# Improving the *nifH* gene reference database

Master 1 Internship

Rémi Legrand

June 9, 2023

Nitrogen is a main block of life however it is a limiting resource on most ocean surfaces and hence nitrogen-fixating bacteria (diazotrophs) have a key role because they are the only lifeform that can metabolize di-nitrogen into ammonium with the Nitrogenase protein. By studying the populations of diazotrophs we can gain a better understanding of past, present, and future climate. The characterization of these populations is made using a reference database of the *nifH* gene, coding for a sub-unit of the Nitrogenase. However, the current database is incomplete and the annotation process results in a lot of unannotated sequences. We present preliminary work to automatically create a new reference database by collecting the annotated *nifH* sequences present in databases such as NCBI, UK-PROT, and Swiss-Prot.

## Contents

**Declaration of Originality** I, Rémi Legrand, hereby confirm that I am the sole author of the written work here enclosed and that I have compiled it in my own words. Parts excepted are corrections of form and content by my supervisor. All information derived from other sources has been duly acknowledged in the references, and due credit is given to the parties involved. No part of this work was previously presented for any other diploma at any other institution.

# 1. Introduction

## 1.1. Why study dinitrogen-fixing bacteria?

Nitrogen is a main block of life since it is present in DNA and proteins. However, Nitrogen is limited in most of the ocean surface [Singh et al., 2012].

Even though the N2 gas constitutes around 78% of our atmosphere and is saturated in seawater, the di-nitrogen fixation process is to a large extent mediated by diazotrophs in the oceans [Knapp, 2012]. Using the nitrogenase enzyme, diazotrophs break the strong triple bond between the two N atoms. This enzyme is encoded by *nif* genes [Stacey et al., 1992].

Hence, reactive nitrogen sources such as dinitrogen (N2) fixation act as natural fertilizing processes [Zehr and Capone, 2020]. Additionally, diazotrophs have a key role in the Nitrogen cycle and is thus “linked to the fixation of atmospheric carbon dioxide and export of carbon from the ocean’s surface” [Zehr and Ward, 2002].

Biological nitrogen fixation is catalyzed by the nitrogenase protein [Stacey et al., 1992]. It is an enzyme that reduces N2 to ammonium (NH3) and is composed of two subunits (see Figure 1):

- the Heterotetrameric core (MoFe protein) encoded by the *nifD* and the *nifK* genes;
- the Dinitrogenase reductase (Fe protein) encoded by the *nifH* gene.

Both Nitrogenase’s subunit proteins are highly conserved but the Fe protein encoded by *nifH* is the most highly conserved across the microbial community [Zehr et al., 2003].

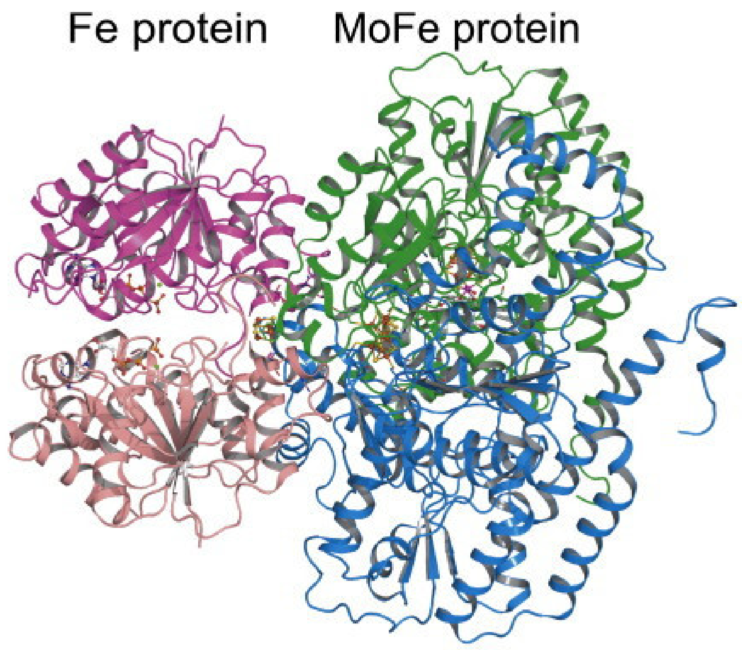

**Figure 1:** 3D structure of the Nitrogenase (from Seefeldt et al. [2013])

## 1.2. Why *nifH*?

Usually, to characterize a population, we use the 16S subunit of the polymerase. However, in the case of nitrogen-fixating microbial species, there are some genetic divergences of *nifH* between the species. Moreover, the “16S rRNA genes do not correlate well at sequence dissimilarity values used commonly to define microbial species” whereas *nifH* does [Gaby and Buckley, 2014]. Hence the *nifH* gene is commonly used to characterize a population of Nitrogen fixating bacteria.

## 1.3. Goals of the internship

To characterize the studied population, a reference database containing the taxonomy and the sequences of the *nifH* gene for different species is needed. The first *nifH* database was created in 2014 and continuously updated until 2017 by the Zehr Lab [Heller et al., 2014]. It was then normalized and kept updated manually by Moynihan [2020].

