Q2Pipe V0.95.5 User guide

Natural Resources Canada

Canadian Forestry Service

Government of Canada

https://github.com/NRCan/Q2Pipe

Authors:

Patrick Gagné, M.Sc.

Christine Martineau, PhD

November 28th, 2023

Contents

What is Q2Pipe?

Q2Pipe is a powerful and flexible pipeline designed to streamline and automate the microbiome analysis process. It serves as a user-friendly interface for conducting complex microbial community analyses, specifically designed to work seamlessly with Qiime2 (Bolyen, et al., 2019), a popular bioinformatics platform for microbiome research.

What it does:

- Q2Pipe provides a comprehensive workflow for microbiome analysis, guiding users through every step, from data import to metrics and tables generation for statistical analyses.
- It automates routine tasks and ensures that best practices in data processing and analysis are followed.
- Users can customize and fine-tune various analysis parameters to meet their specific research needs.

Who should use it:

- Researchers and scientists working on microbiome projects who want to simplify and expedite their data analysis.
- Those with varying levels of expertise, from beginners to experienced bioinformaticians, as Q2Pipe offers both automation and customization options.
- Anyone looking to conduct microbiome analyses using Qiime2 and benefit from a streamlined and user-friendly workflow.

In essence, Q2Pipe is a valuable tool that empowers researchers to efficiently process and interpret microbiome data, making it accessible to a broader range of scientists and facilitating more robust microbiome research.

Q2Pipe Installation and preparation

To download Q2Pipe in your terminal's current folder, use this command:

```
git clone https://github.com/NRCan/Q2Pipe.git
```

You should get this message:

```
Warning: Permanently added 'github.com,140.82.113.3' (RSA) to the list of known
hosts.
remote: Enumerating objects: 174, done.
remote: Counting objects: 100% (174/174), done.
remote: Compressing objects: 100% (122/122), done.
remote: Total 174 (delta 123), reused 103 (delta 52), pack-reused 0
Receiving objects: 100% (174/174), 34.05 KiB | 741.00 KiB/s, done.
Resolving deltas: 100% (123/123), done.
```

Then type the following command:

```
ls ./Q2Pipe
```

You should obtain this (Filenames and count can differ from the picture)

```
convert_manifest_to_sing.sh          qiime2_step2_denoise.sh
create_manifest_direct.sh            qiime2_step3_feature_filtering.sh
optionfile_qiime2_default.txt        qiime2_step4_denovo_clustering.sh
optionfile_qiime2_default_sing.txt   qiime2_step5_classifier_training.sh
qiime2_singularity_manual_mode.sh    qiime2_step6_classification.sh
qiime2_step1.5_denoise_evaluation.sh qiime2_step7_taxa_filtering.sh
qiime2_step10_metrics.sh             qiime2_step8_rarefaction_curve.sh
qiime2_step10_metrics_norarefy.sh    qiime2_step9_rarefy.sh
qiime2_step11_export.sh              testlist.txt
qiime2_step1_import.sh
```

Create a Q2Pipe environment variable for easier use (optional but highly recommended)

Because it can be straining to always enter the full program path for each step, You can make it easier by creating an environment variable as a shortcut for the program path.

Note that this whole guide consider you have followed this step (if not, you must replace $Q2P in each command by the full path to the Q2Pipe folder.

To create the variable, use the following command:

```
export Q2P="$PWD/Q2Pipe"
```

You are now able to call each Q2Pipe steps using $Q2P/ before your step name. It is important to note that this is NOT persistent on the system (closing the terminal will flush this variable), so to make it permanent, use this command:

```
echo "export Q2P="$PWD/Q2Pipe"" >> $HOME/.bashrc
```

It is **VERY IMPORTANT** to use >> and not > (using the latter will destroy your .bashrc and can cause a lot of problem on your system)

If you encounter a problem, open an issue on the Q2Pipe Github repo (https://github.com/NRCan/Q2Pipe/issues)

Q2Pipe is now ready to work on your system.

Installation dependent parameters

Apptainer

Apptainer (formerly known as Singularity) is based on the same containerisation technology as Docker but for Unix-Based system, it is also easier to use and supported on most public HPC like the Digital Research Alliance of Canada "DRAC" (formerly known as Compute Canada). It is also the official supported installation for Q2Pipe. To use it, Apptainer software must be installed on your system (if you are on DRAC, you must load the apptainer module). To install it, follow the instruction on this page: https://docs.sylabs.io/guides/3.10/admin-guide/admin_quickstart.html#installation-from-source

Once Apptainer is installed and ready, you have two options:

1. If you have admin right, you can build the container using the provided recipe in the Q2Pipe repository you cloned in the last step using this command:

```
sudo apptainer build qiime2_202X_X_q2p0955.sif $Q2P/Apptainer_Qiime2_NRCan.recipe
```

2. If you don't have admin right, use this command to pull the prebuilt image from the ghcr.io repository (make sure to use the correct version from the package).

```
apptainer pull q2pipe_qiime_latest.sif
oras://ghcr.io/patg13/q2pipe/qiime2_q2pipe:latest
```

Once it is done, you should have a `q2pipe_qiime_latest`.sif file available, this is your Apptainer image where every program needed to run Q2Pipe is available (including Qiime2). To use this image in Q2Pipe, you must edit your option file and fill the APPTAINER_COMMAND line like this (make sure you use the correct path where your image is located):

APPTAINER_COMMAND="apptainer exec --cleanenv --env MPLCONFIGDIR=/tmp,TMPDIR=/tmp /home/ubuntu/qiime2_apptainer/`q2pipe_qiime_latest.sif` "

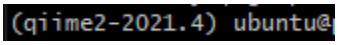
NOTE: if you are running Q2Pipe on a WSL (Windows subsystem for Linux), you must add "-B /run/shm:/run/shm" without the quotes to the apptainer command. This will make sure Apptainer correctly bind the WSL shared-memory folder.

Anaconda3

If you use the conda installation, it is mandatory to activate your Qiime2 environment before launching Q2Pipe. To do this, you can use this command (replace X by the number of your Qiime2 version):

```
conda activate qiime2-202X.X
```

You should now have a qiime2-202X.X prefix in your terminal:



For this type of installation, there is no special parameters to define in the option file.

Updating the pipeline

To check if you have the latest version, launch these commands:

```
cd $Q2P
git fetch
git status
```

If you see “Your branch is behind 'origin/master' by X commits”, you must update Q2Pipe

To update Q2Pipe with the latest version on the repository, use these commands:

```
cd $Q2P
git pull
```

IMPORTANT: Any modification you make on the Q2Pipe files will cause a merge conflict and prevent the pipeline from updating.

