Metagenomics WGS Lab

BCB 5250 Introduction to Bioinformatics II

Spring 2020

Tae-Hyuk (Ted) Ahn
Department of Computer Science
Program of Bioinformatics and Computational Biology
Saint Louis University

SAINT LOUIS
UNIVERSITY
EST. 1818

Tutorial

- This tutorial is mostly referenced from https://github.com/LangilleLab/microbiome_helper/wiki/Metagenomics-Tutorial-(Humann2)
- This tutorial is set-up to walk you through the process of determining the taxonomic and functional composition of several metagenomic samples. It covers the use of Metaphlan2 (for taxonomic classification), HUMAnN2 (for functional classification), and STAMP (for visualization and statistical evaluation).
- Throughout the tutorial, there are several questions to ensure that you are understanding the process (and not just copy and pasting). Submit your answers for the questions.

Reference

- You will be using a subsampled version of the metagenomics dataset from Schmidt et al. (PLoS ONE 2014) that investigated changes in the oral microbiome associated with oral cancers (EBI PRJEB4953). **Note: since these files have been drastically subsampled they do not represent typical metagenomics raw files.** For instance, many more taxonomic and functional classifications would be output if the full datasets were used. However, due to the shallow read depth the running time of the workflow is extremely fast!

Virtual Environment

- It will be good for you to setup virtual environment to install multiple Python packages with avoiding dependency problems.
- There are multiple ways for it:
  - https://realpython.com/python-virtual-environments-a-primer/
  - https://docs.python.org/3/tutorial/venv.html
- A simple way is as below:

$ pyvenv bcb5250_env

$ source bcb5250_env/bin/activate

$ pip install --upgrade pip

$ pip install AnyPackageYouNeed (Cython, biom-format, biopython, numpy, kneaddata, humann2, and so on..)

Explore Samples

- Copy the tutorial data set and save them into your local directory, unzip the files, and enter this folder. The dataset is located at hopper: /public/ahnt/courses/bcb5250/metagenomics_wgs_lab/mgs_tutorial_Oct2017.zip
- You will have noticed that this map file contains four columns with the first being sample ids, then sample type (Cancer or Normal), sex, and the individual. Samples nearby an oral tumor and distal from oral tumors were taken for each individual.
- **Q1**) How many samples are there in total (you can either look at the map.txt file or count the FASTQ files)?
- **Q2**) How many samples are there of each sample type?

Pre-processing

- Two major pre-processing steps need to be performed on shotgun metagenomics sequences: quality filtering of reads and screening out of contaminant reads.
- Trimmomatic is one popular tool for filtering and trimming next-generation sequencing data. Typically you would first explore the quality of your data to choose sensible cut-offs using FASTQC or a similar alternative. Below we will run Trimmomatic on our test file with some typical cut-offs.
- It is also important to screen out contaminant sequences. This is because the strength of shotgun metagenomics is also its weakness: all DNA in your samples will be sequenced with similar efficiencies. This is a problem when host microbiome samples are taken since it's possible to get substantial amounts of host DNA included in your raw sequences. You should screen out these contaminant sequences before running taxonomic and functional classification. It's also a good idea to screen out PhiX sequences in your data: this virus is a common sequencing control since it has such a small genome.

Pre-processing

- kneadData is a helpful wrapper of both Trimmomatic and Bowtie2. You can use the below command to run kneaddata on a **single** sample.
- Install kneadData: $ pip install kneaddata
- You can download specific database (human, mouse, SILVA, and so on).

Download the database

It is recommended that you download the human (hg37_and_human_contamination) reference database (approx. size = 3.5 GB). However, this step is not required if you are using your own custom reference database or if you will not be running with a reference database.

This database is based on the Decoy Genome (http://www.cureffi.org/2013/02/01/the-decoy-genome/) and contaminants taken from "Human contamination in bacterial genomes has created thousands of spurious proteins" (Salzberg et. al. 2019)

- `$ kneaddata_database --download human_genome bowtie2 $DIR`
- When running this command, $DIR should be replaced with the full path to the directory you have selected to store the database.

