Quantitative Metabarcoding Analysis Using Claident

Akifumi S. Tanabe (Graduate School of Life Sciences, Tohoku University)

2025-05-24

1 Quantitative Metabarcoding Analysis Using Claident

Claident is a suite of sequence-analysis programs for metabarcoding and DNA barcoding that I develop and maintain. The name is pronounced "Clai-den" (the final *t* is silent) and is derived from "CLAssification" and "IDENTification". Its main differences from the MiFish pipeline (Sato *et al.* 2018; Zhu *et al.* 2023) are:

- Supports data not only from fish metabarcodes amplified with the MiFish primers (Miya *et al.* 2020, 2015) but from any gene locus of any organism or virus.
- Handles both non-quantitative and quantitative metabarcoding (Ushio *et al.* 2018a).
- Offers more flexible and detailed analyses.
- Provides no web service; all processing will be done on your own computer.
- Requires more prerequisite knowledges and equipments to use.

This textbook explains how to install Claident and how to perform quantitative metabarcoding using internal-standard DNA. A support page for this textbook is available at:

- https://github.com/astanabe/eDNAmanual

Sample data, example files, and this manuscript are placed there.

For additional information on Claident, see:

- https://www.claident.org/

The instructions below assume you are **already comfortable working in a Linux or macOS terminal environment**. If not, please acquire the necessary skills beforehand.

1.1 Operating Environment and Installation of Claident

Claident is designed to run on:

- Debian 11 or later (including WSL environment on Windows)
- Ubuntu 20.04 or later (including WSL environment on Windows)

- Linux Mint 20 or later
- Red Hat Enterprise Linux 8 or later
- AlmaLinux 8 or later (including WSL environment on Windows)
- Rocky Linux 8 or later
- macOS with Homebrew installed
- macOS with MacPorts installed

If you use Windows, install WSL and Ubuntu by following Microsoft's official instructions:

- https://learn.microsoft.com/ja-jp/windows/wsl/install

Note that Ubuntu on Windows can use at most about 250 GB of disk space and only half of the physical memory (as of this writing). Large-scale analyses may therefore exceed those limits. Whenever possible, use a dedicated analysis machine with plenty of RAM and a fast SSD, because species identification with large reference databases is both memory- and I/O-intensive.

Claident cannot be installed correctly if environment-modifying software such as Anaconda or Miniconda is active. Temporarily disable them, or log in with a user account where they are not used, before installation.

On Debian / Ubuntu / Linux Mint / Ubuntu on Windows, run the following commands in a terminal to install Claident:

```
sudo apt install wget
mkdir temporary
cd temporary
wget https://www.claident.org/installClaident_Debian.sh
wget https://www.claident.org/installOptions_Debian.sh
wget https://www.claident.org/installUCHIMEDB_Debian.sh
wget https://www.claident.org/installDB_Debian.sh
sh installClaident_Debian.sh
sh installOptions_Debian.sh
sh installUCHIMEDB_Debian.sh
sh installDB_Debian.sh
cd ..
rm -rf temporary
```

On macOS, first install Homebrew as described at the following URL:

- https://brew.sh/

Then execute the following commands on Terminal to install Claident:

```
brew install wget
mkdir temporary
cd temporary
wget https://www.claident.org/installClaident_macOSHomebrew.sh
wget https://www.claident.org/installOptions_macOSHomebrew.sh
wget https://www.claident.org/installUCHIMEDB_macOSHomebrew.sh
wget https://www.claident.org/installDB_macOSHomebrew.sh
sh installClaident_macOSHomebrew.sh
sh installOptions_macOSHomebrew.sh
```

```
sh installUCHIMEDB_macOSHomebrew.sh
sh installDB_macOSHomebrew.sh
cd ..
rm -rf temporary
```

If you can reach the outside network only through a proxy server, set the environment variables like below before running the installation scripts:

```
export http_proxy=http://proxyaddress:portnumber
export https_proxy=http://proxyaddress:portnumber
export ftp_proxy=http://proxyaddress:portnumber
```

If the proxy requires password authentication, run the following instead:

```
export http_proxy=http://username:password@proxyaddress:portnumber
export https_proxy=http://username:password@proxyaddress:portnumber
export ftp_proxy=http://username:password@proxyaddress:portnumber
```

By default, the scripts will install Claident under "`/usr/local`". To install Claident to another location, set the `PREFIX` environment variable beforehand like below:

```
export PREFIX=/home/tanabe/claident20240101
```

Claident will then be installed under "`/home/tanabe/claident20240101`". If you change the installation path, "`install_path/bin`", which contains the executable commands of Claident, is not on your `PATH` environment variable, and analysis commands of Claident cannot be executed without the full path. Therefore, you need to add it to PATH environment variable before running Claident as the following:

```
export PATH=/home/tanabe/claident20240101/bin:$PATH
```

If typing this every time is inconvenient, append the line to the end of "`~/.bashrc`" (or your shell's startup file); it will then run automatically whenever you open a terminal.

Using different installation directories lets you keep multiple Claident versions on the same machine. Each command refers to the configuration file "`~/.claident`", so switching versions also requires replacing that file. A template is located at "`install_path/share/claident/.claident`"; copying it over "`~/.claident`" completes the switch. If you truly need multiple versions, it is often simplest to create separate user accounts, install each version within the respective home directory, and switch users when you wish to change versions.

1.2 Overall Workflow and Prerequisites for Data Analysis

Data analysis with Claident is carried out in the following steps.

1. Demultiplexing
2. Merging paired-end reads
3. Removal of low-quality reads (Edgar & Flyvbjerg 2015)
4. Denoising (Callahan *et al.* 2016)

5. First chimera removal (Edgar 2016; Edgar *et al.* 2011; Rognes *et al.* 2016)
6. Clustering of internal-standard sequences (Edgar 2010; Rognes *et al.* 2016)
7. Second chimera removal (Edgar *et al.* 2011; Rognes *et al.* 2016)
8. Removal of index-hopping (Esling *et al.* 2015)
9. Decontamination using negative controls
10. Molecular identification (Tanabe & Toju 2013)
11. Construction and processing of the OTU composition table
12. Coverage-based rarefaction (Chao & Jost 2012)
13. Estimation of DNA concentration using the read counts of internal-standard DNA (Ushio *et al.* 2018a)

The final OTU table can be visualized, summarized, and tested in R or another statistical environment. Claident itself does not provide statistical analysis functions for OTU tables.

Claident can handle most metabarcoding data, but in this textbook I assume data that meet the conditions below. Data that do not meet all of them can still be analyzed, but are outside the scope of this explanation.

- Environmental DNA samples obtained by filtering water and extracting DNA from the filter, together with field blanks as negative controls.
- Library preparation as follows:
  - Add several internal-standard DNAs of known concentration and perform tailed PCR with the MiFish primers (1st PCR).
  - Use the 1st-PCR product as template and perform tailed PCR with index primers (2nd PCR).
  - The resulting library is a dual-index library with an index at each end.
- Pool the 2nd-PCR products of all samples and sequence them on an Illumina sequencer **using a dedicated run or an entire lane**.
  - Paired-end sequencing with overlap, so the reads can be merged.
  - The run data including .bcl files or undemultiplexed FASTQ files including the index reads are available.

Accordingly, for every sample and blank you must know the following:

- Whether it is a sample or a blank
- Volume of filtered water
- Volume of extracted DNA solution (the buffer volume used in the final elution, not the recovered volume)
- Internal-standard DNA sequences
- Concentration of internal-standard DNA
- The part of the 1st-PCR primers that is read first by the sequencer
- The part of the 2nd-PCR primers that is read as the index

For cases such as single-end reads, paired-end reads without overlap, libraries prepared without adding internal-standard DNA, or only demultiplexed FASTQ files being available, please refer to the shell scripts on the following page.

- https://github.com/astanabe/ClaidentTutorial

If you have no field blanks or too few of them, you can substitute extraction blanks or 1st-PCR blanks; however, you cannot combine field blanks with other blanks. **You need at least 10 blanks** — that is, ten or more field blanks, **or** ten or more extraction blanks, **or** ten or more 1st-PCR blanks.

Primers for the 1st PCR have already been developed, such as MiFish (Miya *et al.* 2020, 2015), MiDeca (Komai *et al.* 2019), MiMammal (Ushio *et al.* 2017), MiBird (Ushio *et al.* 2018b), Amph16S (Sakata *et al.* 2022), and MtInsects-16S (Takenaka *et al.* 2023). Choose an appropriate primer set according to the target taxon. If you try to design new primers, collect sequences for the target taxon and locus from public databases, find a highly conserved region flanking a sufficiently variable region of suitable length, and design primers there. In addition, it is common to append about six `N` bases to the beginning of each 1st-PCR primer so that the sequencer starts with a diverse base composition; otherwise, the fluorescence signal may saturate on Illumina sequencers. Some oligo synthesis services generate primers in which most of the `N` become `T`, so choose the vendor carefully.

Ready-made index primers for the 2nd PCR are available from Illumina and other suppliers. Some sets are also provided at the URLs below.

- https://github.com/astanabe/TruSeqStyleIndexPrimers
- https://github.com/astanabe/NexteraStyleIndexPrimers

Because a low base diversity in the index region also causes sequencing errors, examine the combinations of indices carefully. Ideally, the proportion of `A/C` to `G/T` should be close to 1:1 at every position, and caution is required when only a few samples are pooled. To use Claident to detect and remove index-hopping, each sample needs "10 or more unused index combinations that share one of the two indices".

Prepare the internal-standard DNA solution by dissolving synthesized DNA in TE buffer and adjusting it to the desired concentration by absolute quantitation with a fluorescent assay or digital PCR, followed by dilution and mixing. Double-stranded DNA synthesis services include Strings DNA Fragments (ThermoFisher) and gBlocks (Integrated DNA Technologies). To design an internal-standard DNA sequence, collect insert regions that can be amplified by the chosen primer set, mutate more than 10 % of the bases at random while keeping the GC content unchanged, and attach the primer sequences to both ends. Ideally, the sequence should differ by $\geq$ 10 %, preferably $\geq$ 15 %, from any known organism. Example sequences for the MiFish primers are listed in Appendix S1 of Ushio *et al.* (2022).

From here on I assume **data from one dedicated run or one dedicated lane**. If you have data from multiple runs or lanes, perform the analysis up to index-hopping removal **separately for each run or lane**, and then merge the results with the `clclassseqv` command.

1.2.1 About "Sample IDs" in Claident

The concept of a sample ID in Claident is as follows. In metabarcoding you may sequence the same primer product of the same sample in multiple runs, or multiple primer products of the same sample in one sequencing run. To distinguish these cases, write the sample ID in the form:

```
RunID__MaterialID__PrimerID
```

RunID is any string that you will specify later as an option of the analysis commands and should identify the sequencing run or lane. PrimerID is a string specified in one of the files described later and identifies the primer set used (e.g. `MiFish` for MiFish primer set). MaterialID is the usual sample ID assigned to the physical material. If the MaterialID is the same while RunID or PrimerID differs, the underlying template DNA is the same. Because the physical sample and Claident's notion of a sample are not necessarily one-to-one, this naming scheme lets you see which physical sample corresponds to a sample ID at a glance.

