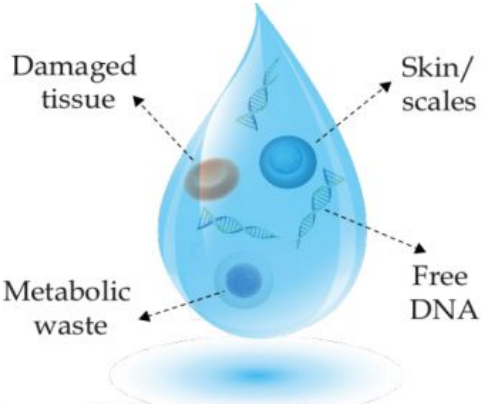
P.E.M.A.

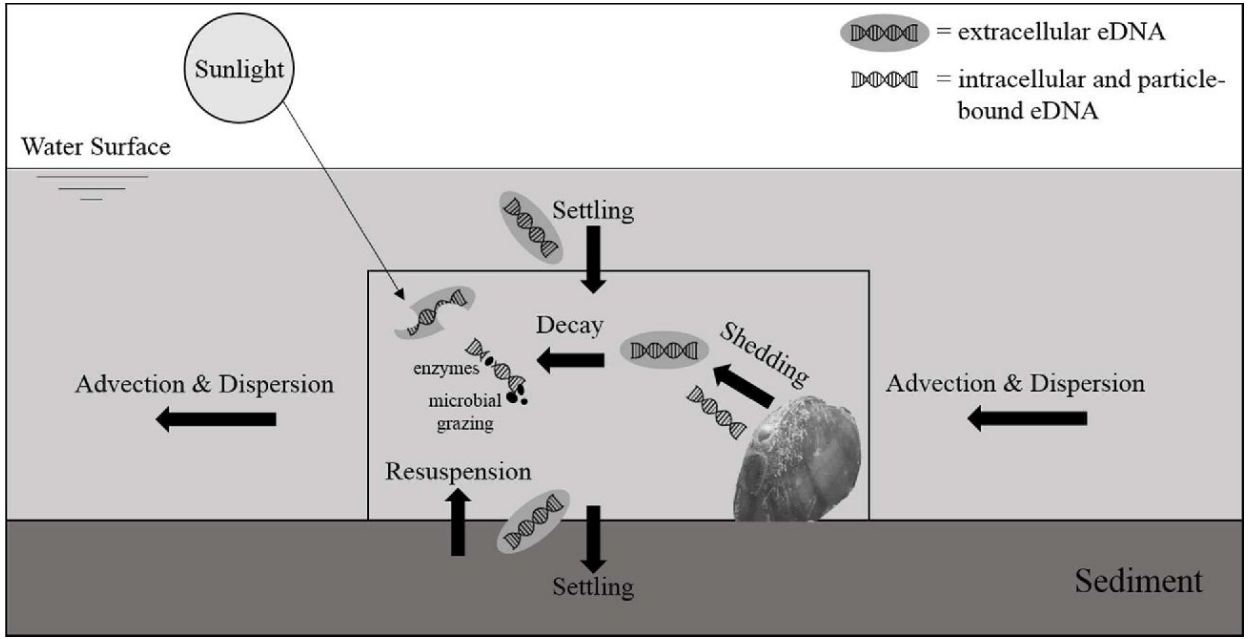
a pipeline for eDNA metabarcoding analysis

By:
Haris Zafeiropoulos

ink well by Daria Moskvina from the Noun Project

environmental DNA (eDNA)





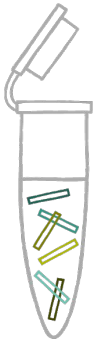
Metabarcoding

is a relatively recent approach that focuses on the simultaneous detection of a large number of species, that uses universal PCR primers to identify DNA from a mixture of organisms.

1 › 2 › 3 › 4

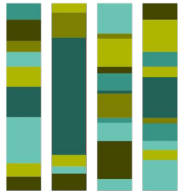
Environmental samples

DNA extraction



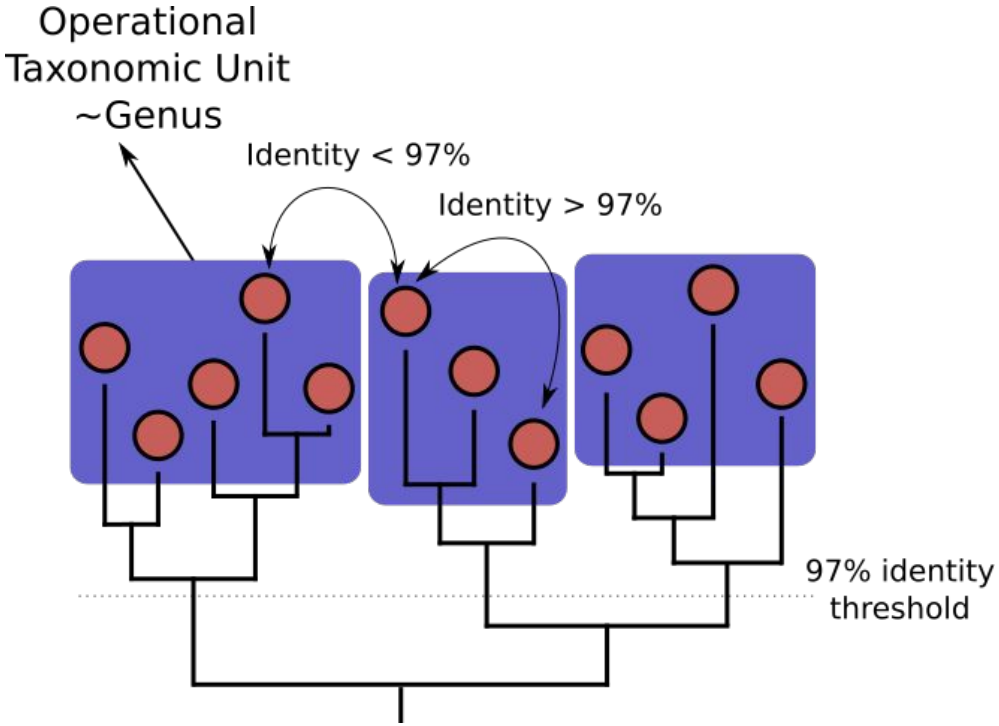
16S amplification

Sequencing

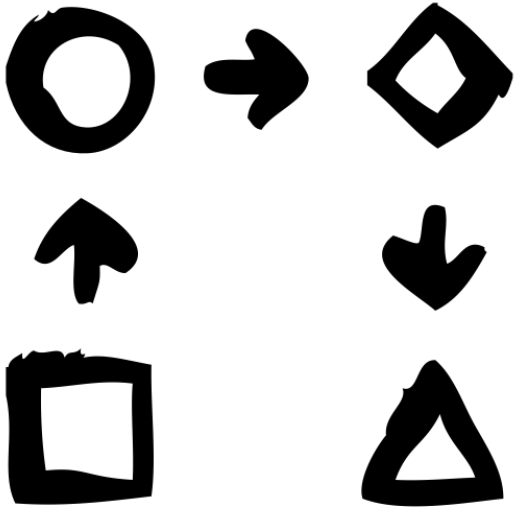


Results
(microbial diversity, etc.)

