TIC

TAXONOMY INFORMED CLUSTERING

Documentation

Version Alpha

Antonios Kioukis

November 15, 2021

Contents

1 Introduction

In 16S rRNA sequence-based diversity analysis, a common practice is the clustering of the sequences based on similarity cutoffs to obtain groups reflecting molecular species, genera or families. Due to the size of the available data, greedy algorithms are preferred for their time efficiency. Such algorithms rely only on pairwise similarities and tend to cluster together sequences with diverse phylogenetic background. Taxonomic classifiers use position specific taxonomic information in assigning probable taxonomy to a given sequence. We developed a tool that uses classifier assigned taxonomy to restrict the clustering among sequences that share the same taxonomic path. We tested "Taxonomy Informed Clustering" with the sequences from SILVA release 138* and show that TIC outperforms greedy algorithms like Usearch and Vsearch in terms of clusters purity and entropy**. We offer a complete and automated pipeline for use in diversity analysis context, based on this concept, and a pipeline for 16S rRNA amplicon dataset processing. This implementation first process raw reads down to denoised amplicons, taxonomically classify them and apply TIC to cluster them further to sOTUs, gOTUs and fOTUs. The resulting tables offer more accurate insights at different evolutionary levels views that will be useful in microbiome research.

2 Organization

TIC-Pipeline is composed of 4 steps that can be run independently or as a set.

1. Samples-Processing
2. Taxonomy-Classification
3. Taxonomy-Informed-Clustering
4. Results_Reporting

Running them in the given order simplifies the process as the outputs of each step are often the inputs of the next. There is also an extra folder where the original data is recommended to be placed to keep the analysis of one study in a compact and organized structure. Inside the Original-Data folder are contained the template files that served as basis for the TIC-Pipeline presentation that can be used for training purposes. Before running any script, please make sure you have read and fully understood the corresponding section of the documentation for each step.

3 Options Reference

3.1 Samples-Processing

SAMPLES_PROCESS_STEP:
Flag (YES/NO) enables the step.

USER_FASTQ_FOLDER:
the full path of the FASTQ files.

TRIM_SCORE:
the minimum quality score of the FASTQ read last position [3-20] [recommended 20]

MAXDIFF:
This is the maximum mismatches during merging of reads allowed.

MINPCTID:
Minimum %identity of alignment. Default 90. Consider decreasing if you have long overlaps.

MINMERGELEN:
200 or 380 or 250 the minimun length allowed after pairing of reads.

MAXMERGELEN:
260 or 440 or 310 the maximun length allowed after pairing of reads.

FORWARD_TRIM:
length of trimming at the forward side of a read [5-25][recommended 10].

REVERSE_TRIM:
length of trimming at the reverse side of a read [5-25][recommended 10].

EXPECTED_ERROR_RATE:
the maximum rate of expected errors allowed in the assemblied paired end reads 0.01 is 1% .

This is the first step of the TIC-Pipeline. It produces denoised sequencess from your FASTQ files. The FASTQ files should be in the directory specified by the option USER_FASTQ_FOLDER. In the same directory, a 'mapping_file.ssv' must also exist. This file is space-seperated and contains the information of which FASTQ files are paired, and what sample they should be assigned to. Please refer to the next table for a short example (the header line is only for comprehension, and should not be included):

Sample Name	Paired Flag	Forward File	Reverse File
SRR2127221	2	SRR2127221_1.fastq	SRR2127221_2.fastq
ERR8971239	1	ERR8971239_1.fastq	

The ASV-Creation step contains the quality and chimera filtering, trimming and error identification part of the TIC-Pipeline. We propose that you trim the bases on the FASTQ file that are of lower than 20 on the quality score (TRIM_SCORE option). The options for the merging of reads are self-explanatory. You can specify the maximum mismatches during the merging of reads and also the percentage of identity of alignment (MAXDIFF and MINPCTID respectively) After the merging of reads, you can control which reads are either too short to be included in the downstream analyses or too long with the options: MINMERGELEN and MAXMERGELEN. After the reads are produced, the pipeline continues by trimming the forward and reverse side of each read, this is done principally to remove primers. The EXPECTED_ERROR_RATE option sets the maximum rate of expected errors allowed in the assemblied paired end reads. All these stages are run by vsearch v2. [2].