However, the current database leaves a considerable amount of unannotated sequences, and the annotation rate increases with the taxonomic level. The goal of my internship is to improve the current *nifH* database by using some other databases such as NCBI [Sayers et al., 2022] and write a code to automatize the update.

### 1.4. Organization of the report

During my internship, I first reproduced the processing and annotation of the sequences proposed by Benavides et al. [2022] with DADA2 in order to evaluate the annotation rate of the dataset of the study. The methodology for building, collecting and processing the samples is described in Section 2.1, 2.2, and 2.3 while the outcome of my replication attempt is presented in Section 3.1.

Then, I tried to improve the reference database of the *nifH* gene by rebuilding from the NCBI databases. These databases and how to interact with them are described in Section 2.4 and 2.5 while a synthetic presentation of the challenges I faced and how I proceeded is given in Section 3.2.

Section 4 concludes this report with some perspectives opened by my work.

## 2. Methods

### 2.1. Collecting environmental data: Metabarcoding

Metabarcoding is a popular approach to characterize the taxonomic diversity of a population using environmental DNA. In our case, we will use the *nifH* gene even though the process is generic.

The metabarcoding approach can be divided into the following steps:

1. Sample the environment we are interested in, ocean or seawater in our case;
2. Extract the DNA and amplify the region we are interested in using Polymerization Chain Reaction (PCR) with the appropriate primers [Angel et al., 2018];
3. Sequence the data using high-throughput sequencing (such as Illumina sequencing);
4. Process the samples through an adequate pipeline that matches them with a reference database to obtain the annotation of the sequences.

Although the first step typically involves costly field expeditions, the second and third ones are generally conducted by private companies. The last step is usually conducted by the researchers themselves.

## 2.2. *nifH* amplicon datasets used

In the lab I had access to datasets obtained from two different expeditions:

1. The **DUPE** dataset [Benavides et al., 2022] has been created by sequencing samples coming from two locations in the South Pacific. It is constituted of 12 samples collected with sediment traps at 170, 270 and 1000m.
2. The **TONGA** dataset [Filella et al., 2022] is based on samples from the Tonga trench volcanic arc region and the South Pacific Gyr. It is constituted of 20 samples.

During my internship, I focused on the **DUPE** dataset as it was complete.

## 2.3. Overview of the DADA2 Workflow

DADA2 [Callahan et al., 2016] is a pipeline used to clean and annotate the genomic data coming from a population. It uses R and it sufficiently light to run on a laptop.

Figure 2 shows the generic workflow of DADA2. I adpated the implementation given by Cabral [2017] to use it on the **DUPE** dataset. The code can be found in the appendix (see Section A.1).

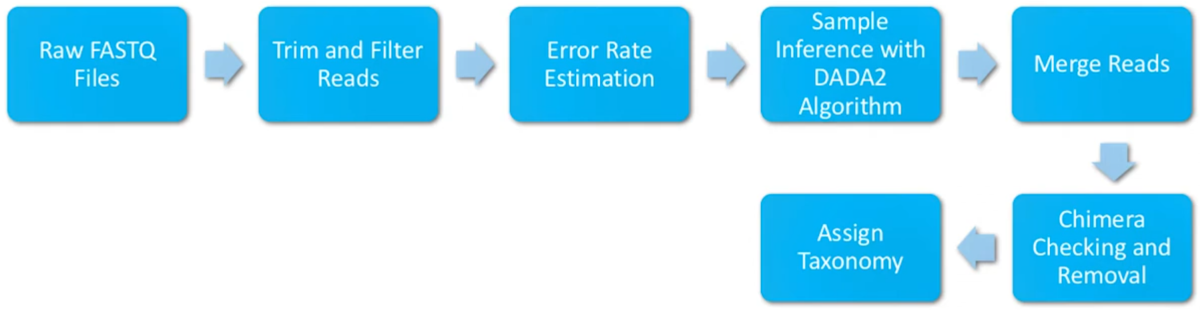

**Figure 2:** DADA2 Workflow (from Cabral [2017])

The next subsections describe every stage of the pipeline.

### 2.3.1. Raw FASTQ Files

The FASTQ format is a text file format used to store at the same time the biological sequences and the associated quality score. It is the standard output given by high-throughput sequencers. The first stage is thus to import the Raw FASTQ files. Paired-end sequencing is used in most cases, hence the analyst has to associate the reads pairwise.

### 2.3.2. Trim and Filter Reads

This step aims to remove the sequencing errors and artifacts and to keep only high-quality sequences. Using the plot of the quality score the analyst determines the trimming cut-offs (usually the lowest accepted quality score is 30), making sure that for each pair the forward and reverse sequences overlap by at least 20bp (base pairs). This will enable DADA2 to successfully merge the two sequences in a further step.

### 2.3.3. Error Rate Estimation

DADA2 builds an error model by learning the sequencing errors from the quality-filtered reads. The error model is then used to correct errors in the reads, improving the sequence accuracy.

In general, for the error plot, the frequency of the errors rate decreases as the quality score increases. Then the analyst should check that the error frequency observed in the data fits the expected error rate predicted by the Q-score.

To this end, I plotted the error frequency rates of the different base transitions of the **DUPE** dataset (see Figure 3). The black line represents what we observe in the data, and the red line is what we expect from the Q-score. The fitting of these two curves varies from one base transition to another, however the precise analysis of these graphics is beyond the scope of this intership.

