OTU Picking

Contents

Back to Table of Contents

All of the code in this page is meant to be run on the command line.

Getting usage info from QIIME scripts

First let's move to the Guerrero Negro data directory. This directory contains sequences from different depths in the Guerrero Negro microbial in Mexico (Harris JK **et al.** ISME J. 2013 Jan;7(1):50-60).

```
# To be run on the command line
# you will need to put your correct path here
cd /your/path/to/8992repo/data/guerreronegro
```

We are going to run closed reference OTU picking, but first let's get usage information and list of options (be sure QIIME is loaded first)

```
pick_closed_reference_otus.py -h
```

```
## Usage: pick_closed_reference_otus.py [options] {-i/--input_fp INPUT_FP -o/--output_dir OUTPUT_DIR}
##
## [] indicates optional input (order unimportant)
## {} indicates required input (order unimportant)
##
##
## This script picks OTUs using a closed reference and constructs an OTU table.
## Taxonomy is assigned using a pre-defined taxonomy map of reference sequence OTU
## to taxonomy. If full-length genomes are provided as the reference sequences,
## this script applies the Shotgun UniFrac method.
##
## **Note:** If most or all of your sequences are failing to hit the reference,
```

```
## your sequences may be in the reverse orientation with respect to your reference
## database. To address this, you should add the following line to your parameters
## file (creating one, if necessary) and pass this file as -p:
##
## pick_otus:enable_rev_strand_match True
##
## Be aware that this doubles the amount of memory used.
##
##
##
## Example usage:
## Print help message and exit
##  pick_closed_reference_otus.py -h
##
## Pick OTUs, assign taxonomy, and create an OTU table against a reference set of OTUs. ALWAYS SPECIFY A
##  pick_closed_reference_otus.py -i $PWD/seqs.fna -r $PWD/refseqs.fna -o $PWD/otus_w_tax/ -t $PWD/taxa
##
## Pick OTUs and create an OTU table against a reference set of OTUs without adding taxonomy assignments
##  pick_closed_reference_otus.py -i $PWD/seqs.fna -r $PWD/refseqs.fna -o $PWD/otus/
##
## Pick OTUs, assign taxonomy, and create an OTU table against a reference set of OTUs using usearch_ref
##  pick_closed_reference_otus.py -i $PWD/seqs.fna -r $PWD/refseqs.fna -o $PWD/otus_usearch/ -p $PWD/use
##
## Pick OTUs using usearch_ref, assign taxonomy, and create an OTU table against a reference set of OTUs
##  pick_closed_reference_otus.py -i $PWD/seqs.fna -r $PWD/refseqs.fna -o $PWD/otus_usearch_ref/ -p $PWD
##
## Pick OTUs, assign taxonomy, and create an OTU table against a reference set of OTUs using sortmerna.
##  pick_closed_reference_otus.py -i $PWD/seqs.fna -r $PWD/refseqs.fna -o $PWD/otus_sortmerna/ -p $PWD/s
##
## Options:
##   --version             show program's version number and exit
##   -h, --help            show this help message and exit
##   -v, --verbose         Print information during execution -- useful for
##                         debugging [default: False]
##   -r REFERENCE_FP, --reference_fp=REFERENCE_FP
##                         The reference sequences [default:
##                         /macqiime/anaconda/lib/python2.7/site-packages/qiime_d
##                         efault_reference/gg_13_8_otus/rep_set/97_otus.fasta].
##                         NOTE: If you do not pass -r to this script, you will
##                         be using QIIME's default reference sequences. In this
##                         case, QIIME will copy the corresponding reference tree
##                         to the output directory. This is the tree that should
##                         be used to perform phylogenetic diversity analyses
##                         (e.g., with core_diversity_analyses.py).
##   -p PARAMETER_FP, --parameter_fp=PARAMETER_FP
##                         path to the parameter file, which specifies changes to
##                         the default behavior. See
##                         http://www.qiime.org/documentation/file_formats.html
##                         #qiime-parameters . [if omitted, default values will
##                         be used]
##   -t TAXONOMY_FP, --taxonomy_fp=TAXONOMY_FP
##                         the taxonomy map [default:
##                         /macqiime/anaconda/lib/python2.7/site-packages/qiime_d
##                         efault_reference/gg_13_8_otus/taxonomy/97_otu_taxonomy
```

```
##                           .txt]
##   -s, --assign_taxonomy
##                           Assign taxonomy to each sequence using
##                           assign_taxonomy.py (this will override --taxonomy_fp,
##                           if provided) [default: False]
##   -f, --force             Force overwrite of existing output directory (note:
##                           existing files in output_dir will not be removed)
##                           [default: none]
##   -w, --print_only        Print the commands but don't call them -- useful for
##                           debugging [default: False]
##   -a, --parallel          Run in parallel where available [default: False]
##   -O JOBS_TO_START, --jobs_to_start=JOBS_TO_START
##                           Number of jobs to start. NOTE: you must also pass -a
##                           to run in parallel, this defines the number of jobs to
##                           be started if and only if -a is passed [default: 1]
##   --suppress_taxonomy_assignment
##                           skip the taxonomy assignment step, resulting in an OTU
##                           table without taxonomy (this will override
##                           --taxonomy_fp and --assign_taxonomy, if provided)
##                           [default: False]
##
##   REQUIRED options:
##     The following options must be provided under all circumstances.
##
##     -i INPUT_FP, --input_fp=INPUT_FP
##                           the input sequences [REQUIRED]
##     -o OUTPUT_DIR, --output_dir=OUTPUT_DIR
##                           the output directory [REQUIRED]
```