If you prepare technical replicates, treat them as separate samples when they remain distinct throughout DNA extraction, library preparation, and sequencing; otherwise treat them as one sample. When treating them as separate, it is useful to append `-R1`, `-R2`, and so on to the rnd of MaterialID to indicate the replicate.

Note that `__` (two or more consecutive underscores) cannot appear in RunID, PrimerID, or MaterialID. Allowed characters are alphanumerics, hyphens, and underscores only. Using other characters may cause unexpected errors.

1.2.2 About OTUs and ASVs

Amplicon Sequence Variant (ASV) or Exact Sequence Variant (ESV) is a set of sequences that are identical — or inferred to be identical — in all positions. Operational Taxonomic Unit (OTU), as the name implies, is any classification unit defined at the analyst's discretion. It is a common misconception that OTUs are "units obtained by clustering sequences at a given similarity", but that contradicts the literal meaning of the term. If you decide to analyze ASVs as your classification units, those ASVs are OTUs. In Claident, OTU is ASV in most cases, but in the following, I will use OTU wherever OTU may not coincide with ASV.

1.2.3 Required Files and Directory Structure

This section explains the files that must be prepared before analysis. File names are arbitrary, but the names assumed in later commands are shown.

1.2.3.1 Blank List (blanklist.txt)

This is a text file in which one blank's sample ID is written per line. It must be written in the following format.

```
RunID__BlankMaterialID1__PrimerID
RunID__BlankMaterialID2__PrimerID
RunID__BlankMaterialID3__PrimerID
```

Claident recognizes the items listed in this file as blanks. You may omit RunID and PrimerID and write them in the following format.

```
BlankMaterialID1
BlankMaterialID2
BlankMaterialID3
```

1.2.3.2 Filtered Water Volume Table (watervoltable.tsv)

This is a tab-delimited text file in which one sample ID and the filtered water volume are written per line, separated by tab characters. If multiple filters were used and you want to record them separately, list multiple numbers separated by tabs. (They are summed when concentration is estimated.)

```
RunID__SampleMaterialID1__PrimerID  1000  1000
RunID__SampleMaterialID2__PrimerID  1000  500
RunID__SampleMaterialID3__PrimerID  1500
RunID__BlankMaterialID1__PrimerID   500
RunID__BlankMaterialID2__PrimerID   500
RunID__BlankMaterialID3__PrimerID   500
```

You may omit RunID and PrimerID and write them as follows.

```
SampleMaterialID1  1000  1000
SampleMaterialID2  1000  500
SampleMaterialID3  1500
BlankMaterialID1   500
BlankMaterialID2   500
BlankMaterialID3   500
```

These numbers are used to estimate the DNA concentration in the original environmental water sample. Any unit may be used, but milliliters are recommended unless there is a special reason. You can append any string after a trailing tab, so you may write the unit there. However, the program does not convert different units to a common unit.

1.2.3.3 Extracted DNA Solution Volume Table (solutionvoltable.tsv)

This is a tab-delimited text file in which one sample or blank ID and the volume of the extracted DNA solution are written per line, separated by tab characters. If multiple filters were used and multiple DNA solutions were obtained, list multiple numbers separated by tabs. (They are summed when concentration is estimated.)

```
RunID__SampleMaterialID1__PrimerID  200  200
RunID__SampleMaterialID2__PrimerID  200  200
RunID__SampleMaterialID3__PrimerID  200
RunID__BlankMaterialID1__PrimerID   200
RunID__BlankMaterialID2__PrimerID   200
RunID__BlankMaterialID3__PrimerID   200
```

You may omit RunID and PrimerID and write them as follows.

```
SampleMaterialID1  200  200
SampleMaterialID2  200  200
SampleMaterialID3  200
BlankMaterialID1   200
BlankMaterialID2   200
BlankMaterialID3   200
```

These numbers are used to estimate the total number of DNA copies in the extracted DNA solution. Any unit may be used, but microliters are recommended unless there is a special reason. You can append any string after a trailing tab, so you may write the unit there. However, the program does not convert different units to a common unit.

Note that this is the buffer volume used for DNA elution, not the volume actually recovered in DNA extraction. For example, if you add 200 µL of elution buffer to a spin column or magnetic beads and finally obtain 190 µL of DNA extract, ten more microliters of DNA extract actually exist but were not recovered, so 200 µL should be written here. It is assumed that the wash buffer used before DNA elution has been completely removed.

1.2.3.4 Internal-Standard DNA Sequences (standard.fasta)

This is a FASTA file containing the internal-standard DNA sequences. Multiple sequences can be listed. Below is an example FASTA file containing four internal-standard DNA sequences.

```
>MiFish_STD_01
CACCGCGGTTATACGACAGGCCCAAGTTGAACGCAGTCGGCGTAAAGAGTGGTTAAAAG...
>MiFish_STD_02
CACCGCGGTTATACGACAGGCCCAAGTTGATCTTGAACGGCGTAAAGAGTGGTTAGATT...
>MiFish_STD_03
CACCGCGGTTATACGACAGGCCCAAGTTGAAGCGACGCGGCGTAAAGAGTGGTTATCAC...
>MiFish_STD_04-2
CACCGCGGTTATACGACAGGCCCAAGTTGAGATCCCACGGCGTAAAGAGTGGTTAGAAC...
```

Internal-standard DNAs are identified based on these sequences. The sequences may include or omit the primer-annealing regions that were used when ordering from the synthesis service. Sequence names must match those in the internal-standard DNA concentration table described below.

1.2.3.5 Internal-Standard DNA Concentration Table (stdconctable.tsv)

This is a tab-delimited text file in which the concentration of each internal-standard DNA added in the 1st PCR is listed for each sample. Prepare a table like the following.

```
samplename                          MiFish_STD_01 MiFish_STD_02 MiFish_STD_03 MiFish_STD_04-2
RunID__SampleMaterialID1__PrimerID 5             10            20            40
RunID__SampleMaterialID2__PrimerID 5             10            20            40
RunID__SampleMaterialID3__PrimerID 5             10            20            40
RunID__BlankMaterialID1__PrimerID  5             10            20            40
RunID__BlankMaterialID2__PrimerID  5             10            20            40
RunID__BlankMaterialID3__PrimerID  5             10            20            40
```

You may omit RunID and PrimerID and write them as follows.

```
samplename        MiFish_STD_01 MiFish_STD_02 MiFish_STD_03 MiFish_STD_04-2
SampleMaterialID1 5             10            20            40
SampleMaterialID2 5             10            20            40
SampleMaterialID3 5             10            20            40
BlankMaterialID1  5             10            20            40
```

```
BlankMaterialID2  5           10          20          40
BlankMaterialID3  5           10          20          40
```

The unit of concentration is copies per microliter. This assumes that equal volumes of sample DNA solution and internal-standard DNA solution were added for the 1st PCR. Therefore, if twice the volume of internal-standard DNA solution as sample DNA solution was added, double the numbers. If the sample DNA solution was diluted tenfold and an equal volume of internal-standard DNA solution was added, multiply the numbers by ten. If both the sample DNA solution and the internal-standard DNA solution were diluted tenfold before mixing, use the numbers as they are. Internal-standard DNA names must match the names in the internal-standard DNA sequence file.

1.2.3.6 Primer Regions Read First by the Sequencer (forwardprimer.fasta and reverseprimer.fasta) These FASTA files contain the parts of the forward and reverse primers of the 1st PCR that are actually sequenced. Remove the regions where the index primers of the 2nd PCR anneal so that only the regions read by the sequencer remain. For example, if MiFish-U-F `ACACTCTTTCCCTACACGACGCTCTTCCGATCTNNNNNNGTCGGTAAAACTCGTGCCAGC` is used as the forward primer in the 1st PCR, write `NNNNNNGTCGGTAAAACTCGTGCCAGC` as the sequence. Multiple primer sequences can be listed in each file, but the first forward primer is paired only with the first reverse primer, and combinations with the remaining reverse primers are not considered. Degenerate base codes such as `R`, `Y`, `M`, `K`, and `N` can be used in the sequences. When slightly different primer sequences such as MiFish variants are mixed, align them and write the degenerate consensus sequence. For example, if MiFish-E-v2, MiFish-U and MiFish-U2 are mixed, the contents of the forward primer sequence file "`forwardprimer.fasta`" will be as follows.

```
>MiFish
NNNNNNNGYYGGTAAAWCTCGTGCCAGC
```

The sequences used to build this degenerate consensus are shown below (aligned for readability).

```
>MiFish-E-F-v2
NNNNNNRGTTGGTAAATCTCGTGCCAGC
>MiFish-U-F
 NNNNNNGTCGGTAAAACTCGTGCCAGC
>MiFish-U2-F
 NNNNNNGCCGGTAAAACTCGTGCCAGC
```

The reverse primer file "`reverseprimer.fasta`" will be as follows.

```
>MiFish
NNNNNNNCATAGKRGGGTRTCTAATCCYMGTTTG
```

The sequences used to build this degenerate consensus are shown below (aligned for readability).

```
>MiFish-E-R-v2
NNNNNNGCATAGTGGGGTATCTAATCCTAGTTTG
>MiFish-U-R
 NNNNNNCATAGTGGGGTATCTAATCCCAGTTTG
>MiFish-U2-R
 NNNNNNCATAGGAGGGTGTCTAATCCCCGTTTG
```

The sequence names in these files are used as the PrimerID portion of the Claident's sample IDs, so they must match the PrimerID used in the other files described above.

1.2.3.7 Index Regions (index1.fasta and index2.fasta) These FASTA files must contain only the index regions of the index primers used in the 2nd PCR. Index2 (the i5 index) lies in the forward index primer, and its read orientation depends on the instrument. Index1 (the i7 index) lies in the reverse index primer and is read in the orientation opposite to the primer sequence ordered. The index sequences listed in "`SampleSheet.csv`" for Illumina sequencers are already standardized for read orientation, so you can copy them directly. The reverse index file "`index1.fasta`" will look like the following.

```
>SampleMaterialID1
ACCTGCAA
>SampleMaterialID2
GTTCCTTG
>SampleMaterialID3
CCAGATCT
>BlankMaterialID1
AAGTGTGA
>BlankMaterialID2
CCATGATC
>BlankMaterialID3
TCATGTCT
```

The forward index file "`index2.fasta`" is almost the same except for the sequences. Ensure that the sequence names correspond to the MaterialID and that the order of the sequences is exactly the same.

To apply the index-hopping removal function, **information on every index used in the sequencing run is required**. Even if some samples or negative controls were discarded after problems were discovered during preparation or sequencing, the data for those samples contain information needed for index-hopping removal. Therefore, do not delete such data until index-hopping removal is finished.