### 2.3.4. Dereplicate Reads

The analyst then needs to condense the data by collapsing together all reads encoding the same sequence to reduce redundancy. This will simplify and speed up the computation.

### 2.3.5. Sample Inference with DADA2 Algorithm

The next step consists in filtering out the artifacts generate by the PCR. To do that, the analyst tests the null hypothesis that the sequence is too abundant in the sample to be solely explained

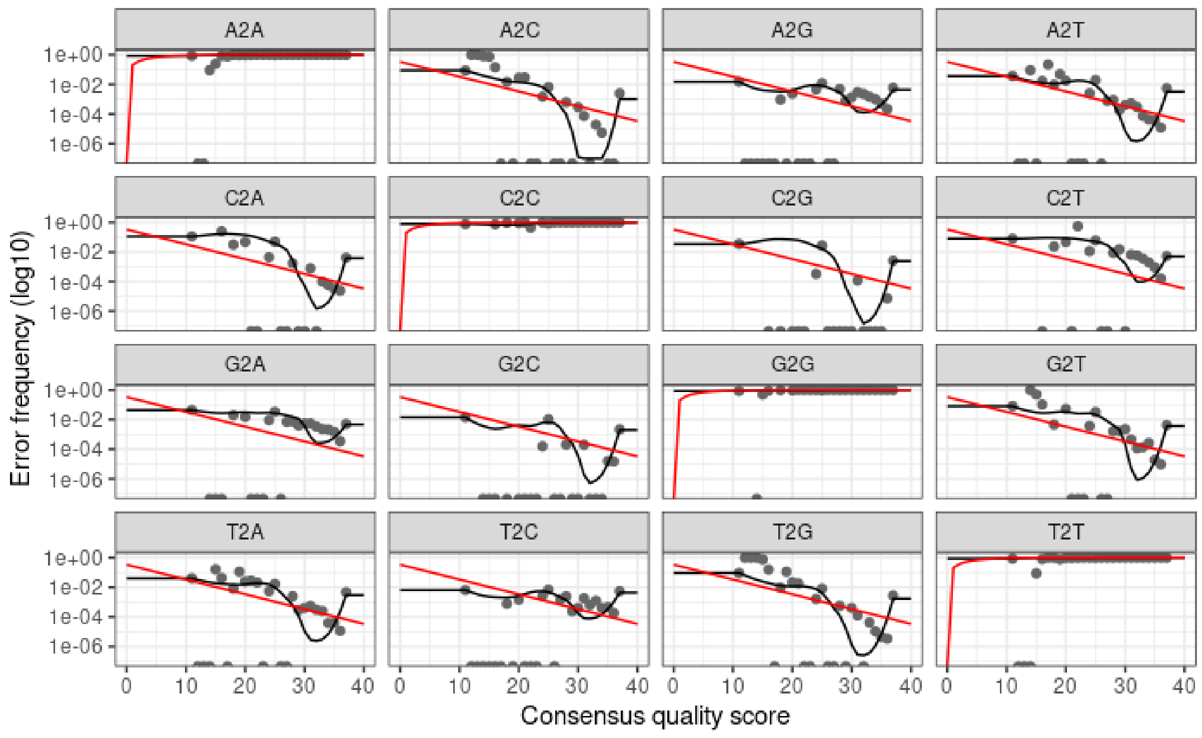

**Figure 3:** Error rate estimation in the **DUPE** dataset

by errors in the data set. Hence, a low p-value sequence can be considered as a real sequence that is not caused by random errors. On the contrary, if the sequence has a high p-value it would likely be caused by errors and will not be kept. This is the last denoising step for the reverse and forward reads.

### 2.3.6. Merge Reads

Once the sample sequences in the forward and reverse reads have been inferred independently, it is time to merge them, throwing out the pairs of reads that do not match. It will return a data frame corresponding to each successfully merged sequence.

### 2.3.7. Chimera Checking and Removal

A chimera is a fusion of two or more parent sequences and can be created during the PCR amplification process. As we want to characterize the sequences coming from the population, fusions of sequences can bias the results and hence need to be removed. As chimeras are relatively rare, the pipeline searches for them amongst the least abundant reads. To spot the chimera sequences, the pipeline performs multiple sequence alignments starting from the least abundant read, and matching them with more abundant reads. The pipeline computes all possible sequence alignment for each combination of reads. When a chimera is detected, it is removed directly from the sequence table.

In general, more than 90% of the unique sequences identified are bimeras and most of the total reads shouldn't be identified as chimeras.

If there are too many chimeras, the analyst should check if the 20 first base pairs of the reads have been trimmed. Indeed, they contain primers that can artificially increase the number of chimeras. Otherwise, the analyst should try to trim more of the low-quality base pairs.

#### 2.3.8. Assign Taxonomy

This is the part I have been mainly working on. The analyst should provide as an input a reference database in the FASTA format. FASTA is a text file format used to store biological sequences (nucleic or proteic) and is a standard format used in bioinformatics. The first line starts with ">" followed by a description of the sequence (i.e., Domain / Phylum / Class / Order / Family / Genus in our case), and the second line is the corresponding DNA or amino acids sequence (i.e., DNA sequences of the *nifH* gene in our case).