Preparing your analysis

Manifest File(s)

In the context of Qiime2 and microbiome analysis, a manifest CSV file is a critical component used to link sequence data (typically in the form of Fastq files) to the metadata associated with each sample in a sequencing experiment. Each row in the manifest file corresponds to a unique sample in your sequencing dataset.
The file should contain information about each sample, such as its name, associated barcode (if used), and the file paths to the forward and reverse read Fastq files.

To generate your manifest file(s), you can use the auto-generation script available in $Q2P; it will make your life a lot easier, especially if you have many samples. To use the script, launch this command in your sample’s folder (do not forget to change the argument according to your sample names):

```
$Q2P/create_manifest_direct.sh .fastq.gz _L001_R1_001 _L001_R2_001 _S[0-9]+
```

Here is an explanation on how this command work and how to customise it for your samples:

Main command	Sample extension	Forward Suffix	Reverse Suffix	Secondary cleanup
$Q2P/create_manifest_direct.sh	.fastq.gz	_L001_R1_001	_L001_R2_001	_S[0-9]+

For example, if this is your sample filename: **82-16S_S23_L001_R1_001.fastq.gz,** your sample ID will be **82-16S_S23**, the forward suffix will be **_L001_R1_001**, the reverse suffix will be **_L001_R2_001**, and the extension will be **.fastq.gz**.

The secondary cleanup is optional and will remove the **_S23** from the sample name. Be careful however; if you have a _S followed by a number in your target sample name (ex: 82**_S43**FRTR_16S_S345_L001_R1_001), using _S[0-9]+ will cause a conflict and the resulting sample name will be 82. In this case, you should use **L001_R1_001** and **L001_R2_001** as suffixes and _S[0-9]+**_** as secondary cleanup (this will use the tailing underscore as a matching guide)

A suffix must always be the same on ALL samples (so S23_L001_R1_001 is NOT a correct suffix). Same thing for the extension.

You must adjust the parameters of the create_manifest_direct command according to your sample names.

After the command, you should have a manifest.csv file in your current folder (you can edit it or use it as-is, but you must make sure the sample ID in your manifest file match those in your metadata file)

Q2Pipe will use the manifest to find your samples, so NEVER move your samples, doing so will break your manifest file.

According to Qiime2 guidelines, each sequencing run should be denoised separately. Combining run before denoising can interfere with the Dada2 error modeling system and eliminate good sequence (or keep wrong ones) in the dataset.

In Q2Pipe, you can separate your runs by creating a manifest file for each sequencing run you have and then specify them in the option file (separate them by a comma). Q2Pipe will take care of the rest. Note that you will have to launch the merging step before running step 3.

You can also keep the denoising folder (will be created during the importation step) and use It in other analysis, but you must make sure your denoising parameters are the same. The merging step contain a compatibility check that will stop the merging if the denoising parameters used are not the same in each run.

Metadata File

Your metadata file contains relevant information about each of your samples (e.g. treatment, sampling location, type of material, etc.). This file is formatted as a TSV (Tabulation-Separated Values) document, where each metadata is represented in a separate column. The initial line of this file serves as the header and must list the names of all metadata. It is essential to avoid using spaces in these names, as they can potentially disrupt parts of the data processing pipeline, particularly when executing Qiime2 commands. To replace spaces, you can employ an underscore character (_) instead.

In the event of multiple sequencing runs, it is crucial not to generate separate metadata files. Although not mandatory, it is advisable to incorporate a column indicating the sequencing run from which each sample is coming, this will help you determine if there is a run effect during the merge.

Option file

The Q2Pipe option file, also known as the configuration file or parameter file, serves as a crucial component in the Q2Pipe workflow, particularly when using Qiime2 for microbiome analysis. The primary purpose of the Q2Pipe option file is to specify and control various parameters, settings, and configurations that dictate how the microbiome analysis is conducted. Here's a detailed explanation of its purpose:

1. **Configuration Management**: The option file allows you to define and manage all the configuration settings for your microbiome analysis in a structured manner. It serves as a central location to specify a wide range of parameters, from data input sources to analysis parameters and outputs.
2. **Input Data Specification**: You use the option file to specify the locations and formats of your input data, including the manifest file (containing sample information and file paths) and the metadata file (containing sample-specific information). It ensures that Q2Pipe knows where to find the necessary data for analysis.
3. **Parameter Customization**: You can customize various parameters for each step of the microbiome analysis workflow, such as trimming, filtering, denoising, rarefaction, and diversity calculations. This customization allows you to fine-tune the analysis based on your specific dataset and research objectives.
4. **Software Tools Integration**: You can configure settings related to the integration of various software tools used in the analysis, such as Qiime2 itself and other auxiliary tools like FunGuild or Falco. This ensures that these tools are invoked correctly and that their results are appropriately integrated into the analysis.
5. **Parallelization and Multithreading**: The option file may provide options to specify the number of threads or CPU cores to be used for parallel processing, which can significantly speed up the analysis, particularly for computationally intensive steps.
6. **Documentation and Communication**: The option file can serve as documentation for your analysis, providing a record of all the settings and parameters used. It also aids in communication with collaborators, as you can easily share and reproduce the analysis by sharing the option file.

In summary, the Q2Pipe option file is a pivotal component in the Q2Pipe workflow, as it provides the means to tailor, orchestrate, and execute a microbiome analysis based on your specific research needs, data, and objectives. It centralizes the control of the entire analysis process, making it more manageable, reproducible, and customizable.

First, you must make a copy of the default option file in your current directory by launching this command:

```
cp $Q2P/optionfile_q2pipe_default.txt .
```

To configure your analysis, you will need to make modifications directly in the terminal using a text editor like Nano or your preferred graphical interface text editor. Pay attention to the use of uppercase letters, which indicate specific options for Q2Pipe, while lowercase parameters correspond to Qiime2 command parameters.

If you require more detailed information on these commands, please consult the Qiime2 documentation. Remember to replace underscores (_) with hyphens (-) in option names, for example, "p_max_depth" in Qiime2 becomes "p-max-depth."

Additionally, be aware that if a parameter in the option file has a default value in Qiime2, it is usually already configured with that default value. If you have an older option file designed for a previous version of Q2Pipe, you can utilize the option file update script. This script will add any missing parameters to a copy of the old file, making it compatible with the latest version of Q2Pipe. Keep in mind that the position of lines in the new file may appear unusual, but this should not impact functionality.

To utilize this tool, use the following command, replacing "old_optionfile.txt" with your own file.

```
$Q2P/update_optionfile.sh old_optionfile.txt
```

This will create a old_optionfile.txt.new file which will be compatible with the current Q2Pipe version.

DISCLAIMER: This script will soon be replaced by a better version that will keep options in their own section. The current script will work, but it will sometime place new options in strange places of the file.