1.2.3.8 Undemultiplexed FASTQ When you outsource sequencing, you usually receive demultiplexed FASTQ files based on "`SampleSheet.csv`". However, Illumina's demultiplexing program may fail when too many samples are multiplexed in one run or one lane, does not take index base quality into account, allows single mismatches, and discards reads with unused index combinations, making index-hopping detection impossible. For these reasons, demultiplexing with Claident's built-in program `clsplitseq` is recommended. To demultiplex with `clsplitseq`, install Illumina's BCL Convert on a Linux machine and generate undemultiplexed FASTQ files that include the index reads from the run data. Locating the working directory on a fast SSD is strongly recommended.

You can download BCL Convert from the following URL.

- https://jp.support.illumina.com/sequencing/sequencing_software/bcl-convert.html

The latest version at the time of writing is v4.3.6. Up to v4.2.4, anyone could download the software, but v4.3.6 requires user registration and login to Illumina Basespace, a cloud service provided by Illumina, and the serial number of the Illumina sequencer being used to download the software. If you have a FASTQ file obtained from an Illumina sequencer, you can open the FASTQ file with a text editor or `less` command and find the serial number of the sequencer in the sequence data (the string between "@" and the first ":" in the line starting with "@" indicating the sequence name is the serial number of the sequencer), so you can use it to download v4.3.6. In most cases, v4.2.4 will work fine, so the following instructions are for v4.2.4, but for v4.3.6, there are no differences other than the version number.

On Debian, Ubuntu, Linux Mint, or Ubuntu on Windows, download the distribution file for Oracle Linux 8 labeled "(Oracle 8)" and place it in the working directory "`workingdirectory`" and execute the following commands in a terminal to install it.

```
sudo apt install rpm2cpio cpio pstack
cd workingdirectory
mkdir temporary
cd temporary
rpm2cpio ../bcl-convert-4.2.4-2.el8.x86_64.rpm | cpio  -id
sudo mkdir -p /usr/local/bin
sudo cp usr/bin/bcl-convert /usr/local/bin/
sudo mkdir -p /var/log/bcl-convert
sudo chmod 777 /var/log/bcl-convert
cd ..
rm -rf temporary bcl-convert-4.2.4-2.el8.x86_64.rpm
```

This program is not available for macOS. To run it on macOS, install a virtual machine, install Ubuntu on it, and then install BCL Convert to that Ubuntu system.

To generate undemultiplexed FASTQ files with BCL Convert, create a file named "`Dummy.csv`" containing the following with a text editor.

```
[Header]
FileFormatVersion,2
[BCLConvert_Settings]
OverrideCycles,Y150N1;I8;I8;Y150N1
CreateFastqForIndexReads,1
[BCLConvert_Data]
Lane,Sample_ID,index,index2
1,Dummy,CCCCCCCC,CCCCCCCC
```

The above "`Dummy.csv`" can be created without a text editor by executing the following command in the terminal.

```
echo '[Header]
FileFormatVersion,2
[BCLConvert_Settings]
OverrideCycles,Y150N1;I8;I8;Y150N1
CreateFastqForIndexReads,1
[BCLConvert_Data]
Lane,Sample_ID,index,index2
1,Dummy,CCCCCCCC,CCCCCCCC' > Dummy.csv
```

This file assumes an 8-base dual index and 151 cycles for both forward and reverse reads (the last cycles are discarded on both forward and reverse reads). If the index length or number of cycles is different, it must be changed accordingly. A dummy sample named `Dummy` with both index1 and index2 set to `CCCCCCCC` is included because BCL Convert produces an error if no sample lines are present. If **any** of the samples where both index1 and index2 are `CCCCCCCC`, index1 is `CCCCCCCC`, or index2 is `CCCCCCCC` exist, rewrite them to appropriate sequences. If no such sample where both index1 and index2 are `CCCCCCCC` exists but some reads appear in the `Dummy` FASTQ files, they are probably caused by sequencing errors and can be ignored. If the index length is not 8 bases, change both the `OverrideCycles` line and the length of `C` characters. For example, for a 10-base dual index with 301 cycles forward and reverse (discarding the last cycle), use `OverrideCycles,Y300N1;I10;I10;Y300N1` and `CCCCCCCCCC`, respectively.

Specify this file with `--sample-sheet` to disable BCL Convert's internal demultiplexing and to create undemultiplexed FASTQ files. The following command will output the undemultiplexed FASTQ files to the directory "`01_undemultiplexed`".

```
bcl-convert \
--sample-sheet Dummy.csv \
--bcl-input-directory RunDataDirectory \
--output-directory 01_undemultiplexed
```

Here, `RunDataDirectory` is the directory that contains the run data copied from the sequencer or from the analysis computer attached to the sequencer. It should contain a "`Data`" directory. Copy this directory in advance to the machine on which BCL Convert is installed. By default, BCL Convert automatically determines the number of CPUs to use (it uses all available CPUs).

The above example assumes an instrument with a single lane. For instruments with multiple lanes, increase the number of dummy sample lines in the `[BCLConvert_Data]` section of "`Dummy.csv`" to match the number of lanes, as shown below.

```
[Header]
FileFormatVersion,2
[BCLConvert_Settings]
OverrideCycles,Y150N1;I8;I8;Y150N1
CreateFastqForIndexReads,1
[BCLConvert_Data]
Lane,Sample_ID,index,index2
1,Dummy1,CCCCCCCC,CCCCCCCC
2,Dummy2,CCCCCCCC,CCCCCCCC
3,Dummy3,CCCCCCCC,CCCCCCCC
4,Dummy4,CCCCCCCC,CCCCCCCC
```

You can restrict the output to a specific lane by adding the `--bcl-only-lane` option. For example, `--bcl-only-lane 1` will output the first lane only. If you omit this option, data for all lanes will be written to separate files.

Running BCL Convert as described above will produce the following four files, in addition to the `Dummy` files, when only lane 1 is output.

`Undetermined_S0_L001_I1_001.fastq.gz` Undemultiplexed FASTQ for index1 (8 bases long)

Undetermined_S0_L001_I2_001.fastq.gz Undemultiplexed FASTQ for index2 (8 bases long)

Undetermined_S0_L001_R1_001.fastq.gz Undemultiplexed FASTQ for the forward reads of the inserts (150 bases long)

Undetermined_S0_L001_R2_001.fastq.gz Undemultiplexed FASTQ for the reverse reads of the inserts (150 bases long)

Here "L001" represents the lane number, so the file names differ for other lanes.

1.2.3.9 Directory Structure Before starting analysis with Claident, your working directory should contain the following files and directories.

- Working directory
  - blanklist.txt
  - watervoltable.tsv
  - solutionvoltable.tsv
  - standard.fasta
  - stdconctable.tsv
  - forwardprimer.fasta
  - reverseprimer.fasta
  - index1.fasta
  - index2.fasta
  - 01_undemultiplexed (directory)
    - Undetermined_S0_L001_I1_001.fastq.gz
    - Undetermined_S0_L001_I2_001.fastq.gz
    - Undetermined_S0_L001_R1_001.fastq.gz
    - Undetermined_S0_L001_R2_001.fastq.gz

1.3 Processing of Nucleotide Sequence Data

This section explains how to process the actual nucleotide sequence data. All commands should be executed in the terminal. It is assumed that the working directory is the current directory. Replace the placeholder `NumberOfCPUcores` in command options with the integer number of CPU cores to use during processing. Files that have already been explained earlier are not described again here. Several steps access the disk heavily, so processing is greatly affected if the working directory is on a slow disk. Therefore, we strongly recommend placing the working directory on a fast SSD.

1.3.1 Demultiplexing with clsplitseq

Run the following command to perform demultiplexing.

```
clsplitseq \
--runname=RunID \
--forwardprimerfile=forwardprimer.fasta \
--reverseprimerfile=reverseprimer.fasta \
--truncateN=enable \
--index1file=index1.fasta \
--index2file=index2.fasta \
--minqualtag=30 \
--compress=xz \
--seqnamestyle=illumina \
--numthreads=NumberOfCPUcores \
01_undemultiplexed/Undetermined_S0_L001_R1_001.fastq.gz \
01_undemultiplexed/Undetermined_S0_L001_I1_001.fastq.gz \
01_undemultiplexed/Undetermined_S0_L001_I2_001.fastq.gz \
01_undemultiplexed/Undetermined_S0_L001_R2_001.fastq.gz \
02_demultiplexed
```

The meaning of each command-line option is as follows.

`--runname` Give any RunID which can be set arbitrarily by the user

`--forwardprimerfile` Forward primer sequence file

`--reverseprimerfile` Reverse primer sequence file

`--truncateN` Whether to exclude the leading cluster of `N` bases in the primer sequence when calculating primer similarity

`--index1file` Reverse index sequence file

`--index2file` Forward index sequence file

`--minqualtag` Lower limit of quality value for index sequences

`--compress` Compression format to use (select from GZIP | BZIP2 | XZ | DISABLE)

`--seqnamestyle` Sequence read name/label format (select from ILLUMINA | MGI | OTHER | NOCHANGE)

After the command-line options, specify the input files and the output folder.

Note that the input files must be specified in the following order.

1. Undemultiplexed FASTQ of the forward read of the insert.
2. Undemultiplexed FASTQ of index1.
3. Undemultiplexed FASTQ of index2.
4. Undemultiplexed FASTQ of the reverse read of the insert.

This order matches the order in which an Illumina sequencer reads the data in dual-index paired-end mode.

Because this command uses primer sequences as well as index sequences for demultiplexing, it can demultiplex more finely than when only index sequences are used. Therefore, even if amplicons from other primers are mixed into the data, they can be separated as long as the primer sequences are sufficiently different.

This command also outputs sequences whose MaterialID is an “unused index combination”. Those samples will be used later in the index-hopping detection and removal step.

In the output files, the parts matching the primer sequences have been removed. Those parts are primer nucleotide sequences and are not actual biological sequences.

If the data size is large, this step takes a very long time.

If you do not have undemultiplexed FASTQ and only have demultiplexed FASTQ, you can use `cltruncprimer` instead. Except that the `--minqualtag` option is ineffective and that the input is the folder containing the demultiplexed FASTQ, the usage is the same as the `clsplitseq` command. However, the MaterialID in the index sequence file must be included in the file name of each demultiplexed FASTQ. Because demultiplexed FASTQ does not contain all sequences with "unused index combinations", index-hopping detection cannot be applied. Even if you saved sequences with unused index combinations in advance, Claident has no way to recognize them as "unused index combinations", so support is impossible.

Even when demultiplexing was performed with `clsplitseq`, index-hopping removal cannot be applied unless the library preparation ensured that each sample has ten or more "unused index combinations that share one of the two indices". To apply index-hopping removal, all of the following must be satisfied. Library preparation in which each sample has ten or more unused index combinations sharing one index, sequencing on a dedicated run or lane, and demultiplexing with `clsplitseq` must all be satisfied.

1.3.2 Concatenating Paired-End Reads with clconcatpairv

After demultiplexing, concatenate paired-end reads with the following command.

```
clconcatpairv \
--mode=ovl \
--compress=xz \
--numthreads=NumberOfCPUcores \
02_demultiplexed \
03_concatenated
```

The meaning of each command-line option is as follows.