If you are running with bmtagger instead of bowtie2, then download the bmtagger database instead of the bowtie2 database with the following command.

- `$ kneaddata_database --download human_genome bmtagger $DIR`
- When running this command, $DIR should be replaced with the full path to the directory you have selected to store the database.

The human transcriptome (hg38) reference database is also available for download (approx. size = 254 MB).

- `$ kneaddata_database --download human_transcriptome bowtie2 $DIR`

The SILVA Ribosomal RNA reference database is also available for download (approx. size = 11 GB).

- `$ kneaddata_database --download ribosomal_RNA bowtie2 $DIR`

The mouse (C57BL) reference database is also available for download (approx. size = 3 GB).

- `$ kneaddata_database --download mouse_C57BL bowtie2 $DIR`

Pre-processing

- I downloaded the human (hg37_and_human_contamination) reference database at /public/ahnt/courses/bcb5250/metagenomics_wgs_lab/bowtie2db/. The index name is hg37dec_v0.1
- For testing, you just run the kneaddata with the raw_data_example files as below:

```
$ kneaddata -i raw_data_example/p144C_R1.fastq -i raw_data_example/p144C_R2.fastq -o kneaddata_out -db /public/ahnt/courses/bcb5250/metagenomics_wgs_lab/bowtie2db/hg37dec_v0.1 --trimmomatic /public/ahnt/courses/bcb5250/metagenomics_wgs_lab/Trimmomatic-0.39 -t 8 --trimmomatic-options "SLIDINGWINDOW:4:20 MINLEN:50" --bowtie2-options "--very-sensitive --dovetail" --remove-intermediate-output
```

Q3: Run the FASTQC for each paired-end sample. Run MultiQC. Run kneaddata using your script. Then, run FASTQC again with the pre-processed samples. Run MultiQC. Provide your scripts and MultiQC output.

Pre-processing

The options above are:

- -i Input FASTQ file. This option is given twice for paired-end data.
- -o Output directory.
- -db Path to bowtie2 database.
- --trimmomatic Path to Trimmomatic folder containing jarfile.
- -t 8 Number of threads to use (8 in this case).
- --trimmomatic-options Options for Trimmomatic to use, in quotations ("SLIDINGWINDOW:4:20 MINLEN:50" in this case). See Trimmomatic website for more options.
- --bowtie2-options Options for bowtie2 to use, in quotations. The options "--very-sensitive" and "--dovetail" in this set alignment parameters to be very sensitive and sets cases where mates extend past each other to be concordant (i.e. they will be called as contaminants and be excluded).
- --remove-intermediate-output Intermediate files, including large FASTQs, will be removed.