Common analysis parameters

The "ANALYSIS_NAME" serves as the prefix for your analysis, and it will be included at the beginning of every filename generated by Q2Pipe. Therefore, it's important to choose a meaningful and descriptive name for your analysis. Here are some suggestions for formatting the analysis name:

ProjectName_Gene_Date: This format includes the project name, the specific gene or target of the analysis, and the date. For example, "MyProject_ABCGene_20230922."

ProjectName_SampleType_Date: If your analysis is focused on different sample types, you can include that information in the analysis name. For example, "ProjectXYZ_WaterSoil_20230922."

When selecting an analysis name, please adhere to the following guidelines:

- Avoid using spaces, special characters (e.g., ! /), or hyphens (-) in the analysis name.
- Instead of spaces, you can use underscores (_) to separate words in the name.

By following these guidelines, you'll ensure that your analysis name is both informative and compatible with Q2Pipe's naming conventions.

The "NB_THREADS" parameter should be set to the number of virtual machine CPUs or threads available for your analysis. If you have a virtual machine with 8 CPUs, you should set "NB_THREADS" to 7. This subtraction of 1 from the maximum available threads is commonly done to optimize performance and prevent oversaturation of the CPU resources, ensuring smoother execution of your analysis tasks.

The “CLASSIFIER_NB_THREADS” parameter is explained in step 6

The "APPTAINER_COMMAND" option is a special parameter for Apptainer type installation, See Apptainer Installation section.

The "METADATA_FILE_PATH" is the path to your metadata text file (in TSV format), it can be a relative path or an absolute path.

The "MANIFEST_FILE_PATH" is the path to your manifest text file (in CSV format), it can be a relative path or an absolute path.

The "TEMPORARY_DIRECTORY" option is used to specify the location where Qiime2 will temporarily store intermediate data during analysis. By default, Qiime2 utilizes the "/tmp" directory for this purpose. However, depending on the size of your analysis or the available space on your system, you may encounter a "No space left on device" error. To avoid this issue, you can change the "TEMPORARY_DIRECTORY" option to point to a different drive or directory with sufficient space for temporary storage.

Launching your analysis

With your data manifest and metadata files prepared, you are now ready to launch the pipeline. Ensure that you have the necessary environment set up depending on your installation method (Anaconda3 or Apptainer).

For Anaconda3 Installation:

1. Make sure your Qiime2 environment is loaded. You can activate it using the following command:

```
conda activate qiime2-202X.X
```

2. Confirm that you have the required dependencies installed and properly configured within your Qiime2 environment.

For Apptainer Installation:

1. Ensure that your Apptainer image is up to date.
2. Confirm that the Apptainer command is correctly specified in your option file. It should point to the Apptainer installation location.

Once you have completed these preparation steps, you can proceed with launching the pipeline using the appropriate command specified in your setup.

Q2Pipe Step 1: Importation (mandatory)

Step description

Q2Pipe's first step involves importing your entire dataset into a Qiime2 Archive (QZA) file format. This is done to prepare your data for further analysis within the Qiime2 framework. Additionally, Q2Pipe offers two optional analysis tools, namely Figaro (Weinstein, Prem, Jin, Tang, & Bhasin, 2019) and Falco (Brandine & Smith, 2021) which can be used to assist in guiding the determination of optimal trimming parameters for subsequent steps in the analysis pipeline.

1. **Importing Data into QZA Archive:** The initial step of Q2Pipe focuses on importing all your raw data into a Qiime2-compatible format, making it ready for Qiime2 analysis. This is a crucial preprocessing step to ensure that Qiime2 can work with your dataset effectively.

2. **Optional Analysis with Figaro and Falco:** Figaro and Falco are tools that can be employed to help determine the best trimming parameters for your data. These tools can provide insights into the quality of your sequencing reads and assist in deciding how much trimming is required before proceeding with downstream analysis. Using Figaro and Falco can enhance the quality of your analysis results.

By ensuring proper data import and utilizing these optional tools, you can set the stage for a successful Qiime2 analysis.

It's important to note that Q2Pipe utilizes modified versions of Figaro (modified version) and Falco (modified version), which have been specifically adapted to work more effectively within the Q2Pipe framework. These modifications are designed to enhance compatibility, performance, or functionality to better suit the needs of users conducting their analysis pipelines with Q2Pipe.

Figaro

FIGARO will quickly analyze error rates in a directory of FASTQ files to determine optimal trimming parameters for high-resolution targeted microbiome sequencing pipelines, such as those utilizing DADA2 and Deblur. Here's a description of the different parameters you can set in Q2Pipe:

1. **figaro_trim_offset**: This parameter represents an offset value that will be subtracted from the longest sequence length. The purpose of this subtraction is to ensure that all reads end up having the same length. Importantly, this operation only affects the copies of the reads, not the original reads themselves. It's a way to standardize the length of reads so Figaro can process them correctly.

2. **f_amplicon_size**: This parameter specifies the expected amplicon sizes for your sequencing data. You can provide a single size or a range of expected sizes for your amplicons. As an example, if your expected amplicon size is 374, you could use this range: 368, 370, 372, 374, 376, 378, 380. These sizes should exclude the primer sequences. This will run Figaro prediction algorithm for each size and allows you to make more informed decisions about how much trimming is needed to preprocess your data effectively while retaining as much high quality sequence information as possible. It's important to note that running Figaro for each specified amplicon size can significantly increase the duration of the analysis. This is because each size requires a separate run of Figaro, and the time required for analysis will increase linearly with the number of sizes you specify.

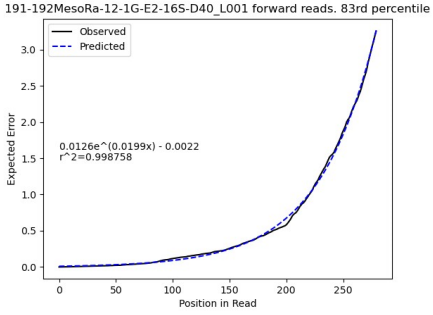
3. **f_forward_primer_len**: This parameter indicates the length of the forward primer used in your sequencing experiment. Knowing their length helps with accurate trimming of the sequences.
4. **f_reverse_primer_len**: Similar to the forward primer length, this parameter specifies the length of the reverse primer used in your sequencing experiment.