`--mode` Specify whether the data are overlapped paired-end or non-overlapped paired-end (select from OVL | NON)

`--compress` Compression format to use (select from GZIP | BZIP2 | XZ | DISABLE)

After the command-line options, specify the input folder and the output folder.

1.3.3 Removing Low-Quality Reads with clfilterseqv

The following command calculates the expected number of errors from quality values and removes low-quality reads. (Edgar & Flyvbjerg 2015)

```
clfilterseqv \
--maxqual=41 \
--minlen=100 \
--maxlen=250 \
--maxnee=2.0 \
```

```
--maxnNs=0 \
--compress=xz \
--numthreads=NumberOfCPUcores \
03_concatenated \
04_filtered
```

The meaning of each command-line option is as follows.

`--maxqual` Upper limit for quality values.
Values above this are set to this value

`--minlen` Lower limit for sequence length

`--maxlen` Upper limit for sequence length

`--maxnee` Upper limit for expected errors

`--maxnNs` Upper limit for the number of `N` bases in a sequence

`--compress` Compression format to use (select from GZIP | BZIP2 | XZ | DISABLE)

After the command-line options, specify the input folder and the output folder.

A maximum quality value is specified here because sequences with excessively high quality values sometimes cause errors during the denoising step described below. Sequences with many expected errors or containing `N` bases are excluded for the same reason. Upper and lower limits for sequence length are determined based on the expected insert length. If you wish to decide the upper limit for expected errors and the length limits from the data, the output of the `clcalcfastqstatv` command may be useful.

1.3.4 Denoising with cldenoiseseqd

Apply denoising based on DADA2 (Callahan *et al.* 2016) with the following command.

```
cldenoiseseqd \
--pool=pseudo \
--numthreads=NumberOfCPUcores \
04_filtered \
05_denoised
```

The meaning of each command-line option is as follows.

`--pool` Specify the sample pooling method (select from ENABLE | DISABLE | PSEUDO)

After the command-line options, specify the input folder and the output folder.

Enabling pooling improves denoising efficiency, but computation time grows rapidly as the number of samples increases. Disabling pooling reduces the denoising efficiency, so we use the pseudo-pooling method provided by the DADA2 developers here. For details on pseudo-pooling, refer to the official DADA2 website.

1.3.5 First Chimera Removal with clremovechimev

The following command applies chimera removal using both *de novo* chimera detection based on the UCHIME3 algorithm (Edgar 2016) and reference-based chimera detection based on UCHIME algorithm (Edgar *et al.* 2011) implemented in VSEARCH (Rognes *et al.* 2016).

```
clremovechimev \
--mode=both \
--uchimedenovo=3 \
--referencedb=cdu12s \
--addtoref=standard.fasta \
05_denoised \
06_chimeraremoved
```

The meanings of the command line options are as follows.

`--mode` Operating mode (select from BOTH | DENOVO | REF)
`--uchimedenovo` UCHIME de novo version (select from 1 | 2 | 3)
`--referencedb` Reference sequence database
`--addtoref` Reference sequence file that will be appended to the database

After the options specify the input folder followed by the output folder.

When `--mode=both` is specified Claident performs both reference-free *de novo* chimera removal and reference-based chimera removal then keeps only the sequences judged as non-chimeric by both methods. Among the three UCHIME de novo versions UCHIME3 is optimized for denoised sequences so `--uchimedenovo` is set to 3. You must specify a reference database for the reference-based method. The following databases are installed automatically by the Claident installer.

cdu12s For mitochondrial 12S region
cdu16s For mitochondrial 16S region
cducox1 For mitochondrial COX1 (COI) region
cducytb For mitochondrial Cyt-b region
cdudloop For mitochondrial D-loop (control region)
cdumatk For chloroplast matK region
cdurbcl For chloroplast rbcL region
cdutrnhpsba For chloroplast trnH-psbA region

The reference databases for chimera-removal are located under "`install_path/share/claident/uchimedb`". You can list the contents of that folder to see which databases are available.

For bacterial 16S, SILVA SSURef or SSUParc is recommended, and for fungal ITS, the UNITE's "Full UNITE+INSD dataset for eukaryotes" is recommended. Because the MiFish primers amplify part of the mitochondrial 12S region we use `cdu12s` here. The databases whose names start with cdu were constructed by me by extracting the target region from full-length or nearly full-length mitochondrial or chloroplast genomes in public databases under the

assumption that such long sequences are unlikely to be chimeric.

When PCR is performed with internal-standard DNA, chimeras can form not only between internal-standard DNAs but also between internal-standard DNA and biological DNA, in addition to chimeras between biological DNAs. Therefore, we append the internal-standard DNA sequences in "`standard.fasta`" to the reference database by `--addtoref` to improve detection sensitivity.

If no suitable reference database exists and you performed PCR with a mixture of internal-standard DNAs, specify "`standard.fasta`" to `--referencedb` and set `--mode=both`. In that case `--addtoref` is unnecessary.

If no suitable reference database exists and you did not add internal-standard DNA during library preparation, run chimera removal with `--mode=denovo`. `--addtoref` is also unnecessary in that situation. However if your laboratory has previously prepared libraries containing internal-standard DNA, the amplified internal-standard DNA may contaminate new PCRs. In that case, you might need to treat the data as though internal-standard DNA had been added.

1.3.6 Clustering Internal-Standard Sequences with clclusterstdv

The following command clusters sequences corresponding to the internal-standard DNA using the UCLUST algorithm (Edgar 2010) implemented in VSEARCH (Rognes *et al.* 2016).

```
clclusterstdv \
--standardseq=standard.fasta \
--minident=0.9 \
--numthreads=NumberOfCPUcores \
06_chimeraremoved \
07_stdclustered
```

The meanings of the command line options are as follows.

`--standardseq` Internal-standard DNA sequence file

`--minident` Lower similarity threshold above which a sequence is regarded as internal-standard DNA

After the options, specify the input folder followed by the output folder.

If the maximum similarity between the internal-standard sequences and biological sequences is low (below 0.85), a threshold around 0.90 is sufficient. The MiFish internal-standard sequences listed in Appendix S1 of Ushio *et al.* (2022) satisfy this condition. If the similarity between internal-standard DNA and biological DNA is high (0.85 or above) and the synthesis error rate of the internal-standard DNA is expected to be low you may raise the threshold to around 0.97. Judge whether the synthesis error rate is low from the manufacturer's stated error rate and synthesis method. If unsure, vary the threshold from 0.90 to 0.97 in steps of 0.01, find the point at which the read count classified as internal-standard DNA changes abruptly and set the threshold to the smaller value at that change. If no abrupt change is observed the synthesis error rate is probably very high or biological DNA similar to the internal-standard are present which makes quantification impossible. In that case you need to resynthesize new internal-standard DNA and repeat the entire procedure. Because internal-standard DNA cannot be distinguished from biological DNA, the data cannot be used even for non-quantitative metabarcoding.

If internal-standard DNA was not added during library preparation, skip this step. However if library preparation is performed in a laboratory where internal-standard DNA was previously used, the workspace may be contaminated with amplified internal-standard DNA that could enter new PCRs. In that case, you might need to treat the data as though internal-standard DNA had been added.

1.3.7 Second Chimera Removal with clremovechimev

The following command applies reference-based chimera detection and removal using the UCHIME algorithm (Edgar *et al.* 2011) implemented in VSEARCH (Rognes *et al.* 2016).

```
clremovechimev \
--mode=ref \
--referencedb=cdu12s \
--addtoref=07_stdclustered/stdvariations.fasta \
--numthreads=NumberOfCPUcores \
07_stdclustered \
08_chimeraremoved
```

The meanings of the command line options are as follows.

`--mode` Select the operating mode (select from BOTH | DENOVO | REF)
`--referencedb` Reference sequence database
`--addtoref` Reference sequence file that will be appended to the database

After the options, specify the input folder followed by the output folder.

`--mode=ref` performs reference-based chimera removal. For details on the reference databases see the section on the first chimera removal.

When PCR is performed with internal-standard DNA, chimeras can form not only between internal-standard DNAs but also between internal-standard DNA and biological DNA, in addition to chimeras between biological DNAs. Therefore, we append the sequences judged to be internal-standard DNA "`07_stdclustered/stdvariations.fasta`" to the reference database by `--addtoref` to improve detection sensitivity. We use "`07_stdclustered/stdvariations.fasta`" instead of "`standard.fasta`" because the former includes sequences containing synthesis errors which allows us to detect chimeras between erroneous internal-standard DNAs and chimeras between erroneous internal-standard DNA and biological DNA.

If no suitable reference database exists and internal-standard DNA was added in library preparation, specify "`07_stdclustered/stdvariations.fasta`" to `--referencedb` and omit `--addtoref`.

If no suitable reference database exists and internal-standard DNA was not added skip this step. Again if your lab may be contaminated with internal-standard DNA you may still perform this analysis.

1.3.8 Removing Index-Hopping with clremovecontam

The following command removes index-hopping following the approach of Esling *et al.* (2015).

```
clremovecontam \
--test=thompson \
--ignoresamplelist=blanklist.txt \
--index1file=index1.fasta \
--index2file=index2.fasta \
--numthreads=NumberOfCPUcores \
08_chimeraremoved \
09_hoppingremoved
```

The meanings of the command line options are as follows.

`--test` Statistical test to use (select from THOMPSON | BINOMIAL)

`--ignoresamplelist` Text file listing sample IDs to be excluded from processing

`--index1file` Reverse index sequence file

This is the same file that was supplied to `clsplitseq`

`--index2file` Forward index sequence file

This is the same file that was supplied to `clsplitseq`

After the options, specify the input folder followed by the output folder.

For each sample, Claident compares the read count of each ASV in "unused index combinations sharing one index with sample" with the read count in that sample. If the sample count is not an outlier, the reads are judged to have arisen from index-hopping and their counts are set to 0.

Blanks are excluded with `--ignoresamplelist` because removing index-hopping from blanks would discard the information needed for the decontamination step below.

Skip this step if the requirements for index-hopping removal are not fulfilled.

1.3.9 Decontamination Using Negative Controls with clremovecontam

The following command estimates the DNA concentration of each ASV in samples and field blanks then sets the counts to 0 for samples whose concentration is not an outlier compared with concentrations of blanks.

```
clremovecontam \
--test=thompson \
--blanklist=blanklist.txt \
--ignoreotuseq=standard.fasta \
--stdconctable=stdconctable.tsv \
--solutionvoltable=solutionvoltable.tsv \
--watervoltable=watervoltable.tsv \
--numthreads=NumberOfCPUcores \
09_hoppingremoved \
10_decontaminated
```

The meanings of the command line options are as follows.

`--test` Statistical test to use (select from THOMPSON | BINOMIAL)

`--blanklist` Text file listing blanks' sample IDs

`--ignoreotuseq` FASTA file of OTU sequences that should be excluded from decontamination
`--stdconctable` Tab-delimited internal-standard DNA concentration table
`--solutionvoltable` Tab-delimited extracted DNA solution volume table
`--watervoltable` Tab-delimited filtered water volume table

After the options, specify the input folder followed by the output folder.