`pick_closed_reference_otus.py` is a "workflow" script that runs other scripts.

Ask pick_closed_reference_otus.py to print the other commands it would run. The `-f` tells it to force overwriting the directory closedref, in case that directory is already there. The `-r` tells it where the reference sequences are. The `-t` tells it where the reference taxonomy map is.

```
pick_closed_reference_otus.py -i seqs.fna -o closedref -f -r ../ref/greengenes/97_otus.fasta -t ../ref/g
```

```
## pick_otus.py -i seqs.fna -o closedref/uclust_ref_picked_otus -r ../ref/greengenes/97_otus.fasta -m uc
## make_otu_table.py -i closedref/uclust_ref_picked_otus/seqs_otus.txt -t ../ref/greengenes/97_otu_taxon
```

Picking closed reference OTUs with QIIME

Now actually run the script. The `-v` tells it to be "verbose", printing updates as it runs. Use `time` to report how long it takes.

```
time pick_closed_reference_otus.py -i seqs.fna -o closedref -r ../ref/greengenes/97_otus.fasta -t ../ref
```

```
## Pick OTUs
## pick_otus.py -i seqs.fna -o closedref/uclust_ref_picked_otus -r ../ref/greengenes/97_otus.fasta -m uc
## Make OTU table
## make_otu_table.py -i closedref/uclust_ref_picked_otus/seqs_otus.txt -t ../ref/greengenes/97_otu_taxon
```

```
##
## real 0m33.782s
## user 0m28.957s
## sys  0m2.688s
```

Picking closed-reference OTUs with NINJA-OPS

Compare to NINJA-OPS. Installation instructions are at https://github.com/GabeAl/NINJA-OPS. Note NINJA-OPS is much faster.

```
time python /path/to/your/ninja/directory/bin/ninja.py -i seqs.fna -o ninja
```

```
## Ninja 1.3.1
## Ninja database directory is is/Users/danknights/Copy/research/ninja/src/databases/greengenes97
## Running Ninja filter...
## "/Users/danknights/Copy/research/ninja/src/bin/ninja_filter_mac" "seqs.fna" "/Users/danknights/Copy/
## Not using paired-end reads
## Performing NINJA replicon-denoising at 1 compacted reads.
## WARNING: Found 1 sequences with ambiguity (1 ambiguous bases).
## Number of sequences: 27389
## Total reads considered: 27388
##
## DONE SORTING.
## 18 Samples found.
## Ninja filter time: 0.0992071628571
## Running Bowtie...
## bowtie2-align-s --no-head -x /Users/danknights/Copy/research/ninja/src/databases/greengenes97/greeng
##
## Bowtie time: 4.18873381615
## Running Ninja parse...
## "/Users/danknights/Copy/research/ninja/src/bin/ninja_parse_filtered_mac" "/Users/danknights/Copy/docs
## Opened /Users/danknights/Copy/docs/teaching/2016-spring/MiCE 8992/repo/data/guerreronegro/ninja/ninja
## Total OTUs available: 99322
## Number of unique samples: 18, max reads: 14792
## Total reads expanded: 23141
## Run complete.
## Ninja parse time: 0.11067199707
##
## real 0m4.485s
## user 0m11.643s
## sys  0m0.471s
```