(M)OTU: splitting the world into.. (s)p(ea)[ie]c(i)es



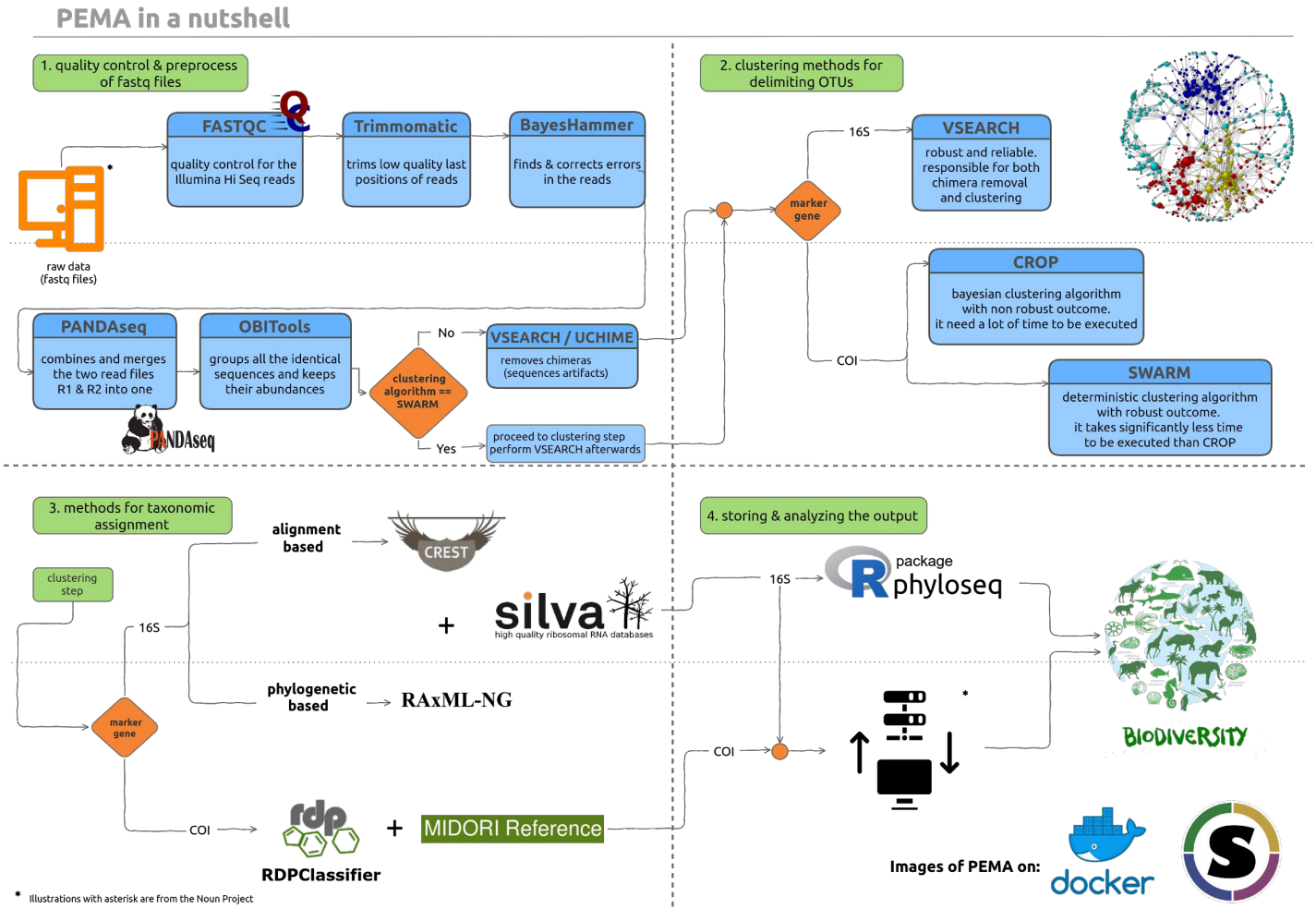
P.E.M.A: a Pipeline for Environmental DNA Metabarcoding Analysis



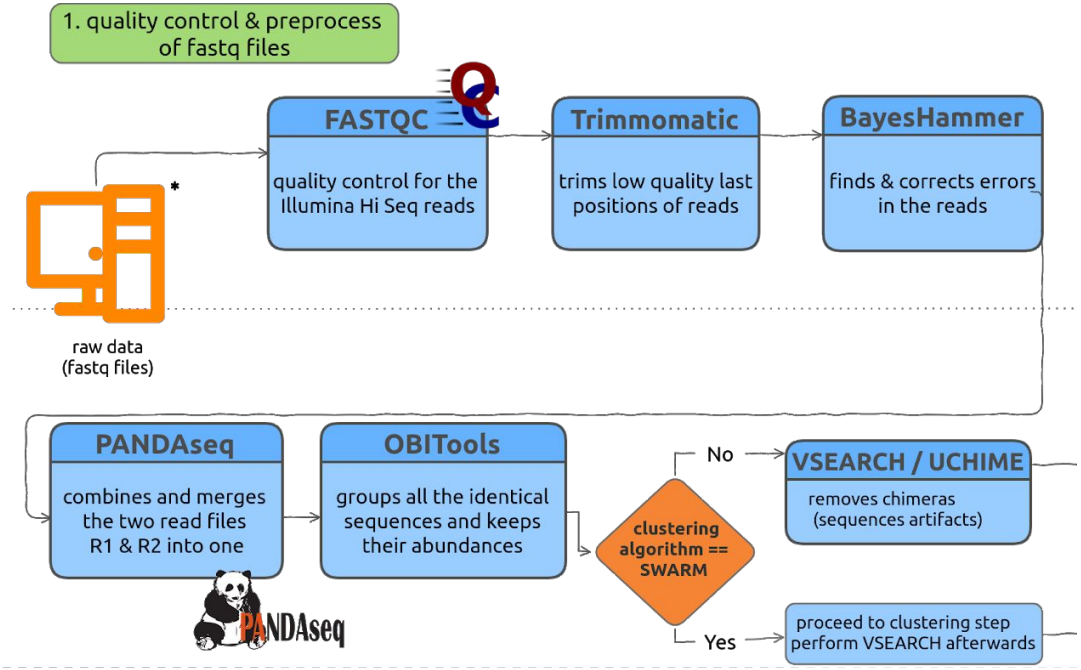
From the Illumina sequencer output to the analysis of eDNA samples to make conclusions about the ecosystem they came from, **a series of tasks** need take place

Aim of Part II is the building of a pipeline that implements all these tasks automatically, both for the 16S marker gene and COI case

Millions of reads: what could be done with so many of them?