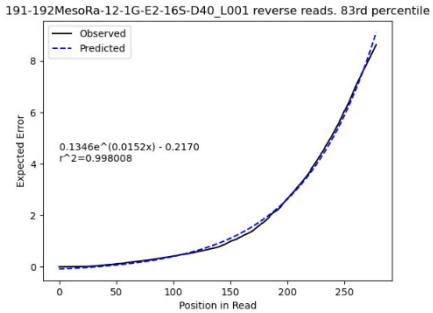
Figaro output will appear in the newly created manifest folder (one folder will be created by specified manifest) and will contain a summary file of the figaro analysis for each size:

{"trimPosition": [**242, 177**], "maxExpectedError": [2, 2], "readRetentionPercent": 85.49, "score": 83.489}

{"trimPosition": [**232, 187**], "maxExpectedError": [2, 3], "readRetentionPercent": 85.21, "score": 81.457}

Figaro output will also contain the plots for expected error values over the course of the amplicon for both forward and reverse directions:





These will have the exponential regression model included as well as its r-squared.

This model appears to generally run a very high r-squared value. If the model has an r-squared below 0.95, something is not right.

In this exemple, since the r-squared is > 0.95 and the maximal expected error is not too high, 242,177 would be a good value for the dada2 trimming. Always keep in mind that this is prediction software, the outputted value is a suggestion and could be wrong depending on your dataset. You should ALWAYS confirm the values by looking at the quality graphs in the importation output. Never trust it blindly.

Falco

Falco is an emulation of the widely used FastQC software, designed to check large sequencing reads for common issues and quality problems. What sets Falco apart is its implementation in C++, which provides improved speed and efficiency, making it well-suited for handling large datasets. Additionally, Falco has been specifically modified for compatibility with Q2Pipe, enabling multithreaded capabilities to enhance its performance in the context of Q2Pipe's analysis pipeline. Here’s a description of the different Falco parameters:

1. **falco_right_trim**: This parameter is used to trim the end of the reads, primarily to reduce false-positive detections of quality issues by Falco. You can specify the number of bases to trim from the right end of the reads. If you set it to 0, it means no trimming will be performed. It's

important to note that the trimming specified is not applied to the original reads or their copies. Instead, this trimming is performed during the Falco analysis itself.

2. **FALCO_COMBINED_RUN**: When set to "true," this parameter indicates that you want to create combined R1 and R2 files and run them through Falco as a combined analysis. This can help identify and flag issues across entire sequencing runs, providing insights into potential problems that may affect multiple samples. Keep in mind that this option may consume more time and disk space compared to running Falco on individual samples. It's good to know that the resulting quality plots generated by Falco will be very similar, if not identical, to Qiime2 quality plots. This similarity allows users to leverage these quality plots in the same way as those generated by Qiime2.

3. **CLEAN_FALCO_OUTPUT**: If set to "true," this parameter instructs the pipeline to delete the exportation files generated during the Falco analysis after the analysis is complete. This is done to save disk space while retaining the results of the analysis for reference. You should only set this to false for debugging or troubleshooting purposes.

Falco module will output html and summary text file for each sample inside a manifest that you can consult to ascertain data quality before running the denoising step. You should consult the Falco documentation to get a better understanding of how to interpret the results and the possible impact on your analysis.

Multithreading

The importation step is multithreaded (NB_THREADS option), it will distribute threads evenly among the manifest files according to following this formula:

Threads_per_manifest = floor(NB_THREADS / number_of_manifests)

This consideration becomes especially significant when you run Figaro and/or Falco, as these tools are engineered to harness the power of multiple threads. However, if your task involves solely importing data without running Figaro or Falco, then the number of threads utilized will be limited to the number of manifests in your analysis.

Running command

To execute the first step of Q2Pipe, you can use the following command, ensuring that you replace "optionfile_q2pipe_default.txt" with the actual name of your option file:

```
$Q2P/q2pipe_step1_import.sh optionfile_q2pipe_default.txt
```

Following this step, as mentioned previously, you should visualize the QZV output for each run to determine the optimal quality trimming parameters for the denoising step. Use the command provided earlier to visualize the QZV files using a GUI environment:

```
qiime tools view FILENAME.qzv
```

This visualization step will assist you in making informed decisions about quality trimming parameters for subsequent analysis steps in your pipeline.

Q2Pipe Step 2: Denoising (mandatory)

Step description

The denoising step in Q2Pipe incorporates the Qiime2 DADA2 implementation to perform several key operations on your sequencing data:

1. **Trimming**: It trims both the forward and reverse reads, which involves removing low-quality bases from the ends of the reads to improve data quality.
2. **Filtering**: In addition to trimming, the denoising step incorporates a filtering process. Filtering is essential for removing sequences that do not meet specific quality criteria or are considered noise. This ensures that only high-quality and relevant sequences are retained for further analysis.
3. **Merging**: It merges the trimmed forward and reverse reads to create longer, high-quality sequences resulting in your amplicon.
4. **Chimera Removal**: The step also includes chimera removal, which identifies and eliminates chimeric sequences from your data. Chimeras are artifacts created during PCR amplification.
5. **ASV/Feature Formation**: It forms ASVs (Amplicon Sequence Variants) or features.

If you are working with multiple sequencing runs, it is possible to specify different truncation lengths for each run. However, it's important to note that you cannot select different primer trimming parameters because doing so can make the runs incompatible for merging (different amplicon length).

CutAdapt

Additionally, if your target gene primers require more complex trimming beyond simple positional trimming, the denoising step incorporates the Qiime2 implementation of CutAdapt. This is particularly useful for sequencing targets like the fungal ITS, which are known for their variable length and primer-related length variations.

If you are using CutAdapt to remove the primers from your sequencing reads, you should set the parameters "p_trim_left_f" and "p_trim_left_r" to 0 by default. However, there is an exception to this rule; you may consider adjusting these parameters to values different from 0 if you have identified low-quality regions at the 5' end of your reads. While this is a relatively rare occurrence, it's not impossible.

Additionally, please be aware of the distinction between positional trimming and the use of CutAdapt. When utilizing CutAdapt for primer removal, it's important to consider that it will truncate your sequences prior to denoising, altering the quality framework of your reads. As a result, if you opt for CutAdapt, it is necessary to utilize the "CA_FORCE_INTERRUPTION" parameter to halt the analysis before denoising. You should then choose your trimming parameters based on the information available in the "manifest.import_CA.qzv" files generated by Cut-Adapt. Once you identify the correct parameter, you can switch back "CA_FORCE_INTERRUPTION" to false so Q2Pipe can proceed with the denoising.

It's crucial to note that if you are employing positional trimming instead, you should NOT subtract the length of your primers from your trimming parameters. Qiime2's operation priority ensures that the sequences are truncated before the primers are removed in positional trimming scenarios.