Then DADA2 will infer the taxonomy and build the annotation based on the sequence similarity.

## 2.4. NCBI Databases

The National Center of Biotechnology Information (NCBI [Sayers et al., 2022]), is an American institute developing software to analyze genome data. NCBI is not itself a database but it aggregates many databases such as GenBank, RefSeq, and PubMed which are the most famous ones. Here I will only present the main ones that are relevant for our study.

The **gene** database integrates information from a wide range of species. A record may include nomenclature, Reference Sequences (RefSeqs), maps, pathways, variations, phenotypes, and links to the genome, phenotype, and locus-specific resources worldwide.

The **proteins** database is a collection of sequences from several sources, including translations from annotated coding regions in GenBank, RefSeq, and TPA, as well as records from SwissProt, PIR, PRF, and PDB. Protein sequences are the fundamental determinants of biological structure and function.

It is really interesting in our case because the *nifH* gene is coding for a protein and hence we can access the taxonomy and many other interesting information on different databases. The downside is that we obtain the amino acid sequences and not the DNA ones and there are possibly many duplicates.

The **Identical Protein Groups (IPG)** database takes in my opinion the best of both worlds: we obtain all the sequences clustered when Amino Acids sequences are identical, we can obtain the ID of the corresponding DNA sequences and it gathers the information from different databases such as Swissprot, GenBank, PIR, etc.

### 2.5. Accessing the NCBI library from Python through *Entrez*

*Entrez*, Global Query Cross-Database Search System, [for Biotechnology Information, 2004] is a searching tool provided by NCBI enabling one to browse through the information of over 20 databases such as Swiss-Prot, PRF, and PIR-International.

It is an indexing and retrieving system gathering data from various sources and handing it back in a uniform format such as FlatFile, Fasta, or XML. Thanks to it, it is possible to retrieve information without going through the NCBI website, which makes it easier to automatize when working on large datasets like the one we want to build.

We used the Extensible Markable Language (**XML**) format, which is a markup language and a file format for storing. It is made to be readable by both humans and machines and it structures the information. In the case of NCBI, it is used to gather all the concerning a sequence, such as the sequences, the authors, the paper it has been published in, the sequence, the taxonomy, etc. in the same file.

## 3. Results

I started by adapting the R DADA2 [Callahan et al., 2016] pipeline implemented by Cabral [2017] to compute the annotation rates as a function of the taxonomic rank of sequences. Applying this to the DUPE database [Benavides et al., 2022] revealed a rather low annotation rate, as explained in more details in Section 3.1).

Therefore, I decided to try to improve the reference database by rebuilding it from the NCBI databases. This has been done using Python scripts that I fully developed myself. To do so I explored four different approaches that did not lead to a satisfying result and I will present my last attempt in Section 3.2.

## 3.1. Evaluation of the annotation rates

In order to check the annotation rate produced by the current reference database, I ran the DADA2 pipeline for the DUPE dataset (see Section A.1). The outcome is represented Figure 4, which depicts the annotation rate of the processed sequences where the low quality reads and chimeras have already been removed. We can see that the annotation rate decreases along the Taxonomic rank. The annotation rate is high at the Kingdom level (89.6%), however, it quickly drops to 24.1% for the Order level.

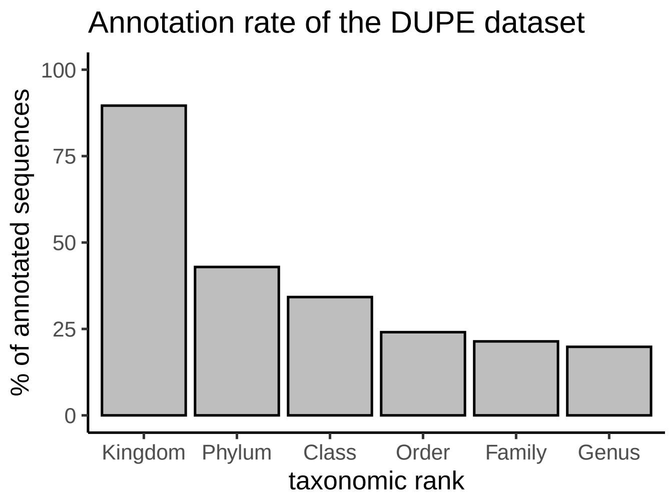

**Figure 4:** Annotation rate of the sequences coming from the DUPE expedition after being processed by the DADA2 pipeline using the current reference database.

With such a level of annotation, it can be tough to characterize the population because we need to assume that the annotation rate of the sequences is uniformly distributed over the species.

If the diversity of a species is well represented in the reference database, then it will well cover the diversity of the sequences and the most of the sequences from the species will be annotated when aligned over the reference database. On the opposite, if there are few similar sequences for a given species, then it is most likely that the sequences diverging too much from the pool of reference sequences will not be annotated, leading to a non-uniform annotation rate.

Considering all the annotated sequences would allow to annotate the sequences to the best of our capacity. It will however not solve the disparity of sequence annotation because not all the species are studied to the same extent and thus their diversiy will not be as well represented.

