DADA2 and Phyloseq pipelines

Description

This pipeline has been written from the DADA2 script developped by Francois Keck and the files conversion for Phyloseq of the ANF MetaBioDiv workshop.

It uses the reference database manager for R to access the rbcL 312 database for DADA2, so no local database is needed.

Other elements have been written from the official DADA2 documentation and official tutorial.

Directories

The `DADA2.R` and `Phyloseq.R` scripts used by the pipeline are located in the `scripts` directory, and paths are set up to run them from this location. The raw sequencing reads should be stored in the `data` directory. The pipeline will create a `results` and a `plots` directories to store the outputs.

If you are working with **RStudio**, the default working directory might not be the `scripts` directory, and might prevent the pipeline to work. The method to fix this problem is detailed in the *How to run the pipeline?* tutorial, point number 5.

How to run the pipeline?

1. Download the scripts

a. Windows operating system

If you are working in a Windows environment the pipeline can be downloaded on your local computer using the Github repository. Go to the Github page and press the **<> Code** button located at the top of the screen. You can then copy/paste the directories and files downloaded in the folder of your choice.

b. UNIX-based environment (Mac OS or Linux)

On a BASH shell similar to the linux terminal used to access a remote server, you can use *Git* to install the pipeline.
To install *Git*, run:

```
conda install anaconda::git
```

Then install the pipeline with:

```
git clone git@github.com:ThibauldMichel/DADA2_Phyloseq.git
```

c. Directories

In the pipeline you will find a `scripts` directory countaining two R scripts: `DADA2.R` and `Phyloseq.R`. We will run them in this order: DADA2 to identify species in our samples, and Phyloseq to plot the results. Phyloseq will be using the outputs of DADA2 located in the `results` directory as inputs.

2. Install dependancies

The scripts will install R dependancies needed by the pipeline. However, a recent version of **cutadapt** is needed. Check the cutadapt website for installation instructions.

a. Windows operating system

The path to the cutadapt executable should be provided between double quotes line 55 of the `scripts/DADA2.R` file as follow:

```
cutadapt <- "C:/path/to/cutadapt/executable"
```

A simple method to do this is to download the Github executable *cutadapt.exe* located here and to provide the link to this file line 55 of the `scripts/DADA2.R` file. By default it is the following line, where you whould replace *usr* by your user name.

```
cutadapt <- "C:/Users/usr/Downloads/cutadapt.exe">
```

b. UNIX-based environment (Mac OS or Linux)

The path of cutadapt should be provided as well.

```
cutadapt <- "/path/to/cutadapt/executable"
```

Alternatively, you may choose to put cutadapt in the $PATH. To do this, open the file `bashrc` with the following command:

```
gedit ~/.bashrc
```

Then paste the path to cutadapt as following.

```
export PATH=$PATH:/dir_containing_cutadapt
```

Then the script should run without modifications.

3. Set up primers removal

The `DADA2.R` script incorporate a primer removal step from the official DADA2 ITS Pipeline Workflow.

The base set of primers used in the pipeline are designed to target a 312bp barcode located on a rbcL plastid gene described by from Vasselon et al. 2017.

If you are using another set of primers, replace the sequence *Forward* (FWD) and *Reverse* (REV) in the `DADA2.R` script lines 36 and 40.

Each primers sequences has to be between double quotes, and different primers has to be separated by a comma. In the base pipeline, 3 *Forward* primers and 2 *Reverse* are used as follow.

```
FWD <- c("AGGTGAAGTAAAAGGTTCWTACTTAAA",
         "AGGTGAAGTTAAAGGTTCWTAYTTAAA",
         "AGGTGAAACTAAAGGTTCWTACTTAAA")

REV <- c("CCTTCTAATTTACCWACWACTG",
         "CCTTCTAATTTACCWACAACAG")
```

4. Prepare the reads

In Next Generation Sequencing (NGS) data sets, two type of reads are provided in different files. Reads *Forward*, labelled R1 and *Reverse*, labelled R2. For each sample, both files R1 and R2 have to be put in the `data` directory in a compressed format, ending with `fastq.gz`.

```
{ID sample number}_L{Sequencing lane number}_R1_001.fastq.gz
{ID sample number}_L{Sequencing lane number}_R2_001.fastq.gz
```

5. Check the pipeline path

If you are working on **RStudio**, the woking directory path might not be located in the `scripts` directory as it is expected when simply running the R script out of RStudio.