Multithreading

The importation step is multithreaded (NB_THREADS option), it will distribute threads evenly among the manifest files according to following this formula:

If **CONCURRENT_JOBS > 1:**

Threads_per_manifest = floor(NB_THREADS / CONCURRENT_JOBS*)

* For best result, **CONCURRENT_JOBS** should be equal to the number of manifests to process

If CONCURRENT_JOBS = 1:

Threads_per_manifest* = **NB_THREADS**

*Manifest will be processed in serial mode (one at the time)

Running command

Once you have identified the optimal parameters for your data and updated your option file accordingly, you can initiate this step using the following command:

```
$Q2P/q2pipe_step2_denoise.sh optionfile_q2pipe_default.txt
```

Note that this step can be time-consuming and may require a significant amount of time to complete, especially if you have a large number of reads/runs and depending on the parameters you have chosen.

If you have only one manifest file, the analysis will display the mean sample frequency. This information is provided to assist you in calculating the appropriate minimal frequency value for step 3.

Q2Pipe Step 2.5: Multiple run merging (mandatory if manifest count > 1)

Step description

During this step in the analysis process, the objective is to merge multiple sequencing runs while ensuring that all runs are compatible with each other. Compatibility is achieved by utilizing the same primer denoising parameters for each run. While you have the flexibility to employ slightly different truncation parameters for each run, it is advisable to keep these parameters very close to each other. This approach minimizes the likelihood of introducing a potential run effect that could negatively impact your results.

It's important to highlight that forcing the merge in cases of incompatibility is possible but strongly discouraged. While there may be an option available to override compatibility checks, it is NOT recommended to use it. Maintaining compatibility ensures the integrity of your analysis and helps avoid potential issues that may arise from merging datasets with different denoising parameters.

Multithreading

The merge step supports multithreading to expedite the generation of merge plots. However, it's important to note that using fewer threads, including just one thread, should have a minimal impact on the speed of this step. The merge plot generation process is not computationally heavy, and the use of additional threads primarily aims to improve efficiency when handling larger datasets or more complex operations. Therefore, you can adapt the number of threads to your available resources without significant concerns about speed degradation.

Running command

You can launch this step by using this command:

```
$Q2P/q2pipe_step2.5_run_merging.sh optionfile_q2pipe_default.txt
```

The step will produce combined rep-seqs and feature-table files that will be used by the next step. It will also produce a merge plot.

The "mergeplot.qzv" file can be used to examine whether there is evidence of a run effect. If your features are distinctly separated between the runs in the visualization, it may indicate the presence of a run effect. However, it's essential to interpret this information within the context of your specific sample and study design, as the presence of a run effect can be influenced by various factors related to your biological samples and sequencing process.

This step will also output the mean sample frequency of your merged runs. This information is provided to assist you in calculating the appropriate minimal frequency value for next step.

Q2Pipe Step 3: Low abundance and rare feature Filtering (mandatory, but can be deactivated with an option)

Step description

Before initiating this step, it's advisable to visualize the "$ANALYSIS_NAME.table-dada2.qzv" file and take note of the sample mean frequency. You have the flexibility to set a threshold that determines the minimum occurrence of a variant (ASV) and, if desired, the minimum number of samples in which the variant must appear to be preserved. For example. One potential option is to eliminate any ASVs with a frequency lower than 0.05% of the average sample depth (**p_min_frequency** option).

You can also filter your ASV depending on the number of samples it can be found (**p_min_sample** option). For example, a p_min_sample = 2 would remove any ASV that are not in at least two samples.

It's worth mentioning that Q2Pipe also provides these calculated values for you at the end of Step 2 and 2.5, simplifying the process of determining the minimal frequency value. This information is valuable for removing low frequency ASV.

Frequency per sample

	Frequency
Minimum frequency	87,624.0
1st quartile	109,739.0
Median frequency	129,981.0
3rd quartile	142,116.0
Maximum frequency	180,451.0
Mean frequency	128,524.52380952382

It this example, if you want a 0.05% threshold, the correct calculation is 128,524 * 0.0005, which is 64.26, but because Qiime2 feature-filtering does not support float value, you can floor the value to **64.**

This step is typically mandatory, you have the option to use a value of 0 for both "**p_min_frequency**" and "**p_min_samples**" to effectively "skip" the filtering process. In this case, the step will create the necessary files for subsequent steps, but it will not filter the table. The filenames generated may still contain the "filtered" tag, but this is done for compatibility purposes. This flexibility allows you to tailor your analysis according to your specific needs and objectives.

Running command

Once you set your parameters, you can launch this step using this command:

```
$Q2P/q2pipe_step3_feature_filtering.sh optionfile_q2pipe_default.txt
```

Q2Pipe Step 4: Denovo Clustering (mandatory, but can be deactivated with an option)

Step Description

In this step, certain features are grouped together based on their sequence similarity. However, you have the option to bypass this step by using "**NA**" as the similarity threshold. This will generate the necessary files for the next step in the analysis without performing the sequence grouping based on similarity.

It's important to emphasize that running this step effectively "converts" your ASVs into OTUs, which may have implications for your analysis. As a result, it's not recommended to run this step unless you have a specific reason or requirement to do so. The decision to convert ASVs into OTUs should be carefully considered based on your research goals and the implications it may have on downstream analysis and interpretation.

Multithreading

The clustering step is designed to be multithreaded. It can significantly accelerate the clustering process, making it more efficient and reducing the time required for this step.

Running command

To launch this step, use the following command:

```
$Q2P/q2pipe_step4_denovo_clustering.sh optionfile_q2pipe_default.txt
```

Q2Pipe Step 5: Classifier Training (DEPRECATED)

WARNING: STEP DEPRECATED

Step description

Please be aware that this step has been deprecated in Q2Pipe version 0.93.0, and it will be entirely removed in version 0.94.0. You should consider using this step only if you do not have a trained classifier available for the taxonomical classification step. In such cases, this step will be responsible for training the classifier before proceeding with classification.

However, it is strongly recommended that you prepare your classifier outside of Q2Pipe and then submit your trained classifier directly to step 6 using the "CLASSIFIER_DATABASE_PATH" option. This approach provides you with greater flexibility and control over the training process and ensures that you are working with a classifier that meets your specific requirements and criteria.

By following this recommendation, you can optimize your taxonomical classification process and make more informed choices regarding classifier training.

Running command

Once your files are ready, launch this step using this command:

```
$Q2P/q2pipe_step5_classifier_training.sh optionfile_q2pipe_default.txt
```

Important note: This step can take a very long time to complete and is very RAM intensive, make sure your system is powerful enough to launch it.

Q2Pipe Step 6: Taxonomical classification (mandatory)

Step description

This step's primary function is to assign taxonomy to your features using a trained classifier. By default, Qiime2 employs a confidence level of 0.7 during the taxonomic assignment process. The confidence level serves as a threshold or cutoff value, ensuring that taxonomic assignments are made with a certain level of confidence in the accuracy of the assignment. This threshold helps filter out less confident assignments, improving the reliability of the taxonomic annotations. it can be advisable to adjust the confidence level for taxonomic annotation based on the specific gene or marker region being analyzed.