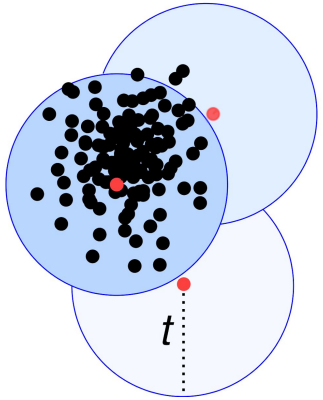


Step 1:
Quality control and pre-processing of raw data



Step 2:
Clustering (M)OTUs

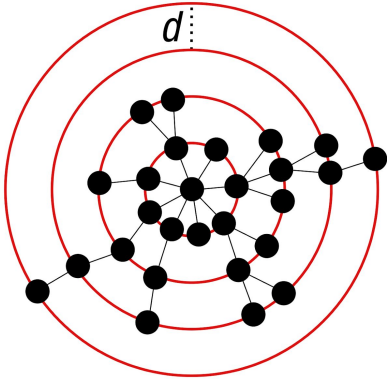
Part II



16S

VSEARCH algorithm

Swarm v2



CROP

COI

$$P(A|B) = \frac{P(B|A) * P(A)}{P(B)}$$

Step 3:
Taxonomic assignment

16S

Alignment based approach

CREST
Classification Resources for Environmental Sequence Tags

Phylogeny based approach

EPA-ng

+

silva
high quality ribosomal RNA databases

COI

Alignment based approach

rdp

+

MIDORI Reference

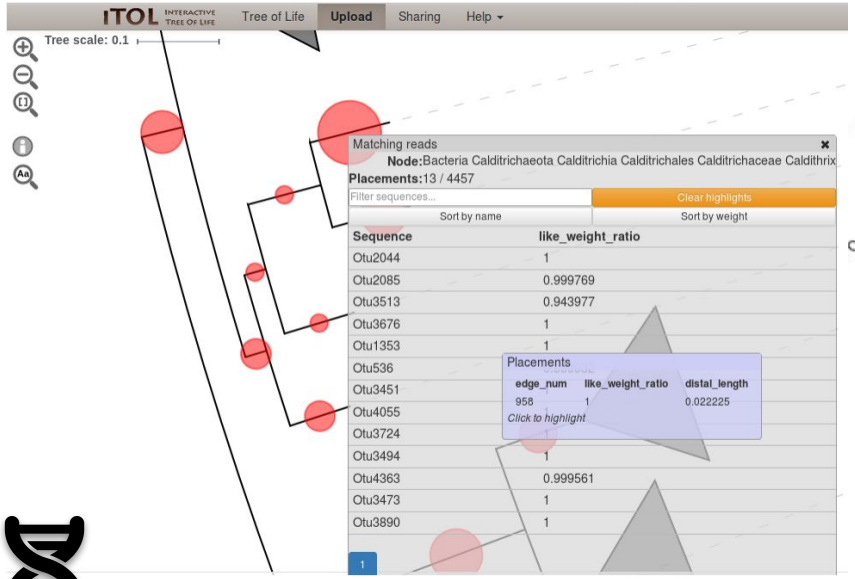
Parameters' file

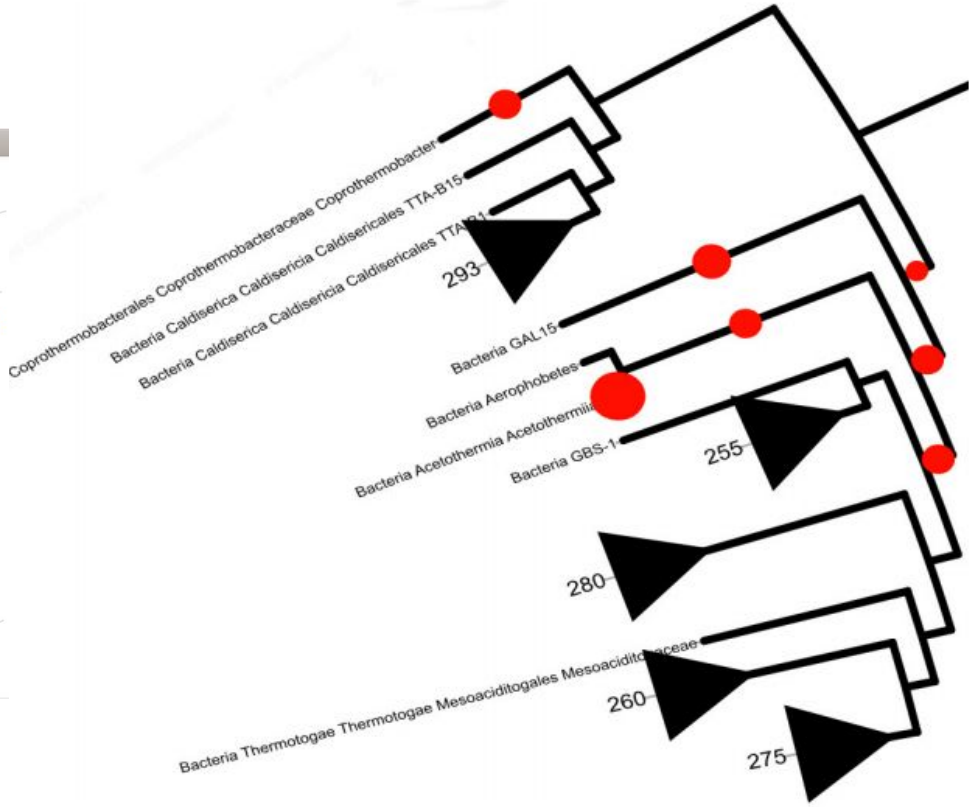
```
#############################################################################################
##############################   P.E.M.A. 's PARAMETERS   ###################################
#############################################################################################
#
#
# In this file there are all the parameters that NEED TO BE ASSIGNED every time you need PEMA to run!
# That does not mean that these parameters that we have her,  are the only  parameters of the tools PEMA uses!
# As you already may know, the combinations are infinite!
# Hence, we encourage you the most to study the manual of each tool and make them as good as possible for your SPECIFIC experiment.
# In each and every variable that you see quotes ( '' ) you have to write inside those the option you choose
# The rest, you need to complete them with a number and never add quotes.
# We chose to have a set of parameters for some tools (mainly for Trimmomatic) as 'by default'. That is because in some cases, plenty of time is required
# in order to have a correct set of those. In the link next to each tool, you can find further information about its parameters.
# YOU NEED TO BE REALLY CAREFUL WHEN YOU FILL THIS FILE !!
#  From each variable you have to leave EXACTLY FOUR (4) BLANKS and then fill the parameter as you wish.
#
## just to test that the parameters are assigned right in our main script:
#
sources my_parameters_work_just_fine!
#
## give in your each uniq experiment a NAME, so a single output file will be created for each of them
outputFile      final_COI_d_10
#
#############################################################
#####################   blastqc    ##########################
#############################################################
#
## no parameters here!
#
#
#############################################################
#################### trimmomatic   ##########################          //      http://www.usadellab.org/cms/uploads/supplementary/Trimmomatic/TrimmomaticManual_V0.32
.pdf
#############################################################
#
## Performs an adaptive quality trim, balancing the benefits
## of retaining longer reads against the costs of retaining
## bases with errors.
## can be either 'Yes' or 'No'
maxInfo Y
#
#
############ for MAXINFO ####################
## Specifies the read length which is likely to allow the
## location of the read within the target sequence to be determined
targetLength    200
#
```

OTU-table for the case of 16S marker gene

OTU	ERR1906855	ERR1906853	ERR1906863	ERR1906856	ERR1906857	ERR1906859	ERR1906858	ERR1906870	ERR1906854	ERR1906867 E
RR1906866	ERR1906865	ERR1906861	ERR1906862	ERR1906864	ERR1906869	ERR1906860	ERR1906868	classification		
Otu4056 8	4	0	0	8	3	6	0	3	0	0
0	0	0	0	0	2	0	Main genome;			
Bacteria;Bacteria (superkingdom);Proteobacteria (superphylum);Proteobacteria;Gammaproteobacteria;Vibrionales;Vibrionaceae;Vibrio										
Otu4057 23	5	1	6	8	0	13	0	2	0	0
0	0	3	0	0	0	0	Main genome;			
Bacteria;Bacteria (superkingdom);FCB group;Caldithrix phylum incertae sedis;Caldithrix class incertae sedis;Caldithrix order incertae sedis;Caldithrix family incert										
ae sedis;Unknown Caldithrix family incertae sedis genus 4										
Otu4054 0	0	0	1	2	0	12	0	0	0	0
0	0	0	0	0	0	0	Main genome;			
Bacteria;Bacteria (superkingdom);Fusobacteria (superphylum);Fusobacteria;Fusobacteriia;Fusobacteriales										
Otu4055 0	1	0	4	9	0	9	0	0	1	0
0	0	0	0	0	0	0	Main genome;			
Bacteria;Bacteria (superkingdom);Proteobacteria (superphylum);Proteobacteria;Deltaproteobacteria;Myxococcales;Unknown Myxococcales family 16										
Otu4052 1	3	2	2	1	0	5	0	1	0	0
0	1	2	0	0	3	0	Main genome;			
Bacteria;Bacteria (superkingdom);Proteobacteria (superphylum);Proteobacteria;Deltaproteobacteria;Myxococcales;VHS-B4-70										
Otu4053 0	0	0	1	4	0	11	0	0	0	13
0	0	0	0	0	0	0	Main genome;			
Bacteria;Bacteria (superkingdom);CPR;Ca. Parcubacteria;Candidatus Falkowbacteria										
Otu4050 5	3	1	5	2	0	9	0	3	0	0
0	0	0	0	0	0	0	Main genome;			
Bacteria;Bacteria (superkingdom);FCB group;Bacteroidetes;Sphingobacteriia;Sphingobacteriales;Unknown Sphingobacteriales family 25										
Otu4051 0	0	0	5	6	0	14	0	0	0	1
0	8	0	1	0	9	0	Main genome;			
Bacteria;Bacteria (superkingdom);Proteobacteria (superphylum);Proteobacteria;Deltaproteobacteria;Desulfobacterales;Desulfobulbaceae;Desulfobulbus										
Otu4058 2	6	0	15	12	12	9	3	4	11	8
2	4	0	0	0	0	2	Main genome;			

MOTUs placements after phylogeny based taxonomic assignment





https://github.com/hariszaf/pema

https://hub.docker.com/r/hariszaf/pema/

hariszaf / pema

Watch 0 | Star 0 | Fork 0

Code | Issues 0 | Pull requests 0 | Projects 0 | Wiki | Insights | Settings

P.E.M.A.: a Pipeline for Environmental DNA Metabarcoding Analysis for the 16S and COI marker genes