Paired-end read files concatenation

- Before running the programs, you have to concatenate the FASTQs together now before running Metaphlan2 and HUMAnN2 since these programs do not use paired-end information.
- $ mkdir kneaddata_out/contam_seq
- $ mv kneaddata_out/*_contam_*fastq kneaddata_out/contam_seq
- $ perl /public/ahnt/courses/bcb5250/metagenomics_wgs_lab/microbiome_helper/concat_paired_end.pl -p 4 --no_R_match -o cat_reads kneaddata_out/*_paired_*.fastq

```
[ahnt@hopper:~/Course/bcb5250/2020S/labs/metagenomics_wgs_lab/mgs_tutorial_Oct2017]$ perl /public/ahnt/courses/bcb5250/metagenomics_wgs_lab/microbiome_h
elper/concat_paired_end.pl -p 4 --no_R_match -o cat_reads kneaddata_out/*_paired_*.fastq
cat kneaddata_out/p156N_R1_kneaddata_paired_1.fastq kneaddata_out/p156N_R1_kneaddata_paired_2.fastq > cat_reads/p156N.fastq
cat kneaddata_out/p136N_R1_kneaddata_paired_1.fastq kneaddata_out/p136N_R1_kneaddata_paired_2.fastq > cat_reads/p136N.fastq
cat kneaddata_out/p136C_R1_kneaddata_paired_1.fastq kneaddata_out/p136C_R1_kneaddata_paired_2.fastq > cat_reads/p136C.fastq
cat kneaddata_out/p143N_R1_kneaddata_paired_1.fastq kneaddata_out/p143N_R1_kneaddata_paired_2.fastq > cat_reads/p143N.fastq
cat kneaddata_out/p156C_R1_kneaddata_paired_1.fastq kneaddata_out/p156C_R1_kneaddata_paired_2.fastq > cat_reads/p156C.fastq
cat kneaddata_out/p153C_R1_kneaddata_paired_1.fastq kneaddata_out/p153C_R1_kneaddata_paired_2.fastq > cat_reads/p153C.fastq
cat kneaddata_out/p146C_R1_kneaddata_paired_1.fastq kneaddata_out/p146C_R1_kneaddata_paired_2.fastq > cat_reads/p146C.fastq
cat kneaddata_out/p143C_R1_kneaddata_paired_1.fastq kneaddata_out/p143C_R1_kneaddata_paired_2.fastq > cat_reads/p143C.fastq
cat kneaddata_out/p144C_R1_kneaddata_paired_1.fastq kneaddata_out/p144C_R1_kneaddata_paired_2.fastq > cat_reads/p144C.fastq
cat kneaddata_out/p146N_R1_kneaddata_paired_1.fastq kneaddata_out/p146N_R1_kneaddata_paired_2.fastq > cat_reads/p146N.fastq
cat kneaddata_out/p153N_R1_kneaddata_paired_1.fastq kneaddata_out/p153N_R1_kneaddata_paired_2.fastq > cat_reads/p153N.fastq
cat kneaddata_out/p144N_R1_kneaddata_paired_1.fastq kneaddata_out/p144N_R1_kneaddata_paired_2.fastq > cat_reads/p144N.fastq
```

Taxonomic and Functional Profiling with MetaPhlAn2 and HUMAnN2 Respectively

- MetaPhlAn2 is a standalone tool and can be run with the metaphlan2.py command. It is easier to run a custom analysis with this script directly. However, since HUMAnN2 runs this script anyway (with default options unless specified otherwise) we can use the output files produced by this pipeline instead, which is simpler!
- However, study and follow the tutorial of Biobakery Metaphlan2 tutotial: https://bitbucket.org/biobakery/biobakery/wiki/metaphlan2#rst-header-input-files
- To begin with make sure the program metaphlahn2 is in your .bashrc. Add the path /public/ahnt/courses/bcb5250/metagenomics_wgs_lab/metaphlan2 into your environment.

Run Humann2

- Note that by default, a pip or other HUMAnN2 install will NOT include the full nucleotide or amino acid search databases needed for raw metagenome or metatranscriptome profiling, as these are quite large and should be installed based on project needs. A typical environment, however, will require one of each, and we default to ChocoPhlAn for the former and UniRef90 for the latter (note that **for the purposes of this tutorial alone, you will download demo databases to reduce run time**):
  - $ humann2_databases --download chocophlan DEMO humann2_database_downloads
  - $ humann2_databases --download uniref DEMO_diamond humann2_database_downloads

- Run Humann2
  - $ humann2 --threads 8 --input cat_reads/p144C.fastq --output humann2_out/

Run Humann2

- **Q4)** What is the relative abundance of organisms unclassified at the species level for the genus *Neisseria* for the sample p144C?

- Run the humann2 for all samples (Do not follow **Running HUMAnN2 on all FASTQs with GNU Parallel**).

- **Q5**) Make you own shell script (run_humann2.sh) to run it. The shell script should get the FASTQ list of files in a directory in a shell script, and run the humann2 iteratively. Copy and paste your shell script to the answer sheet. Use 8 (or 12) threads to run each run.