This article for example conducted a comparative study for different confidence levels using the Qiime2 Naive Bayes classifier for the COI F230 gene. The study found that the highest species-level accuracy was achieved when using the q2-feature-classifier Naive Bayes classifier with a confidence level set to 0.3 for this gene. This result highlights the importance of carefully selecting the confidence threshold for taxonomic assignment, as different genes or marker regions may require different confidence levels to achieve optimal accuracy in species-level classification.

Multithreading

This step support multithreading to accelerate the taxonomy assignment, However, it's crucial to exercise caution when allocating threads, as each additional thread places an additional burden on the system's RAM (Random Access Memory). Excessive thread usage can overload your system's RAM, potentially causing the step to fail.

Given the difficulty in accurately predicting the exact RAM requirements for this step (which depend on factors like classifier size and the number of features to be assigned), it is advisable to limit the number of threads to a conservative value, such as 8 threads or even fewer. This conservative approach helps mitigate the risk of RAM overload, ensuring the step's successful execution without taxing system resources excessively.

Take note that this step has its own thread number specification option (CLASSIFIER_NB_THREADS) to make it easier to use.

Running command

To launch this step, use this command:

```
$Q2P/q2pipe_step6_classification.sh optionfile_q2pipe_default.txt
```

Q2Pipe Step 7: Taxa filtering (mandatory, but can be deactivated with an option)

Step description

In this step, it's crucial to carefully consider what you want to include or exclude in your taxa, as this decision has a significant impact on your analysis. Ensure that the options you choose align with your specific objectives and research questions. Here are some key points to keep in mind:

1. **Inclusion and Exclusion**: You have the flexibility to define what taxa you want to include or exclude in your analysis. This step allows you to specify which taxa are of interest to you and which ones you want to disregard. Make sure your options accurately reflect your intentions.
2. **Matching Criteria**: By default, the matching criteria is set to "contains," which is suitable for most use cases. However, you can also define whether you want the matching to be exact or not. Adjust this setting as needed to meet your specific requirements.
3. **Skipping the Step**: If you do not wish to perform any inclusion or exclusion of taxa at this stage, you can simply leave both the include and exclude options blank. This will effectively skip this step in your analysis.
4. **Metadata Filtering**: Additionally, you have the option to filter your data based on metadata using the "p_where" option in the option file. You can define filtering criteria to include or exclude samples based on specific metadata attributes. Setting "p_where" to an empty string (p_where="") will indicate that no metadata filtering should be applied.
5. If you want to generate phylogenetic tree at this step, you must enable the **GENERATE_PHYLOGENY** parameter in the option file. Note that tis parameter will also be used in step 8

In summary, this step provides flexibility in defining what taxa to include or exclude, how matching is performed, and whether metadata-based filtering should be applied. Ensure that your choices align with your research objectives and the nature of your dataset.

Metadata filtering examples

These examples of metadata filtration parameters can serve as reference points to help you construct your own parameters based on your specific research needs.

p_where="[subject]='subject-1' AND NOT [body-site]='gut'"
p_where="[body-site]='gut' OR [reported-antibiotic-usage]='Yes'"
p_where="[body-site] IN ('left palm', 'right palm')"
p_where="[subject ID]='subject 1'

It is important to respect the double and the single quotes for the filtering to work correctly.
This filtering will be applied **BEFORE** the actual taxa filtering.

Inclusion and exclusion filtering examples

These examples of criteria filtration parameters can serve as reference points to help you construct your own parameters based on your specific research needs:
Ex 1 (with inclusion and exclusion parameters):
p_include=bacteria,archaea
p_exclude=eukaryota,mitochondria,chloroplast

Ex 2 (inclusion parameter only):
p_include=d__Eukaryota
p_exclude=

It's important to note that the "p_include" option will remove every taxa that does not contain (or be an exact match depending on the p_mode option) the inputted name. The "p_exclude" will then proceed to exclude any taxa that contain/match the inputted name among the remaining taxa.
In essence, "p_include" allows you to focus on taxa that contain the specified name while "p_exclude" excludes any exact matches of that name among the remaining taxa. This dual-step process can help you fine-tune which taxa are included or excluded based on your specific criteria and objectives.

Multithreading

In this step, it's important to note that most of the processes do not support multithreading. However, there is an exception: if you have enabled the "GENERATE_PHYLOGENY" parameter, the subsequent phylogenetic tree generation step does support multithreading.

Running command

Once you specified your inclusion/exclusion parameters, you can launch this step with this command:

```
$Q2P/q2pipe_step7_taxa_filtering.sh optionfile_q2pipe_default.txt
```

After completing this step, it's essential to review the barplot output to verify that your chosen parameters were applied correctly and that the results align with your research objectives. This visual inspection allows you to ensure that the taxa inclusion, exclusion, and other filtration criteria have been appropriately implemented and that the output accurately represents your analysis needs.

Before proceeding further, it's important to make a decision regarding whether you want to rarefy your data or not. You can specify this choice using the "SKIP_RAREFACTION" parameter in the option file. If you opt not to perform rarefaction, you can skip to Step 10 in the analysis pipeline. Rarefaction is a process used to standardize sample sizes, and its application can impact downstream analyses, so this decision should align with your research objectives and the nature of your data.

Additionally, you have the option to specify "both" for the "SKIP_RAREFACTION" parameter. When you choose this option, you can proceed with Step 8 and the pipeline will automatically generate result files for both rarefied and unrarefied data at each step. These files will be separated and clearly identifiable in their filenames. This approach provides flexibility and allows you to explore both rarefied and unrarefied data, facilitating more comprehensive analyses and comparisons as needed.

Q2Pipe Step 8: Rarefaction Curve (optional but strongly recommended if you rarefy)

This step can be safely skipped if you do not intend to rarefy your data. You can also launch it whatever you are rarefying your data or not.

Step description

The primary purpose of this step is to create a rarefaction curve, which assists in determining the appropriate sampling depth for subsequent steps, specifically Step 9 and Step 10. In your option file, you should specify a relevant "**p_max_depth**" parameter to guide the rarefaction curve generation.

It's worth noting that you have the flexibility to relaunch this step with different "**p_max_depth**" parameters if needed. However, please be aware that doing so will overwrite the previous curve. If you wish to retain multiple rarefaction curves for reference, remember to rename the "ANALYSIS_NAME.rarefaction_curves_filtered.qzv" file to something distinct before relaunching the step.

Running command

To launch this step, use the following command:

```
$Q2P/q2pipe_step8_rarefaction_curve.sh optionfile_q2pipe_default.txt
```

You can now visualize the rarefaction curve by using the "rarefaction_curves_filtered.qzv" file.