Find out how many sequences were assigned ("Total count") by uclust_ref (QIIME default) and how many with NINJA-OPS using the `biom summarize-table` command.

```
biom summarize-table -i closedref/otu_table.biom > closedref/stats.txt
head closedref/stats.txt
```

```
## Num samples: 18
## Num observations: 1750
## Total count: 22879
```

```
## Table density (fraction of non-zero values): 0.192
##
## Counts/sample summary:
##  Min: 950.0
##  Max: 1758.0
##  Median: 1262.000
##  Mean: 1271.056
```

```
biom summarize-table -i ninja/ninja_otutable.biom > ninja/stats.txt
head ninja/stats.txt
```

```
## Num samples: 18
## Num observations: 2130
## Total count: 23141
## Table density (fraction of non-zero values): 0.193
##
## Counts/sample summary:
##  Min: 960.0
##  Max: 1775.0
##  Median: 1276.000
##  Mean: 1285.611
```

Try NINJA-OPS on the Global Gut data set. It can process 1 million sequences in the Global Gut data set in under 20 seconds on a Macbook.

```
cd ../globalgut
time python /path/to/your/ninja/directory/bin/ninja.py -i seqs.fna -o ninja
```

```
## Ninja 1.3.1
## Ninja database directory is is/Users/danknights/Copy/research/ninja/src/databases/greengenes97
## Running Ninja filter...
## "/Users/danknights/Copy/research/ninja/src/bin/ninja_filter_mac" "seqs.fna" "/Users/danknights/Copy/
## Not using paired-end reads
## Performing NINJA replicon-denoising at 1 compacted reads.
## Number of sequences: 1000000
## Total reads considered: 1000000
##
## DONE SORTING.
## 489 Samples found.
## Ninja filter time: 1.96135497093
## Running Bowtie...
## bowtie2-align-s --no-head -x /Users/danknights/Copy/research/ninja/src/databases/greengenes97/greeng
##
## Bowtie time: 14.7388730049
## Running Ninja parse...
## "/Users/danknights/Copy/research/ninja/src/bin/ninja_parse_filtered_mac" "/Users/danknights/Copy/docs
## Opened /Users/danknights/Copy/docs/teaching/2016-spring/MiCE 8992/repo/data/globalgut/ninja/ninja_otu
## Total OTUs available: 99322
## Number of unique samples: 489, max reads: 117369
## Total reads expanded: 942750
## Run complete.
## Ninja parse time: 0.518316030502
```

```
##
## real 0m17.269s
## user 0m50.408s
## sys  0m1.259s
```

Picking de novo OTUs with QIIME

Run the workflow script, `pick_de_novo_otus.py` with `-w` to print the basic commands it would run.

```
cd ../guerreronegro
pick_de_novo_otus.py -i seqs.fna -o openref -f -w
```

```
## pick_otus.py -i seqs.fna -o openref/uclust_picked_otus
## pick_rep_set.py -i openref/uclust_picked_otus/seqs_otus.txt -f seqs.fna -l openref/rep_set//seqs_rep
## assign_taxonomy.py -o openref/uclust_assigned_taxonomy -i openref/rep_set//seqs_rep_set.fasta
## make_otu_table.py -i openref/uclust_picked_otus/seqs_otus.txt -t openref/uclust_assigned_taxonomy/se
## align_seqs.py -i openref/rep_set//seqs_rep_set.fasta -o openref/pynast_aligned_seqs
## filter_alignment.py -o openref/pynast_aligned_seqs -i openref/pynast_aligned_seqs/seqs_rep_set_align
## make_phylogeny.py -i openref/pynast_aligned_seqs/seqs_rep_set_aligned_pfiltered.fasta -o openref/rep
```