3.2 Taxonomy-Classification

ALIGNMENT_CLASSIFICATION_STEP:
Flag (YES/NO) enables the step.

INPUT_FASTA_ALI_CLASS:
Provide the full path of the TIC-Pipeline. Where the non-chimeric ZOTUs should be at 1.ASV-Creation/good_ZOTUS.fa

OUTPUT_FASTA_ALI_CLASS:
Provide the full path of the TIC-Pipeline. Where the sina-aligned ZOTUs will be.

PDF_REGION_OUTPUT:
The calculation of the SINA regions in order to view it and select which region you want to extract for the clustering process.

EXTRACTION_STEP:
Flag (YES/NO) enables the step.

INPUT_FASTA_EXTRACTION:
/home/antonios/TIC-Pipeline/sina_output.fasta

EXTRACTION_REGION_START:
start of the extraction region from the sina alignment. [0-49999].

EXTRACTION_REGION_END:
end of the extraction region from the sina alignment. [0-49999].

EXTRACTION_REGION_LIMIT:
the minimum number of bases that should be present in the sina aligned region selected, so the sequence is outputed. This number is multiplied by

0.80 to be more generous so keep this in mind.

OUTPUT_FASTA_EXTRACTION:
Provide the full path of the extracted region ASVs with taxonomic information FASTA file.

Description:

ZOTU_CREATION_STEP:
Flag (YES/NO) enables the step.

MIN_ZOTU_SIZE:
The minimum size of a ZOTU to be counted (trades speed for sensitivity)

Description:

This is the second step of the TIC-Pipeline. Every produced ASV is aligned and classified using SINA [5] and the SILVA ARB file as reference. Each ASV is aligned to 50.000 positions and is classified up to the GENUS taxonomic level with the majority voting rule implemented in SINA, first find the 10 closest neighbors of the input sequence in the ARB file and then vote which is the lowest taxonomic rank that the majority of the neighbors' taxonomy agrees. This step also includes a sub-step that sums for each of the 50.000 position how many bases are aligned in that position. This vector is then plotted by the R script 'ggplot_alignment_vector.R' and creates a PDF output in the 'PDF_REGION_OUTPUT' option, that helps you identify which region should be extracted for the next step and how many bases should be at least aligned in that region so that the ASV is included in your clustering 1.

The first time you run this command, it will produce an index of the ARB file. This process is slow but it happens only this one time and is expected by the pipeline.

The paired end reads are checked for chimeras using the SortMeRNA tool [3] with the SILVA bacteria and archaea databases used as references.

For producing the ASVs from the non-chimeric reads you only have to

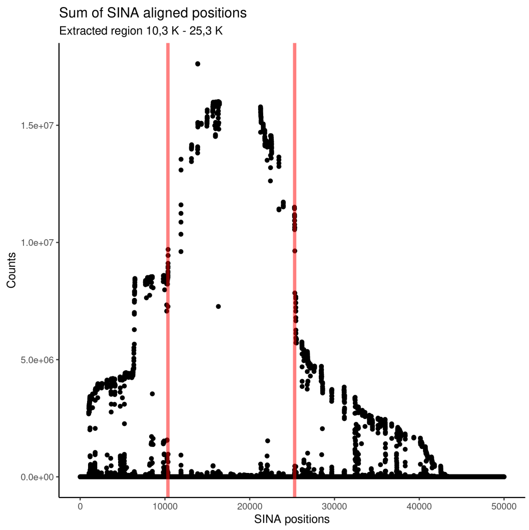


Figure 1: Sum of bases aligned in each position.

specify the MIN_ZOTU_SIZE. Any ZOTU that contains less than this number is discarded. Please keep in mind, when specifying this option that smaller values than the default 4 makes the ASVs more sensitive but a lot slower

3.3 Taxonomy-Informed-Clustering

TAXONOMIC_CLUSTERING_STEP:
Flag (YES/NO) enables the step.

CLUSTERING_DIRECTORY:
Full path directory to save the results of the taxonomic split and taxonomic clustering.

INPUT_FASTA_CLUSTERING:
Full path directory of the FASTA with taxonomic classifications which will be used as input of the step.