Merging Metaphlan2 Results

- Create a directory “metaphlan2_out” to save each Metaphlan2 results (E.g., humann2_out/p144C_metaphlan_bugs_list.tsv). Then merge the results using merge_metaphlan_tables.py.
  - $ merge_metaphlan_tables.py metaphlan2_out/*.tsv > metaphlan2_merged.tsv
- To read this table into STAMP we'll need to convert it be a .SPF file with this command:
  - $ perl /public/ahnt/courses/bcb5250/metagenomics_wgs_lab/microbiome_helper /metaphlan_to_stamp.pl metaphlan2_merged.tsv > metaphlan2_merged.spf
- Notice that the sample names in map.txt don't match the column names in metaphlan2_merged.spf. These need to match for STAMP to read the table. This can be fixed by removing all instances of "_metaphlan_bugs_list"
  - Convert the sample name from “p144C_metaphlan_bugs_list” to “p144C” (i.e., using a sed or awk or VI search/replace command).
- **Q6**) How did you replace "_metaphlan_bugs_list" into NULL? Check the sed command in the web manual, but I hope you work it using your own way. Copy and paste your command into your solution sheet.

STAMP

- https://beikolab.cs.dal.ca/software/STAMP
- Install: https://beikolab.cs.dal.ca/software/Quick_installation_instructions_for_STAMP

STAMP

STAMP is a software package for analyzing taxonomic or metabolic profiles that promotes 'best practices' in choosing appropriate statistical techniques and reporting results. Statistical hypothesis tests for pairs of samples or groups of samples is support along with a wide range of exploratory plots. STAMP encourages the use of effect sizes and confidence intervals in assessing biological importance. A user friendly, graphical interface permits easy exploration of statistical results and generation of publication quality plots for inferring the biological relevance of features in a metagenomic profile. STAMP is open source, extensible via a plugin framework, and available for all major platforms.