Edit

Manage topics

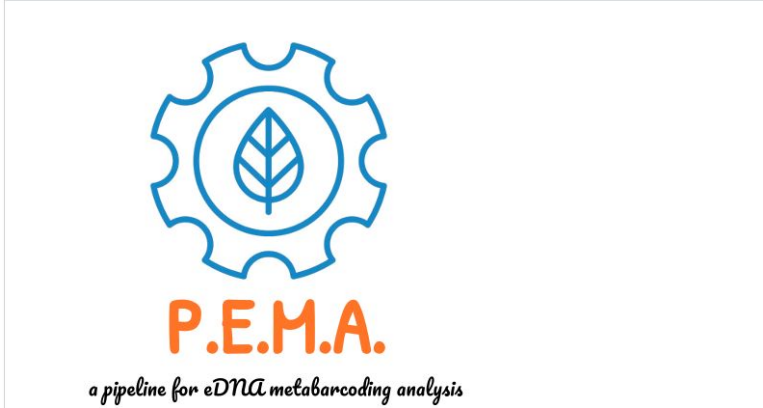
35 commits | 1 branch | 0 releases | 1 contributor

Branch: master | New pull request | Create new file | Upload files | Find file | Clone or download

hariszaf Update README.md — Latest commit f7662ed 8 days ago

README.md	Update README.md
parameters_docker.tsv	Update parameters_docker.tsv

README.md



hariszaf/pema — Explore | Help | Sign up | Sign in

PUBLIC REPOSITORY

hariszaf/pema ☆

Last pushed: 8 days ago

Repo Info | Tags

Short Description

P.E.M.A.: a Pipeline for Environmental DNA Metabarcoding Analysis for the 16S and COI marker genes

Docker Pull Command

```
docker pull hariszaf/pema
```

Full Description

P.E.M.A. is a pipeline for two marker genes, 16S rRNA (microbes) and COI (eukaryotes). As input, P.E.M.A. accepts fastq files as returned by Illumina sequencing platforms. P.E.M.A. processes the reads from each sample and returns an OTU-table with the taxonomies of the organisms found and their abundances in each sample. It also returns statistics and a FASTQC diagram about the quality of the reads for each sample. Finally, in the case of 16S, P.E.M.A. returns alpha and beta diversities, and make correlations between samples. The last step is facilitated by Rhea, a set of R scripts for downstream 16S amplicon analysis of microbial profiles.

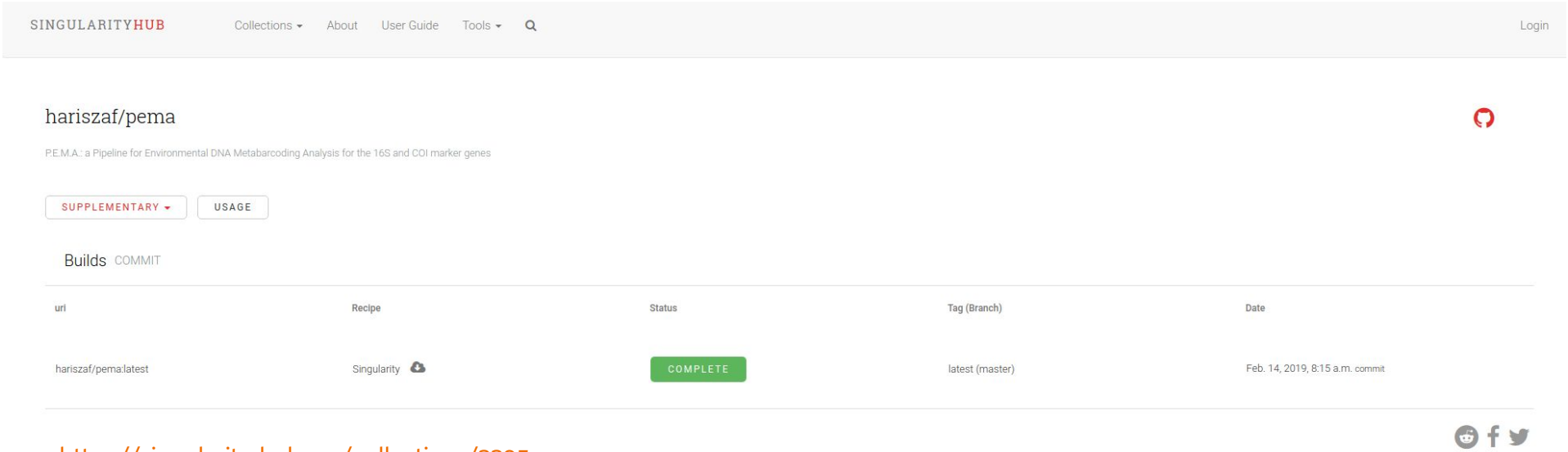
In the COI case, two clustering algorithms can be performed by P.E.M.A. (CROP and SWARM), while in the 16S, two approaches for taxonomy assignment are supported: alignment- and phylogenetic-based. For the latter, a reference tree with 1000 taxa was created using SILVA_132_SSURef, EPA-ng and RaxML-ng.

For more information about how to use P.E.M.A. please see our github repository - https://github.com/hariszaf/pema

Owner

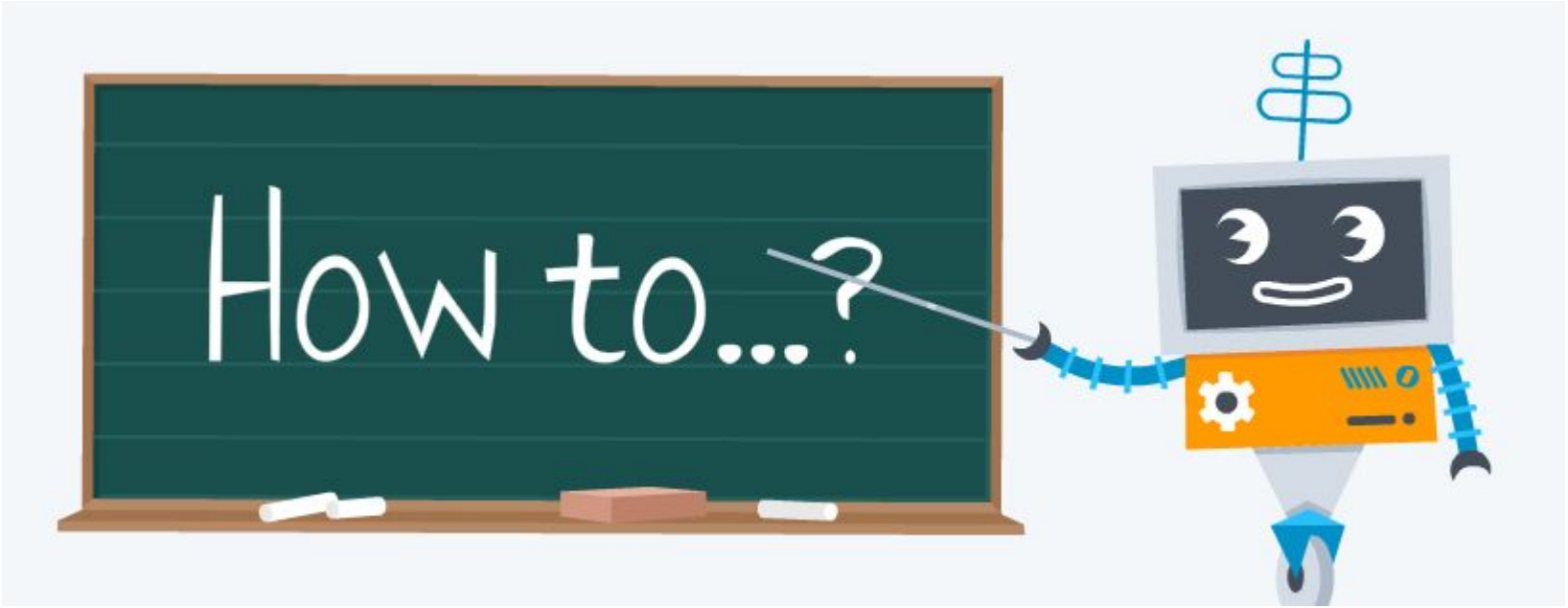
hariszaf

Three commands and a few minutes away!