This is why, I will explain in the next section how to build a more complete reference database to improve the annotation rate even though it will not fully address the diversity representativity problems.

## 3.2. Improvement of the reference database

### 3.2.1. Enriching the database using BLAST

Our initial goal was to start from the current reference database made by Moynihan [2020] and extract the unannotated sequences. It would then have been possible to BLAST [Altschul et al., 1990] them using NCBI in order to extract sequences with a relevant alignment and then enrich back the database. However, it revealed to be a bad idea for the following reasons:

1. This reference database has been manually updated, i.e., some sequences have been inserted afterward, which is makes the quality of the database difficult to guarantee;
2. This reference database is not updated automatically. Unfortunately, the general bacteria taxonomy has recently changed [Hugenholtz et al., 2021] and the reference database had to be manually revised;
3. The first version of this reference database has been created using ARBitrator [Heller et al., 2014]. To the best of my understanding, the construction of the database starts from a selection of a few sequences which are considered to be representative of the diversity of the *nifH* genes. Then the ARBitrator program repeatedly looks for similar sequences in NCBI. This similiarity measure does not control for the *nifH* membership and, as a consequence, the resulting database both contains some non-*nifH* sequences and misses some of the *nifH* sequences. Since the current reference database is based on the one created with ARBitrator, this problem remains.

For all these reasons, building a new database from scratch in a fully automated way seemed more promising than trying to build on top of something I was not fully confident in.

### 3.2.2. Making a new database from scratch

Unlike the approach proposed in ARBitrator [Heller et al., 2014], my goal was to create a program that gathers the sequences and the taxonomy from NCBI (and in the future maybe also from other databases) for **all** and **only** the available and annotated records of the DNA sequences of the *nifH* gene. In a first approach and for scalability reasons, I chose to make a program that fetches the annotated *nifH* sequences only from NCBI.

To do so I used the *Identical Protein Groups*. For a given QUERY, we obtain one report for each identical amino acids sequence, and inside of each report there is the reference of the DNA file, the database it is present in, Start and Stop (most of the time this DNA sequence is part of a bigger DNA sequence) and the strand.

However, *Entrez* does not enable to download the DNA sequences of the IPG reports coming from the IPG database. Hence to obtain the DNA sequences, we need to extract the information from the IPG report and then make a request for the DNA sequence in the gene database.

Thus, from the information contained in the IPG reports, we can download each DNA sequence with the corresponding taxonomy and make a reference database out of it. Running my Python code (see Section A.2 took about 6 hours, downloaded about 400Mb and produced a database with around 3,500 unique sequences.

Unfortunately, I obtained inconsistencies in some reports with sequences of tens of thousand bp long whereas the *nifH* gene is around 900 bp long. I contacted the NCBI support team who gave me some leads: non identical DNA sequences may be present inside the same IPG report, and all reports should thus be downloaded to be included in the reference database. The resulting database could then comprise about 5-10 times more sequences (, whereas the reference database comprises $\approx$ 20,000 non-unique sequences with a mixture of both *nifH* and *non-nifH* sequences).

Although I did not have the time to update my Python code (see Section A.2) after their reply, here is how we should proceed:

1. Gather all the IDs of the IPG reports from a Query request
2. Download the corresponding IPG report
3. Extract all references for the DNA sequences of the IPG report coming from NCBI and fill a table with it.
4. Fetch with *Entrez* the XML report for each sequence in the previous table
    - Different formats can be obtained but the most practical one is XML because it

    contains the taxonomy and the sequence in the same file. Moreover, it has a structure making it easy to access specific data inside the file.
  - As most of the DNA sequences the IPG report is redirecting to are genomes or groups of genes, the DNA sequence, as well as the DNA strand, the starting and stopping nucleotides have to be given.

5. Make a FASTA file by extracting for each file the taxa and the sequence.
6. Normalize the taxonomy
   - Some of the bacteria in the reference have sub-classes that are making the taxonomy non-uniform and should be suppressed with regular expressions.

Adapting the code to take into account the advices from the NCBI support team in steps 1 to 5 is a matter of a few days of work. Step 6 was however not started at all but it should require a few days of work too.

# 4. Discussion and conclusions

Doing a BLAST of the un-annotated sequences over the NCBI databases to then enrich the current reference database with the relevant sequences obtained revealed to be a bad idea for the following reasons: the current database is manually updated and hence might contain some human errors and, due to the way it was first created, it both contains some non-*nifH* sequences and misses some *nifH* sequences.

For these reasons it seemed more reasonable to create a new reference database from scratch by gathering all the annotated DNA sequences of the *nifH* gene in the NCBI databases (the steps are described in section 3.2.2). Although I did not have the time to fully implement it, some encouraging results were obtained. Some minor modifications to the Python code (available in A.2) are still required to automatically build the *nifH* reference database from the databases available on NCBI. It will however only contain the sequences present inside the NCBI databases. The other databases such as UK-Prot and Swiss-Prot do not have the same API as NCBI and will require further developments to retrieve the data and then merge the resulting reference databases.

It is unfortunately too soon to predict by how much the annotation rate of the new database would increase.