We exclude internal-standard sequences with `--ignoreotuseq` to avoid deleting them by mistake.

If only the internal-standard DNA concentration table is available, Claident estimates DNA concentration in the extracted solution instead of the environmental water and decontaminates based on those concentrations. If none of the three tables are available, Claident uses raw read counts for decontamination. When concentration estimates based on internal-standard DNA are used, the method is valid even if library preparation involved concentration normalization. If raw read counts are used, you must ensure that no concentration normalization was applied during library preparation and that the total number of PCR cycles was kept to a minimum so that no sample reached plateau.

In rare cases severe contamination during sampling or laboratory work can cause almost all sequences in samples to be judged contaminants and set to 0. Because such an outcome prevents any community-level analysis, you must exercise great care to avoid contamination from sampling through library preparation.

Sequence processing ends here but you may wish to cluster ASVs further or merge multiple sequencing runs. You can perform additional clustering and merging with the `clclassseqv` command.

From the denoising step onward each output folder contains the following three files. The prefix "〜" is common to all three.

〜.fasta FASTA file of ASV or OTU sequences at that step
〜.otu.gz File recording ASV or OTU membership at that step
〜.tsv Tab-delimited table of ASV or OTU read counts per sample at that step

By following these TSV files you can track the changes introduced by each processing step.

1.4 Molecular Identification

This section describes the molecular identification workflow based on the QCauto method and the 95%-3NN method (Tanabe & Toju 2013). The QCauto method yields very few misidentifications, but lower taxonomic ranks such as species or genus are prone to be labelled "unidentified". The 95%-3NN method can usually identify down to low taxonomic ranks such as species or genus, but it tends to produce more misidentifications when the reference sequence database is not well curated. When MiFish metabarcoding is conducted on Japanese freshwater or coastal samples, the reference sequence database maintained by the Chiba Prefectural Museum group is well curated, so the 95%-3NN method rarely causes problems. However, when other reference databases have insufficient coverage, we recommend using the QCauto results.

Before proceeding, create an output directory for molecular identification with the following command.

```
mkdir 11_taxonomy
```

1.4.1 Reference Sequence Databases for Molecular Identification

Claident ships with many reference sequence databases for molecular identification. The bundled databases are named as follows.

`Taxon_Locus_TaxonomicRankOfReferences`

`Taxon_Locus` can be one of the following.

overall　All organisms and all loci
animals_12S　Animals 12S
animals_16S　Animals 16S
animals_COX1　Animals COX1 (COI)
animals_CytB　Animals CytB
animals_D-loop　Animals D-loop
animals_mt　Animal mitochondrial DNA
eukaryota_LSU　Eukaryotes LSU (28S)
eukaryota_SSU　Eukaryotes SSU (18S)
fungi_all　All loci of Fungi
fungi_ITS　Fungi ITS
plants_cp　Plant chloroplast DNA
plants_matK　Plant matK
plants_rbcL　Plant rbcL
plants_trnH-psbA　Plant trnH-psbA
prokaryota_16S　Prokaryotes 16S
prokaryota_all　All loci of Prokaryotes

`TaxonomicRankOfReferences` can be one of the following.

class　Contains reference sequences identified to class or lower (available only for overall)
order　Contains reference sequences identified to order or lower (available only for overall)
family　Contains reference sequences identified to family or lower (available only for overall)
genus　Contains reference sequences identified to genus or lower
species_wsp　Contains reference sequences identified to species or lower even if species name contains “sp.”
species　Contains reference sequences identified to species or lower, but sequences whose species name ends with “sp.” are excluded
species_wosp　Contains reference sequences identified to species or lower, but sequences whose species name contains “sp.” are excluded
genus_man　Contains reference sequences identified to genus or lower and where the genus name is not empty

species_wsp_man　Contains reference sequences identified to species or lower. Sequences whose species name contains “sp.” are included, but sequences with an empty genus name are excluded

species_man　Contains reference sequences identified to species or lower, but sequences whose species name ends with “sp.” or whose genus name is empty are excluded

species_wosp_man　Contains reference sequences identified to species or lower, but sequences whose species name contains “sp.” or whose genus name is empty are excluded

Reference databases for molecular identification are located in “`install_path/share/claident/blastdb`”, so list contents of this folder provides which databases are installed.

Because there are so many databases, choosing the best one can be difficult, and the optimal choice depends on the target taxon and research goals. When MiFish metabarcoding is applied to Japanese freshwater or coastal samples and you also wish to identify non-animal or non-mitochondrial sequences, we recommend “`overall_species_wsp`”. However, the `overall`-series databases are very large, so machines with little RAM may run out of memory. In such cases you will not be able to identify non-animal or off-target sequences, but “`animals_12S_species_wsp`” or “`animals_mt_species_wsp`” are good alternatives. When genus-level identification is critically important for fungi or bacteria, a database that ends with “`_species_wsp_man`” may produce better results. If you are unsure which database to use, you can perform identification with several databases and later merge the results to take the best assignment for each OTU.

1.4.2 Building a Cache Database with clmakecachedb

For molecular identification using Claident, generating a cache database is highly recommended. The following command will generate a cache database for the decontaminated sequences.

```
clmakecachedb \
--blastdb=animals_mt_species_wsp \
--ignoreotuseq=standard.fasta \
--numthreads=NumberOfCPUcores \
10_decontaminated/decontaminated.fasta \
11_taxonomy/cachedb_species_wsp
```

The meanings of the command line options are as follows.

`--blastdb`　Reference sequence database to use for molecular identification

`--ignoreotuseq`　OTUs that match sequence names contained in the specified FASTA sequence file will be ignored

After the options, specify the input file followed by the output folder.

This step may consume large amounts of memory. While it is running, open another terminal and monitor free memory with `top`. If the machine runs out of RAM, press Ctrl+C to abort, switch to a smaller database, or increase the memory.

1.4.3 Molecular Identification with the QCauto Method

1.4.3.1 Retrieving Neighborhood Sequences with clidentseq

The following command retrieves the neighborhood sequences from the cache database based on the QCauto method.

```
clidentseq \
--method=QC \
--blastdb=11_taxonomy/cachedb_species_wsp \
--ignoreotuseq=standard.fasta \
--numthreads=NumberOfCPUcores \
10_decontaminated/decontaminated.fasta \
11_taxonomy/neighborhoods_qcauto_species_wsp.txt
```

The meanings of the command line options are as follows.

`--method` Molecular identification algorithm to use

`--blastdb` Cache database or reference database to use for molecular identification

`--ignoreotuseq` OTUs that match sequence names contained in the specified FASTA sequence file will be ignored

After the options, specify the input file followed by the output file.

1.4.3.2 Assigning Taxonomy with classigntax

The following command assigns a taxon to each OTU using the LCA algorithm (Huson *et al.* 2007) based on the identification information of the acquired neighborhood sequences.

```
classigntax \
--taxdb=animals_mt_species_wsp \
11_taxonomy/neighborhoods_qcauto_species_wsp.txt \
11_taxonomy/taxonomy_qcauto_species_wsp.tsv
```

The meanings of the command line options are as follows.

`--taxdb` Taxonomic database corresponding to the `--blastdb` used for `clmakecachedb`

After the options, specify the input file followed by the output file.

The output is a tab-delimited file containing one identification result per OTU.

In the LCA algorithm, the taxonomic rank is raised until all neighborhood sequences support the same taxon, and taxonomic ranks that do not satisfy the condition are "unidentified". In other words, it adopts the strict consensus taxonomy of the neighborhood sequence. In this method, if even one misidentified sequence is contaminated in the neighborhood sequences, it will no longer be identified. Here, sequences that support the identification result are called "supporters" and sequences that do not support the identification result are called "opposers". For `classigntax`, you can add `--maxpopposer=0.05 --minsoratio=19` to the command line option to allow the

presence of “opposers” up to 5% and adopt a 95% majority rule consensus taxonomy. If misidentified sequences are contaminated with neighborhood sequences, query sequence can still be identified by using this method if only a small percentage of the misidentified sequence is present. The `--minsoratio=19` option sets the minimum ratio of the number of “supporters” to the number of “opposers” to 19, and it is necessary because there can be sequences that are neither supporters nor opposers due to the lack of information on that taxonomic rank.

1.4.4 Molecular Identification with the 95%-3NN Method

1.4.4.1 Retrieving Neighborhood Sequences with clidentseq

The following command retrieves the neighborhood sequences from the cache database based on the 95%-3NN method.

```
clidentseq \
--method=3,95% \
--blastdb=11_taxonomy/cachedb_species_wsp \
--ignoreotuseq=standard.fasta \
--numthreads=NumberOfCPUcores \
10_decontaminated/decontaminated.fasta \
11_taxonomy/neighborhoods_95p3nn_species_wsp.txt
```

1.4.4.2 Assigning Taxonomy with classigntax

The following command assigns a taxon to each OTU using the LCA algorithm (Huson *et al.* 2007) based on the identification information of the acquired neighborhood sequences.

```
classigntax \
--taxdb=animals_mt_species_wsp \
--minnsupporter=3 \
11_taxonomy/neighborhoods_95p3nn_species_wsp.txt \
11_taxonomy/taxonomy_95p3nn_species_wsp.tsv
```

The meanings of the command line options are as follows.

`--minnsupporter` Minimum number of neighborhood sequences that must support the result

In this method Claident retrieves up to the top three reference sequences whose identity is 95% or higher and then assigns taxonomy with the LCA algorithm (Huson *et al.* 2007).

1.4.5 Reusing Identification Results with clmakeidentdb

The following commands can be used to create a database of molecular identification results by the QCauto method.

```
clmakeidentdb \
--append \
11_taxonomy/neighborhoods_qcauto_species_wsp.txt \
11_taxonomy/qcauto_species_wsp.identdb
```

The following commands can also be used to create a database of molecular identification results by the 95%-3NN

method.

```
clmakeidentdb \
--append \
11_taxonomy/neighborhoods_95p3nn_species_wsp.txt \
11_taxonomy/95p3nn_species_wsp.identdb
```

The meanings of the command line options are as follows.

`--append` Append to the output file if it already exists

After the options, specify the input file followed by the output file.

Each output file stores molecular identification results obtained by `clidentseq`. If you supply such a database to `clmakecachedb` and `clidentseq` with `--identdb`, OTUs already present in this database are skipped, which saves computation time. Because different methods and/or databases can naturally yield different molecular identification results, do not mix results from different methods or databases when appending the results to the existing file.

1.4.6 Merging Multiple Identification Results with clmergeassign

The above analysis should have yielded at least the molecular identification results by the QCauto method and the 95%-3NN method for each OTU. In some cases, more molecular identification results may have been obtained for the same OTU by performing molecular identifications in multiple databases. You can merge such multiple identification results by accepting non-conflicting taxonomy of the lowest rank for each OTU. The following command prefers the conservative QCauto result but replaces it with a 95%-3NN result when that result is not in conflict and reaches a lower taxonomic rank.

```
clmergeassign \
--preferlower \
--priority=descend \
11_taxonomy/taxonomy_qcauto_species_wsp.tsv \
11_taxonomy/taxonomy_95p3nn_species_wsp.tsv \
11_taxonomy/taxonomy_merged.tsv
```

The meanings of the command line options are as follows.