To be sure it is the case, go to the **RStudio** tab `Session -> Set Working Directory -> To Source File Location` and clic on this later option.

You can now check you are in the `scripts` directory with the command:

```
getwd()
```

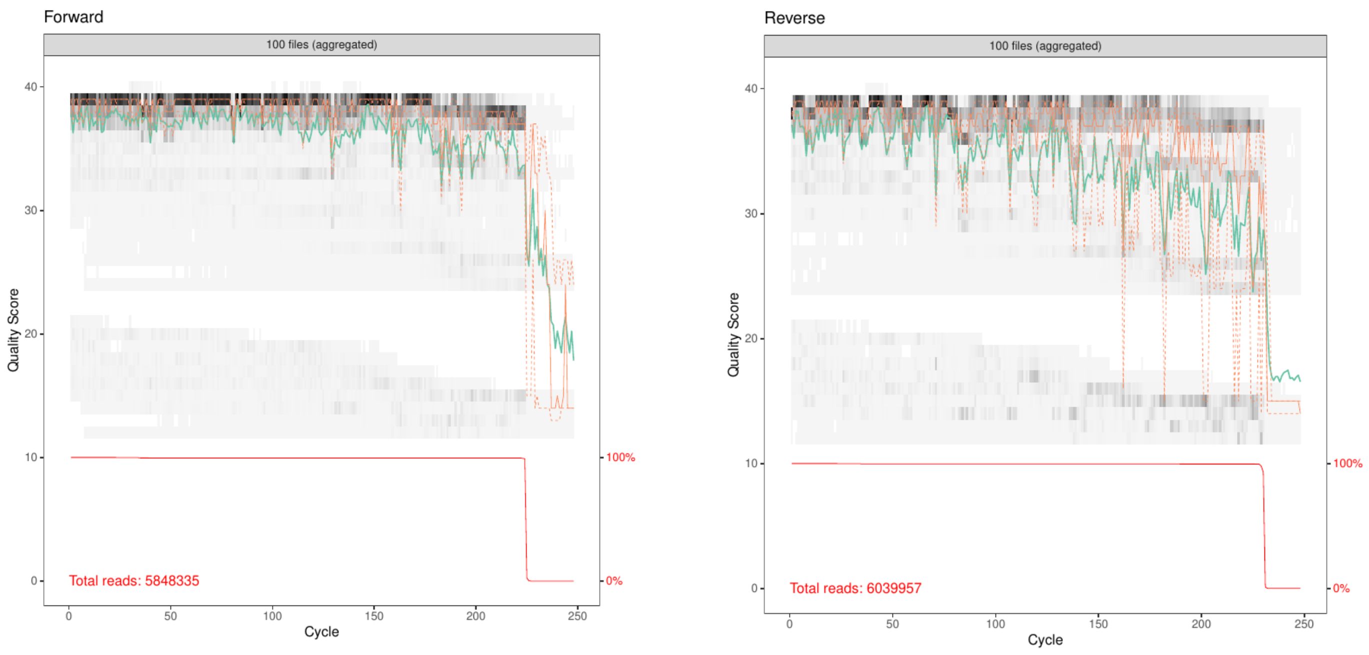
6. Run the Quality Check (QC)

The pipeline can now be run up to the Quality Check steps, enabling to vizualize the quality profiles of *Forward* and *Reverse* reads, necessary to adapt the trimming parameters in subsequent steps.

For this, run all the commands located in the **SET UP THE ENVIRONMENT**, **REMOVAL OF PRIMERS**, and **QC CHECK** parts of the `DADA2.R` script.