## References

S. F. Altschul, W. Gish, W. Miller, E. W. Myers, and D. J. Lipman. Basic local alignment search tool. *Journal of Molecular Biology*, 215(3):403–410, October 1990. ISSN 0022-2836. doi: 10.1016/S0022-2836(05)80360-2.

Roey Angel, Maximilian Nepel, Christopher Panhölzl, Hannes Schmidt, Craig W. Herbold, Stephanie A. Eichorst, and Dagmar Woebken. Evaluation of Primers Targeting the Diazotroph Functional Gene and Development of NifMAP – A Bioinformatics Pipeline for Analyzing nifH Amplicon Data. *Frontiers in Microbiology*, 9:703, April 2018. ISSN 1664-302X. doi: 10.3389/fmicb.2018.00703. URL https://www.ncbi.nlm.nih.gov/pmc/articles/PMC5936773/.

Mar Benavides, Sophie Bonnet, Frédéric A. C. Le Moigne, Gabrielle Armin, Keisuke Inomura, Søren Hallstrøm, Lasse Riemann, Ilana Berman-Frank, Emilie Poletti, Marc Garel, Olivier Grosso, Karine Leblanc, Catherine Guigue, Marc Tedetti, and Cécile Dupouy. Sinking Trichodesmium fixes nitrogen in the dark ocean. *The ISME Journal*, 16(10):2398–2405, October 2022. ISSN 1751-7370. doi: 10.1038/s41396-022-01289-6. URL https://www.nature.com/articles/s41396-022-01289-6. Number: 10 Publisher: Nature Publishing Group.

Damien Cabral. Microbiome/Metagenome Analysis Workshop: DADA2 - YouTube, November 2017. URL https://www.youtube.com/watch?v=wV5_z7rR6yw.

Benjamin J. Callahan, Paul J. McMurdie, Michael J. Rosen, Andrew W. Han, Amy Jo A. Johnson, and Susan P. Holmes. DADA2: High-resolution sample inference from Illumina amplicon data. *Nature Methods*, 13:581–583, 2016. doi: 10.1038/nmeth.3869. URL https://doi.org/doi:10.18129/B9.bioc.dada2.

Alba Filella, Lasse Riemann, France Van Wambeke, Elvira Pulido-Villena, Angela Vogts, Sophie Bonnet, Olivier Grosso, Julia M. Diaz, Solange Duhamel, and Mar Benavides. Contrasting Roles of DOP as a Source of Phosphorus and Energy for Marine Diazotrophs. *Frontiers in Marine Science*, 9, 2022. ISSN 2296-7745. URL https://www.frontiersin.org/articles/10.3389/fmars.2022.923765.

National Center for Biotechnology Information. Entrez molecular sequence database system. http://www.ncbi.nlm.nih.gov/Entrez/, February 2004. Originally published on https://www.ncbi.nlm.nih.gov/Web/Search/entrezfs.html.

John Christian Gaby and Daniel H. Buckley. A comprehensive aligned nifH gene database: a multipurpose tool for studies of nitrogen-fixing bacteria. *Database: The Journal of Biological Databases and Curation*, 2014:bau001, 2014. ISSN 1758-0463. doi: 10.1093/database/bau001.

Philip Heller, H. Tripp, Kendra Turk-Kubo, and Jonathan Zehr. ARBitrator: A software pipeline for on-demand retrieval of auto-curated nifH sequences from GenBank. *Bioinformatics (Oxford, England)*, 30, July 2014. doi: 10.1093/bioinformatics/btu417.

Philip Hugenholtz, Maria Chuvochina, Aharon Oren, Donovan H. Parks, and Rochelle M. Soo. Prokaryotic taxonomy and nomenclature in the age of big sequence data. *The ISME Journal*, 15(7):1879–1892, July 2021. ISSN 1751-7370. doi: 10.1038/s41396-021-00941-x. URL https://www.nature.com/articles/s41396-021-00941-x. Number: 7 Publisher: Nature Publishing Group.

Angela Knapp. The sensitivity of marine N2 fixation to dissolved inorganic nitrogen. *Frontiers in Microbiology*, 3, 2012. ISSN 1664-302X. URL https://www.frontiersin.org/articles/10.3389/fmicb.2012.00374.

Molly Moynihan. nifHdada2: nifH dada2 beta version (v1.0.1). Zenodo. https://doi.org/10.5281/zenodo.3958370, July 2020. Originally published on https://github.com/moyn413/nifHdada2.

Eric W. Sayers, Evan E. Bolton, J. Rodney Brister, Kathi Canese, Jessica Chan, Donald C. Comeau, Ryan Connor, Kathryn Funk, Chris Kelly, Sunghwan Kim, Tom Madej, Aron Marchler-Bauer, Christopher Lanczycki, Stacy Lathrop, Zhiyong Lu, Francoise Thibaud-Nissen, Terence Murphy, Lon Phan, Yuri Skripchenko, Tony Tse, Jiyao Wang, Rebecca Williams, Barton W. Trawick, Kim D. Pruitt, and Stephen T. Sherry. Database resources of the national center for biotechnology information. *Nucleic Acids Research*, 50(D1):D20–D26, January 2022. ISSN 1362-4962. doi: 10.1093/nar/gkab1112.