`--preferlower` Prefer results that reach lower taxonomic ranks

`--priority` Priority of the input files (select from ASCEND | DESCEND | EQUAL | formula)

To write an formula, assign numerical values beginning with 0 in the order in which they are specified as the input files, then construct formula such as "`0<1=2<3<4`"

This preference takes precedence over `--preferlower`

After the options, specify the multiple input files followed by the output file.

If `--priority=descend` is specified, the earlier input file has priority over the later.

1.4.7 Filling missing taxonomic ranks with `clfillassign`

The output of `classigntax` as it is leaves the taxonomic ranks without identification information blank. The following command can fill in all such blanks.

```
clfillassign \
--fullfill=enable \
11_taxonomy/taxonomy_merged.tsv \
11_taxonomy/taxonomy_merged_filled.tsv
```

The meanings of the command line options are as follows.

`--fullfill` Whether to fill in all taxonomic ranks supported by Claident, including taxonomic ranks that do not exist in the file (select from ENABLE | DISABLE)

After the options, specify the input file followed by the output file.

Missing ranks are filled as follows. If lower ranks information exist, their values propagate upward. If no lower rank information exists, the lowest rank information is copied and prefixed with "`unidentified`". For example, if order is "`Foo`" and infraorder is "`Bar`" but suborder, which is a rank between an order and an infraorder, is blank, suborder becomes "`Bar`". If parvorder and all lower ranks are blank, they become "`unidentified Bar`".

1.5 Generating OTU Composition Tables

The OTU composition table referred to here is a table of read counts for each OTU in each sample. It can be represented in the following format:

```
samplename  OTU1  OTU2  OTU3  OTU4
Sample1     3813   130  1949 34959
Sample2    18389    19   194  1948
Sample3       18     1   148   184
```

This table serves as the source data for statistical analysis. Here, I will explain the preprocessing required before entering into actual statistical analysis.

Before that, create an output directory for the OTU composition table in the working directory with the following command:

```
mkdir 12_community
```

Also, the OTU composition table that serves as the starting point for processing already exists as "`10_decontaminated/decontaminated.tsv`", so copy it to the directory created above with the following command:

```
cp \
10_decontaminated/decontaminated.tsv \
12_community/sample_otu_matrix_all.tsv
```

1.5.1 Processing OTU Composition Tables with clfiltersum

With the following command, you can create a table containing only internal standard OTUs (other OTUs will be excluded):

```
clfiltersum \
--otuseq=standard.fasta \
12_community/sample_otu_matrix_all.tsv \
12_community/sample_otu_matrix_standard.tsv
```

The meaning of the command line options is as follows:

`--otuseq` Extracts OTU data that matches sequence names included in the specified FASTA sequence file

Following the command line options, specify the input file and output file.

By executing the following command, you can create a table of OTUs for the taxonomic group specified by `--includetaxa` (fish in this case) based on molecular identification results:

```
clfiltersum \
--negativeotuseq=standard.fasta \
--taxfile=11_taxonomy/taxonomy_merged_filled.tsv \
--includetaxa=class,Hyperoartia,class,Myxini,class,Chondrichthyes \
--includetaxa=superclass,Actinopterygii,order,Coelacanthiformes \
--includetaxa=subclass,Dipnomorpha \
12_community/sample_otu_matrix_all.tsv \
12_community/sample_otu_matrix_fishes.tsv
```

The meaning of the command line options is as follows:

`--negativeotuseq` Excludes OTU data that matches sequence names included in the specified FASTA sequence file

`--taxfile` Tab-delimited text file of molecular identification results (in the output format of `classigntax`)

`--includetaxa` Extracts OTU data for the corresponding taxonomic names

It is also possible to limit the taxonomic rank in which taxonomic names are searched

Multiple specifications are possible

By replacing `--includetaxa` with `--excludetaxa` as shown below, you can create a table of non-fish OTUs:

```
clfiltersum \
--negativeotuseq=standard.fasta \
--taxfile=11_taxonomy/taxonomy_merged_filled.tsv \
--excludetaxa=class,Hyperoartia,class,Myxini,class,Chondrichthyes \
--excludetaxa=superclass,Actinopterygii,order,Coelacanthiformes \
--excludetaxa=subclass,Dipnomorpha \
12_community/sample_otu_matrix_all.tsv \
12_community/sample_otu_matrix_nonfishes.tsv
```

Let's try doing the same thing in a different way. The following command extracts only the OTU names from the fish OTU table and saves them to "`12_community/fishotus.txt`":

```
head -n 1 12_community/sample_otu_matrix_fishes.tsv \
| perl -ne '@row=split(/\t/);shift(@row);print(join("\n",@row)."\n");' \
> 12_community/fishotus.txt
```

Since `clfiltersum` has an option to extract OTUs whose names are not included in a given text file, you can create a table of non-fish OTUs as follows using the file created above:

```
clfiltersum \
--negativeotuseq=standard.fasta \
--negativeotulist=12_community/fishotus.txt \
12_community/sample_otu_matrix_all.tsv \
12_community/sample_otu_matrix_nonfishes2.tsv
```

The meaning of the command line options is as follows:

`--negativeotulist` Text file containing a list of OTU names to be excluded

1.5.2 Coverage-based Rarefaction of OTU Composition Tables with clrarefysum

While community ecological analysis can be performed with OTU composition tables, the varying coverage (completeness of sampling surveys) among samples may lead to misinterpretation where high-coverage samples with inherently fewer species appear to have more species than low-coverage samples that actually have more species. Coverage-based rarefaction is a process that avoids such problems by equalizing coverage between samples (Chao & Jost 2012). Note that rarefaction can refer to either "obtaining rarefied OTU composition tables" or "obtaining rarefaction curves", but in this textbook, please consider it to refer to the former.

As a method for performing coverage-based rarefaction, there is an approach that estimates coverage based on "the number of OTUs observed only once in that sample (singletons)" and "the number of OTUs observed only twice in that sample (doubletons)" (Chao & Jost 2012). However, in metabarcoding data, these numbers are not considered sufficiently reliable due to the presence of large amounts of sequencing errors (Chiu & Chao 2016). While one might think that denoised data would be problem-free, there is currently insufficient evidence for this. Chiu & Chao (2016) proposed a method to correct the number of singletons even in data with such sequencing errors, and this method can be applied by enabling `correct_singletons` when rarefying with the `phyloseq_coverage_raref()` function in the metagMisc R package.

However, the method of Chiu & Chao (2016) is premised on non-denoised data. When denoising is applied, ASVs with low read counts (including singletons) are either discarded during processing or considered to be derived from sequencing errors of neighboring ASVs with higher read counts, resulting in a dramatic reduction from non-denoised data. Therefore, applying the method of Chiu & Chao (2016) directly to denoised data is considered problematic.

Here, $(1 - \text{slope of rarefaction curve})$ can be regarded as coverage itself (Chao & Jost 2012). Based on this idea, Claident supports rarefaction that equalizes the slope at the endpoints of rarefaction curves between samples. As it uses the slope of rarefaction curves where the influence of singleton counts is reduced, it can be expected to be more robust against sequencing errors than methods that estimate coverage from singleton and doubleton counts. However, since computational complexity increases, optimization using parallelization has been implemented in

Claident.

The following command removes samples with fewer than 1000 reads, calculates coverage for each remaining sample, and performs rarefaction on all samples to align with the same coverage as the sample with the lowest coverage. However, if the coverage of the sample with the lowest coverage is less than 0.99, it aligns to 0.99, and samples with coverage less than 0.99 are removed. Because rarefaction involves randomly discarding reads, results may vary with repetition. Therefore, rarefaction is performed 10 times, and each result is saved.

```
clrarefysum \
--minpcov=0.99 \
--minntotalseqsample=1000 \
--nreplicate=10 \
--numthreads=NumberOfCPUcores \
12_community/sample_otu_matrix_all.tsv \
12_community/sample_otu_matrix_all_rarefied
```

The meaning of the command line options is as follows:

`--minpcov` Lower limit of coverage to align (samples below this are discarded)
`--minntotalseqsample` Lower limit of read count before rarefaction (samples below this are discarded)
`--nreplicate` Number of rarefaction replicates

Following the command line options, specify the input file and output file prefix.

The output files generated after execution are as follows:

`output_file_prefix-r[number].tsv` Tab-delimited text of rarefied OTU composition table
`output_file_prefix_inputpcov.tsv` Tab-delimited text of coverage estimates for each input sample
`output_file_prefix_inputnseq.tsv` Tab-delimited text of total read counts for each input sample
`output_file_prefix_outputpcov.tsv` Tab-delimited text of coverage estimates for each output sample
`output_file_prefix_outputnseq.tsv` Tab-delimited text of total read counts for each output sample

After rarefaction is complete, extract only internal standard OTUs from all 10 replicates with the following command:

```
for n in `seq -w 1 10`
do clfiltersum \
--otuseq=standard.fasta \
12_community/sample_otu_matrix_all_rarefied-r$n.tsv \
12_community/sample_otu_matrix_standard_rarefied-r$n.tsv
done
```

The following command extracts only fish OTUs:

```
for n in `seq -w 1 10`
do clfiltersum \
--negativeotuseq=standard.fasta \
--taxfile=11_taxonomy/taxonomy_merged_filled.tsv \
--includetaxa=class,Hyperoartia,class,Myxini,class,Chondrichthyes \
--includetaxa=superclass,Actinopterygii,order,Coelacanthiformes \
--includetaxa=subclass,Dipnomorpha \
12_community/sample_otu_matrix_all_rarefied-r$n.tsv \
```

```
12_community/sample_otu_matrix_fishes_rarefied-r$n.tsv
done
```

The following command extracts non-fish OTUs:

```
for n in `seq -w 1 10`
do clfiltersum \
--negativeotuseq=standard.fasta \
--taxfile=11_taxonomy/taxonomy_merged_filled.tsv \
--excludetaxa=class,Hyperoartia,class,Myxini,class,Chondrichthyes \
--excludetaxa=superclass,Actinopterygii,order,Coelacanthiformes \
--excludetaxa=subclass,Dipnomorpha \
12_community/sample_otu_matrix_all_rarefied-r$n.tsv \
12_community/sample_otu_matrix_nonfishes_rarefied-r$n.tsv
done
```

In the above example, coverage-based rarefaction is performed using OTU composition tables of all taxonomic groups, and fish OTUs and non-fish OTUs are separated after rarefaction. However, if there is no interest in non-fish from the beginning, or if non-fish are considered likely to be contamination based on prior knowledge, it might be better to perform coverage-based rarefaction using only fish OTU composition tables.