Now actually run the script. Note that the OTU picking step is quick on this data set. The other steps will take a long time (more than 10 minutes). So instead, kill the script by typing `ctrl-c`.

```
pick_de_novo_otus.py -i seqs.fna -o openref -f -v
```

Instead let's just run the pick_otus.py step, with several different OTU picking methods.

```
time pick_otus.py -i seqs.fna -o uclust -m uclust
make_otu_table.py -i uclust/seqs_otus.txt -o uclust/otu_table.biom
biom summarize-table -i uclust/otu_table.biom | head -n 5
```

```
##
## real 0m2.519s
## user 0m2.194s
## sys  0m0.267s
## Num samples: 18
## Num observations: 4074
## Total count: 27389
## Table density (fraction of non-zero values): 0.118
```

```
time pick_otus.py -i seqs.fna -o swarm -m swarm
make_otu_table.py -i swarm/seqs_otus.txt -o swarm/otu_table.biom
biom summarize-table -i swarm/otu_table.biom | head -n 5
```

```
## Traceback (most recent call last):
##   File "/macqiime/anaconda/bin/pick_otus.py", line 1004, in <module>
##     main()
##   File "/macqiime/anaconda/bin/pick_otus.py", line 995, in main
##     result_path=result_path, log_path=log_path)
##   File "/macqiime/anaconda/lib/python2.7/site-packages/qiime/pick_otus.py", line 774, in __call__
```

```
##     HALT_EXEC=False)
##   File "/macqiime/anaconda/lib/python2.7/site-packages/bfillings/swarm_v127.py", line 294, in swarm_c
##     clusters = swarm(seq_path)
##   File "/macqiime/anaconda/lib/python2.7/site-packages/bfillings/swarm_v127.py", line 77, in __call_
##     super(Swarm, self).__call__(seq_path)
##   File "/macqiime/anaconda/lib/python2.7/site-packages/burrito/util.py", line 285, in __call__
##     'StdErr:\n%s\n' % open(errfile).read())
## burrito.util.ApplicationError: Unacceptable application exit status: 1
## Command:
## cd "/Users/danknights/Copy/docs/teaching/2016-spring/MiCE 8992/repo/data/guerreronegro/"; swarm -d 1
## StdOut:
##
## StdErr:
## Swarm 1.2.19 [Jun  2 2015 14:40:16]
## Copyright (C) 2012-2014 Torbjorn Rognes and Frederic Mahe
## https://github.com/torognes/swarm
##
## Please cite: Mahe F, Rognes T, Quince C, de Vargas C, Dunthorn M (2014)
## Swarm: robust and fast clustering method for amplicon-based studies.
## PeerJ 2:e593 http://dx.doi.org/10.7717/peerj.593
##
## CPU features:      mmx sse sse2 sse3 ssse3 sse4.1 sse4.2 popcnt avx avx2
## Database file:     /tmp/SwarmExactMatchFiltert2lkor.fasta
## Output file:       /tmp/temp_otumap_2UK2rV.swarm
## Resolution (d):    1
## Threads:           1
## Algorithm:         regular
## Scores:            match: 5, mismatch: -4
## Gap penalties:     opening: 12, extension: 4
## Converted costs:   mismatch: 9, gap opening: 12, gap extension: 7
##
## Reading database:  0%Error: Illegal character 'N' in sequence on line 2
##
##
##
## real 0m1.372s
## user 0m1.114s
## sys  0m0.234s
## Error in make_otu_table.py: option -i: file does not exist: 'swarm/seqs_otus.txt'
##
## If you need help with QIIME, see:
## http://help.qiime.org
## Usage: biom summarize-table [options] {-i/--input-fp INPUT-FP}
##
## [] indicates optional input (order unimportant)
## {} indicates required input (order unimportant)
##
## Provides details on the observation counts per sample, including summary statistics, as well as metac
##
## Example usage:
## Print help message and exit
##  biom summarize-table -h
##
## Basic script usage: Write a summary of table.biom to table_summary.txt
```

```
##  biom summarize-table -i table.biom -o table_summary.txt
##
## biom summarize-table: error: option -i: file does not exist: 'swarm/otu_table.biom'
```

```
time pick_otus.py -i seqs.fna -o cdhit -m cdhit
make_otu_table.py -i cdhit/seqs_otus.txt -o cdhit/otu_table.biom
biom summarize-table -i cdhit/otu_table.biom | head -n 5
```

```
##
## real 0m29.098s
## user 0m27.270s
## sys  0m0.925s
## Num samples: 18
## Num observations: 4520
## Total count: 27389
## Table density (fraction of non-zero values): 0.121
```

```
time pick_otus.py -i seqs.fna -o sumaclust -m sumaclust
make_otu_table.py -i sumaclust/seqs_otus.txt -o sumaclust/otu_table.biom
biom summarize-table -i sumaclust/otu_table.biom | head -n 5
```

```
##
## real 1m1.659s
## user 0m59.526s
## sys  0m0.902s
## Traceback (most recent call last):
##   File "/macqiime/anaconda/bin/make_otu_table.py", line 119, in <module>
##     main()
##   File "/macqiime/anaconda/bin/make_otu_table.py", line 115, in main
##     write_biom_table(biom_otu_table, opts.output_biom_fp)
##   File "/macqiime/anaconda/lib/python2.7/site-packages/qiime/util.py", line 569, in write_biom_table
##     "Attempting to write an empty BIOM table to disk. "
## qiime.util.EmptyBIOMTableError: Attempting to write an empty BIOM table to disk. QIIME doesn't suppo
## Usage: biom summarize-table [options] {-i/--input-fp INPUT-FP}
##
## [] indicates optional input (order unimportant)
## {} indicates required input (order unimportant)
##
## Provides details on the observation counts per sample, including summary statistics, as well as meta
##
## Example usage:
## Print help message and exit
##  biom summarize-table -h
##
## Basic script usage: Write a summary of table.biom to table_summary.txt
##  biom summarize-table -i table.biom -o table_summary.txt
##
## biom summarize-table: error: option -i: file does not exist: 'sumaclust/otu_table.biom'
```