Lance C. Seefeldt, Zhi-Yong Yang, Simon Duval, and Dennis R. Dean. Nitrogenase reduction of carbon-containing compounds. *Biochimica et Biophysica Acta (BBA) - Bioenergetics*, 1827 (8):1102–1111, August 2013. ISSN 0005-2728. doi: 10.1016/j.bbabio.2013.04.003. URL https://www.sciencedirect.com/science/article/pii/S0005272813000741.

Arvind Singh, Naveen Gandhi, and R. Ramesh. Contribution of atmospheric nitrogen deposition to new production in the nitrogen limited photic zone of the northern Indian Ocean.

*Journal of Geophysical Research: Oceans*, 117(C6), 2012. ISSN 2156-2202. doi: 10.1029/2011JC007737. URL https://onlinelibrary.wiley.com/doi/abs/10.1029/2011JC007737. _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1029/2011JC007737.

Gary Stacey, Robert H. Burris, and Harold J. Evans. *Biological Nitrogen Fixation*. Springer Science & Business Media, April 1992. ISBN 978-0-412-02421-4.

Jonathan P. Zehr and Douglas G. Capone. Changing perspectives in marine nitrogen fixation. *Science (New York, N.Y.)*, 368(6492):eaay9514, May 2020. ISSN 1095-9203. doi: 10.1126/science.aay9514.

Jonathan P. Zehr and Bess B. Ward. Nitrogen Cycling in the Ocean: New Perspectives on Processes and Paradigms. *Applied and Environmental Microbiology*, 68(3):1015–1024, March 2002. doi: 10.1128/AEM.68.3.1015-1024.2002. URL https://journals.asm.org/doi/10.1128/AEM.68.3.1015-1024.2002. Publisher: American Society for Microbiology.

Jonathan P. Zehr, Bethany D. Jenkins, Steven M. Short, and Grieg F. Steward. Nitrogenase gene diversity and microbial community structure: a cross-system comparison. *Environmental Microbiology*, 5(7):539–554, July 2003. ISSN 1462-2912. doi: 10.1046/j.1462-2920.2003.00451.x.

# A. Appendix

## A.1. DADA2 Pipeline

Load DADA2

```r
library(dada2); packageVersion("dada2")
```

Raw FASTQ files

```r
# Specify path where FASTQ files are located
path <- "../data/dupe_train" # change to the directory where the FASTQ files are located after unzipping

# Sort the files to ensure reverse and forward reads are in the same order
fnFs <- sort(list.files(path, pattern="_R1_001.fastq"))
fnRs <- sort(list.files(path, pattern="_R2_001.fastq"))

# Extract sample names, assuming filenames have format: SAMPLENAME_XXX.fastq
sample.names <- sapply(strsplit(fnFs, "_"), `[`, 1) # here I don't get the end of the code but it retrieve the name of the samples

# Specify the full path to the fnFs and fnRs
fnFs <- file.path(path, fnFs)
fnRs <- file.path(path, fnRs)
```

Trim the reads

```r
# Plot average quality score
plotQualityProfile(fnFs[1:2])# I don't know what is negative but the curve is weird
plotQualityProfile(fnRs[1:6])


# Create a new file path to store filtered and trimmed reads
filt_path <- file.path(path, "filtered") # place the filtered files in a "filtered" subdirectory

# Rename filtered files
filtFs <- file.path(filt_path, paste0(sample.names, "_F_filt.fastq.gz"))
```

```r
filtRs <- file.path(filt_path, paste0(sample.names, "_R_filt.
    fastq.gz"))
```

```r
# Quality Filtering and trimming
out <- filterAndTrim(fnFs, filtFs, fnRs, filtRs, truncLen=c
    (250,250),
                maxN=0, maxEE=c(2,2), truncQ=2, rm.phix=TRUE,
                compress=TRUE, multithread=TRUE)

head(out)
```

Error rate estimation

```r
# Estimate the error model for DADA2 algorithm using reverse
    reads
errF <- learnErrors(filtFs, multithread=TRUE)
# do the same on the reverse read and it will take more cycles
    because the reverse reads are worse
errR <- learnErrors(filtRs, multithread=TRUE)

# Plot error rates for all possible bases transitions
plotErrors(errF, nominalQ=TRUE) # black:observed error rate, red
    :expected
plotErrors(errR, nominalQ=TRUE) # black:observed error rate, red
    :expected
```

Dereplicate the reads

```r
# Dereplicate FASTQ files to speed up computation
derepFs <- derepFastq(filtFs, verbose=TRUE)
derepRs <- derepFastq(filtRs, verbose=TRUE)

# Name the derep-class objects by the sample names
names(derepFs) <- sample.names
names(derepRs) <- sample.names
```

Sample inference using the DADA2 algorithm

```r
# Apply core sequence-variant inference algorithm to reverse
    reads
dadaFs <- dada(derepFs, err=errR, multithread=TRUE)
dadaRs <- dada(derepRs, err=errR, multithread=TRUE)
```