It is important to note that what can be achieved with these coverage-based rarefaction methods, whether metagMisc or Claident, is merely "equalization of sequencing coverage for communities". "Equalization of coverage for communities of sampled water", "equalization of coverage for communities of DNA collected on filtration filters", or "equalization of coverage for communities of DNA solution input to PCR" are not performed. In metabarcoding, there are numerous sampling steps, i.e., "extracting a portion", so sequencing coverage is not the only aspect where uniformity becomes problematic. However, subsequent analyses are performed under the assumption that all of these are saturated (coverage is approximately 1.0). If any abnormal results are obtained, it might be necessary to consider the possibility that this assumption is not satisfied.

1.5.3 DNA Concentration Estimation Using clestimateconc and Internal Standard DNA Read Counts

The following command estimates the DNA concentration of OTUs in environmental water samples based on internal standard DNA read counts with known concentrations:

```
clestimateconc \
--stdtable=12_community/sample_otu_matrix_standard.tsv \
--stdconctable=stdconctable.tsv \
--solutionvoltable=solutionvoltable.tsv \
--watervoltable=watervoltable.tsv \
--numthreads=NumberOfCPUcores \
12_community/sample_otu_matrix_fishes.tsv \
12_community/sample_otu_matrix_fishes_concentration.tsv
```

The meaning of the command line options is as follows:

`--stdtable` Tab-delimited text of internal standard OTU read count table (not required if internal standard OTU read counts are included in the input file)

`--stdconctable` Tab-delimited text of internal standard DNA concentration table

`--solutionvoltable` Tab-delimited text of extracted DNA solution volume table

`--watervoltable` Tab-delimited text of filtered water volume table

Following the command line options, specify the input file and output file.

To estimate DNA concentrations for data with 10 rarefaction replicates, execute the following command:

```
for n in `seq -w 1 10`
do clestimateconc \
--stdtable=12_community/sample_otu_matrix_standard_rarefied-r$n.tsv \
--stdconctable=stdconctable.tsv \
--solutionvoltable=solutionvoltable.tsv \
--watervoltable=watervoltable.tsv \
--numthreads=NumberOfCPUcores \
12_community/sample_otu_matrix_fishes_rarefied-r$n.tsv \
12_community/sample_otu_matrix_fishes_rarefied-r$n_concentration.tsv
done
```

In data where coverage is not equalized, the reliability of estimated DNA concentrations varies between samples. Because it is not possible to perform analyses that consider the variation in reliability of values from data containing only DNA concentration information, it would often be better to use DNA concentration data estimated after rarefaction when conducting analyses utilizing DNA concentrations. However, depending on the analysis method, DNA concentration data estimated from original data before rarefaction might be more suitable in some cases.

1.5.4 Creating Species Composition Tables from OTU Composition Tables

While OTU composition tables are suitable for community ecological analysis, species composition tables or genus composition tables may be more understandable for plotting and other purposes. In such cases, species composition tables or genus composition tables can be created from OTU composition tables and molecular identification results. The following command creates a species composition table from a fish OTU composition table:

```
clsumtaxa \
--taxfile=11_taxonomy/taxonomy_merged_filled.tsv \
--targetrank=species \
--taxnamereplace=enable \
--fuseotu=enable \
--numbering=enable \
--sortkey=abundance \
12_community/sample_otu_matrix_fishes.tsv \
12_community/sample_species_matrix_fishes.tsv
```

The meaning of the command line options is as follows:

`--taxfile` Tab-delimited text file of molecular identification results (in the output format of `classigntax`)

`--targetrank` Use information from the specified taxonomic rank

`--taxnamereplace` Replace spaces and colons used in output OTU names with underscores (select from ENABLE | DISABLE)

`--fuseotu` Whether to merge OTUs with the same taxonomic group name (select from ENABLE | DISABLE)
If DISABLE is selected, output OTU names will be "`input_OTU_name:taxonomic_group_name`" and

composition content is not changed

If DISABLE is selected and `--taxnamereplace` is also enabled, output OTU names will be "`input_OTU_name_taxonomic_group_name`"

`--numbering` Whether to add numbers as prefixes to output OTU names in sort order (select from ENABLE | DISABLE)

If ENABLE is selected and there are 100 output OTUs, numbers from `001` to `100` with aligned width are added separated by colon "`:`"

If `--taxnamereplace` is also enabled, separation is by underscore "`_`" instead of colon

If both `--fuseotu` and `--taxnamereplace` are also enabled, the format becomes "`number_taxonomic_group_name`"

If `--fuseotu` is also enabled and `--taxnamereplace` is disabled, the format becomes "`number:taxonomic_group_name`"

If `--fuseotu` is disabled and `--taxnamereplace` is enabled, the format becomes "`number_input_OTU_name_taxonomic_g`

If both `--fuseotu` and `--taxnamereplace` are disabled, the format becomes "`number:input_OTU_name:taxonomic_grou`

`--sortkey` Key to determine sort order (select from ABUNDANCE | RANKNAME)

RANKNAME should be "`familyname`", "`classname`", "`"species group name"`" (quote if spaces are included), etc.

Following the command line options, specify the input file and output file.

The following command creates a species composition table from an OTU composition table with DNA concentrations as values:

```
clsumtaxa \
--taxfile=11_taxonomy/taxonomy_merged_filled.tsv \
--targetrank=species \
--taxnamereplace=enable \
--taxranknamereplace=enable \
--fuseotu=enable \
--numbering=enable \
--sortkey=abundance \
12_community/sample_otu_matrix_fishes_concentration.tsv \
12_community/sample_species_matrix_fishes_concentration.tsv
```

Note that when `--fuseotu` is enabled, OTUs are merged based only on taxonomic group names, so even with `--targetrank=species`, species named "`unidentified higher_taxonomic_group_name`" will exist, and multiple species may be merged into these. This is because OTUs that could not be identified at low taxonomic ranks were assigned as "`unidentified higher_taxonomic_group_name`" by `clfillassign`. Therefore, this results in species composition tables that include OTUs where multiple species are incorrectly merged. While such species composition tables can be used for plotting, for statistical analysis, use OTU composition tables with ASVs or OTUs clustered based on sequence similarity as units.

1.6 Towards Community Ecological Analysis Using OTU Composition Tables

Through the content covered so far, OTU composition tables necessary for community ecological analysis can be obtained, but there should be at least four types: non-rarefied read count data, rarefied read count data, non-rarefied

DNA concentration data, and rarefied DNA concentration data. These need to be used appropriately depending on the purpose and analysis methods. Here, we briefly introduce packages that are useful when conducting community ecological analysis in R (R Core Team 2023) using OTU composition tables.

First, non-rarefied read count data is used for estimating and plotting rarefaction curves and Hill numbers (effective number of species) (Chao *et al.* 2014). The following R packages will be helpful:

- vegan https://github.com/vegandevs/vegan
- iNEXT https://github.com/JohnsonHsieh/iNEXT (Hsieh *et al.* 2016)

Rarefied read count data can be used for most analyses that do not require quantitative comparison among samples (cluster analysis, NMDS, PerMANOVA, community phylogenetic analysis). Investigating the following R packages is recommended:

- vegan https://github.com/vegandevs/vegan
- picante https://cran.r-project.org/web/packages/picante/ (Kembel *et al.* 2010)
- MicEco https://github.com/Russel88/MicEco
- iNEXT.beta3D https://github.com/KaiHsiangHu/iNEXT.beta3D (Chao *et al.* 2023)
- bipartite https://github.com/biometry/bipartite
- pvclust https://github.com/shimo-lab/pvclust (Suzuki & Shimodaira 2006)
- mpmcorrelogram https://cran.r-project.org/web/packages/mpmcorrelogram/
- boral https://cran.r-project.org/web/packages/boral/ (Hui 2016)
- gllvm https://github.com/JenniNiku/gllvm (Niku *et al.* 2019)

DNA concentration data can be used for analytical methods that require quantitative comparison between samples. Instead, methods that require integer values cannot be applied. The following R packages enable time series causal inference:

- rEDM https://ha0ye.github.io/rEDM/ (Ye & Sugihara 2016)
- rUIC https://github.com/yutakaos/rUIC (Osada *et al.* 2023)

When targeting spatial analysis, Joint Species Distribution Modeling enables simultaneous estimation of habitat suitability for multiple species and identification of important areas with high diversity. The following R packages support fitting such complex models:

- jSDM https://ecology.ghislainv.fr/jSDM/ (Warton *et al.* 2015)
- HMSC https://github.com/hmsc-r/HMSC (Tikhonov *et al.* 2020)

Methods for estimating networks of inter-OTU relationships from OTU composition have also been actively developed in recent years. The following are R packages that support estimation and visualization of inter-OTU relationship networks:

- SpiecEasi https://github.com/zdk123/SpiecEasi (Kurtz *et al.* 2015)
- NetCoMi https://github.com/stefpeschel/NetCoMi (Peschel *et al.* 2021)
- ggClusterNet https://github.com/taowenmicro/ggClusterNet (Wen *et al.* 2022)

Many of the R packages introduced here and the methods implemented in them are relatively new, and I cannot claim to have fully grasped them all. In particular, please carefully examine the properties of data required as prerequisites for each method (presence/absence, integer or decimal or real values, whether intra-sample or inter-sample quantitative properties exist, etc.) in papers and manuals before use.

Finally, 土居・岡村 (2011), 門脇 (2016), and Kadowaki (2023) provide introductory explanations of community ecological analysis in R, so I recommend you to read them.

References

Callahan, Benjamin J., McMurdie, Paul J., Rosen, Michael J., Han, Andrew W., Johnson, Amy Jo A. & Holmes, Susan P. (2016) DADA2: High-resolution sample inference from Illumina amplicon data. *Nature Methods* 13, 581–583. https://doi.org/10.1038/nmeth.3869

Chao, Anne, Gotelli, Nicholas J., Hsieh, T. C., Sander, Elizabeth L., Ma, K. H., Colwell, Robert K. & Ellison, Aaron M. (2014) Rarefaction and extrapolation with Hill numbers: a framework for sampling and estimation in species diversity studies. *Ecological Monographs* 84, 45–67. https://doi.org/10.1890/13-0133.1

Chao, Anne & Jost, Lou (2012) Coverage-based rarefaction and extrapolation: standardizing samples by completeness rather than size. *Ecology* 93, 2533–2547. https://doi.org/10.1890/11-1952.1

Chao, Anne, Thorn, Simon, Chiu, Chun-Huo, Moyes, Faye, Hu, Kai-Hsiang, Chazdon, Robin L., Wu, Jessie, Magnago, Luiz Fernando S., Dornelas, Maria, Zelený, David, Colwell, Robert K. & Magurran, Anne E. (2023) Rarefaction and extrapolation with beta diversity under a framework of Hill numbers: The iNEXT.beta3D standardization. *Ecological Monographs* 93, e1588. https://doi.org/10.1002/ecm.1588

Chiu, Chun-Huo & Chao, Anne (2016) Estimating and comparing microbial diversity in the presence of sequencing errors. *PeerJ* 4, e1634. https://doi.org/10.7717/peerj.1634

Edgar, Robert C. (2010) Search and clustering orders of magnitude faster than BLAST. *Bioinformatics* 26, 2460–2461. https://doi.org/10.1093/bioinformatics/btq461

Edgar, Robert C. (2016) UCHIME2: improved chimera prediction for amplicon sequencing., 074252.