Customizing workflow scripts

We can customize workflow scripts using a parameter file. You will need some kind of text editor to edit the parameter file, such as Notepad or Notepad++ for Windows, TextWrangler or Sublime for Mac, or Emacs

or vi for the Mac/Linux command line. For more information, please see the official QIIME explaining parameter files here: http://qiime.org/documentation/qiime_parameters_files.html.

For example, let's assume we want to customize the OTU picking method used by (`pick_otus.py`) as part of the `pick_de_novo_otus.py` workflow, changing it from the default `uclust` to `swarm`. To find the parameter name, we can either run `pick_otus.py -h` or read the description at qiime.org: http://qiime.org/scripts/pick_otus.html (found by Googling **QIIME pick_otus.py**). We see that the parameter we need is called `otu_picking_method`, and that the valid options are `sortmerna, mothur, trie, uclust_ref, usearch, usearch_ref, blast, usearch61, usearch61_ref, sumaclust, swarm, prefix_suffix, cdhit, uclust`.

Therefore we need to add `pick_otus:otu_picking_method swarm` to a text file. This can be done with emacs as follows, if you have emacs on your system:

```
emacs parameters-swarm.txt
```

Then add these lines to the top of the file:

```
pick_otus:otu_picking_method swarm
pick_otus:similarity .99
```

Then save the file with `ctrl-x ctrl-s`, then exit with `ctrl-x ctrl-c`.

Now we can run the workflow script with the custom parameters file:

```
pick_de_novo_otus.py -i seqs.fna -o swarm99 -p parameters-swarm.txt
```

Making OTU tables human-readable

You can convert OTU tables to tab-delimited output using `biom convert`:

```
biom convert -i closedref/otu_table.biom -o closedref/otu_table.txt --to-tsv
```

The output file can now be opened in Excel.

Follow-up exercises

1. How can you use `filter_otus_from_otu_table.py` to determine the number of singleton OTUs produced by each OTU picking method?
2. How many OTUs remain from each method after filtering singletons?
3. Run OTU picking on another data set from EBI or QIITA.
4. Try (loading OTU tables into R)[]