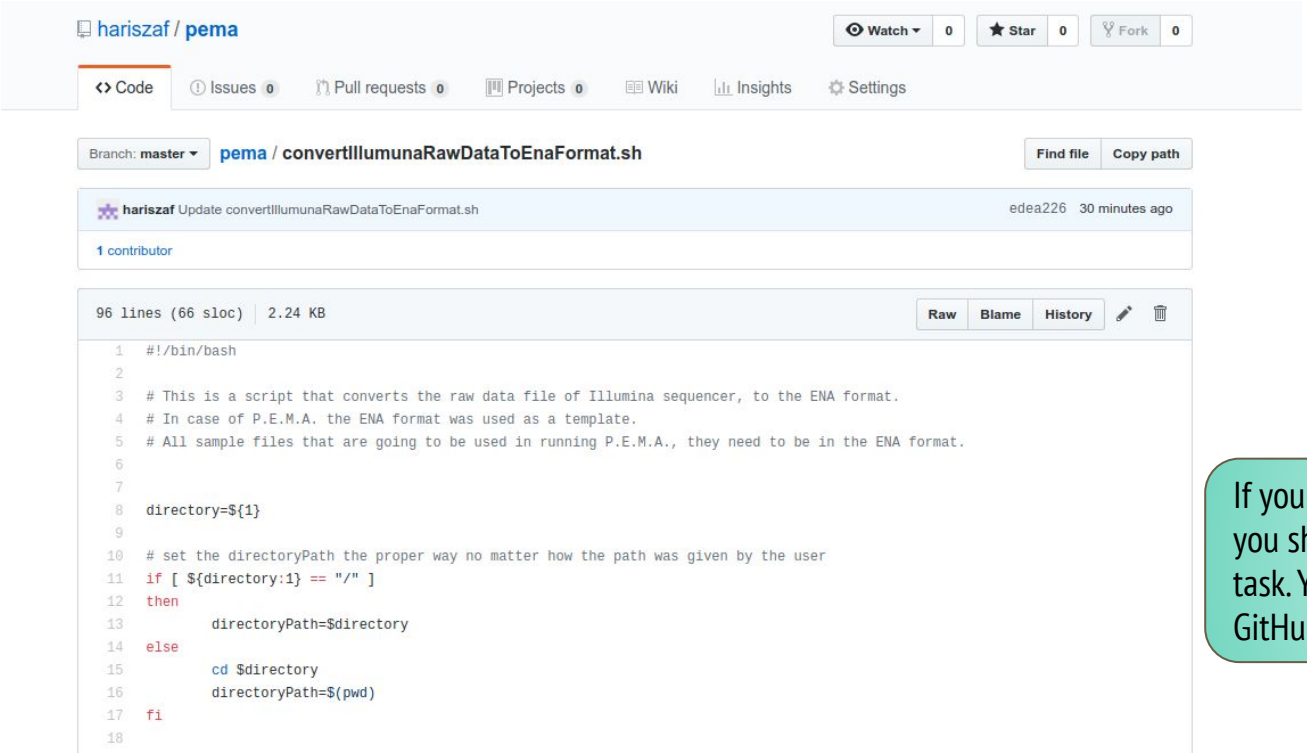
https://singularity-hub.org/collections/2295

Finally..



run P.E.M.A. ?

P.E.M.A. and input format!



If your raw data are not in ENA format, then you should use a script we have made for this task. You can find it in “pema” repository on GitHub.

On GitHub you can get all the info you need. However, let's do it together!

Step 1

Make sure that you have Singularity on your HPC environment:

```
$singularity --version
```

All version ≧2.6 are ok. In case you don't have Singularity, ask your IT group to do this for you.

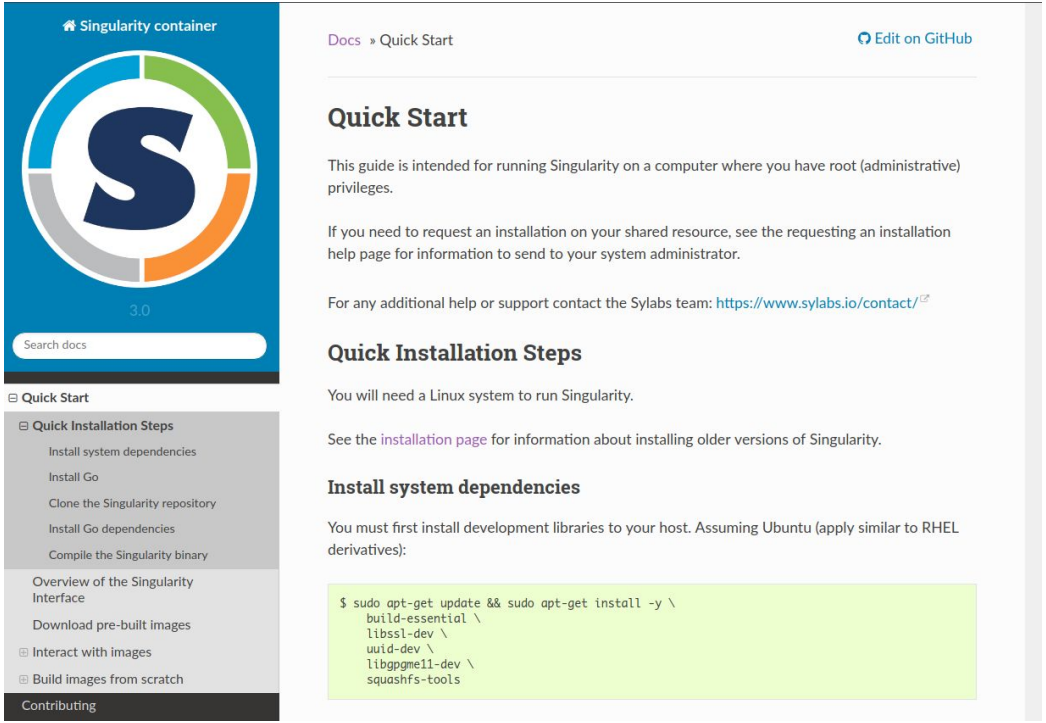
Get a P.E.M.A. container using this command:

```
singularity pull shub://hariszaf/pema
```

You are free to save it wherever on your home folder if you want to.
We suggest to rename it to "***pema.simg***". To do so, you just need to run this command

```
$mv hariszaf-pema-master-latest.simg pema.simg
```

Some info for your sys admin



You can get Singularity 3.0 from here.

