tree_graphlan.py -t output/RAxML_bestTree.t__SGB1877.StrainPhlAn4.tre.metadata -m subjectID
```

Sources

- https://github.com/biobakery/biobakery/wiki/strainphlan4
- https://github.com/biobakery/MetaPhlAn/wiki/StrainPhlAn-4.1
- https://github.com/biobakery/MetaPhlAn/wiki/Strain-Sharing-Inference