The cutadapt step in **REMOVAL OF PRIMERS** and `plotQualityProfile()` in **QC CHECK** are the most time-consuming steps and can last for tens of minutes, depending of the size of the dataset and RAM available.

The pipeline will output graphs about the average error rate observed in the *Forward* and *Reverse* reads in the `plots` directory.



QC check of the reads Forward and Reverse.

We will use the average QC check to estimate how to trim our reads. Look at the plot, and locate the **Cycle** value (x-axis) for which the **Quality Score** (y-axis) start to drop for Forward and Reverse reads.

In the exemple above the Quality Score drops at 220 for Forward reads, and 230 for Reverse reads. Therefore, we will trim our reads at these values.

You can now input these values in the `DADA2.R` script, line 115. The two values are given toghether in the `truncLen` option of the function `filterAndTrim()`. In our case, with the previous scores of 220 and 230 we estimated, the command would be:

```
out_2 <- filterAndTrim(fas_Fs_process, fas_Fs_filtered, fas_Rs_process,
fas_Rs_filtered,
                       truncLen = c(220, 230))
```

7. Filter and trim

Once the parameters of `filterAndTrim()` modified, all the commands of the **FILTER AND TRIM** part of the pipeline can be run. The `filterAndTrim()` command is time-consuming and took more than 10 minutes to give an output with the ACA_2018 data set (1.4GB of raw reads).

8. Process the rest of the DADA2 pipeline

Learning error rate

The DADA2 algorithm will learn the error rate specific to the data set with a parametric error model, which is going to be used in subsequent steps to remove ambiguous reads.
The `leanErrors()` command is time-consuming as well, and will take tens of minutes to run for both *Forward* and *Reverse* sets.

Dereplication, remove chimera, and reads statistics

Subsequently, you might run the **DEREPLICATION, SAMPLE INFERENCE & MERGE PAIRED READ**, **CONSTRUCT SEQUENCE TABLE**, **REMOVE CHIMERA**, **TRACK READS THROUGH THE PIPELINE** to finish processing the reads and make the seqtab ASVs table.

Assign taxonomy

During the **ASSIGN TAXONOMY** step, a two-steps ASVs identification will attribute taxonomy to each ASVs with `assignTaxonomy()`, then use a more stringent species assignement with `assignSpecies()`.

Data saving and plot

Run the rest of the `DADA2.R` script to save the pipeline outputs in the `results` directory. These include traditional DADA2 output (from fkeck Github):

- `track_reads.csv` provide the number of reads remaining after different steps of filtering.
- `seq_nochim_tax.csv` show the outputs of the basic DADA2 **assignTaxonomy()** command, with ASVs sequence, taxonomic assignement, and bootstraps values for each taxonomic rank.
- `seq_nochim_exact_sp.csv` show the more stringent species-level taxonomic attribution from the **assignSpecies()** command.The file include ASVs sequence, and Genus plus Species identification.

In addition, several files formated specifically for the Phyloseq and BLAST pipeline processing:

- `taxonomy.csv` is similar to **seq_nochim_tax.csv**, but include an unique ASV number for this data set (numerated ASV_0001, ASV_0002, ASV_0003, etc...). Used as input for the Phyloseq module of the pipeline.
- `asv_table.csv` show the number of ASV per samples. Each rows represent an ASV and each column is a sample. Used as input for the Phyloseq module of the pipeline.
- `asv.fasta` is a FASTA-formated file incorporing ASVs names and sequences. Used as input for the BLAST module of the pipeline.

9. Phyloseq pipeline

We will plot the species abundance distribution from this data set using the `Phyloseq.R` script, located in the `script` directory, as the `DADA2.R` script.

As the `Phyloseq.R` script is using files produced in step 8 of the pipeline as in input, it does not require to run in the same R session than the `DADA2.R` script.

You can run all the commands located in the script, which are not requiring editing. The species abundance plots should be saved in the `plots` directory.