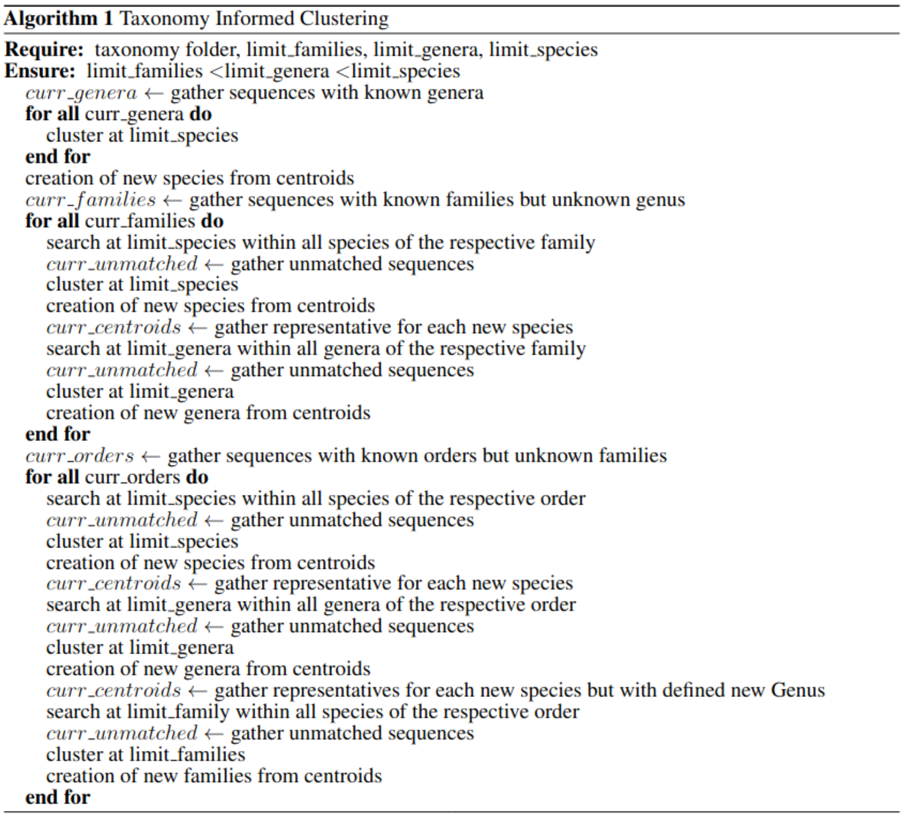
FAMILY_IDENTITY:
Percentage identity on the Family level. [0-GENERA_IDENTITY].

GENERA_IDENTITY:

Percentage identity on the GENERA level. [FAMILY_IDENTITY-SPECIES_IDENTITY].

SPECIES_IDENTITY:

Percentage identity on the GENERA level. [0-100].



```
Algorithm 1 Taxonomy Informed Clustering
Require: taxonomy folder, limit_families, limit_genera, limit_species
Ensure: limit_families <limit_genera <limit_species
  curr_genera ← gather sequences with known genera
  for all curr_genera do
    cluster at limit_species
  end for
  creation of new species from centroids
  curr_families ← gather sequences with known families but unknown genus
  for all curr_families do
    search at limit_species within all species of the respective family
    curr_unmatched ← gather unmatched sequences
    cluster at limit_species
    creation of new species from centroids
    curr_centroids ← gather representative for each new species
    search at limit_genera within all genera of the respective family
    curr_unmatched ← gather unmatched sequences
    cluster at limit_genera
    creation of new genera from centroids
  end for
  curr_orders ← gather sequences with known orders but unknown families
  for all curr_orders do
    search at limit_species within all species of the respective order
    curr_unmatched ← gather unmatched sequences
    cluster at limit_species
    creation of new species from centroids
    curr_centroids ← gather representative for each new species
    search at limit_genera within all genera of the respective order
    curr_unmatched ← gather unmatched sequences
    cluster at limit_genera
    creation of new genera from centroids
    curr_centroids ← gather representatives for each new species but with defined new Genus
    search at limit_family within all species of the respective order
    curr_unmatched ← gather unmatched sequences
    cluster at limit_families
    creation of new families from centroids
  end for
```

Figure 2: Pseudocode of the TIC algorithm.