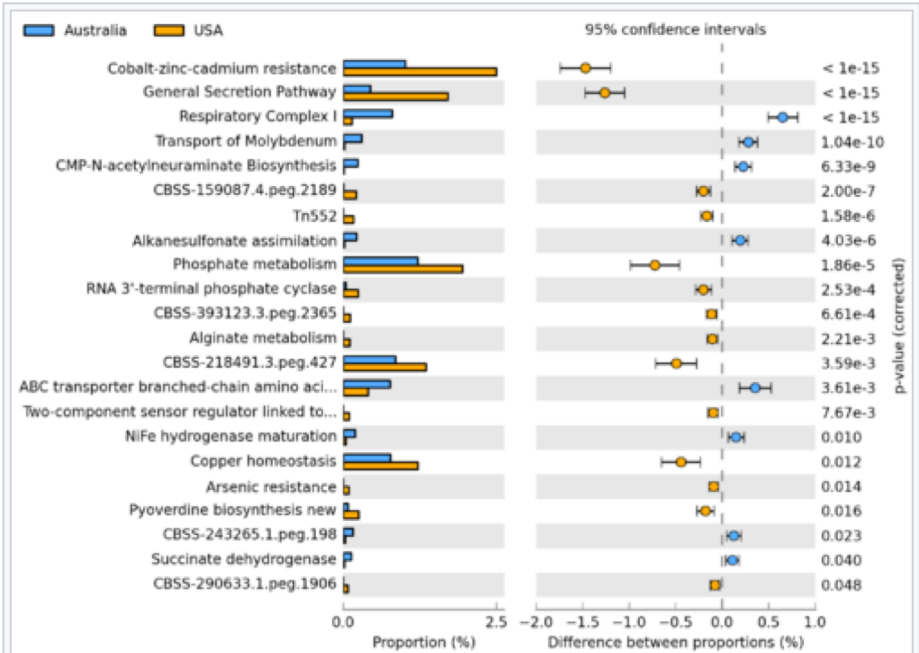
Announcements

- June 26, 2015: STAMP v2.1.3 released. Minor bug fix to scatter plot to properly handle profiles contained a single feature.
- June 15, 2015: STAMP v2.1.2 released. Minor enhancements to extended error bar, heatmap, and profile bar plots.
- June 7, 2015: STAMP v2.1.1 released. Resolves issue with v2.1.0 installation.
- June 4, 2015: STAMP v2.1.0 released. Resolves the _hierarchy_wrap issue. Requires numpy >= 1.9.1, scipy >= 0.15.1, matplotlib >= 1.4.2.
- Previous announcements

Documentation

- Quick installation instructions (Microsoft Windows, Linux, Apple's Mac OS X)
- User's Guide
- Google Group
- FAQs
- Version history

Downloads



Using STAMP to identify SEED subsystems which are differentially abundant between *Candidatus* Accumulibacter phosphatis sequences obtained from a pair of enhanced biological phosphorus removal (EBPR) sludge metagenomes(data originally described in Parks and Beiko, 2010).

Analyzing Metaphlan2 output with STAMP

- STAMP takes two main files as input the profile data which is a table that contains the abundance of features (i.e. taxonomic or functions) and a group metadata file which provides more information about each of the samples in the profile data file.
- The metadata file is the map.txt file. This file is present already.
- Follow the web tutorial.

Q7) How much of the variation is explained by PC3?

Q8) Which genus is significant based on its *raw* P-value?

Q9) Save this boxplot of the Q9 genus by going to File -> Save plot and attach here

Analyzing a Subset of the HUMAnN2 output

- Similar analyses can be run on the pathway abundances identified with HUMAnN2 as above. To begin with we can join all individual tables into a single table as above.
  - $ humann2_join_tables --input humann2_out/ --file_name pathabundance --output humann2_pathabundance.tsv
- Then we'll want to normalize each sample into relative abundance (so that the counts for each sample sum to 100).
  - $ humann2_renorm_table --input humann2_pathabundance.tsv --units relab --output humann2_pathabundance_relab.tsv
- We could read this entire table (after running the reformatting commands below) and run similar analyses as we did above.
- However, since HUMAnN2 yields functions stratified by taxa we could limit ourselves just to those functions that were found within the genus Streptococcus in order to focus on the above finding with the Metaphlan2 data. You could once again use grep to slice out only those pathways that are linked to this genus specifically.
- In addition, more focused analyses could be based off of prior knowledge. For instance, pathways related to sugar degradation are especially of interest in the oral microbiome due to its relationship with dental health. We will test for differences between cancer and normal samples in STAMP similar to the earlier analysis based on this approach.
- First an unstratified version of the pathway abundance table needs to be generated (the below command will also create a stratified version only in the same directory):
  - $ humann2_split_stratified_table --input humann2_pathabundance_relab.tsv --output ./
- Then we can parse out the headerline and any lines matching the pathway of interest (LACTOSECAT-PWY: lactose and galactose degradation I).
  - $ head -n 1 humann2_pathabundance_relab_unstratified.tsv >> humann2_pathabundance_relab_LACTOSECAT-PWY.tsv
  - $ grep "LACTOSECAT-PWY" humann2_pathabundance_relab_unstratified.tsv >> humann2_pathabundance_relab_LACTOSECAT-PWY.tsv
- Using sed to make string replacements we can also format the header-line for input into STAMP.
  - $ sed 's/_Abundance//g' humann2_pathabundance_relab_LACTOSECAT-PWY.tsv > humann2_pathabundance_relab_LACTOSECAT-PWY.spf
  - $ sed -i 's/# Pathway/MetaCyc_pathway/' humann2_pathabundance_relab_LACTOSECAT-PWY.spf
- **Q10)** Run a two-sided Welch's t-test between sample types for this pathway. What is the P-value?