Merge reads

```r
# Merge denoised reads
mergers <- mergePairs(dadaFs, derepFs, dadaRs, derepRs, verbose=
    TRUE) # will only merge perfectly overlapped seq so can
    addmaxMismatch 1 or 2 if lots of reads are not merging

# Inspect the merger data.frame from the first sample
head(mergers[[1]])

seqtab <- makeSequenceTable(mergers)
dim(seqtab)

table(nchar(getSequences(seqtab)))
```

Chimera checking and removal

```r
# Perform de novo chimera sequence detection and removal
seqtab.nochim <- removeBimeraDenovo(seqtab, method="consensus",
    multithread=TRUE, verbose=TRUE)
dim(seqtab.nochim)


# Calculate the proportion of the non-chimeric RSVs (reads)
sum(seqtab.nochim)/sum(seqtab)
```

Assign taxonomy

```r
# Assign taxonomy using RDP database (greengenes and Silva also
    available)
# This is performed in two steps: this first one assigns phylum
    to genus
taxa <- assignTaxonomy(seqtab.nochim,"../data/nifH_dada2_phylum_
    v1.1.0.fasta", multithread=TRUE) # database we want to use
unname(head(taxa))

class(taxa)

write.table(taxa, '../data/taxa.txt') # To save the table
```

## A.2. Implementation with Python

Import packages

```python
# Import packages
from Bio import Entrez
from Bio import SeqIO


# Set email adress
Entrez.email = "remi.legrand38@gmail.com.com"
```

Gather ID of the IPG report from a QUERY

```python
# Function to search data on NCBI
def search_id(query, number_seq):
    handle = Entrez.esearch(db="ipg", term=query,
                            retmax=number_seq)  # number of
                                sequences
    record = Entrez.read(handle)
    handle.close()
    return record["IdList"]
```

```python
# Create a list containing the ID
query = "((nifH[Gene Name]) AND 100:350[Sequence Length]) NOT
    uncultured"
seq_id = search_id(query, 20000)
print(seq_id)
len(seq_id)
```

Download the IPG reports

```python
def ipg_report(ipg_accession):
    # Use the efetch function to retrieve the IPG record in XML
        format
    handle = Entrez.efetch(db="ipg", id=ipg_accession,
                           rettype="ipg", retmode="text")
    ipg_record = handle.read()

    # Save the IPG record to a file
    filename = f"ipg_report/ipg_{ipg_accession}.txt"
    with open(filename, "wb") as file:
        file.write(ipg_record)

    print(f"IPG record saved to '{filename}'")
```

```python
for id in seq_id:
    ipg_report(id)
```

Extract the first line of the IPG report present in NCBI and put it in a panda data frame

```python
import pandas as pd
import numpy as np


gene_list = pd.DataFrame(columns=["accession", "start", "stop",
    "strand"])

for name in seq_id:
    file = "ipg_report/ipg_"+name+".txt"
    ipg = pd.read_csv(file, sep="\t")

    insdc = np.array(ipg.loc[:, "Source"] == "INSDC", dtype="
        bool")
    refseq = np.array(ipg.loc[:, "Source"] == "RefSeq", dtype="
        bool")

    in_ncbi = ipg.loc[refseq | insdc, :]
    in_ncbi = in_ncbi.reset_index()
    url = str()
    if len(in_ncbi) != 0:
        name = str(in_ncbi.loc[0, "Nucleotide Accession"])
        start = str(in_ncbi.loc[0, "Start"])
        stop = str(in_ncbi.loc[0, "Stop"])
        if str(in_ncbi.loc[0, "Strand"]) == "+":
            strand = "1"
        elif str(in_ncbi.loc[0, "Strand"]) == "-":
            strand = "2"
        gene_list.loc[len(gene_list)] = [name, start, stop,
            strand]
    else:
        print(file+" is not in NCBI")

gene_list
```

Fetch with Entrez the INSDSeq XML report for each line the the previous table and download it

```python
from Bio import Entrez
```

```python

Entrez.email = "remi.legrand38@gmail.com.com"


def download_insdseq_xml(index):
    accession = gene_list.loc[index, "accession"]
    start = gene_list.loc[index, "start"]
    stop = gene_list.loc[index, "stop"]
    strand = gene_list.loc[index, "strand"]
    handle = Entrez.efetch(db='nucleotide', id=accession,
       rettype='gb',
                                  retmode='xml', seq_start=start,
                                     seq_stop=stop, strand=strand)
    # Save the XML data to a file
    with open(f'insdseq_xml_report/{accession}_INSDSeq.xml', 'wb
       ') as file:
        file.write(handle.read())

    print('INSDSeq XML report downloaded successfully.')


for i in range(len(gene_list)):
    download_insdseq_xml(i)
```

Make a FASTA file by extracting for each file the taxa and the sequence

```python
import os
import xml.etree.ElementTree as ET

files = os.listdir('insdseq_xml_report')

with open('reference_db_ipg.fasta', 'w') as file:
    for xml_file in files:
        path = 'insdseq_xml_report/' + xml_file
        tree = ET.parse(path)
        root = tree.getroot()
        try:
            tax = root.find("./GBSeq/GBSeq_taxonomy").text
            seq = root.find("./GBSeq/GBSeq_sequence").text
        except AttributeError:
            print("in", path, "sequence or taxonomy not found")
        file.write(">" + tax + "\n")
        file.write(seq + "\n")
```