Q2Pipe Step 9: Rarefy data (mandatory if you rarefy)

You can skip this step if you do not intend to rarefy your data.

Step description

The objective at this step is to identify the optimal balance between minimizing sample loss (any samples with less features than the rarefaction level will be removed) and preserving the biological richness of the dataset. The feature count for each sample can be found in the "filtered_table.qzv" file.

In summary, the goal is to select a rarefaction level situated within a stable region of the rarefaction curve. This ensures that the chosen level is neither excessively high, which could result in the loss of important samples, nor too low, which might compromise the representation of biological diversity. Achieving this balance is essential for obtaining reliable and meaningful results in subsequent analysis steps.

Running command

To launch this step, use the following command:

```
$Q2P/q2pipe_step9_rarefy.sh optionfile_q2pipe_default.txt
```

Q2Pipe Step 10: Metrics generation (mandatory)

Step description

In this step, various metrics are generated based on the user's choices, specifically the "alpha_metrics" and "beta_metrics" parameters in the option file. The metrics generated depend on these selections. If you have rarefied your data, the step will use the rarefied feature table for these calculations. If not, it will use the taxa-filtered table.

For a comprehensive list of available metrics, you can refer to the following links:

- Alpha Diversity Metrics: Alpha Diversity Metrics List
- Beta Diversity Metrics: Beta Diversity Metrics List

It's essential to ensure that your metric choices are selected from these lists to prevent any issues during the step. Using metrics outside of these lists may lead to errors and crashes. If you encounter such issues, it's advisable to delete the step's output folder before retrying with valid metric selections.

By default, Q2Pipe generates the same diversity metrics as the "core-metrics" command in Qiime2.

Multithreading

The metrics generation step in Q2Pipe is designed to leverage multithreading so that multiple threads can be used to simultaneously generate diversity metrics, significantly reducing the time required for this step.

Each metric generation process is independent of the others, and Q2Pipe will assign one thread per metric. It will continue to allocate threads for each metric generation process until it reaches the specified "NB_THREADS" parameter. After reaching this thread limit, the pipeline will wait for a process to finish before starting another. This multithreading approach optimizes the use of computational resources and expedites the analysis, making the metric generation process more efficient.

Running command

To launch this step, use the following command:

```
$Q2P/q2pipe_step10_metrics.sh optionfile_q2pipe_default.txt
```

This step will generate qza and qzv files according to you selected metrics and will export the qza files into a human readable format (usually TSV)

Q2Pipe Step 11: Exportation (mandatory)

Step description

This step plays a crucial role in generating important result files and may include secondary modules for more advanced result generation tasks, such as R-scripts and file formatting. The generated files encompass:

1. ASV Tables: These tables contain the occurrence count and taxonomical information for each sample for each ASV/Feature, providing insights into the distribution of features across samples.

2. "Seqs" ASV Table: This specific table has the same information as the regular ASV table but contains the DNA sequence of each ASV/Feature. This information can be valuable for downstream analyses that require sequence-level data.

These files collectively serve as the foundation for subsequent analyses and offer comprehensive information about the features and taxonomic composition of your microbiome or metagenomic dataset.

In addition to the core functionalities, this step also includes optional modules that you can choose to use based on your specific analysis requirements, including ANCOM analysis and FUNGuild (more to come)

ANCOM Analysis module

ANCOM (Analysis of Composition of Microbiomes) is a statistical method for identifying differentially abundant features in microbiome data. This module allows you to perform ANCOM analysis to uncover significant differences in feature abundance across groups or conditions.

To execute the module, you can specify the relevant metadata columns (comma-separated list) that you want to use for generating ANCOM graphs using the "m_metadata_column" parameter in the option file. Alternatively, you can choose to skip this step altogether by setting "GENERATE_ANCOM" to false.

When the ANCOM analysis is executed, the results will be stored in a dedicated folder named "ANALYSIS_NAME_ANCOM." This folder will contain the ANCOM analysis outputs, including graphs and other relevant data, providing you with insights into differential abundance patterns across your metadata categories.

FUNGuild (ITS functional annotation) module

Within this step, you have the option to utilize FUNGuild for the functional annotation of fungal ITS sequences. To use FUNGuild, it's necessary to install the custom FUNGuild package, which you can access from the following link: Custom FUNGuild Repository.

When installing the custom FUNGuild package, make sure it is accessible from anywhere on your system. **If you're using the Apptainer installation of Q2Pipe, there's no need to perform this installation step separately because FUNGuild is already available within Apptainer.**

However, if you are using the anaconda Qiime2 installation, you must follow these steps to install FUNGuild:

Please note that if you are using the Anaconda version of Qiime2, you may need to repeat these steps each time you update Qiime2.

Clone the repository on your machine:

```
git clone https://github.com/Patg13/FUNGuild.git
cd FUNGuild
```

Copy the guild script to your current qiime2 environment (replace 202X.X by your current version of Qiime2)

```
cp Guilds_v1.1.py /home/Ubuntu/anaconda3/envs/qiime2-202X.X/bin
chmod 775 /home/Ubuntu/anaconda3/envs/qiime2-202X.X/bin/Guilds_v1.1.py
```

NOTE: In the anaconda version, you must redo these steps after every Qiime2 update

Once you have successfully installed FUNGuild, you can activate the "GENERATE_FUNGUILD" option in your Q2Pipe option file and specify the database you wish to use. It's essential to use this custom version of FUNGuild as it supports the use of a local database, which is typically more convenient for analysis. However, if you prefer to use the original version, you can specify "fungi" as the database name but ensure that the "guild" script has internet access since the original version relies on internet connectivity for its functionality.

Results extraction form

The result extraction feature in Q2Pipe is designed to simplify the process of automatically isolating specific important results from an analysis. To use this feature, follow these steps:

1. Fill out the Q2Pipe result extraction form, which is provided in the repository. In this form, you can select the results files you want to isolate during the last step of Q2Pipe.
2. Save the completed form in TSV format (tabulation-separated values).
3. Adjust the "EXTRACTION_FORM_PATH" option in your Q2Pipe option file to point to the location of the saved TSV file.
4. After the analysis is complete, a folder named "ANALYSIS_NAME_form_extraction" will be created. This folder will contain the results files you selected in the form, along with a partial form that explains the content of each of these files.

By using this feature, you can efficiently extract and organize specific results files that are most relevant to your analysis, streamlining the post-analysis data retrieval process.

Running command

To launch this step, use the following command:

```
$Q2P/q2pipe_step11_export.sh optionfile_q2pipe_default.txt
```

Q2Pipe Fullauto mode

The "fullauto" method in Q2Pipe is primarily intended for preliminary analysis or reproducing a completed analysis efficiently. This method utilizes a special script that automates the entire pipeline by calling each step sequentially using a single comprehensive option file.