Description:

This is the third step of the TIC-Pipeline. Usearch when clustering does not take into account differences in the importance or mutation rates of bases. It simply calculates how similar are the sequence as a percentage and if it is greater than a specific threshold the sequences are pooled together or are split. By following the TIC-Pipeline Each ASV now has a designated taxonomy. To avoid the above caveat of usearch, we split the ASVs into seperate files bases on their last known taxonomy level. This counteracts some greediness of usearch by clustering smaller parts of the ASVs alone rather than as a whole. Additionally our complex taxonomically informed clustering, tries to identify misses from the sina classification process. Each ASV that misses one or multiple taxonomic levels is searched against the sequences included in the taxonomic clade of the tree, and only if it does not match with any of those sequences, it creates new families, genera etc 3.

Since there is no specific percentage similarity cutoff for the differentiation of Phyla, Classes and Orders, TIC only produces new families, genera and species and if higher taxonomic levels are missing it pools them together in the UNKPHULUM, UNKCLASS and UNKORDER, respectively.

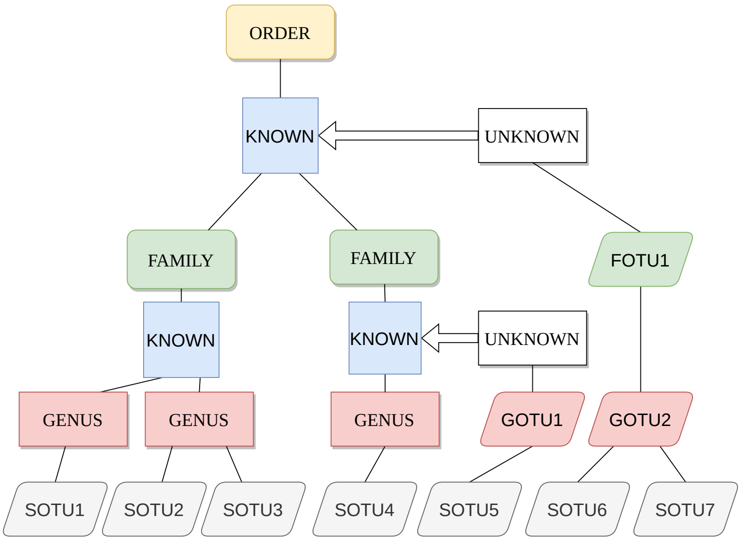


Figure 3: High level abstraction of the TIC algorithm.

3.4 Results-Processing

RESULTS_CREATION_STEP:
Flag (YES/NO) enables the step.

OUTPUT_FOLDER:
Full path of the directory where all results will be saved.

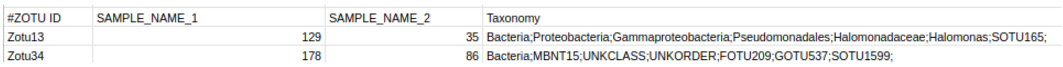
OUTPUT_ZOTU_FASTA_WITH_TAXONOMY:
Name of the output FASTA file containing the ASVs along with their full taxonomic information.

OUTPUT_ZOTU_TABLE:
Name of the output TAB file containing the ASVs along with their full taxonomic information and number of reads in each sample of the study.

OUTPUT_SOTU_FASTA_WITH_TAXONOMY:
Name of the output FASTA file containing the SOTU along with their full taxonomic information.

Description:

This is the forth and final step of the TIC-Pipeline. In this step, we gather all outputs produced in the previous process and try to create simple graphical representations of your data. All outputs are placed in the new folder specified by OUTPUT_FOLDER in the config_options.txt The denoised sequences are assigned to the full taxonomies produced from TIC and written to the file specified by OUTPUT_ZOTU_FASTA_WITH_TAXONOMY. The OUTPUT_ZOTU_TABLE contains the number of reads of each sample in your study as well as their full taxonomy 4. The ”sotu_sizes.tab” contains information answers the question how many denoised sequences are part of each SOTU. You can find each denoised sequence is mapped to what sotu by using the ”denoised_map_sotu.tab”. ”sotus_with_taxonomy.tab” quantifies how many reads of each input sample are present in each denoised sequence. The centroid sequences for each SOTU are written in the file specified by OUTPUT_SOTU_FASTA_WITH_TAXONOMY option in the configuration file. Mapping the relationship between species, genera and families is important to the end-users. The files ”sotus_to_gotus_map.tab”, ”gotus_to_fotus_map.tab” detail those ties in TAB-seperated format. A graphlan taxonomic tree [1] as well as a Krona [4] plot are created and written in the output folder. All red clades in the taxonomic tree represent novel families, orders etc. 5.