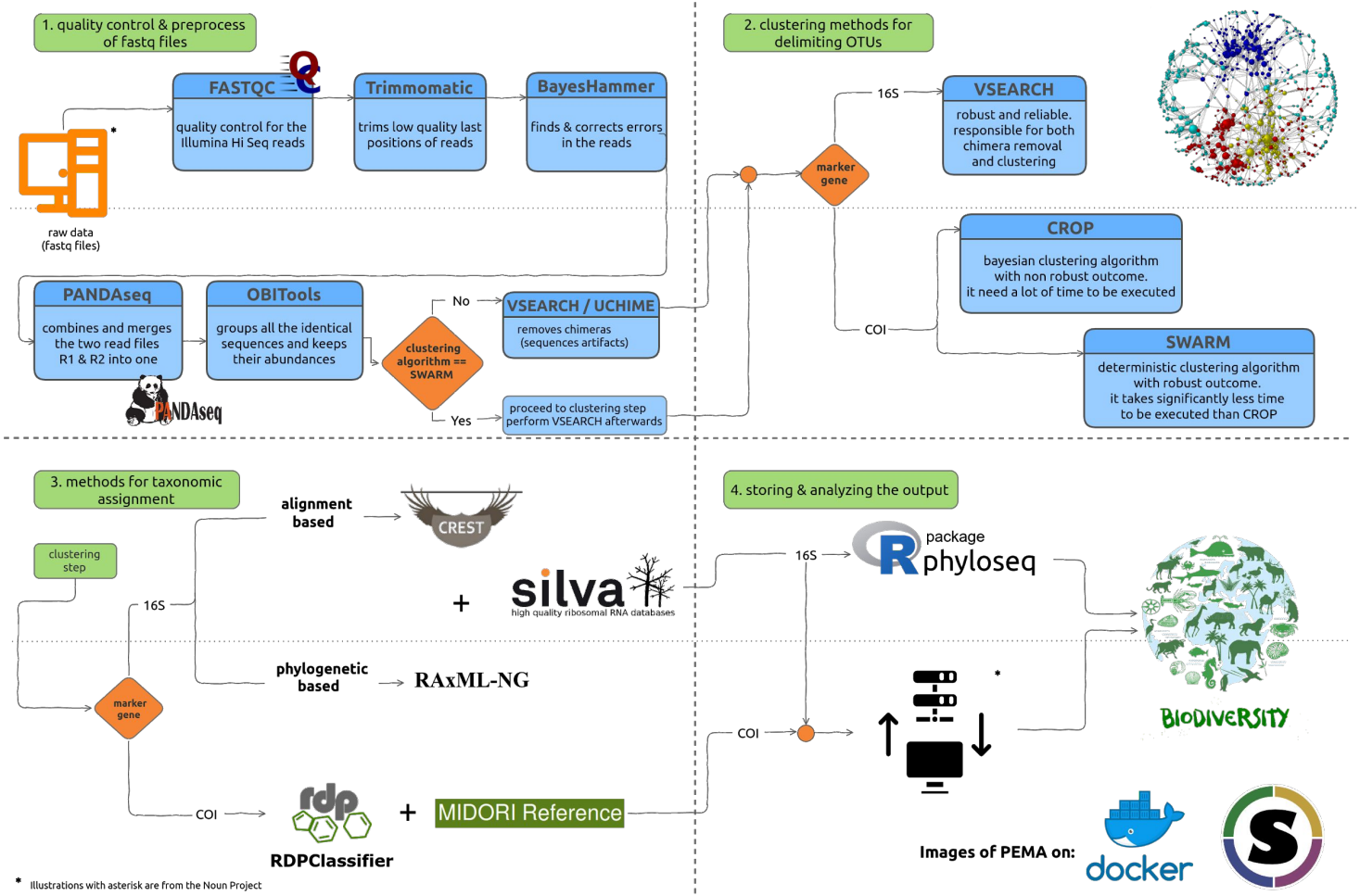
Singularity allows container - based technology to be merged with HPC.

However, it can also be used for servers or any other computer.

Please sys - admin, do not panic!
Singularity is just an app!!

PEMA in a nutshell

If it was not the Singularity, you would have to install everything you see on this image!



On GitHub you can get all the info you need. However, let's do it together!

- Create a folder for your analysis. Let's call it **"analysis folder"**.

  ```
  mkdir analysis_folder
  ```

Step 2

- Download the ***parameters.tsv*** file from github and then make a copy of that on your analysis folder using this command:

  ```
  scp -P <port_number> /<path_on_your_pc>parameters.tsv <user>@<cluster_ip>://<path_to_your_>analysis_folder
  ```

  Otherwise, you can download it from github directly on the server with a command equivalent to wget and then adjust:

  ```
  Wget https://raw.githubusercontent.com/hariszaf/pema/master/parameters.tsv
  ```

- Add your raw data in a subfolder of the "analysis folder" which it has to be called **"mydata"**.

```
scp -P <port_number> /<path_to_rawdata/* <user>@<cluster_ip>://<path_to_your_>/analysis_folder/  mydata
```

In the end, you need to have something like this:

```
haris@zorba:~/metabar_pipeline/PEMASingularity/testFile$ ls
mydata  parameters.tsv
```

mydata is a subfolder containing **only** your raw data and it needs always to be called like that

Here is the dataset we will work with

In `/home1/haris/mikroviokosmos/analysis_folder/mydata`
you can find the raw data we will use for this tutorial.

3.147.175 sequences from 7 samples
(this is only a part from the initial dataset, in order to be on time)

These samples are from the *Pavloudi et al.* (2017) dataset and are published in ENA.
"*Sediment microbial taxonomic and functional diversity in a natural salinity gradient challenge Remane's "species minimum" concept.*" is the publication that was the result of this study.

Step 3

Create your sbatch file: **touch pema_job.sh**
and edit it using any editor of your choice

```
#!/bin/bash

#SBATCH --partition=<specify_the_cluster_partition_where_your_job_will_be_submitted>
#SBATCH --nodes=<number_of_nodes_your_job_will_allocate>
#SBATCH --nodelist=<optional_specify_the_nodes_that_will_be_allocated>
#SBATCH --ntasks-per-node=<maximum_ntasks_be_invoked_on_each_core>
#SBATCH --mem=<optional_memory_per_node_specification>
#SBATCH --job-name="<give_a_name_to_your_job>"
#SBATCH --output=myanalysis.output
#SBATCH --mail-user=<put_here_your_emai_adress>
#SBATCH --mail-type=ALL
#SBATCH --requeue

singularity run -B /<path_of_your_analysis_folder/ :/mnt
/<path_to_your_PEMA_container>/ pema.simg
```

This is your **sbatch file,** in case that your HPC uses **SLURM**, that allows you to submit a job to a cluster. If you are using another cluster management and job scheduling system, your sbatch will have a different format.

Here is an example.

```
#!/bin/bash

#SBATCH --partition=batch
#SBATCH --nodes=1
#SBATCH --nodelist=node-12
#SBATCH --ntasks-per-node=20
#SBATCH --mem=
#SBATCH --job-name="PEMA_tutorial"
#SBATCH --output=myPEMAanalysis.output
#SBATCH --mail-user=haris.zafr@gmail.com
#SBATCH --mail-type=ALL
#SBATCH --requeue
```


```
singularity run -B /home1/haris/mikroviokosmos/analysis_folder/:/mnt
/home1/haris/mikroviokosmos/pema.simg
```

And what if there is no cluster?

```
singularity run -B /<path_of_your_analysis_folder/:/mnt
/<path_to_your_PEMA_container>/pema.simg
```

For Singularity there is no need for a cluster!

It is going to run exactly in the same way as before, but this time there is no need of the “sbatch file”.

A simple .sh script will do the job!

Photo by: Emmanouela Panteri, IMBBC

Step 3

Run this command:

```
sbatch pema_job.sh
```

and your job has now be submitted.

The **sbatch** command is again a **SLURM** command and if your HPC uses something else you need to do submit your job the way it does so.

Then run:

```
squeue
```

to see that your analysis is **up and running!**

Special thanks to:

Evangelos Pafilis (for being the brain)

IMBBC HPC group (for being the muscle)

Christina Pavloudi (for being the brain and the muscle)

Ha Quoc Viet (for the patience)

Office No 40 (for being the best and for the coffee!) and George the.. photoshop-er

Thank you for your attention!

Lab42 Open

IMBBC
INSTITUTE OF MARINE BIOLOGY, BIOTECHNOLOGY AND AQUACULTURE

hcmr
ΕΛΚΕΘΕ

Some extras...

About how PEMA and its algos actually work!

Making phylogeny based assignment possible

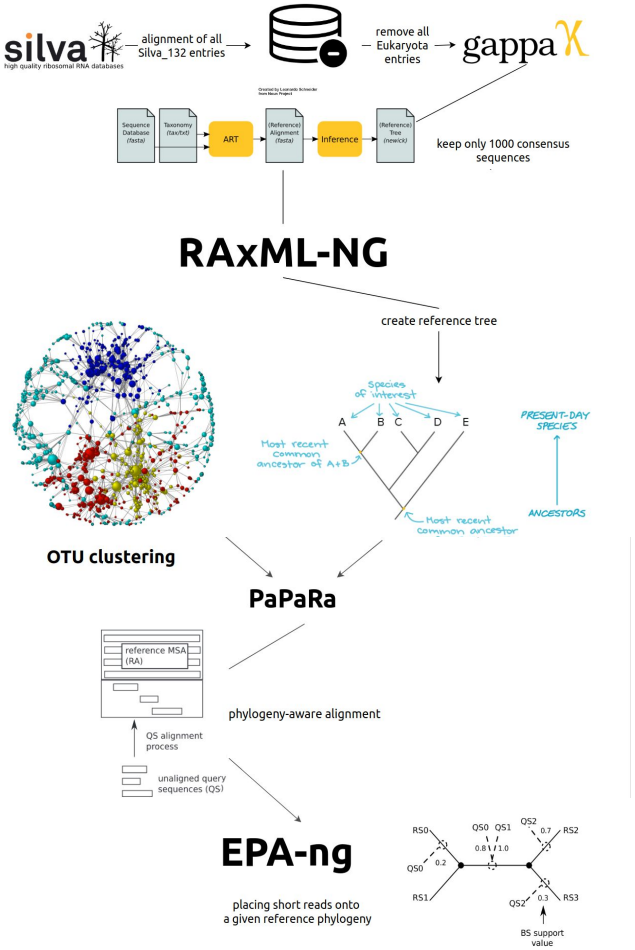
- Gathering all Bacteria and Archaea sequences from the alignment of **SILVA_132_SSURef** database, removing Eukaryota.