Edgar, Robert C. & Flyvbjerg, Henrik (2015) Error filtering, pair assembly and error correction for next-generation sequencing reads. *Bioinformatics* 31, 3476–3482. https://doi.org/10.1093/bioinformatics/btv401

Edgar, Robert C., Haas, Brian J., Clemente, Jose C., Quince, Christopher & Knight, Rob (2011) UCHIME improves sensitivity and speed of chimera detection. *Bioinformatics* 27, 2194–2200. https://doi.org/10.1093/bioinformatics/btr381

Esling, Philippe, Lejzerowicz, Franck & Pawlowski, Jan (2015) Accurate multiplexing and filtering for high-throughput amplicon-sequencing. *Nucleic Acids Research* 43, 2513–2524. https://doi.org/10.1093/nar/gkv107

Hsieh, T. C., Ma, K. H. & Chao, Anne (2016) iNEXT: an R package for rarefaction and extrapolation of species

diversity (Hill numbers). *Methods in Ecology and Evolution* 7, 1451–1456. https://doi.org/10.1111/2041-210X.12613

Hui, Francis K. C. (2016) boral – Bayesian Ordination and Regression Analysis of Multivariate Abundance Data in r. *Methods in Ecology and Evolution* 7, 744–750. https://doi.org/10.1111/2041-210X.12514

Huson, Daniel H., Auch, Alexander F., Qi, Ji & Schuster, Stephan C. (2007) MEGAN analysis of metagenomic data. *Genome Research* 17, 377–386. https://doi.org/10.1101/gr.5969107

Kadowaki, Kohmei (2023) A primer of community ecology using the R language. *Population Ecology* 65, 240–256. https://doi.org/10.1002/1438-390X.12158

Kembel, Steven W., Cowan, Peter D., Helmus, Matthew R., Cornwell, William K., Morlon, Helene, Ackerly, David D., Blomberg, Simon P. & Webb, Campbell O. (2010) Picante: R tools for integrating phylogenies and ecology. *Bioinformatics* 26, 1463–1464. https://doi.org/10.1093/bioinformatics/btq166

Komai, Tomoyuki, Gotoh, Ryo O., Sado, Tetsuya & Miya, Masaki (2019) Development of a new set of PCR primers for eDNA metabarcoding decapod crustaceans. *Metabarcoding and Metagenomics* 3, e33835. https://doi.org/10.3897/mbmg.3.33835

Kurtz, Zachary D., Müller, Christian L., Miraldi, Emily R., Littman, Dan R., Blaser, Martin J. & Bonneau, Richard A. (2015) Sparse and Compositionally Robust Inference of Microbial Ecological Networks. *PLOS Computational Biology* 11, e1004226. https://doi.org/10.1371/journal.pcbi.1004226

Miya, Masaki, Gotoh, Ryo O. & Sado, Tetsuya (2020) MiFish metabarcoding: a high-throughput approach for simultaneous detection of multiple fish species from environmental DNA and other samples. *Fisheries Science* 86, 939–970. https://doi.org/10.1007/s12562-020-01461-x

Miya, M., Sato, Y., Fukunaga, T., Sado, T., Poulsen, J. Y., Sato, K., Minamoto, T., Yamamoto, S., Yamanaka, H., Araki, H., Kondoh, M. & Iwasaki, W. (2015) MiFish, a set of universal PCR primers for metabarcoding environmental DNA from fishes: detection of more than 230 subtropical marine species. *Royal Society Open Science* 2, 150088. https://doi.org/10.1098/rsos.150088

Niku, Jenni, Hui, Francis K. C., Taskinen, Sara & Warton, David I. (2019) gllvm: Fast analysis of multivariate abundance data with generalized linear latent variable models in r. *Methods in Ecology and Evolution* 10, 2173–2182. https://doi.org/10.1111/2041-210X.13303

Osada, Yutaka, Ushio, Masayuki & Michio, Kondoh (2023) A unified framework for nonparametric causality detection., 2023.04.20.537743.

Peschel, Stefanie, Müller, Christian L, von Mutius, Erika, Boulesteix, Anne-Laure & Depner, Martin (2021) NetCoMi: network construction and comparison for microbiome data in R. *Briefings in Bioinformatics* 22, bbaa290. https://doi.org/10.1093/bib/bbaa290

R Core Team (2023) R: A Language and Environment for Statistical Computing. https://www.R-project.org

Rognes, Torbjørn, Flouri, Tomáš, Nichols, Ben, Quince, Christopher & Mahé, Frédéric (2016) VSEARCH: a versatile open source tool for metagenomics. *PeerJ* 4, e2584. https://doi.org/10.7717/peerj.2584

Sakata, Masayuki K., Kawata, Mone U., Kurabayashi, Atsushi, Kurita, Takaki, Nakamura, Masatoshi, Shirako, Tomoyasu, Kakehashi, Ryosuke, Nishikawa, Kanto, Hossman, Mohamad Yazid, Nishijima, Takashi, Kabamoto, Junichi, Miya, Masaki & Minamoto, Toshifumi (2022) Development and evaluation of PCR primers for environmental DNA (eDNA) metabarcoding of Amphibia. *Metabarcoding and Metagenomics* 6, e76534. https://doi.org/10.3897/mbmg.6.76534

Sato, Yukuto, Miya, Masaki, Fukunaga, Tsukasa, Sado, Tetsuya & Iwasaki, Wataru (2018) MitoFish and MiFish Pipeline: A Mitochondrial Genome Database of Fish with an Analysis Pipeline for Environmental DNA Metabarcoding. *Molecular Biology and Evolution* 35, 1553–1555. https://doi.org/10.1093/molbev/msy074

Suzuki, Ryota & Shimodaira, Hidetoshi (2006) Pvclust: an R package for assessing the uncertainty in hierarchical clustering. *Bioinformatics* 22, 1540–1542. https://doi.org/10.1093/bioinformatics/btl117

Takenaka, Masaki, Yano, Koki, Suzuki, Tomoya & Tojo, Koji (2023) Development of novel PCR primer sets for DNA barcoding of aquatic insects, and the discovery of some cryptic species. *Limnology* 24, 121–136. https://doi.org/10.1007/s10201-022-00710-5

Tanabe, Akifumi S. & Toju, Hirokazu (2013) Two New Computational Methods for Universal DNA Barcoding: A Benchmark Using Barcode Sequences of Bacteria, Archaea, Animals, Fungi, and Land Plants. *PLOS ONE* 8, e76910. https://doi.org/10.1371/journal.pone.0076910

Tikhonov, Gleb, Opedal, Øystein H., Abrego, Nerea, Lehikoinen, Aleksi, de Jonge, Melinda M. J., Oksanen, Jari & Ovaskainen, Otso (2020) Joint species distribution modelling with the r-package Hmsc. *Methods in Ecology and Evolution* 11, 442–447. https://doi.org/10.1111/2041-210X.13345

Ushio, Masayuki, Fukuda, Hisato, Inoue, Toshiki, Makoto, Kobayashi, Kishida, Osamu, Sato, Keiichi, Murata, Koichi, Nikaido, Masato, Sado, Tetsuya, Sato, Yukuto, Takeshita, Masamichi, Iwasaki, Wataru, Yamanaka, Hiroki, Kondoh, Michio & Miya, Masaki (2017) Environmental DNA enables detection of terrestrial mammals from forest pond water. *Molecular Ecology Resources* 17, e63–e75. https://doi.org/10.1111/1755-0998.12690

Ushio, Masayuki, Furukawa, Saori, Murakami, Hiroaki, Masuda, Reiji & Nagano, Atsushi J. (2022) An efficient early-pooling protocol for environmental DNA metabarcoding. *Environmental DNA* 4, 1212–1228. https://doi.org/10.1002/edn3.337

Ushio, Masayuki, Murakami, Hiroaki, Masuda, Reiji, Sado, Tetsuya, Miya, Masaki, Sakurai, Sho, Yamanaka, Hiroki, Minamoto, Toshifumi & Kondoh, Michio (2018a) Quantitative monitoring of multispecies fish environmental DNA using high-throughput sequencing. *Metabarcoding and Metagenomics* 2, e23297. https://doi.org/10.3897/mbmg.2.23297

Ushio, Masayuki, Murata, Koichi, Sado, Tetsuya, Nishiumi, Isao, Takeshita, Masamichi, Iwasaki, Wataru & Miya, Masaki (2018b) Demonstration of the potential of environmental DNA as a tool for the detection of avian species. *Scientific Reports* 8, 4493. https://doi.org/10.1038/s41598-018-22817-5

Warton, David I., Blanchet, F. Guillaume, O'Hara, Robert B., Ovaskainen, Otso, Taskinen, Sara, Walker, Steven C. & Hui, Francis K. C. (2015) So Many Variables: Joint Modeling in Community Ecology. *Trends in Ecology & Evolution* 30, 766–779. https://doi.org/10.1016/j.tree.2015.09.007

Wen, Tao, Xie, Penghao, Yang, Shengdie, Niu, Guoqing, Liu, Xiaoyu, Ding, Zhexu, Xue, Chao, Liu, Yong-Xin, Shen, Qirong & Yuan, Jun (2022) ggClusterNet: An R package for microbiome network analysis and modularity-based multiple network layouts. *iMeta* 1, e32. https://doi.org/10.1002/imt2.32

Ye, Hao & Sugihara, George (2016) Information leverage in interconnected ecosystems: Overcoming the curse of dimensionality. *Science* 353, 922–925. https://doi.org/10.1126/science.aag0863

Zhu, Tao, Sato, Yukuto, Sado, Tetsuya, Miya, Masaki & Iwasaki, Wataru (2023) MitoFish, MitoAnnotator, and MiFish Pipeline: Updates in 10 Years. *Molecular Biology and Evolution* 40, msad035. https://doi.org/10.1093/molbev/msad035

土居秀幸・岡村寛 (2011) 生物群集解析のための類似度とその応用: R を使った類似度の算出、グラフ化、検定. 日本生態学会誌 61, 3–20. https://doi.org/10.18960/seitai.61.1_3

門脇浩明 (2016) メタゲノムデータを用いた群集統計解析法：レアファクションから仮説検定まで. 日本生態学会第 63 回大会講演資料. https://www.fifthdimension.jp/wiki.cgi?page=%BC%AB%CD%B3%BD%B8%B2%F12016%A1%A7%A5%E1%A5%BF%A5%D0%A1%BC%A5%B3%A1%BC%A5%C7%A5%A3%A5%F3%A5%B0%A1%A6%B4%C4%B6%ADDNA%A5%D0%A1%BC%A5%B3%A1%BC%A5%C7%A5%A3%A5%F3%A5%B0%B2%F2%C0%CF%A4%CE%B5%BB%CB%A1