#ZOTU ID	SAMPLE_NAME_1	SAMPLE_NAME_2	Taxonomy
Zotu13	129	35	Bacteria;Proteobacteria;Gammaproteobacteria;Pseudomonadales;Halomonadaceae;Halomonas;SOTU165;
Zotu34	178	86	Bacteria;MBNT15;UNKCLASS;UNKORDER;FOTU209;GOTU537;SOTU1599;

Figure 4: Two lines from the ASV tab file.

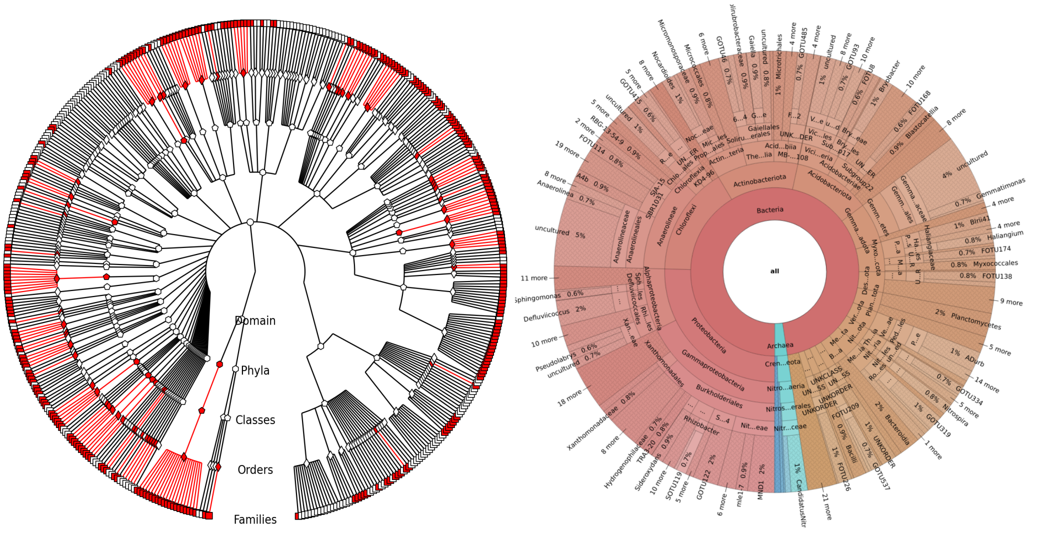


Figure 5: Plots produced from TIC-Pipeline. **(A)** Graphlan plot depicting the taxonomic tree of the denoised sequences after TIC incorporating both novel(red) and known(black) clades up to the family level. **(B)** Krona plot quantifying the size of each taxonomy in the merged study samples. Contains novel and known taxonomies as produced by sina classifier and TIC.

Bibliography

[1] F. Asnicar, G. Weingart, T. L. Tickle, C. Huttenhower, and N. Segata. Compact graphical representation of phylogenetic data and metadata with graphlan. *PeerJ*, 3:e1029, 2015.

[2] R. C. Edgar. Search and clustering orders of magnitude faster than blast. *Bioinformatics*, 26(19):2460–2461, 2010.

[3] E. Kopylova, L. Noé, and H. Touzet. Sortmerna: fast and accurate filtering of ribosomal rnas in metatranscriptomic data. *Bioinformatics*, 28(24): 3211–3217, 2012.

[4] B. D. Ondov, N. H. Bergman, and A. M. Phillippy. Interactive metagenomic visualization in a web browser. *BMC bioinformatics*, 12(1):1–10, 2011.

[5] E. Pruesse, J. Peplies, and F. O. Glöckner. Sina: accurate high-throughput multiple sequence alignment of ribosomal rna genes. *Bioinformatics*, 28 (14):1823–1829, 2012.