- Using **art** algorithm to keep only **1.000** “consensus” sequences that retain as much of the diversity of the initial sequence dataset **(reference multiple sequence alignment)**.

- Using **RAxML-ng** to build the **reference tree**.

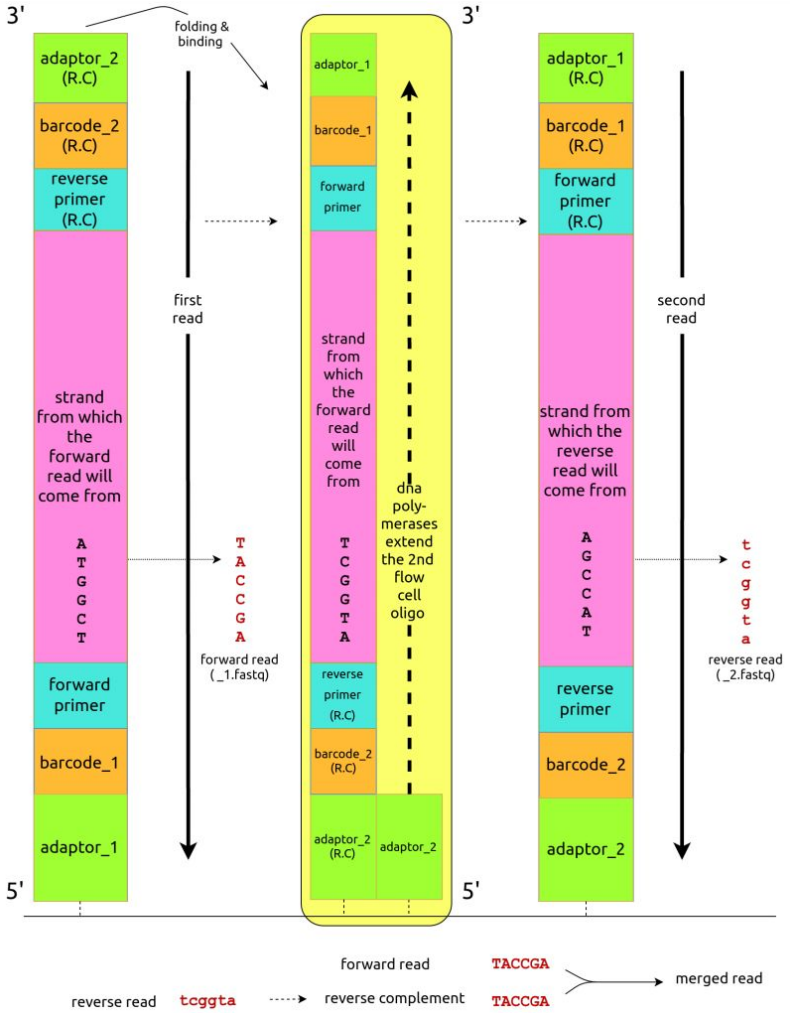
- With **PaPaRa**, a multiple sequence alignment using both the reference and the query is built and then manually, only the **query** part of PaPaRa ‘s output is kept.

- Using **Evolutionary Placement Algorithm - next generation** the queries are placed on the reference tree.



Illumina sequencing for paired-end samples

extras



PANDAseq - PAired-eND Assembler for Illumina sequences

extras

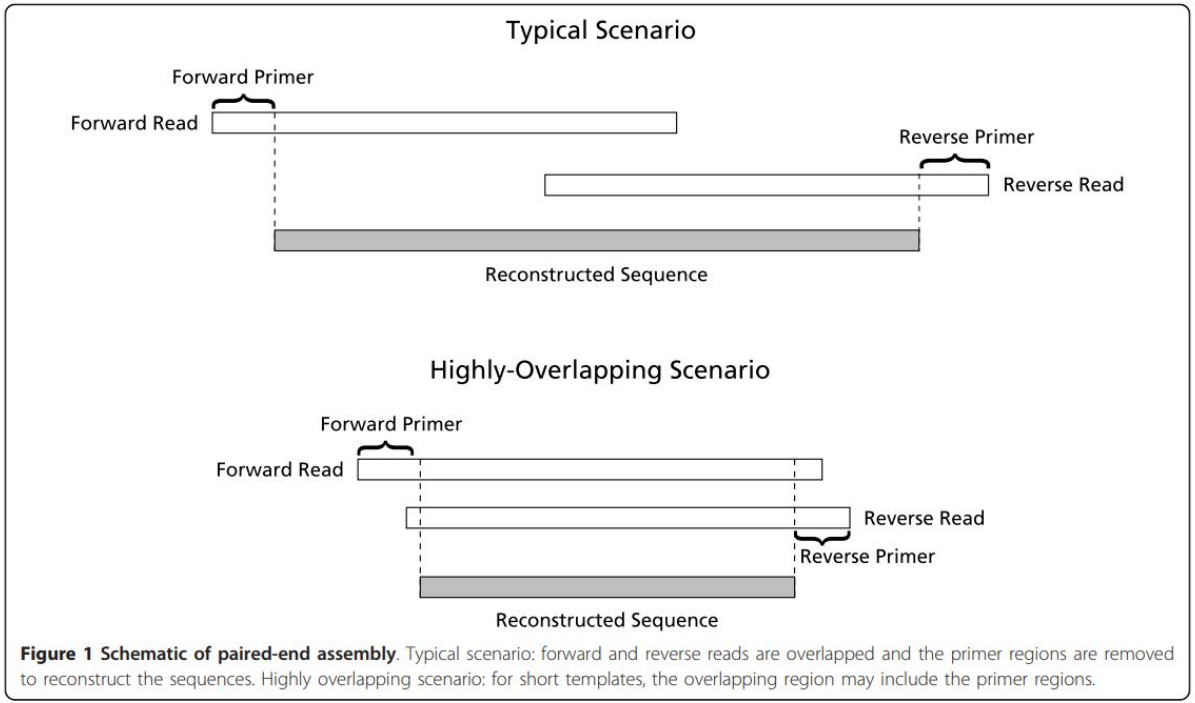


Figure 1 Schematic of paired-end assembly. Typical scenario: forward and reverse reads are overlapped and the primer regions are removed to reconstruct the sequences. Highly overlapping scenario: for short templates, the overlapping region may include the primer regions.