Key features of the "fullauto" method include:

1. Checkpointing: The script has built-in checkpointing functionality, allowing it to skip steps that have already been completed. This is particularly useful in case of errors or interruptions during analysis.
2. Step Invalidation: If you need to restart an analysis from a specific point, the script can delete checkpoint files associated with previous steps, ensuring a clean restart.
3. Step-Level Control: Each step in the pipeline generates a q2pipe_stepX.DONE file, which the script recognizes. If you wish to rerun a particular step, you can delete the corresponding checkpoint file and relaunch the script.

To launch the fullauto script, use this command:

```
$Q2P/q2pipe_fullauto.sh optionfile_q2pipe_default.txt
```

Ensure that your option file is complete, including details such as the manifest, metadata, classifier, Apptainer command, and other necessary parameters.

Additionally, you can specify at which step the "fullauto" mode should begin by adding the step number at the end of the command, like this:

```
$Q2P/q2pipe_fullauto.sh optionfile_q2pipe_default.txt 8
```

will skip steps 1-7 and initiate the pipeline from step 8 onwards, without checking for checkpoint files or file validity. This approach can be particularly useful when you have manually launched a partial analysis and wish to automate the remaining steps, such as running rarefaction steps multiple times with different depth values.

By specifying the starting step number, you can easily customize the workflow to suit your specific analysis requirements, making the "fullauto" script a flexible tool for automating Qiime2 analyses efficiently.

Q2Pipe Tools

This section encompasses various tools that exist outside of the Q2Pipe workflow but can still offer valuable insights or execute specific operations on your sample data.

Denoise evaluation

Relevant options (excluding Qiime2 parameters)

DENOISE_EVALUATION_SAMPLE_SIZE: The step will randomly select this number of samples from the evaluation manifest file to generate your test subset (way faster then doing the test on everything)

EVAL_MANIFEST_FILE_PATH: Manifest file to use for checking

TESTFILE_PATH: Full path to your test file

FORCE_RESAMPLING: If you relaunch the step, it will check if a subset already exist ($ANALYSIS_NAME.eval_manifest.temp) and use it. If you switch this option to true, it will erase it and create a new one

If you prefer to manually select the subset of samples yourself, you can follow these steps:

1. Set the "DRY_RUN" parameter to true in your option file.
2. Launch the corresponding step. This will create a temporary evaluation manifest file named "YOUR_ANALYSIS_NAME.eval_manifest.temp."
3. Remove unwanted samples from this temporary manifest file. Keep the first line intact.
4. Open your original manifest file and copy the samples you want to include in the new file. Make sure to copy both the forward and reverse samples for each.
5. The pipeline will take care of the rest but ensure that the "FORCE_RESAMPLING" parameter is set to false.

This approach allows you to manually curate the subset of samples you want to include in your analysis. It can be particularly useful for fine-tuning trimming parameters before performing the full denoising step, as it helps you evaluate the impact of different parameters on a smaller subset of data.

To create a test file, follow the syntax provided in your data, ensuring that it includes the necessary information about samples, forward and reverse sequences, and any other relevant details for your analysis.

```
test_param:--p-trim-left-f 19 --p-trim-left-r 20 --p-trunc-len-f 218 --p-trunc-len-r 212 --p-max-ee-f 2 --p-max-ee-r 2 --p-n-reads-learn 99000
test_param2:--p-trim-left-f 18 --p-trim-left-r 20 --p-trunc-len-f 218 --p-trunc-len-r 212 --p-max-ee-f 2 --p-max-ee-r 2 --p-n-reads-learn 99000
test_param3:--p-trim-left-f 17 --p-trim-left-r 20 --p-trunc-len-f 218 --p-trunc-len-r 212 --p-max-ee-f 2 --p-max-ee-r 2 --p-n-reads-learn 99000
test_param4:--p-trim-left-f 16 --p-trim-left-r 20 --p-trunc-len-f 218 --p-trunc-len-r 212 --p-max-ee-f 2 --p-max-ee-r 2 --p-n-reads-learn 99000
#mjb_param11:--p-trim-left-f 22 --p-trim-left-r 20 --p-trunc-len-f 218 --p-trunc-len-r 212 --p-max-ee-f 2 --p-max-ee-r 2 --p-n-reads-learn 990000
```

When creating a test file, it's essential to ensure that each test within the file has a unique name to prevent result overwriting. You have the flexibility to use any parameters that are compatible with the DADA2 **denoise-paired** step. However, please be cautious and avoid modifying input and output parameters, as altering these can lead to program crashes and unintended behavior.

Focus on adjusting parameters related to denoising, trimming, filtering, and other DADA2-specific options to fine-tune the behavior of this step during your analysis. This allows you to experiment with different settings on a smaller subset of data before applying them to the full denoising step, helping you optimize your analysis process effectively.

Once your test file is ready, you can launch the step using this command:

```
$Q2P/q2pipe_step1.5_denoise_evaluation.sh optionfile_q2pipe_default.txt
```

Each test you perform using the test file will generate a series of result files in the "YOUR_ANALYSIS_NAME_DenoiseTest_Results" folder. These result files include QZA, QZV, and TSV files, which contain various data and metrics related to the denoising step for each test.

To identify which test parameters performed the best, you can consult these result files and analyze the metrics and data they contain. You can compare the performance of different parameter settings to determine which one meets your analysis goals.

As an additional note, there is mention of an "AutoPlotting" script (UNFINISHED FEATURE) that can be used to automatically generate a series of graphs from the TSV files to visually identify the best results. However, it's important to check if this feature is available and functional in the current version of Q2Pipe.

Once you've identified the best parameters that yield optimal results, you can input these parameters into your option file and proceed with step 2 of your analysis. This iterative testing and optimization process helps ensure that you fine-tune your analysis for the best outcomes.

References

Bolyen, E., Rideout, J. R., Dillon, M. R., Bokulich, N. A., Abnet, C. C., Al-Ghalith, G. A., . . . Caporaso, J. G. (2019). Reproducible, interactive, scalable and extensible microbiome data science using QIIME 2. *Nature Biotechnology, 37*(8), 852-857. Retrieved 10 30, 2023, from https://nature.com/articles/s41587-019-0209-9

Brandine, G. d., & Smith, A. D. (2021). Falco: high-speed FastQC emulation for quality control of sequencing data. *F1000Research*, 8:1874.

Weinstein, M. M., Prem, A., Jin, M., Tang, S., & Bhasin, J. M. (2019). FIGARO: An efficient and objective tool for optimizing microbiome rRNA gene trimming parameters. *bioRxiv*, 610394. Retrieved 10 30, 2023, from https://biorxiv.org/content/10.1101/610394v1

Licence

MIT License