Bayes-Hammer

extras

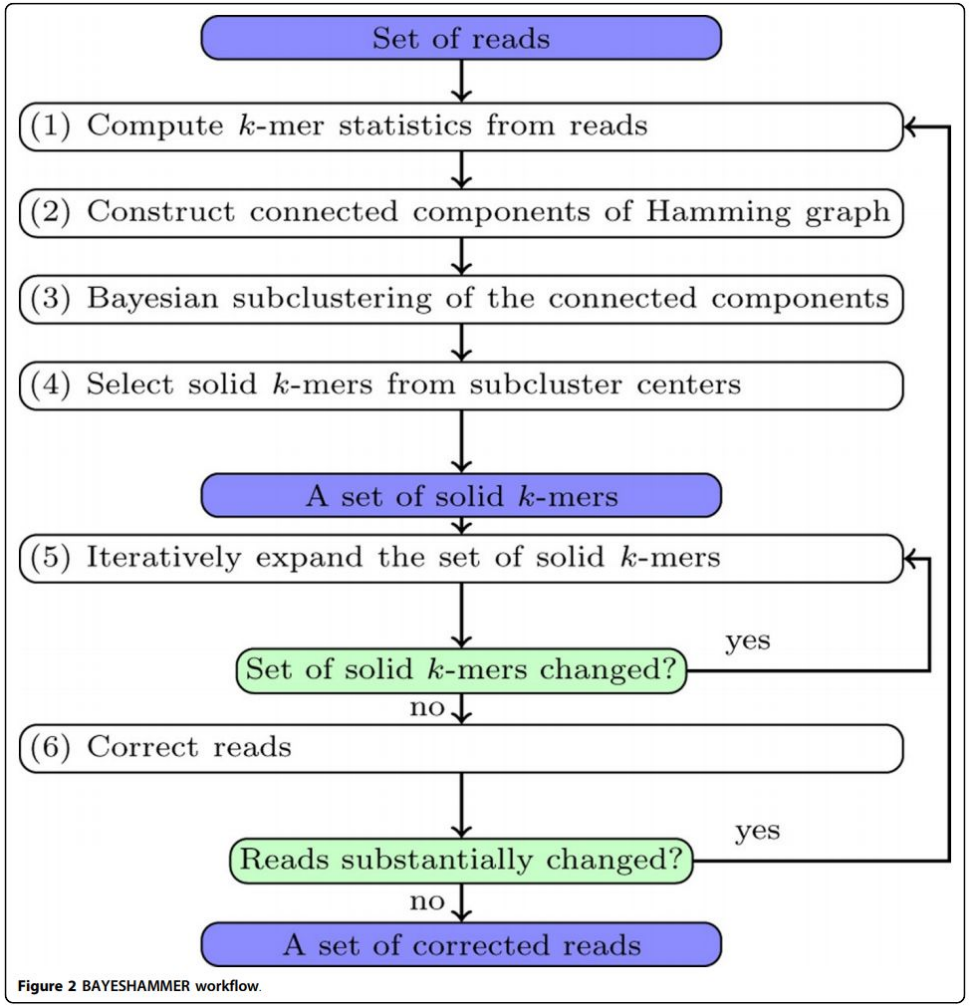
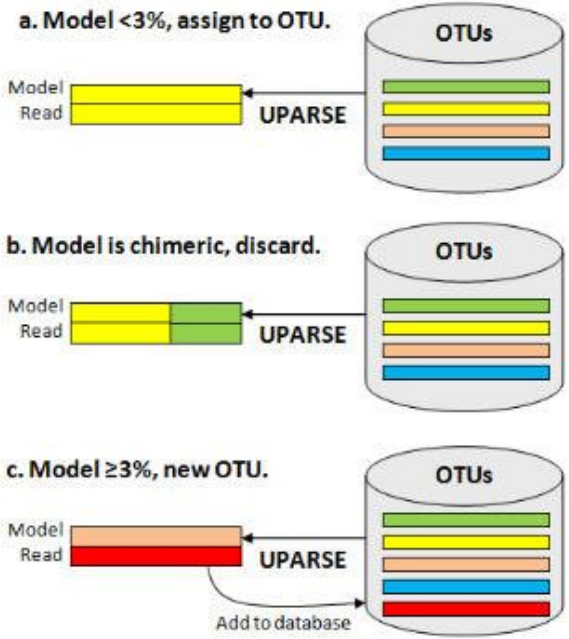
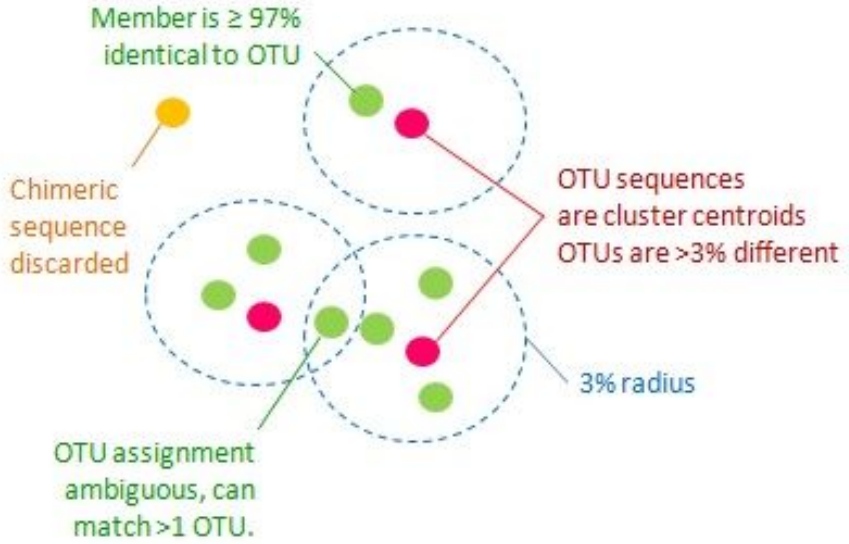


Figure 2 BAYESHAMMER workflow.

USEARCH - UPARSE algorithm

extras





Swarm V2.

extras

Let us consider a nucleic sequence S made of As, Cs, Gs and Ts. A “microvariant” is a sequence with one difference (d = 1) to the original sequence S. How many distinct microvariants can derive from S? In a sequence S of length L, each position can be substituted with 3 different bases, so there are 3L possible microvariants due to substitutions. Each position in S can be deleted once, so there are L possible microvariants due to deletions. Insertions are more complicated. An insertion can happen before or after each position in the sequence S, and four different nucleotides can be inserted resulting in 4(L + 1) microvariants. However, some insertions will result in the same microvariant: for example, inserting a “G” before or after a “G” will result in the same sequence “GG.” As that situation arises for all positions in S but one, the maximum number of distinct insertions is not 4(L + 1), but 3(L + 1) + 1 = 3L + 4. In total, the maximum number of microvariants that can be obtained from a given sequence S of length L is 3L + L + 3L + 4 = 7L + 4.

As stated above, different sequence modifications can produce the same microvariant. The final number of distinct microvariants depends on the number of homopolymer stretches in the sequence. **The number of distinct microvariants that can be obtained from a sequence S of length L then varies from 6L + 5 to 7L + 4.**

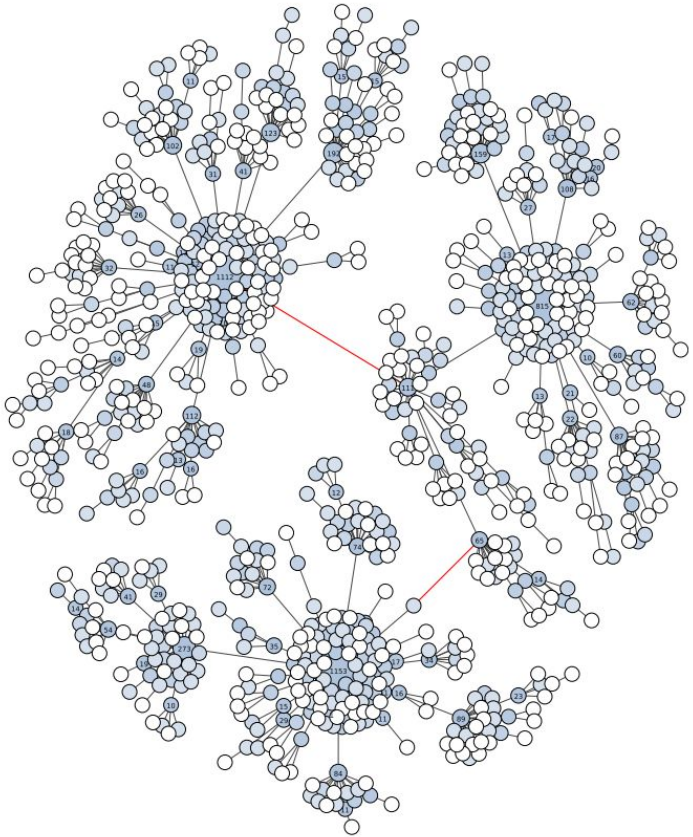


Figure 3 **Graphical representation of an OTU produced by Swarm (breaking and grafting phases deactivated) when clustering the BioMarKs 18S rRNA V4 dataset (amplicons are appr. 380 bp in length).** Nodes represent amplicons. Node size, color and text annotations represent the abundance of each amplicon. Edges represent one difference (substitution, deletion or insertion); the length of the edges carries no information. The red-colored edges indicate where Swarm’s breaking phase cuts when it is not deactivated, resulting into three high abundant OTUs, each being assigned to a different taxa of Cnidaria (Metazoa).